Ferdowsi University of Mashhad

Iranian Journal of Numerical Analysis and Optimization

Volume 12 , Number 2

Summer 2022

Serial Number: 22

Ferdowsi University of Mashhad, Iran

In the Name of God

Iranian Journal of Numerical Analysis and Optimization (IJNAO)

This journal is authorized under the registration No. 174/853 dated 1386/2/26 (2007/05/16), by the Ministry of Culture and Islamic Guidance.

Volume 12, Number 2, Summer 2022

ISSN-Print: 2423-6977, **ISSN-Online:** 2423-6969

Publisher: Faculty of Mathematical Sciences, Ferdowsi University of Mashhad

Published by: Ferdowsi University of Mashhad Press

Printing Method: Electronic

Address: Iranian Journal of Numerical Analysis and Optimization

Faculty of Mathematical Sciences, Ferdowsi University of Mashhad

P.O. Box 1159, Mashhad 91775, Iran.

Tel. : +98-51-38806222 , **Fax:** +98-51-38807358

E-mail: ijnao@um.ac.ir

Website: http://ijnao.um.ac.ir

This journal is indexed by:

- Scopus
- Zentralblatt
- ISC
- SID
- Civilica
- Magiran
- DOAJ
- OAJI
- AcademicKeys
- COPE
- Mendeley
- Academia.edu
- Linkedin

- The Journal granted the International degree by the Iranian Ministry of Science, Research, and Technology.

Iranian Journal of Numerical Analysis and Optimization

Volume 12, Number 2, Summer 2022

Ferdowsi University of Mashhad - Iran

Iranian Journal of Numerical Analysis and Optimization

This journal is published under the auspices of Ferdowsi University of Mashhad

* Full Professor

** Associate Professor

We would like to acknowledge the help of Miss Narjes khatoon Zohorian in the preparation of this issue.

Letter from the Editor–in–Chief

I would like to welcome you to the Iranian Journal of Numerical Analysis and Optimization (IJNAO). This journal is published two issues per year and supported by the Faculty of Mathematical Sciences at the Ferdowsi University of Mashhad. Faculty of Mathematical Sciences with three centers of excellence and three research centers is well-known in mathematical communities in Iran.

The main aim of the journal is to facilitate discussions and collaborations between specialists in applied mathematics, especially in the fields of numerical analysis and optimization, in the region and worldwide.

Our vision is that scholars from different applied mathematical research disciplines, pool their insight, knowledge and efforts by communicating via this international journal.

In order to assure high quality of the journal, each article is reviewed by subject-qualified referees.

Our expectations for IJNAO are as high as any well-known applied mathematical journal in the world. We trust that by publishing quality research and creative work, the possibility of more collaborations between researchers would be provided. We invite all applied mathematicians especially in the fields of numerical analysis and optimization to join us by submitting their original work to the Iranian Journal of Numerical Analysis and Optimization.

Ali R. Soheili

Contents

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 250–276
DOI: 10.22067/ijnao.2021.71891.1053
https://ijnao.um.ac.ir/

A fourth-order optimal numerical approximation and its convergence for singularly perturbed time delayed parabolic problems

J. Mohapatra* and L. Govindarao

Abstract

This paper presents a numerical solution for a time delay parabolic problem (reaction-diffusion) containing a small parameter. The numerical method combines the implicit Crank–Nicolson scheme for the time derivative on the uniform mesh and the central difference scheme for the spatial derivative on the Shishkin-type meshes. It is shown to be second-order uniformly convergent in time and space. Then Richardson extrapolation technique is applied to enhance the accuracy from second-order to fourth-order. The error analysis is carried out, and the method is proved to be uniformly convergent. These two methods are applied to two test examples in support of the theoretical results.

AMS subject classifications (2020): 65M06, 65M12.

Keywords: Time delayed parabolic problem; Boundary layer; Post processing technique; Singular perturbation.

---

* Corresponding author

Received 7 August 2021; revised 10 November 2021; accepted 20 November 2021

Jugal Mohapatra
Department of Mathematics, National Institute of Technology Rourkela, 769008, India, e-mail: jugal@nitrkl.ac.in

Lolugu Govindarao
Department of Mathematics, Amrita School of Engineering Coimbatore, Amrita Vishwa Vidyapeetham, Amrita University, Ettimadai, Tamilnadu, India, e-mail: l_govindarao@cb.amrita.edu

1 Introduction

In singularly perturbed delay partial differential equations (SPPDEs), a small parameter affects the highest derivative of partial differential equations (PDEs) and also contains more than one delay term in the time direction. These SPPDEs have existence in subjects like physical/chemical science, biology, and ecology. Time delay (large) diffusion problems (the present time depends upon the past) appear in mathematical models in population dynamics [13, 26, 17] and also in biological modeling [29].

Consider the following time delay reaction-diffusion initial-boundary-value problems (IBVPs):

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,s}\Big) z(y,\ s) = -\mathfrak{b}(y,\ s) z(y,\ s-\gamma) + \mathfrak{f}(y,s), & (y,s) \in \mathfrak{D}, \\ z(y,s) = \Theta_b(y,\ s), & (y,s) \in \mathfrak{d}_b, \\ z(0,s) = \Theta_l(s), & \text{on} \quad \mathfrak{d}_l = \{(0,s) : 0 \le s \le \mathcal{S}\}, \\ z(1,s) = \Theta_r(s), & \text{on} \quad \mathfrak{d}_r = \{(1,s) : 0 \le s \le \mathcal{S}\}, \end{cases} \tag{1}$$

where $L_{\varepsilon,y} z(y,\ s) = -\varepsilon z_{yy}(y,\ s) + \mathfrak{p}(y) z(y,\ s)$. Here $= (0,\ 1)$, $\mathfrak{D} = \Pi \times (0,\ \mathcal{S}]$, $\mathfrak{d} = \mathfrak{d}_l \cup \mathfrak{d}_b \cup \mathfrak{d}_r$. Moreover, $\gamma > 0$ is a given constant. This model (1) is singularly perturbed, parabolic in nature with $0 < \varepsilon \ll 1$. Moreover, $\mathfrak{d}_r$ and $\mathfrak{d}_l$ are the right and the left sides of the domain $\mathfrak{D}$, and $\mathfrak{d}_b = [0,\ 1] \times [-\gamma,\ 0]$. The functions $\mathfrak{b}(y,\ s)$ $(\mathfrak{b}(y,\ s) > 0)$, $\mathfrak{p}(y)(\mathfrak{p}(y) \ge \beta > 0)$, the source term $\mathfrak{f}(y,\ s)$ on $\overline{\mathfrak{D}}$ and $\Theta_l(s)$, $\Theta_r(s)$, $\Theta_b(y,\ s)$ on $\mathfrak{d}$, are bounded, sufficiently smooth functions. The assumption on the time $\mathcal{S}$ is $\mathcal{S} = m\gamma$ for $m \in \mathcal{N}$. We also assume that suitable compatibility conditions hold true on the given boundary and initial data to ensure a unique solution for (1) that exhibits boundary layers along with end points of y [1, 19]. These conditions are $\Theta_b(0,0) = \Theta_l(0)$, $\Theta_b(1,0) = \Theta_r(0)$, and

$$\frac{d\Theta_l(0)}{ds} - \varepsilon \frac{\partial^2 \Theta_b(0,\ 0)}{\partial y^2} + \mathfrak{p}(0)\Theta_b(0,\ 0) = -\mathfrak{b}(0,0)\Theta_b(0,\ -\gamma) + \mathfrak{f}(0,\ 0),$$
$$\frac{d\Theta_r(0)}{ds} - \varepsilon \frac{\partial^2 \Theta_b(1,\ 0)}{\partial y^2} + \mathfrak{p}(1)\Theta_b(1,\ 0) = -\mathfrak{b}(1,\ 0)\Theta_b(1,\ -\gamma) + \mathfrak{f}(1,\ 0).$$

Due to the small parameter ε involved in the problem (1), the classical methods on equal step length for solving (1) fail to give accurate results. They are mostly unstable and unacceptable [22, 27]. Hence, we choose the fitted mesh idea through the nonuniform mesh as given in [14, 20, 22, 25, 27] and the references therein. One can refer [1, 7, 10, 9, 11, 18, 16, 23] for some order enhancing methods of IBVPs. Some articles are available, which discuss both the analytical and the numerical techniques for SPPDEs in literature. Ansari, Bakr, and Shishkin [1] numerically solved the problem (1) on the Shishkin mesh. Das and Natesan [6] gave details of numerical results for the time delay parabolic convection diffusion problem. Indeed, the methods discussed above

using numerical techniques are of first- or second-order accurate. Hence, order enhancing techniques are needed for the problem (1), which is the main aim of this work.

Richardson extrapolation through averaging the numerical solutions computed on two embedded meshes provides a good approximation to the exact solution. This helps to increase the accuracy and the order of convergence. Mohapatra and Natesan [21] used the extrapolation technique for solving singularly perturbed delay BVPs while Shishkin and Shishkina [28] applied on time dependent IBVPs of reaction-diffusion type. This technique is used in [5] for convection-diffusion singularly perturbed parabolic problems on the adaptive mesh. This article aims to get a fourth-order accurate solution for (1) using the extrapolation technique. Initially, the central difference scheme is used on Shishkin-type meshes and the Crank–Nicolson method on temporal direction on uniform mesh. Here, problem (1) is solved with M number of mesh points in spatial and N number of mesh points in time direction. After that (1) is solved by the above methods with $2M$ and $2N$ number of mesh points. Then the rate of convergence increases from second- to fourth-order globally by taking a proper combination of these two solutions.

The rest portion is arranged as follows. In Section 2, we describe the continuous solution to the problem. Section 3 studies the construction of the numerical schemes. In Section 4, we implement the post-process ideas and the theoretical analysis. Section 5 presents the numerical results through plots and tables. We denote $\mathcal{C}$ and the subscripted $\mathcal{C}$'s as constants, which are positive, independent of the small parameter (ε) and spatial and time mesh sizes. The error is represented in the supremum norm ($\|\cdot\|_\infty$). It is defined as $||\mathfrak{h}||_\infty = \sup_{(y,s)\in\mathfrak{D}} |\mathfrak{h}(y,s)|$ for any function $\mathfrak{h}$ on the domain $\mathfrak{D}$.

2 Analytic solution and its behavior

As the perturbed parameter $\varepsilon \to 0$ in (1), the problem reduces as given below:

$$\begin{cases} \dfrac{\partial z_0(y,\ s)}{\partial s} + \mathfrak{p}(y) z_0(y,\ s) = -\mathfrak{b}(y,\ s) z_0(y, s-\gamma) + \mathfrak{f}(y,\ s), & (y,\ s) \in \mathfrak{D}, \\ z_0(y,\ s) = \theta_b(y,\ s), & (y,\ s) \in \mathfrak{d}_b. \end{cases} \tag{2}$$

It is clear that the solution to (1) has boundary layers on $\mathfrak{d}_l$ and $\mathfrak{d}_r$. The characteristics of (2) are the vertical lines $y = C$, which implies that boundary layers arising in the solution are parabolic type. The operator $\left(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\right)$ in (1) satisfies the following lemma known as the maximum principle.

Lemma 1. Suppose that $\Psi(y,\ s) \in \mathcal{C}^0(\overline{\mathfrak{D}}) \cap \mathcal{C}^2(\mathfrak{D})$ satisfies $\left(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\right)\Psi(y,\ s) \geq 0$ in $\mathfrak{D}$ and that $\Psi(y,\ s) \geq 0$ on $\mathfrak{d}$. Then $\Psi(y,\ s) \geq 0$ for all $(y,\ s) \in \overline{\mathfrak{D}}$.

Proof. The proof of this lemma is available in [1]. □

2.1 Solution decomposition

The continuous solution $z(y,\ s)$ to (1) is decomposed as $z = v_r + v_s$. The regular component v_r expresses as $v_r(y,\ s) = v_{r0}(y,\ s) + \varepsilon v_{r1}(y,\ s),\ (y,\ s) \in \overline{\mathfrak{D}}$, where v_{r0} and v_{r1} are the solutions to the following problem:

$$\begin{cases} (v_{r0})_s(y,\ s) + \mathfrak{p}(y)v_{r0}(y,\ s) = -\mathfrak{b}(y,\ s)v_{r0}(y, s-\gamma) + f(y,\ s),\ (y,\ s) \in \mathfrak{D}, \\ v_{r0}(y,\ s) = \Theta_b(y,\ s),\ (y,\ s) \in \mathfrak{d}_b, \\ \\ \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)v_{r1}(y,\ s) = -\mathfrak{b}(y,\ s)v_{r1}(y, s-\gamma) + (v_{r0})_{yy},\ (y,\ s) \in \mathfrak{D}, \\ v_{r1}(y,\ s) = 0, \quad (y,\ s) \in \Gamma. \end{cases} \tag{3}$$

The regular component $v_r(y,\ s)$ satisfies the following problem:

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)v_r(y,\ s) = -\mathfrak{b}(y,\ s)v(y, s-\tau) + \mathfrak{f}(y,\ s), & (y,\ s) \in \mathfrak{D}, \\ v_r(y,\ s) = \theta_b(y,\ s), & (y,\ s) \in \mathfrak{d}_b, \\ v_r(0,t) = v_{r0}(0,\ s), & \text{on} \quad \mathfrak{d}_l = \{(0,\ s) : 0 \le s \le \mathcal{S}\}, \\ v_r(1,t) = v_{r0}(1,\ s), & \text{on} \quad \mathfrak{d}_r = \{(1,\ s) : 0 \le s \le \mathcal{S}\}, \end{cases} \tag{4}$$

and the singular component $v_s(y,\ s)$ satisfies the PDE

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)v_s(y,\ s) = -\mathfrak{b}(y,\ s)v_s(y, s-\gamma), & (y,\ s) \in \mathfrak{D}, \\ v_s(y,\ s) = 0, & (y,\ s) \in \Gamma_b, \\ v_s(0,\ s) = \Theta_l - v_{r0}(0,\ s), & \text{on} \quad \mathfrak{d}_l = \{(0,\ s) : 0 \le s \le \mathcal{S}\}, \\ v_s(1,\ s) = \Theta_r - v_{r0}(1,\ s), & \text{on} \quad \mathfrak{d}_r = \{(1,\ s) : 0 \le s \le \mathcal{S}\}. \end{cases} \tag{5}$$

Now, we can write $v_s = v_{sl} + v_{sr}$, where v_{sl} is the boundary layer part on $\mathfrak{d}_l$ and v_{sr} is the boundary layer part on $\mathfrak{d}_r$. Here v_{sl} and v_{sr} are defined by

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)v_{sl}(y,\ s) = -\mathfrak{b}(y,\ s)v_{sl}(y,\ s-\gamma), \quad (y,\ s) \in \mathfrak{D}, \\ v_{sl}(0,\ s) = \theta_l - v_{r0}(0,\ s)\ (y,\ s) \in \mathfrak{d}_l\ v_{sl}(y,\ s) = 0,\ (y,\ s) \in \mathfrak{d}_b \bigcup\ \mathfrak{d}_r, \\ \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)v_{sr}(y,\ s) = -\mathfrak{b}(y,\ s)v_{sr}(y,\ s-\gamma), \quad (y,\ s) \in \mathfrak{D}, \\ v_{sr}(1,\ s) = \Theta_r - v_{r0}(1,\ s),\ (y,s) \in \mathfrak{d}_r\ v_{sr}(y,s) = 0,\ (y,s) \in \mathfrak{d}_l \bigcup \mathfrak{d}_b. \end{cases} \tag{6}$$

Theorem 1. For all integers $l > 0$ and $m > 0$ with $0 \le l + 2m \le 8$, v_r and v_s, defined in (3) and (5), respectively, satisfy

$$\left\| \frac{\partial^{l+m} v_s}{\partial y^l \partial s^m} \right\|_\infty \leq \mathcal{C}(1+\varepsilon^{2-l/2}), \quad (y,\ s) \in \mathfrak{D},$$

and

$$\left\| \frac{\partial^{l+m} v_r}{\partial y^l \partial s^m} \right\|_\infty \leq \mathcal{C}\varepsilon^{-l/2}\Big(\exp(-y\sqrt{\beta/\varepsilon}) + \exp(-(1-y)\sqrt{\beta/\varepsilon})\Big), \quad (y,\ s) \in \mathfrak{D},$$

$$\left\| \frac{\partial^{l+m} v_{sl}}{\partial y^l \partial s^m} \right\|_\infty \leq \mathcal{C}\varepsilon^{-l/2}\Big(\exp(-y\sqrt{\beta/\varepsilon}))\Big), \quad (y,\ s) \in \mathfrak{D},$$

$$\left\| \frac{\partial^{l+m} v_{sr}}{\partial y^l \partial s^m} \right\|_\infty \leq \mathcal{C}\varepsilon^{-l/2}\Big(\exp(-(1-y)\sqrt{\beta/\varepsilon})\Big), \quad (y,\ s) \in \mathfrak{D}.$$

Proof. One may refer [1] for the details. □

3 Discretization

On $[0,\ \mathcal{S}]$, the uniform time step Δs is used in the time direction such that $\Pi_s^N = \{s_n = n\Delta s,\quad n = 0,\ldots,N,\quad s_N = \mathcal{S},\quad \Delta s = \mathcal{S}/N\}$, $\Pi_s^p = \{s_j = j\Delta s,\quad j = 0,\ldots,p,\quad s_p = \gamma,\quad \Delta s = \gamma/p\}$. Here, p represents the number of mesh points in $[-\gamma, 0]$ and N denotes the number of mesh points in temporal direction on the interval $[0,\ \mathcal{S}]$. The step size (Δs) satisfies $p\Delta s = \gamma$, where $p > 0$ is an integer $s_n = n\Delta s$, $n \geq -p$.

3.1 Discretization of the spatial domain

Let $\xi = \min\left\{\frac{1}{4},\ \rho_0\sqrt{\varepsilon}\ln M\right\}$, where $\rho_0 \geq 2/\beta$ is a mesh transition parameter. We divide $\overline{\Pi} = [0,1]$ into three subdomains as $\overline{\Pi} = \overline{\Pi_l} \cup \overline{\Pi_c} \cup \overline{\Pi_r}$, where $\Pi_l = [0,\xi], \Pi_c = (\xi, 1-\xi]$, and $\Pi_r = (1-\xi, 1]$. Without loss of generality, we assume that M is even and that $M \geq 8$.

Shishkin mesh(S-mesh)

One can find the construction of the S-mesh in [22, 27] briefly described as follows: let us define S-mesh as $\Pi_y^M = \{y_i \in [0,1], 0 \leq i \leq M\}$, where

$$y_i = \begin{cases} \dfrac{4i\xi}{M} & \text{for} \quad 0 \le i \le M/4, \\ \dfrac{2i(1-2\xi)}{M} & \text{for} \quad \dfrac{M}{4}+1 \le i \le \dfrac{3M}{4}, \\ \dfrac{4i\xi}{M} & \text{for} \quad \dfrac{3M}{4}+1 \le i \le M. \end{cases}$$

If $\xi = \dfrac{1}{4}$, then the mesh has equal step length and otherwise when $\xi = \rho_0\sqrt{\varepsilon}\ln M$, the mesh is changing at near the end of $\mathfrak{b}_l$ and $\mathfrak{b}_r$, where $y_i - y_{i-1} = 4\xi M^{-1}$. Therefore, the mesh is piecewise uniform.

Bakhvalov-Shishkin mesh (B-S-mesh)

The idea of constructing the B-S mesh is available in [4, 27]. The mesh is constructed as follows:

$$y_i = \begin{cases} \dfrac{2}{\beta}\left(-\ln\left(1-2(1-\frac{1}{M})\frac{i}{M}\right)\right), & \text{for} \quad 0 \le i \le M/4-1, \\ y_{M/4-1} + \left(\dfrac{y_{M/4+1}-y_{M/4-1}}{M/2+2}\right)(i-M/4+1) & \text{for} \quad \dfrac{M}{4} \le i \le \dfrac{3M}{4}, \\ 1-\dfrac{2}{\beta}\left(-\ln\left(1-2(1-\frac{1}{M})\frac{M-i}{M}\right)\right) & \text{for} \quad \dfrac{3M}{4}+1 \le i \le M. \end{cases}$$

We define the numerical domain $\mathfrak{D}^M = \Pi_y^M \times \Pi_s^N$ on $\mathfrak{D}$ and $\mathfrak{b}^M = \Pi_y^M \times \Pi_s^p$ on $\mathfrak{b}$.

3.2 Semidiscretization

The Crank–Nicolson scheme for the time variable of (1) is given by

$$\begin{cases} z^{-j} = \Theta_b(y,-s_j) & \text{for} \quad j=0,\dots p, \quad y \in \overline{\mathfrak{D}}, \\ \left(I + \frac{\Delta s}{2}\mathfrak{L}_{\varepsilon,y}\right)z^{n+1} = \frac{\Delta s}{2}\left(-\mathfrak{b}^{n+1}z^{n-p+1} - \mathfrak{b}^n z^{n-p} + \mathfrak{f}^{n+1} + \mathfrak{f}^n\right) + \left(I - \frac{\Delta s}{2}\mathfrak{L}_{\varepsilon,y}\right)z^n, \\ z^{n+1}(0) = \Theta_l(s_{n+1}), \quad z^{n+1}(1) = \Theta_r(s_{n+1}), \end{cases} \tag{7}$$

where $\mathfrak{f}^n = \mathfrak{f}(y,t_n)$, $\mathfrak{b}^n = \mathfrak{b}(y,\ s_n)$, $z^n = z(y,\ s_n)$ is the semidiscrete approximation to $z(y,\ s)$ of (1) at $s_n = n\Delta s$. Let $e^{n+1} = z^{n+1} - \widetilde{z}^{n+1}$ be the local truncation error of (7) and let $\widetilde{z}^{n+1}$ be the solution to

$$\begin{cases} \widetilde{z}^{-j} = \Theta_b(y,-t_j) & \text{for} \quad j=0,\dots p, \quad x \in \overline{\mathfrak{D}}, \\ \left(I + \frac{\Delta s}{2}\mathfrak{L}_{\varepsilon,y}\right)\widetilde{z}^{n+1} = \frac{\Delta s}{2}\left(-\mathfrak{b}^{n+1}\widetilde{z}^{n-p+1} - \mathfrak{b}^n \widetilde{z}^{n-p} + \mathfrak{f}^{n+1} + \mathfrak{f}^n\right) + \left(I - \frac{\Delta s}{2}\mathfrak{L}_{\varepsilon,y}\right)(\widetilde{z}^2)^n, \\ \widetilde{z}^{n+1}(0) = \Theta_l(s_{n+1}), \quad \widetilde{z}^{n+1}(1) = \Theta_r(s_{n+1}). \end{cases}$$

Hence, the global error at s^n is given by $E^n = z(y, s^n) - z^n(y)$.

3.3 Bounds on the solution and its derivatives

Consider the global error $E^{n+1} = e^{n+1} + RE^n$, associated with the scheme (7). The transition operator

$$R = \left(I + \frac{\Delta s}{2}\mathfrak{L}_{\varepsilon,y}\right)^{-1}\left(I - \frac{\Delta s}{2}\mathfrak{L}_{\varepsilon,y}\right),$$

is defined as follows: set $z^n = E_n$ as the initial data with null boundary condition and zero source term $\mathfrak{f}$. After one time step of (7), let RE^n be the solution obtained. Using this, we have $E^{n+1} = \sum_{k=0}^{M} R^{n-k} e_{k+1}$. Thus, we claim

$$||R^j||_\infty \leq \mathcal{C} \quad \text{for all} \quad j = 0, 1, \ldots, n. \tag{8}$$

Then it follows that $\sup_{n\Delta s \leq \mathcal{S}} ||E^{n+1}||_\infty \leq \mathcal{C}(\Delta s)^2$. Hence, the scheme (7) is second-order accurate. It may be noted here that (8) is a stability condition. The details of this argument were given in [3, 8].

Lemma 2. If $\left|\frac{\partial^i}{\partial s^i} z(y,\ s)\right| \leq \mathcal{C}$ for all $(y,\ s) \in \overline{\mathfrak{D}}$, for $0 \leq i \leq 3$, then the local error with the method (7) satisfies

$$\|e^{n+1}\| \leq \mathcal{C}(\Delta s)^3. \tag{9}$$

Proof. It can be shown using the same argument discussed in [3]. □

Theorem 2. The global error estimate E_n associated with (7) is given by $\|E^n\|_\infty \leq \mathcal{C}(\Delta s)^2$ for all $n \leq \mathcal{S}/\Delta s$.

Proof. See [8]. □

Theorem 3. The derivatives of $z(y,\ s)$ satisfy the bounds

$$\left|\frac{\partial^{l+m} z}{\partial y^l \partial s^m}\right| \leq \mathcal{C}\varepsilon^{-l/2}, \quad (y,\ s) \in \mathfrak{D} \quad \text{for all } l > 0, m > 0 \text{ with } 0 \leq l + 2m \leq 8.$$

Proof. Refer to [1] for the details. □

3.4 Totally finite difference scheme

The second-order approximation for the operator is given by

$$D_y^+ D_y^- \omega_j^n = \frac{2}{h_j + h_{j+1}} \left(\frac{\omega_{j+1}^n - \omega_j^n}{h_{j+1}} - \frac{\omega_j^n - \omega_{j-1}^n}{h_j} \right).$$

In time, the backward difference scheme is $D_s^- \omega_j^n = \dfrac{\omega_j^n - \omega_j^{n-1}}{\Delta s}$, where $\omega_j^n = \omega(y_j, s_n)$. To solve (1), the Crank–Nicolson scheme for the time scale and the central difference scheme for the space are combined as

$$2D_t^- Z_i^{n+1} + \mathfrak{L}_\varepsilon Z_i^{n+1} = -\mathfrak{b}_i^{n+1} Z_i^{n-p+1} - \mathfrak{b}_i^n Z_i^{n-p} + \mathfrak{f}_i^n + \mathfrak{f}_i^{n+1} - \mathfrak{L}_\varepsilon Z_i^n \quad (10)$$
$$\text{for} \quad 1 \le i < M.$$

Here,

$$\mathfrak{L}_\varepsilon Z_i^n = -\varepsilon D_y^+ D_y^- Z_i^n + \mathfrak{p}_i Z_i^n, \quad \mathfrak{f}_i^n = \mathfrak{f}(y_i, s_n), \quad \mathfrak{b}_i^n = \mathfrak{b}(y_i, s_n), \quad \mathfrak{p}_i = \mathfrak{p}(y_i).$$

After rearranging the terms in (10) and combining (7), the fully discrete scheme is

$$\begin{cases} \dfrac{1}{2}\left(r_i^- Z_{i-1}^{n+1} + r_i^o Z_i^{n+1} + r_i^+ Z_{i+1}^{n+1} \right) = g_i^M, \quad 1 \le i < M, \\ Z_0^{n+1} = \Theta_l(s_{n+1}), \quad Z_N^{n+1} = \Theta_r(s_{n+1}), \\ Z_i^{-j} = \Theta_b(y_i, -s_j) \quad \text{for} \quad j = 0, \ldots, p \quad \text{and} \quad i = 1 \le i < M, \end{cases} \quad (11)$$

$$r_i^- = \Delta s \Big(-\frac{2\varepsilon}{\widehat{h}_i h_i} \Big), \; r_i^o = \Delta s \Big(\frac{2\varepsilon}{h_{i+1} h_i} + \mathfrak{b}_i^{n+1} \Big) + 1, \; r_i^+ = \Delta s \Big(-\frac{2\varepsilon}{\widehat{h}_i h_{i+1}} \Big),$$

$$g_i^M = \frac{\Delta s}{2} \Big(-\mathfrak{b}_i^{n+1} Z_i^{n-p+1} - \mathfrak{b}_i^n Z_i^{n-p} + \mathfrak{f}_i^{n+1} + \mathfrak{f}_i^n \Big) + \Big(1 - \frac{\Delta s}{2} \mathfrak{L}_\varepsilon \Big) Z_i^n$$
$$\text{for} \;\; 0 < i \le M - 1.$$

The difference equation (11) at $n+1$ time level forms a tridiagonal system of $M-1$ equations with the same number of unknowns. This system has properties like $r_i^- < 0$, $r_i^o > 0$, $r_i^+ < 0$ for $1 \le i \le M-1$. The Thomas algorithm [24] is used to solve the tridiagonal system.

3.5 Error analysis

Theorem 4. Let z be the analytical solution to (1) and let Z be the numerical solution (11). Then on S-mesh, the error of the scheme (11) satisfies the following estimate:

$$\max_{i,n} \left|(z-Z)(y_i,\ s_n)\right| \leq \mathcal{C}\Big((M^{-1}\ln M)^2 + \Delta s^2\Big) \quad \text{for} \quad i = 1,\ldots,M-1.$$

On B-S-mesh, it satisfies

$$\max_{i,n} \left|(z-Z)(y_i,\ s_n)\right| \leq \mathcal{C}\Big(M^{-2} + \Delta s^2\Big) \quad \text{for} \quad i = 1,\ldots,M-1,$$

where $Z(y_i, s_n) = Z_i^n$ for $(y_i, s_n) \in \mathfrak{D}^M$.

Proof. The proof is divided into various steps for each time level. On the first interval $s \in [0,\gamma]$ (the time discretization n varies from 0 to p), the term $\mathfrak{f}(y,\ s) - \mathfrak{c}(y,\ s)\Theta_b(s, s-\gamma)$ in the right side of (1) is independent of ε. Now since $(y_i,\ s_n) \in \mathfrak{D}_1^M = \Pi_y^M \times [0,\gamma)$, using the convergence results given in [3] and Theorem 2, we obtain on S-mesh

$$\max_{i,n} \left|(z-Z)(y_i,\ s_n)\right| \leq \mathcal{C}\Big(M^{-2}\ln^2 M + (\Delta s)^2\Big) \quad \text{for} \quad 0 < i < M.$$

Following a similar procedure done in [31, 30] for the error bounds on B-S mesh, we consider the mesh generating function for B-S mesh $\varsigma(\xi) = (2/\beta)\xi\chi(s)$, where $\chi(s) = -\ln\Big(1 - 2(1-\frac{1}{M})(s)\Big)$. The mesh generating function satisfies $\varsigma(\xi)' \leq \mathcal{C}M$, and assume that $\varepsilon \leq M^{-1}$. The spatial mesh size h_i on the layer region satisfies $h_i \leq \mathcal{C}M^{-1}$ and $h_i \leq \mathcal{C}\varepsilon$, for $i = 0, 1, \ldots, (M/4) - 1$ and for $i = (3M/4) + 1, \ldots, M$. Now using the bounds on the derivative of z given Theorem (3), we get on B-S-mesh,

$$\max_{i,n} \left|(z-Z)(y_i,\ s_n)\right| \leq \mathcal{C}\Big(M^{-2} + \Delta s^2\Big) \quad \text{for} \quad i = 1,\ldots,M-1,$$

Now, the term $z(y,\ s)$ depends on $z(y,\ s-\gamma)$, which is unknown for $s \geq \gamma$. So, the above process is not applicable for $s \geq \gamma$. To get the error over the interval $[\gamma,\ 2\gamma]$, using the convergence results in [1] and Theorem 2, we obtain the desired bound. □

4 Post processing technique

To enhance the order for the difference scheme (10), we use the Richardson extrapolation technique. First, we solve the discrete problems (11) on the fine mesh $\mathfrak{D}^{2M} = \overline{\Pi}_y^{2M} \times \overline{\Pi}_s^{2N}$ with $2M$ and $2N$ mesh intervals in the spatial and time direction respectively, where $\overline{\Pi}_y^{2M}$ is the Shishkin-type mesh and is obtained by halving each mesh interval of $\overline{\Pi}_y^{M}$ with a fixed transition parameter. Clearly, $\mathfrak{D}^M = \{(y_i,\ s_n)\} \subset \mathfrak{D}^{2M} = \{(\overline{y}_i, \overline{s}_n)\}$. Therefore, the corresponding S-mesh is $\overline{\Pi}_y^{2M} = \{\overline{y}_i \in (0,1), 0 \leq i \leq 2M\}$ by

$$\overline{y}_i = \begin{cases} \dfrac{2i\xi}{M} & \text{for } 0 \le i \le M/2, \\ \dfrac{i(1-2\xi)}{M} & \text{for } \dfrac{M}{2}+1 \le i \le \dfrac{3M}{2}, \\ \dfrac{2i\xi}{M} & \text{for } \dfrac{3M}{2}+1 \le i \le 2M, \end{cases}$$

and the B-S-mesh is

$$\overline{y}_i = \begin{cases} \dfrac{2}{\beta}\Big(-\ln\Big(1-2(1-\frac{1}{2M})\frac{i}{2M}\Big)\Big) & \text{for} \quad 0 \le i \le M/2-1, \\ \overline{y}_{M/2-1} + \Big(\dfrac{\overline{y}_{M/2+1}-\overline{y}_{M/2-1}}{M+2}\Big)(i-M/2+1) & \text{for} \quad \dfrac{M}{2} \le i \le \dfrac{3M}{2}, \\ 1-\dfrac{2}{\beta}\Big(-\ln\Big(1-2(1-\frac{1}{2M})\frac{2M-i}{2M}\Big)\Big) & \text{for} \quad i \le \dfrac{3M}{2}+1 \le 2M, \end{cases}$$

and $\overline{s}_n - \overline{s}_{n-1} = \Delta s/2$ for $\overline{s_n} \in \overline{\Pi}_s^{2N}$. Now, from Theorem 4, the error is

$$\begin{aligned}(Z-z)(y_i,\ s_n) &= \mathcal{C}\Big((M^{-1}\ln M)^2 + \Delta s^2\Big) + o\Big((M^{-1}\ln M)^2 + \Delta s^2\Big), \\ &= \mathcal{C}(\tfrac{M^{-1}\xi}{\rho_0\sqrt{\varepsilon}})^2 + \mathcal{C}\Delta s^2 + o\Big((M^{-1}\ln M)^2 + \Delta s^2\Big)\end{aligned} \tag{12}$$

for $(y_i,\ s_n) \in \mathfrak{D}^M$. Let $\overline{Z}(\overline{y}_i, \overline{s}_n)$ be the solution to (11) on the domain $\mathfrak{D}^{2M}$. From Theorem 4, we get

$$(\overline{Z}-z)(\overline{y}_i,\overline{s}_n) = \mathcal{C}\Big((2M)^{-2}(\tfrac{\xi}{\rho_0\sqrt{\varepsilon}})^2 + (\tfrac{\Delta s}{2})^2\Big) + o\Big((M^{-1}\ln M)^2 + (\tfrac{\Delta s}{2})^2\Big) \tag{13}$$

for $(\overline{y}_i, \overline{s}_n) \in \mathfrak{D}^{2M}$. Now, the elimination of $o(M^{-2})$ term from (12) and (13) leads to the following approximation:

$$z(y_i,s_n) - \frac{1}{3}\Big(4\overline{Z}-Z\Big)(y_i,\ s_n) = o\Big((M^{-1}\ln M)^2 + \Delta s^2\Big),\ (y_i,s_n) \in \mathfrak{D}^M. \tag{14}$$

Therefore, we use the extrapolation formula as

$$Z_{extp}(y_i,\ s_n) = \frac{1}{3}\Big(4\overline{Z}-Z\Big)(y_i,\ s_n), \quad (y_i,\ s_n) \in \mathfrak{D}^M. \tag{15}$$

Theorem 5. Let Z_{extp} be the solution by extrapolation technique (15) and let z be the solution to problem (1). Also, assume that $\sqrt{\varepsilon} \le M^{-1}$. Then we have the following error bound on S-mesh

$$\Big|z(y_i,\ s_n) - Z_{extp}(y_i,\ s_n)\Big| \le \mathcal{C}\Big(M^{-4}\ln^4 M + \Delta s^4\Big) \quad \text{for} \quad 1 \le i \le M-1.$$

Proof. The term in right side of (1), $\mathfrak{f}(y,\ s) - \mathfrak{b}(y,\ s)\Theta_b(y,\ s-\gamma)$ is independent of ε in the interval $[0,\gamma]$, where the time discretization parameter n

varies from 0 to p. One can find the complete analysis of the extrapolation technique in [28].

Since $z(y,\ s)$ depends on $z(y,\ s-\gamma)$, which is unknown for $s \geq \gamma$, the approach in [28] is not applicable in the interval $[\gamma, 2\gamma]$ and $s \geq \gamma$. Therefore the following delay parabolic equation involving small parameter considered on domain $\mathfrak{D}_2 = \Pi \times (\gamma, 2\gamma)$:

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big) z(y,\ s) = -\mathfrak{b}(y,\ s) z(y,\ s-\gamma) + \mathfrak{f}(y,\ s), \quad (y,\ s) \in \mathfrak{D}_2, \\ z(y,\ s) = z_b(y,\ s), \quad (y,\ s) \in \mathfrak{D}_1 = \Pi \times (0, \gamma), \\ z(0,\ s) = \Theta_l(s), \quad z(1,\ s) = \Theta_r(s), \quad s \in [\gamma, 2\gamma], \end{cases} \tag{16}$$

where $z_b(y,\ s)$ is a continuous solution in $\mathfrak{D}_1$. Applying the scheme for (16) on $\mathfrak{D}_2$ as given in (10), we have

$$\begin{aligned} 2D_t^- Z_i^{n+1} - \varepsilon \delta_y^2 Z_i^{n+1} + \mathfrak{p}_i Z_i^{n+1} &= -\mathfrak{b}_i^{n+1} Z_i^{n-p+1} - \mathfrak{b}_i^n Z_i^{n-p} \\ &\quad + \mathfrak{f}_i^n + \mathfrak{f}_i^{n+1} - \mathfrak{L}_\varepsilon Z_i^n, \\ Z_0^n &= \Theta_l(s_n), \quad Z_N^n = \Theta_r(s_n), \quad s_n \in [\gamma, 2\gamma], \\ Z_i^{-j} &= Z_b(y_i,\ s_n), \quad (y_i,\ s_n) \in \mathfrak{D}_1^M, \end{aligned} \tag{17}$$

where $\delta_y^2 = D_y^- D_y^+$, $\mathfrak{f}_i^n = \mathfrak{f}(y_i,\ s_n)$, $(y_i,\ s_n) \in \mathfrak{D}_2^M$ and $Z_b^1(\cdot, \cdot)$ is the numerical solution in $\mathfrak{D}_1^M$.

Decompose z in (16) on the domain $\mathfrak{D}_2$ as $z = w_r + w_s$, where w_r is the smooth component and w_s is the singular component. Again we write $w_r = w_{r0} + \varepsilon w_{r1}$, satisfying

$$\begin{cases} \dfrac{\partial w_{r0}(y,\ s)}{\partial s} + \mathfrak{p}(y) w_{r0}(y,\ s) = -\mathfrak{b}(y,\ s) w_{r0}(y,\ s-\gamma) + \mathfrak{f}(y,\ s), \ (y,\ s) \in \mathfrak{D}_2, \\ w_{r0}(y,\ s) = z(y,\ s), \quad (y,\ s) \in \mathfrak{D}_1, \ w_{r0}(0, s) = z(0,\ s), \quad s \in [\gamma, 2\gamma], \end{cases} \tag{18}$$

and

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big) w_{r1}(y, s) = -\mathfrak{b}(y, s) w_{r1}(y, s-\gamma) + (w_{r0})_{yy}, \ (y, s) \in \mathfrak{D}_2, \\ w_{r1}(y,\ s) = 0, (y,\ s) \in \mathfrak{D}_1, \ w_{r1}(0, s) = 0, \ w_{r1}(1, s) = 0, \ s \in [\gamma, 2\gamma]. \end{cases} \tag{19}$$

Therefore

$$\begin{cases} \Big(\dfrac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big) w_r(y,\ s) = -\mathfrak{b}(y,\ s) w_r(y, s-\gamma) + \mathfrak{f}(y,\ s), \ (y,\ s) \in \mathfrak{D}_2, \\ w_r(y,\ s) = z(y,\ s), \quad (y,\ s) \in \mathfrak{D}_1, \\ w_r(0,\ s) = w_{r0}(0, s), \quad w_r(1,\ s) = w_{r0}(1,\ s), \quad s \in [\gamma, 2\gamma]. \end{cases} \tag{20}$$

Then, the singular component w_s satisfies

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)w_s(y,\ s)=-\mathfrak{b}(y,\ s)w_s(y,\ s-\gamma), \quad (y,\ s)\in\mathfrak{D}_2,\\ w_s(y,\ s)=0, \quad (y,\ s)\in\mathfrak{D}_1, w_s(0,s)=\Theta_l(s)-w_{r0}(0,\ s),\\ w_s(1,\ s)=\Theta_r(s)-w_{r0}(1,\ s),\ s\in[\gamma,2\gamma]. \end{cases} \tag{21}$$

Write $w_s = w_{sl} + w_{sr}$, where w_{sl} is the left boundary layer component on $\mathfrak{d}_l$ and w_{sr} is the right boundary layer component on $\mathfrak{d}_r$. Also, w_{sl} and w_{sr} are satisfying the following PDEs:

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)w_{sl}(y,\ s)=-\mathfrak{b}(y,\ s)w_l(y,\ s-\gamma), \ \ (y,\ s)\in\mathfrak{D}_2,\\ w_{sl}(0,\ s)=\Theta_l-w_{r0}(0,\ s)\ s\in[\gamma,2\gamma],\ w_{sl}(y,\ s)=0,\ (y,\ s)\in\mathfrak{D}_{1l},\\ \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)w_{sr}(y,\ s)=-\mathfrak{b}(y,\ s)w_r(y,\ s-\gamma), \quad (y,\ s)\in\mathfrak{D}_2,\\ w_{sr}(1,s)=\Theta_r-w_{r0}(1,s),\ s\in[\gamma,2\gamma],\ w_{sr}(y,s)=0,\ (y,s)\in\mathfrak{D}_{1r}, \end{cases} \tag{22}$$

where $\mathfrak{D}_{1l}=(0,\xi)\times[0,\gamma]$ and $\mathfrak{D}_{1r}=(1-\xi,1)\times[0,\gamma]$. Since, $\mathfrak{D}_2\subset\mathfrak{D}$, the estimates given in Theorem 1, can be used for w_r and w_s. Decompose the numerical solution Z to (17) as $Z=W_r+W_s$, where W_r is the smooth part and W_s is the singular part. Thus

$$\begin{aligned} \Big(D_s^+ + D_s^- + \mathfrak{L}_\varepsilon\Big){W_r}_i^{n+\frac{1}{2}} &= -\mathfrak{b}_i^{n+\frac{1}{2}}{W_r}_i^{n+\frac{1}{2}-p} + f_i^{n+\frac{1}{2}},\ (y_i,\ s_{n+\frac{1}{2}})\in\mathfrak{D}_2^M, \\ {W_r}_i^{\,n} &= Z_b(y_i,\ s_n), \quad (y_i,\ s_n)\in\mathfrak{D}_1^M, \\ {W_r}_0^{\,n} &= w_r(0,\ s_n), \quad {W_r}_N^{\,n} = w_r(1,\ s_n), \quad s_n\in[\gamma,2\gamma], \end{aligned} \tag{23}$$

and therefore, W_s satisfies

$$\begin{aligned} \Big(D_s^+ + D_s^- + \mathfrak{L}_\varepsilon\Big){W_s}_i^{\,n+\frac{1}{2}} &= -b_i^n W_i^{n+\frac{1}{2}-p}, \quad (y_i, s_{n+\frac{1}{2}})\in\mathfrak{D}_2^M, \\ {W_s}_i^{\,n} &= 0, \quad (y_i,\ s_n)\in\mathfrak{D}_1^M, \\ {W_s}_0^{\,n} &= \Theta_l(s_n) - w_r(0,\ s_n), \\ {W_s}_N^{\,n} &= \Theta_r(s_n) - w_r(1,\ s_n), \quad s_n\in[\gamma,2\gamma]. \end{aligned} \tag{24}$$

Now, we write $W_s = W_{sl} + W_{sr}$, where W_{sl} is the boundary layer on $\mathfrak{d}_l^M$ and W_{sr} is the boundary layers on $\mathfrak{d}_r^M$. Hence W_{sl} and W_{sr} are defined by

$$\begin{aligned} &\Big(D_s^+ + D_s^- + \mathfrak{L}_\varepsilon\Big)W_{sl}(y_i,\ s_{n+\frac{1}{2}}) = -\mathfrak{b}_i^n W_{sl}(y_i,\ s_{n+\frac{1}{2}-p}),\ \ (y_i,\ s_{n+\frac{1}{2}})\in\mathfrak{D}_2^M,\\ &W_{sl}(0,s_n)=\Theta_l-w_r(0,\ s_n)\ (y_i,\ s_n)\in\mathfrak{d}_l, \quad W_{sl}(y_i,\ s_n)=0,\ (y_i,\ s_n)\in\mathfrak{D}_{1l}^M,\\ &\Big(D_s^+ + D_s^- + \mathfrak{L}_\varepsilon\Big)W_{sr}(y_i,\ s_{n+\frac{1}{2}}) = -\mathfrak{b}_i^n W_{sr}(y_i,\ s_{n+\frac{1}{2}-p}),\ \ (y_i,\ t_{n+\frac{1}{2}})\in\mathfrak{D}_2^M,\\ &W_{sr}(1,\ s_n)=\Theta_r-w_r(1,\ s_n),\ (y_i,\ s_n)\in\mathfrak{d}_r, \quad W_{sr}=0, \quad (y_i,\ s_n)\in\mathfrak{D}_{1r}^M, \end{aligned} \tag{25}$$

where $\mathfrak{D}_{1l}^M$ is the discretized domain of $\mathfrak{D}_{1l}$ and $\mathfrak{D}_{1r}^M$ is the discretized domain of $\mathfrak{D}_{1l}$. Similarly $\mathfrak{d}_l^M$ is the discretized domain of $\mathfrak{d}_l$ and $\mathfrak{d}_r^M$ is the discretized domain of $\mathfrak{d}_r$. □

4.1 Error bound for W_r

Lemma 3. The local truncation error in $\mathfrak{D}_2^M$ associated to the smooth component satisfies

$$\begin{aligned}\Big(2D_s^- + \mathfrak{L}_\varepsilon\Big)(w_r - W_r)(y_i,\ s_n) \\ = \mathcal{C}\Big((M^{-1}\ln M)^2 + \Delta s^2\Big) + \frac{\varepsilon}{12}(y_i - y_{i-1})^2\frac{\partial^4 w_r}{\partial y^4}(y_i,\ s_{n-\frac{1}{2}}) \\ + \frac{1}{12}\Delta s^2\frac{\partial^3 w_r}{\partial s^3}(y_i,\ s_n) + O(M^{-4} + \Delta s^4).\end{aligned}$$

Proof. Using (20) and (23), we get

$$\begin{aligned}\Big(2D_s^- + \mathfrak{L}_\varepsilon\Big)(w_r - W_r)(y_i,\ s_n) \\ = \mathfrak{b}_i^n(z(y_i,\ s_{n-p-\frac{1}{2}}) - Z_b(y_i,\ s_{n-p-\frac{1}{2}})) + \Big(\frac{\partial}{\partial s} - 2D_s^-\Big)w_r(y_i,\ s_n) \\ + \frac{\partial}{\partial s}w_r(y_i,\ s_{n-1}) - +(\mathfrak{L}_{\varepsilon,y} - \mathfrak{L}_\varepsilon)w_r(y_i,\ s_{n-\frac{1}{2}}).\end{aligned}$$

Now, by using the estimate in Theorem 4 and Taylor's expansion, the desired result can be achieved. Refer to [8] for more details. □

Let the function $E_d(y,\ s)$, for $d = 1, 2$, satisfy the IBVPs (refer approach of Kellar [12]):

$$\begin{cases}\Big(\frac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)E_1(y,\ s) = \frac{\varepsilon}{12}\frac{\partial^4 w_r(y,\ s)}{\partial y^4} & \text{in } \mathfrak{D}, \\ E_1(y,\ s) = 0, & (y,\ s) \in \mathfrak{b}_b, \\ E_1(0,\ s) = 0, \quad E_1(1,\ s) = 0, & s \in [0, \mathcal{S}],\end{cases} \tag{26}$$

$$\begin{cases}\Big(\frac{\partial}{\partial s} + \mathfrak{L}_{\varepsilon,y}\Big)E_2(y,\ s) = \frac{1}{12}\frac{\partial^3 w_r(y,\ s)}{\partial s^3} & \text{in } \mathfrak{D}, \\ E_2(y,\ s) = 0, & (y,\ s) \in \Gamma_b, \\ E_2(0,\ s) = 0, \quad E_2(1,\ s) = 0, & s \in [0, \mathcal{S}].\end{cases} \tag{27}$$

Now E_d can be decomposed as $E_d = \eta_d + \vartheta_d$, where η_d and ϑ_d are the regular and singular layer parts of E_d. Now from Theorem 1, we have

$$\left\|\frac{\partial^{l+m}\eta_d}{\partial y^l \partial s^m}\right\|_\infty \leq \mathcal{C}(1 + \varepsilon^{2-l/2}) \quad for \quad 0 \leq l + 2m \leq 8.$$

One can get these bounds using a similar procedure as done in [1]. Taking $(y,\ s) \in \mathfrak{D}_2$, (26) and (27) reduce to the following IBVPs:

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)E_1(y,\ s)=\dfrac{\varepsilon}{12}\dfrac{\partial^4 w_r(y,\ s)}{\partial y^4} & \text{in } \mathfrak{D}_2,\\ E_1(y,\ s)=E_{1\gamma}, & (y,\ s)\in\mathfrak{D}_1,\\ E_1(0,\ s)=0, \quad E_1(1,\ s)=0, & s\in[\gamma,2\gamma], \end{cases}$$

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)E_2(y,\ s)=\dfrac{1}{12}\dfrac{\partial^3 w_r(y,\ s)}{\partial s^3} & \text{in } \mathfrak{D}_2,\\ E_2(y,\ s)=E_{2\gamma}, & (y,\ s)\in\mathfrak{D}_1,\\ E_2(0,\ s)=0, \quad E_2(1,\ s)=0, & s\in[\gamma,2\gamma], \end{cases}$$

where $E_{1\gamma}(\cdot,\cdot)$ and $E_{2\gamma}(\cdot,\cdot)$ are the respective solutions in $\mathfrak{D}_1$. Therefore, the components η_d and ϑ_d, $d=1,2$, satisfy

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)\eta_1=\dfrac{\varepsilon}{12}\dfrac{\partial^4 w_r(y,\ s)}{\partial y^4} & \text{in } \mathfrak{D}_2,\\ \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)\eta_2=\dfrac{1}{12}\dfrac{\partial^3 w_r(y,\ s)}{\partial s^3} & \text{in } \mathfrak{D}_2,\\ \eta_d(y,\ s)=E_{d\gamma}, & (y,\ s)\in\mathfrak{D}_1 \quad \text{for} \quad d=1,2,\\ \eta_d(0,\ s)=0, \quad \eta_d(1,\ s)=-\vartheta_d(1,\ s), & s\in[\gamma,2\gamma], \end{cases}$$

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)\vartheta_d=0 \quad \text{in } \mathfrak{D}_2\\ \vartheta_d(y,\ s)=0, & (y,\ s)\in\mathfrak{D}_1 \quad \text{for} \quad d=1,2,\\ \vartheta_d(0,\ s)=0, \quad \vartheta_d(1,\ s)=-\eta_d(1,\ s), & s\in[\gamma,2\gamma]. \end{cases} \tag{28}$$

Lemma 4. The local truncation error in $\mathfrak{D}_2^M$ associated with w_r satisfies

$$(w_r-W_r)(y_i,\ s_n)=\mathcal{C}\Big((M^{-1}\ln M)^2+\Delta s^2\Big)+(y_i-y_{i-1})^2\eta_1+\eta_2\Delta s+O(M^{-4}+\Delta s^2).$$

Proof. From Lemma 3 and (28), one can easily get

$$\begin{aligned} \Big(2D_s^-+\mathfrak{L}_\varepsilon\Big)((w_r-W_r)(y_i,\ s_n) &= C\big((M^{-1}\ln M)^2+\Delta s\big)\\ &+h_i^2\left(\Big(\frac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)-(2D_s^-+\mathfrak{L}_\varepsilon)\right)\eta_1\\ &+\Delta s\left(\Big(\frac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)-(2D_s^-+\mathfrak{L}_\varepsilon)\right)\eta_2\\ &+O(h^4+\Delta s^4). \end{aligned} \tag{29}$$

Using the derivative bounds of η_d and from the Taylor's expansion, it follows that for $d=1,2$,

$$\left|h_i^2\left(\Big(\frac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)-(2D_s^-+\mathfrak{L}_\varepsilon)\right)\eta_1+\Delta s\left(\Big(\frac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)-(2D_s^-+\mathfrak{L}_\varepsilon)\right)\eta_2\right|$$

$$\leq \mathcal{C}\Big(h^4 + \Delta s^4\Big). \tag{30}$$

Therefore, from (29) and (30), we have

$$\Big(2D_s^- + \mathfrak{L}_\varepsilon\Big)(w_r - W_r)(y_i,\ s_n) \leq \mathcal{C}\Big((M^{-1}\ln M)^2 + \Delta s^2 + \big(h^4 + \Delta s^4\big)\Big),$$

and, by applying barrier functions and the discrete maximum principle as done in [22], we obtain the following bound:

$$\Big|(w_r - W_r)(y_i,\ s_n)\Big| \leq \mathcal{C}\Big((M^{-1}\ln M)^2 + \Delta s^2\Big) + h_i^2\eta_1 + \eta_2\Delta s^2 + O(h^4 + \Delta s^4).$$

□

Lemma 5. The error associated with W_r after extrapolation satisfies

$$(w_r - W_{rextp})(y_i,\ s_n) \leq \mathcal{C}\Big(M^{-4} + \Delta s^4\Big) \quad \text{for} \quad (y_i,\ s_n)\ \mathfrak{D}_2^M.$$

Proof. From Lemma 4 on the fine mesh $\mathfrak{D}_2^{2N}$, we have

$$(\overline{W_r} - w_r)(y_i,\ s_n) = \mathcal{C}\Big((2M)^{-2}(\ln 2M)^2 + \frac{\Delta s^2}{4}\Big) + \frac{h_i^2}{4}\eta_1 + \eta_2\frac{\Delta s^2}{4} + O(M^{-4} + \Delta s^4). \tag{31}$$

From the extrapolation formula (15), we can write

$$\begin{aligned}(w_r - W_{rextp})(y_i,\ s_n) &= w_r(y_i,\ s_n) - \big(\frac{1}{3}(4\overline{W_r} - W_r)(y_i,\ s_n)\big)\\ &= -\frac{1}{3}\Big(4(\overline{W_r} - w_r) + (W_r - w_r)\Big)(y_i,\ s_n).\end{aligned}$$

By using Lemma 4 and (31), we obtain

$$\begin{aligned}-\frac{1}{3}\Big(4(\overline{W_r} - w_r) + (W_r - w_r)\Big)(y_i,\ s_n) =& \frac{1}{3}\Bigg(- C\Big((M^{-1}\ln 2M)^2 + \Delta s^2\Big)\\ &- h_i^2\eta_1 - \eta_2\Delta s^2\\ &+ C\Big((M^{-1}\ln M)^2 + \Delta s^2\Big)\\ &+ h_i^2\eta_1 + \eta_2\Delta s^2\Bigg) + O(M^{-4} + \Delta s^4)\\ =& O(M^{-4} + \Delta s^4),\end{aligned}$$

which is our desired bound. □

Now, we define the function $F_d = F_{dl} + F_{dr}$, $d = 1, 2$, by the following IBVPs:

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)F_{1l}=\dfrac{\rho_0\varepsilon}{3}\dfrac{\partial^4 w_s(y,\ s)}{\partial y^4}, & (y,\ s)\text{ in }\mathfrak{D}_l=(0,\xi)\times(0,\mathcal{S}],\\ \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)F_{1r}=\dfrac{\rho_0\varepsilon}{3}\dfrac{\partial^4 w_s(y,\ s)}{\partial y^4}, & (y,\ s)\text{ in }\mathfrak{D}_r=(1-\xi,1)\times(0,\mathcal{S}],\\ F_{1l}(y,\ s)=0, & (y,\ s)\in[0,\xi]\times(-\gamma,0),\\ F_{1r}(y,\ s)=0, & (y,\ s)\in[1-\xi,1]\times(-\gamma,0),\\ F_{1l}(0,\ s)=F_{1l}(\xi,\ s)=0, & s\in(0,\mathcal{S}],\\ F_{1r}(1-\xi,\ s)=F_{1r}(1,\ s)=0, & s\in(0,\mathcal{S}], \end{cases} \tag{32}$$

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)F_{2l}=\dfrac{1}{12}\dfrac{\partial^3 w_s(y,\ s)}{\partial s^3}, & (y,\ s)\text{ in }\mathfrak{D}_l=(0,\xi)\times(0,\mathcal{S}],\\ \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)F_{2r}=\dfrac{1}{12}\dfrac{\partial^3 w_s(y,\ s)}{\partial s^3}, & (y,\ s)\text{ in }\mathfrak{D}_r=(1-\xi,1)\times(0,\mathcal{S}],\\ F_{2l}(y,\ s)=0, & (y,\ s)\in[0,\xi]\times(-\gamma,0),\\ F_{2r}(y,\ s)=0, & (y,\ s)\in[1-\xi,1]\times(-\gamma,0),\\ F_{2l}(0,\ s)=F_{1l}(\xi,\ s)=0, & s\in(0,\mathcal{S}],\\ F_{2r}(1-\xi,\ s)=F_{1r}(1,\ s)=0, & s\in(0,\mathcal{S}], \end{cases} \tag{33}$$

The solution F_d, $d=1,2$, to (32) and (33) satisfies the following bounds:

$$\left|\frac{\partial^{l+m}F_d}{\partial y^l\partial s^m}\right|_\infty\le\mathcal{C}\varepsilon^{-l/2}\Big(\exp(-y\sqrt{\beta/\varepsilon})+\exp(-(1-y)\sqrt{\beta/\varepsilon})\Big),\quad (y,\ s)\in\mathfrak{D}.$$

In this context, by considering $s\in(\gamma,2\gamma)$, therefore (32) and (33) reduces to

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)F_{1l}=\dfrac{\rho_0\varepsilon}{3}\dfrac{\partial^4 w_s(y,\ s)}{\partial y^4}, & (y,\ s)\text{ in }\mathfrak{D}_l=(0,\xi)\times(\gamma,2\gamma),\\ \Big(\dfrac{\partial}{\partial s}+\mathbb{L}_{\varepsilon,y}\Big)F_{1r}=\dfrac{\rho_0\varepsilon}{3}\dfrac{\partial^4 w_s(y,\ s)}{\partial y^4}, & (y,\ s)\text{ in }D_r=(1-\xi,1)\times(\gamma,2\gamma),\\ F_{1l}(y,\ s)=F_{1l\gamma}(y,\ s), & (y,\ s)\in[0,\xi]\times(0,\gamma),\\ F_{1r}(y,\ s)=F_{1r\gamma}(y,\ s), & (y,\ s)\in[1-\xi,1]\times(0,\gamma),\\ F_{1l}(0,\ s)=F_{1l}(\xi,t)=0, & s\in(\gamma,2\gamma),\\ F_{1r}(1-\xi,t)=F_{1r}(1,\ s)=0, & s\in(\gamma,2\gamma), \end{cases} \tag{34}$$

$$\begin{cases} \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,x}\Big)F_{2l}=\dfrac{1}{12}\dfrac{\partial^3 w_s(y,\ s)}{\partial s^3}, & (y,\ s)\text{ in }\mathfrak{D}_l=(0,\xi)\times(\gamma,2\gamma),\\ \Big(\dfrac{\partial}{\partial s}+\mathfrak{L}_{\varepsilon,y}\Big)F_{2r}=\dfrac{1}{12}\dfrac{\partial^3 w_s(y,\ s)}{\partial s^3}, & (y,\ s)\text{ in }\mathfrak{D}_r=(1-\xi,1)\times(\gamma,2\gamma),\\ F_{2l}(y,\ s)=F_{2l\gamma}(y,\ s), & (y,\ s)\in[0,\xi]\times(0,\gamma),\\ F_{2r}(y,\ s)=F_{2r\gamma}(y,\ s), & (y,\ s)\in[1-\xi,1]\times(0,\gamma),\\ F_{2l}(0,\ s)=F_{2l}(\xi,\ s)=0, & s\in(\gamma,2\gamma),\\ F_{2r}(1-\xi,\ s)=F_{1r}(1,\ s)=0, & s\in(\gamma,2\gamma), \end{cases} \tag{35}$$

where $F_{kl}(\cdot,\cdot)$, $k=1,2$, are the analytic solutions in $[0,\xi]\times(0,\gamma)$ and $F_{kr}(\cdot,\cdot)$, $k=1,2$, are the analytic solutions in $[1-\xi,1]\times(0,\gamma)$.

Lemma 6. For $(y_i,\ s_n)\ \mathfrak{D}_2^M$, the error associated with W_s satisfies

$$(W_s-w_s)(y_i,\ s_n)=(M^{-1}\ln M)^2F_1(y_i,\ s_n)+\Delta s^2F_2(y_i,\ s_n)+O\Big(M^{-4}\ln^4 M+\Delta s^4\Big).$$

Proof. Write $W_s=W_{sl}+W_{sr}$, where W_{sl} and W_{sr} are defined in (25). Now $W_s-w_s=(W_{sl}-w_{sl})+(W_{sr}-w_{sr})$, where the error $W_{sl}-w_{sl}$ is related to a layer on $\mathfrak{D}_l$ and $W_{sr}-w_{sr}$ on $\mathfrak{D}_r$. These errors can be estimated separately. First, we estimate $W_{sl}-w_{sl}$, from (22) and difference equation (25) and follow the similar procedure of Lemma 4. □

Next, we can estimate the error $W_{sr}-w_{sr}$.

Lemma 7. For $(y_i,\ s_n)\ \mathfrak{D}_2{}^M$, the error associated to W_s after extrapolation satisfies

$$\Big|(w_s-W_{sextp})(y_i,\ s_n)\Big|\le\mathcal{C}\Big(M^{-4}\ln^4 M+\Delta s^4\Big).$$

Proof. From Lemma 6, we get

$$\begin{aligned}(W_s-w_s)= &\quad (M^{-1}\ln M)^2F_1(y_i,\ s_n)+\Delta s^2F_2(y_i,\ s_n)\\ &+O\Big(M^{-4}\ln^4 N+\Delta s^4\Big)\quad\text{ for }\ (y_i,\ s_n)\ \mathfrak{D}_2^M,\end{aligned} \tag{36}$$

and

$$\begin{aligned}(w_s-\overline{W_s})=&\ ((2M)^{-1}\ln 2M)^2F_1(y_i,\ s_n)+\frac{\Delta s^2}{4}F_2(y_i,\ s_n)\\ &+O\Big(M^{-4}\ln^4 M+\Delta s^4\Big)\end{aligned} \tag{37}$$

for $(y_i,\ s_n)\in\mathfrak{D}_2^{2M}$. Eliminating the terms $O(M^{-2})$ and Δs^2 from (36) and (37), the required result is achieved. □

Theorem 6 (Error after extrapolation in $\mathfrak{D}_2^M$)**.** Let Z_{extp} be the extrapolated solution (by technique (15)) for solving (11) on $\mathfrak{D}_2{}^M$ and $\mathfrak{D}_2^{2M}$. Let z be the solution to (1). Assume that $\varepsilon<M^{-2}$. Then we have the following error bound associated with Z_{extp} :

$$\Big|z(y_i,\ s_n) - Z_{extp}(y_i,\ s_n)\Big| \leq \mathcal{C}\Big(M^{-4}\ln^4 M + \Delta s^4\Big) \quad \text{for} \quad 1 \leq i \leq M-1.$$

Proof. We have

$$\begin{aligned}\Big|z(y_i,\ s_n) - Z_{extp}(y_i,\ s_n)\Big| \leq &\Big|w_r(y_i,\ s_n) - W_{rextp}(y_i,\ s_n)\Big| \\ &+ \Big|w_s(y_i,\ s_n) - W_{sextp}(y_i,\ s_n)\Big|\end{aligned}$$

for all $(y_i,\ s_n) \in \mathfrak{D}_2^M$. Combining Lemmas 4 and 7, we can get the required result. □

Remark 1. The error bound on B-S-mesh in D_2^M is

$$\Big|z(y_i,\ s_n) - Z_{extp}(y_i,\ s_n)\Big| \leq \mathcal{C}\Big(M^{-4} + \Delta s^4\Big) \quad \text{for} \quad 1 \leq i \leq M-1.$$

Remark 2.(Error after extrapolation in $\mathfrak{D}^M$) Let Z_{extp} be the extrapolated solution (by technique (15)) for solving (11) on ${\mathfrak{D}_2}^M$ and $\mathfrak{D}_2^{2M}$. Let z be the solution to (1). Then we have the following error bound on S-mesh:

$$\Big|z(y_i,\ s_n) - Z_{extp}(y_i,\ s_n)\Big| \leq \mathcal{C}\Big(M^{-4}\ln^4 M + \Delta s^2\Big) \quad \text{for} \quad 1 \leq i \leq M-1,$$

similarly, on B-S-mesh

$$\Big|z(y_i,\ s_n) - Z_{extp}(y_i,\ s_n)\Big| \leq \mathcal{C}\Big(M^{-4} + \Delta s^4\Big) \quad \text{for} \quad 1 \leq i \leq M-1.$$

5 Numerical experiments

The proposed scheme (10) is tested on two test problems in this section. In this section, all the numerical results are obtained using MATLAB software in 64 GB RAM workstation.

Example 1. Consider

$$\begin{cases} z_s - \varepsilon z_{yy} + 0.5z = -2e^{-1}z(y,\ s-1) + \mathfrak{f}(y,s), \quad (x,\ t) \in (0,\ 1) \times (0,\ 2], \\ z(y,\ s) = e^{-s+y/\sqrt{\varepsilon}} + e^{-s+(1-y)/\sqrt{\varepsilon}}, \quad (y,\ s) \in [0,\ 1] \times [-1,\ 0], \\ z(0,\ s) = e^{-s} + e^{-s+1/\sqrt{\varepsilon}}, z(1,\ s) = e^{-s+1/\sqrt{\varepsilon}} + e^{-s},\ s \in [0,\ 2]. \end{cases} \tag{38}$$

The exact solution for Example 1 is $z(y,\ s) = e^{-s+y/\sqrt{\varepsilon}} + e^{-s+(1-y)/\sqrt{\varepsilon}}$. The maximum pointwise error before and after extrapolation given by

$$E_{\varepsilon}^{M,\Delta s} = \max_{(y_i,\ s_n) \in \mathfrak{D}^M} |z(y_i,\ s_n) - Z^{M,\Delta s}(y_i,\ s_n)|$$

and

$$E_{\varepsilon}^{M,\Delta s} = \max_{(y_i,\ s_n)\in\mathfrak{D}^M} |z(y_i,\ s_n) - Z_{extp}^{M,\Delta s}(y_i,\ s_n)|.$$

The corresponding order of convergence is $P_{\varepsilon}^{M,\Delta s} = \log_2\left(\frac{E_{\varepsilon}^{M,\Delta s}}{E_{\varepsilon}^{2M,\Delta s/2}}\right)$. Here, $z(y_i,\ s_n)$ is the exact solution and $Z^{M,\Delta s}(y_i,\ s_n)$ and $Z_{extp}^{M,\Delta s}(y_i,\ s_n)$ are the numerical solutions before and after extrapolation, respectively.

Example 2. Consider the test problem:

$$\begin{cases} z_s - \varepsilon z_{yy} + \dfrac{(1+y)^2}{2} z = s^3 - z(y, s-1), \quad (y,\ s) \in (0,\ 1) \times (0,\ 2], \\ z(y,\ s) = 0, \quad (y,\ s) \in [0,\ 1] \times [-1,\ 0], \\ z(0,s) = 0, z(1,s) = 0,\ \ s \in [0,\ 2]. \end{cases} \tag{39}$$

Since the exact solution to (2) is not known, we use the idea of double mesh principle to obtain the pointwise errors and to verify the ε-uniform convergence. Define $\widetilde{Z}(y_i,\ s_n)$ as the numerical solution obtained on $\widetilde{\mathfrak{D}}^{2M} = \widetilde{\Pi}_y^{2M} \times \widetilde{\Pi}_s^{2N}$ with $2M$ mesh intervals in space and $2N$ mesh intervals in the s-direction, where $\widetilde{\Pi}_y^{2M}$ is the Shishkin-type mesh as defined Π_y^M with the fixed transition parameter. For each ε, we calculate the maximum pointwise error before and after extrapolation by $\widehat{E}_{\varepsilon}^{M,\Delta s} = \max\limits_{(y_i,\ s_n)\in\mathfrak{D}^M} |Z^{M,\Delta s}(y_i,\ s_n) - \widetilde{Z}^{M,\Delta s}(y_i,\ s_n)|$ and $\widehat{E}_{\varepsilon}^{M,\Delta s} = \max\limits_{(y_i,\ s_n)\in\mathfrak{D}^M} |Z_{extp}^{M,\Delta s}(y_i,\ s_n) - \widetilde{Z}_{extp}^{M,\Delta s}(y_i,\ s_n)|$, respectively. The corresponding order of convergence is obtained by $\widehat{P}_{\varepsilon}^{M,\Delta s} = \log_2\left(\frac{\widehat{E}_{\varepsilon}^{M,\Delta s}}{\widehat{E}_{\varepsilon}^{2M,\Delta s/2}}\right)$. Here, $\widetilde{Z}_{extp}^{M,\Delta s}(y_i,\ s_n)$ is the extrapolation solution obtained by the double mesh principle.

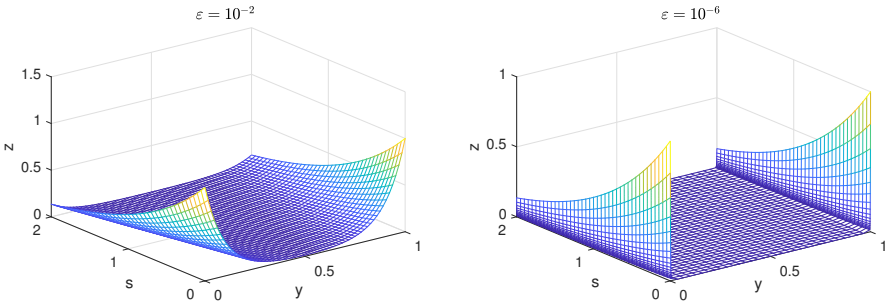


Figure 1: Surface plots of the computed solution on S-mesh for Example 1.

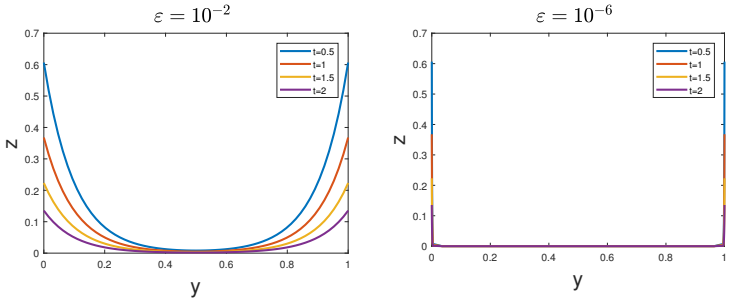
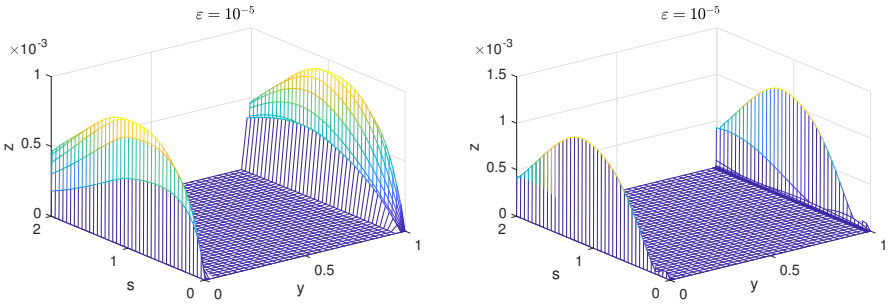


Figure 2: Solution plots of each time on S-mesh for Example 1.



(a) Before extrapolation. *(b) After extrapolation.*

Figure 3: Error graphs of the computed solution on B-S mesh for Example 1.

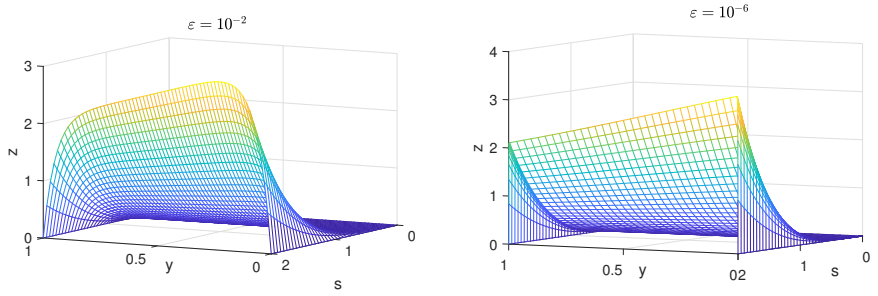
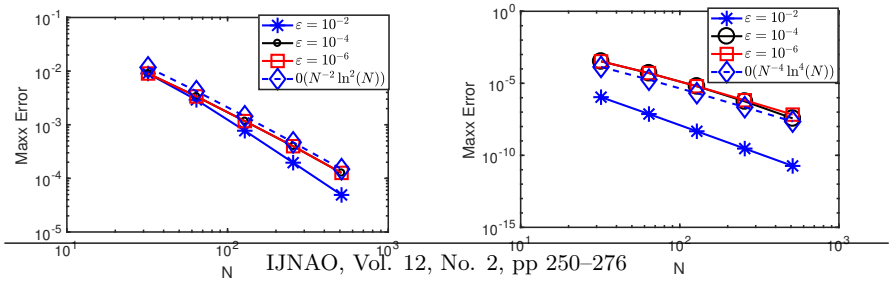


Figure 4: Surface plots of the computed solution on S-mesh for Example 2.



(a) Before extrapolation. *(b) After extrapolation.*

Figure 9: Log-log plots on B-S mesh for Example 2.

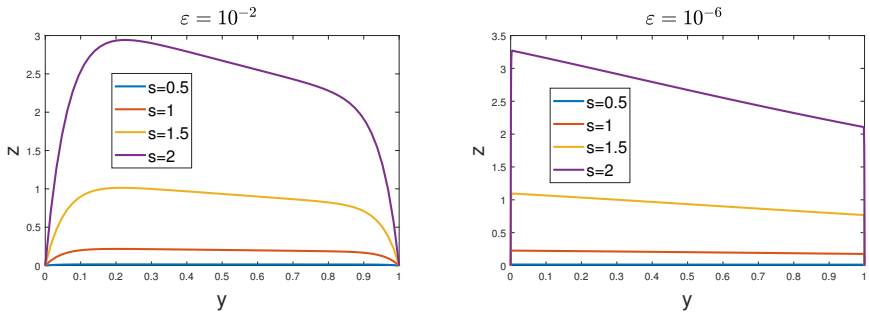
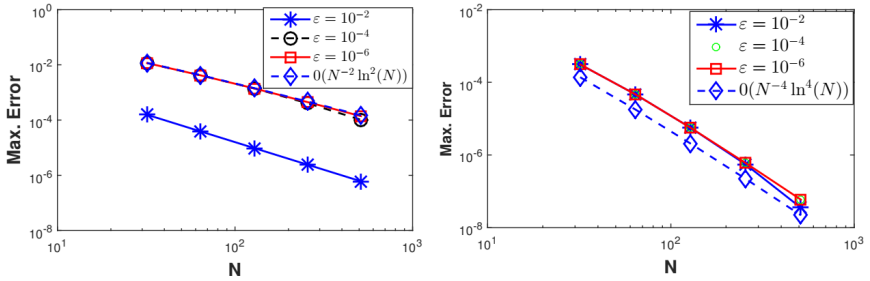
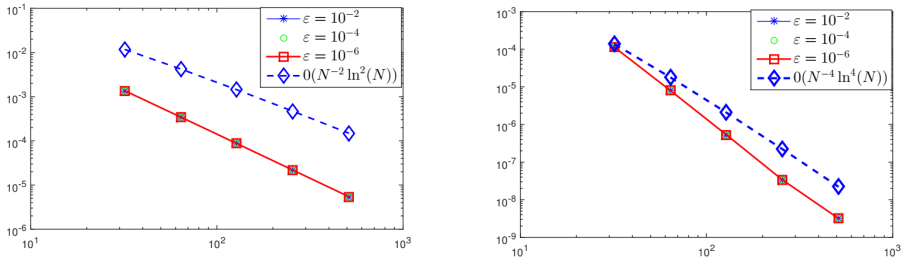


Figure 5: Solution plots on S-mesh for Example 2.



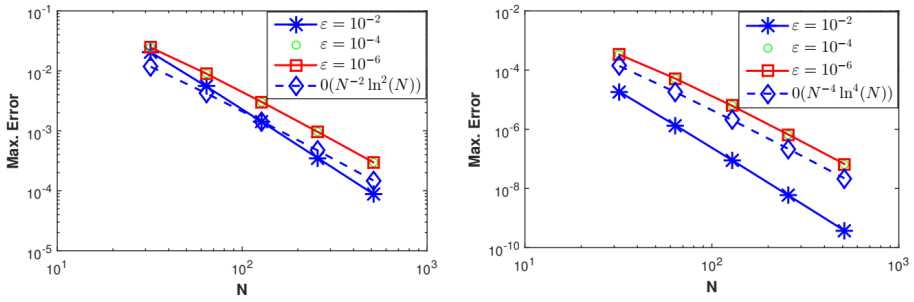
(a) Before extrapolation. *(b) After extrapolation.*

Figure 6: Log-log plots on S-mesh for Example 1.



(a) Before extrapolation. *(b) After extrapolation.*

Figure 7: Log-log plots on B-S mesh for Example 1.



(a) Before extrapolation. *(b) After extrapolation.*

Figure 8: Log-log plots on S-mesh for Example 2.

Table 1: $E_\varepsilon^{M,\Delta s}$ and $P_\varepsilon^{M,\Delta s}$ generated on S-mesh using the proposed method for Example 1.

$M/\Delta s$	Extrapolation	1e-4		1e-6	
		Singular layer	Regular layer	Singular layer	Regular layer
32/10	before	1.15e-2	6.42e-4	1.15e-2	6.42e-4
		1.47	2.13	1.47	2.13
	after	3.09e-4	1.52e-6	3.09e-4	1.52e-6
		2.72	3.56	2.72	3.56
64/40	before	4.15e-3	1.46e-4	4.15e-3	1.46e-4
		1.56	2.25	1.56	2.25
	after	1.33e-6	1.10e-7	1.33e-6	1.10e-7
		3.05	3.78	3.05	3.78
128/160	before	1.40e-3	3.07e-5	1.40e-3	3.07e-5
		1.65	2.38	1.65	2.38
	after	5.61e-6	9.33e-9	5.61e-6	9.33e-9
		3.22	3.89	3.22	3.89
256/640	before	4.46e-4	5.90e-6	4.46e-4	5.90e-6
		1.68	2.48	1.68	2.48
	after	6.01e-7	6.28e-10	6.00e-7	6.28e-10
		3.34	4.27	3.34	4.27
512/2560	before	1.38e-4	1.05e-6	1.38e-4	1.05e-6
	after	5.91e-8	3.25e-11	5.91e-8	3.25e-11

Table 2: $E_\varepsilon^{M,\Delta s}$ and $P_\varepsilon^{M,\Delta s}$ generated on B-S-mesh using the proposed method for Example 1.

ε	Extrapolation	Number of intervals M				
		32/10	64/40	128/160	256/640	512/2560
$1e-6$	before	4.9562e-3	1.3618e-3	3.4560e-4	8.6895e-5	2.1776e-5
		1.8637	1.9783	1.9918	1.9965	
	after	5.3442e-4	4.4861e-5	3.0539e-6	1.9448e-7	1.2234e-8
		3.5744	3.8767	3.9730	3.9906	
$1e-8$	before	4.9562e-3	1.3618e-3	3.4560e-4	8.6895e-5	2.1776e-5
		1.8637	1.9783	1.9918	1.9965	
	after	5.3442e-4	4.4861e-5	3.0539e-6	1.9448e-7	1.2234e-8
		3.5744	3.8767	3.9730	3.9906	
$1e-12$	before	4.9562e-3	1.3618e-3	3.4560e-4	8.6895e-5	2.1776e-5
		1.8637	1.9783	1.9918	1.9965	
	after	5.3442e-4	4.4861e-5	3.0539e-6	1.9448e-7	1.2234e-8
		3.5744	3.8767	3.9730	3.9906	

Table 3: $\widehat{E}_\varepsilon^{M,\Delta s}$ and $\widehat{P}_\varepsilon^{M,\Delta s}$ generated on S-mesh using the proposed method for Example 2.

$N/\Delta s$	Extrapolation	1e-4		1e-6		1e-8	
		Inner layer	Outer layer	Inner layer	Outer layer	Inner layer	Outer layer
32/10	before	9.021e-3	2.800e-3	9.021e-3	3.100e-3	9.021e-3	3.100e-3
		1.633	2.001	1.418	2.015	1.418	2.015
	after	1.850e-5	6.976e-6	3.344e-4	7.103e-6	3.344e-4	7.103e-6
		3.788	3.982	2.704	3.970	2.704	3.970
64/40	before	2.907e-3	7.068e-4	3.374e-3	7.669e-4	3.374e-3	7.669e-4
		1.911	2.001	1.528	2.000	1.528	2.000
	after	1.339e-6	4.414e-7	5.132e-5	4.530e-7	5.132e-5	4.530e-7
		3.878	4.06	3.037	3.925	3.037	3.925
128/160	before	7.729e-4	1.765e-4	1.169e-3	1.917e-4	1.169e-3	1.917e-4
		1.418	2.002	1.633	2.000	1.6337	2.000
	after	9.108e-8	2.631e-8	6.249e-6	2.981e-8	6.249e-6	2.981e-8
		3.949	4.024	3.489	4.011	3.489	4.011
256/640	before	1.959e-4	4.404e-5	3.979e-4	4.793e-5	3.979e-4	4.793e-5
		1.980	2.010	1.655	2.000	1.655	2.000
	after	5.897e-9	1.617e-9	5.565e-7	1.848e-9	5.565e-7	1.848e-9
		3.974	4.162	3.920	4.291	3.920	4.291
512/2560	before	4.909e-5	1.093e-5	1.263e-4	1.198e-5	1.263e-4	1.198e-5
	after	3.752e-10	1.778e-11	3.675e-8	9.440e-11	3.675e-8	9.440e-11

Table 4: $\widehat{E}_{\varepsilon}^{M,\Delta s}$ and $\widehat{P}_{\varepsilon}^{M,\Delta s}$ generated on B-S-mesh using the proposed method for Example 2.

ε	Extrapolation	Number of intervals M				
		32/10	64/40	128/160	256/640	512/2560
$1e-6$	before	1.3012e-2	3.5208e-3	9.2402e-4	2.3350e-4	5.8667e-5
		1.8859	1.9299	1.9845	1.9928	
	after	5.4243e-4	4.6020e-5	3.1547e-6	2.0324e-7	1.2801e-8
		3.5591	3.8667	3.9562	3.9889	
$1e-8$	before	1.3012e-2	3.5208e-3	9.2402e-4	2.3350e-4	5.8667e-5
		1.8859	1.9299	1.9845	1.9928	
	after	5.4243e-4	4.6020e-5	3.1547e-6	2.0324e-7	1.2801e-8
		3.5591	3.8667	3.9562	3.9889	
$1e-12$	before	1.3012e-2	3.5208e-3	9.2402e-4	2.3350e-4	5.8667e-5
		1.8859	1.9299	1.9845	1.9928	
	after	5.4243e-4	4.6020e-5	3.1547e-6	2.0324e-7	1.2801e-8
		3.5591	3.8667	3.9562	3.9889	

The surface plots are plotted for $\varepsilon = 10^{-2}$ and $\varepsilon = 10^{-6}$ in Figure 1 for Example 1 and Figure 4 for Example 2 on S-mesh, respectively. Figures 2 and 5 display the solution for different values ε with respect to time for Example 1 and Example 2, respectively. From these figures, one can visualize that the boundary layers appear at $y = 0$ and $y = 1$. The error plots are given in Figure 3 for Example 1 on the B-S mesh before and after extrapolation. These graphs show that the error is less after extrapolation compared to before extrapolation. To see the numerical convergence rate graphically, $E_{\varepsilon}^{M,\Delta s}$ are plotted in the log-log scale before extrapolation in Figures 6(a), 7(a),8(a), and 9(a) and after extrapolation in Figures 6(b), 7(b),8(b), and 9(b). Moreover, $E_{\varepsilon}^{M,\Delta s}$ and $P_{\varepsilon}^{M,\Delta s}$ by the proposed scheme for Example 1 are presented in Tables 1 and 2 on S-mesh and B-S-mesh, respectively. Similar results for Example 2 are shown in Tables 3 and 4. From these tables, one can conclude that B-S-meshes are more accurate, and the rate of convergence is more compared to the S-mesh. One can notice that the numerical results are well by our theoretical findings.

References

1. Ansari, A.R., Bakr, S.A., and Shishkin, G.I. *A parameter-robust finite difference method for singularly perturbed delay parabolic partial differential equations*, J. Comput. Appl. Math. 205(1) (2007), 552–566.

2. Chandru, M., Prabha, T. and Shanthi, V. *A hybrid difference scheme for a second-order singularly perturbed reaction-diffusion problem with non-smooth data*, Int. J. Appl. Comput. Math. 1(1) (2015), 87–100.

3. Clavero, C., Gracia, J.L. and Jorge, J.C. *High-order numerical methods for one-dimensional parabolic singularly perturbed problems with regular*

layers, Numer. Methods Partial Differential Equations 21 (2005), 149–169.

4. Constantinou, P. and Xenophontos, C. *Finite element analysis of an exponentially graded mesh for singularly perturbed problems*, Comput. Methods Appl. Math. 15(2) (2015), 135–143.

5. Das, P. and Natesan, S. *Richardson extrapolation method for singularly perturbed convection-diffusion problems on adaptively generated mesh*, CMES Comput. Model. Eng. Sci. 90(6) (2013), 463–485.

6. Das, A. and Natesan, S. *Second-order uniformly convergent numerical method for singularly perturbed delay parabolic partial differential equations*, Int. J. Comput. Math. 95(3) (2018), 490–510.

7. Geetha, N., Tamilselvan, A. and Subburayan, V. *Parameter uniform numerical method for third order singularly perturbed turning point problems exhibiting boundary layers*, Int. J. Appl. Comput. Math. 2(3) (2016), 349–364.

8. Govindarao L. and Mohapatra J. *A second-order numerical method for singularly perturbed delay parabolic partial differential equation*, Eng. Comput. 36(2) (2019), 420–444.

9. Govindarao, L., Mohapatra, J. and Das, A. *A fourth-order numerical scheme for singularly perturbed delay parabolic problem arising in population dynamics*, J. Appl. Math. Comput. 63 (2020), 171–195.

10. Govindarao, L., Sahu, S.R. and Mohapatra, J. *Uniformly convergent numerical method for singularly perturbed time delay parabolic problem with two small parameters*, Iran. J. Sci. Technol. Trans. A Sci. 43(5) (2019), 2373–2383.

11. Gupta, V., Kumar, M. and Kumar, S. *Higher order numerical approximation for time dependent singularly perturbed differential-difference convection-diffusion equations*, Numer. Methods Partial Differential Equations 34(1) (2018), 357–380.

12. Keller, H.B. *Numerical Methods for Two-point Boundary Value Problems*, Dover, New York, 1992.

13. Kuang, Y. *Delay Differential Equations with Applications to Population Biology*, Academic Press, New York, 1993.

14. Kudu, M. *A parameter uniform difference scheme for the parameterized singularly perturbed problem with integral boundary condition*, Adv. Difference Equ. 2018(1), (2018), 1–12.

15. Kumar, D., *A parameter-uniform scheme for the parabolicsingularly perturbed problem with a delay in time*, Numer. Methods Partial Differ. Equ., 37(1) (2021), 626–642.

16. Kumar, D. and Kumari, P. *A parameter-uniform numerical scheme for the parabolic singularly perturbed initial boundary value problems with large time delay*, Appl. Math. Comput. 59(1) (2019), 179–206.

17. Kumar, D. and Kumari, P. *Parameter-uniform numerical treatment of singularly perturbed initial-boundary value problems with large delay*, Appl.Numer. Math. 153 (2020), 412–429.

18. Kumar, M. and Sekhara Rao, S.C. *High order parameter-robust numerical method for time dependent singularly perturbed reaction-diffusion problems*, Computing 90(1-2) (2010), 15–38.

19. Ladyzenskaja, O.A., Solonnikov, V.A. and Uraluceva N.N. *Linear and Quasilinear Equations of Parabolic type*, (Vol. 23), American Mathematical Soc, 1968.

20. Mohapatra, J. *Equidistribution grids for two-parameter convection–diffusion boundary-value problems*, J. Math. Model. 2(1) (2014), 1–21.

21. Mohapatra, J. and Natesan, S. *Uniformly convergent second-order numerical method for singularly perturbed delay differential equations*, Neural Parallel Sci. Comput. 30 (2008), 353–370.

22. Miller, J.J.H., O'Riordan, E. and Shishkin, G.I. *Fitted Numerical Methods for Singular Perturbation Problems*, World Scientific, Singapore, 1996.

23. Salama, A. A. and Al-Amerya, D.G. *A higher order uniformly convergent method for singularly perturbed delay parabolic partial differential equations*, Int. J. Comput. Math. 12(94) (2017), 2520–2546.

24. Raji Reddy, N. and Mohapatra, J. *An efficient numerical method for singularly perturbed two point boundary value problems exhibiting boundary layers*, Natl. Acad. Sci. Lett. 4(38) (2015), 355–359.

25. Selvi, P.A. and Ramanujam, N. *An iterative numerical method for singularly perturbed reaction-diffusion equations with negative shift*, J. Appl. Comput. Math. 296 (2016), 10–23.

26. Sedighi, H.M. and Bozorgmehri, A. *Dynamic instability analysis of doubly clamped cylindrical nanowires in the presence of Casimir attraction and surface effects using modified couple stress theory*, Acta Mechanica 227(6) (2016), 1575–1591.

27. Shishkin, G.I. and Shishkina, L.P. *Difference Methods for Singular Perturbation Problems*, Chapman and Hall/CRC, New York, 2008.

28. Shishkin, G.I. and Shishkina, L.P. *A richardson scheme of the decomposition method for solving singularly perturbed parabolic reaction-diffusion equation*, Comp. Math. and Math. Phy. 50(12) (2010), 2003–2022.

29. Stein, R.B., *Some models of neuronal variability*, Biophys. J. 7 (1967), 37–68.

30. Zheng, Q., Li, X. and Gao, Y. *Uniformly convergent hybrid schemes for solutions and derivatives in quasilinear singularly perturbed BVPs*, Appl. Numer. Math. 871 (2015), 46–59.

31. Zheng, Q. and Wang, Y. Y., *The midpoint upwind scheme on the Bakhvalov-Shishkin mesh for parabolic singularly perturbed problems*, In Journal of Physics: Conference Series, 2012(1) (2021), P. 012047, IOP Publishing.

How to cite this article
J. Mohapatra and L. Govindarao A fourth-order optimal numerical approximation and its convergence for singularly perturbed time delayed parabolic problems. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 250-276. doi: 10.22067/ijnao.2021.71891.1053.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 277–300
DOI: 10.22067/ijnao.2021.72366.1058
https://ijnao.um.ac.ir/

How to cite this article
Research Article

Sixth-order compact finite difference method for solving KDV-Burger equation in the application of wave propagations

K. Aliyi Koroche* and H. Muleta Chemeda

Abstract

Sixth-order compact finite difference method is presented for solving the one-dimensional KdV-Burger equation. First, the given solution domain is discretized using a uniform discretization grid point in a spatial direction. Then, using the Taylor series expansion, we obtain a higher-order finite difference discretization of the KdV-Burger equation involving spatial variables and produce a system of nonlinear ordinary differential equations. Then, the obtained system of a differential equation is solved by using the fourth-order Runge–Kutta method. To validate the applicability of proposed techniques, four model examples are considered. The stability and convergent analysis of the present method is worked by using von Neumann stability analysis techniques by supporting the theoretical and mathematical statements in order to verify the accuracy of the present solution. The quality of the attending method has been shown in the sense of root mean square error L_2 and point-wise maximum absolute error L_∞. This is used to show, how the present method approximates the exact solution very well and how it is quite efficient and practically well suited for solving the KdV-Burger equation. Numerical results of considered examples are presented in terms of L_2 and L_∞ in tables. The graph of obtained present numerical and its exact solution are also presented in this paper. The present approximate numeric solvent in the table and graph shows that the numerical solutions are in good agreement with the exact solution of the given model problem. Hence the technique is reliable and capable for solving the one-dimensional KdV-Burger equation.

* Corresponding author

Received 18 September 2021; revised 16 December 2021; accepted 18 December 2021

Kedir Aliyi Koroche
Ambo University, College of Natural and Computational Sciences, Department of Mathematics, Ambo, Ethiopia. gmail: kediraliyi39@gmail,com

Hailu Muleta Chemeda
Jimma University College of Natural Sciences Department of Mathematics, Jimma, Ethiopia. gmail: mulatah@gmail.com

AMS subject classifications (2020): 104A58; 104A62; 104A66; 101D44; 101D20; 101D32; 101E12.

Keywords: KdV-equation; Compact finite difference method; Stability Analysis; Convergent of method; Root mean square error; Maximum absolute error.

1 Introduction

Partial differential equations (PDEs) are the mathematical equations that are significant in modeling physical phenomena that occur in nature. Applications of PDEs can be found in physics, engineering, mathematics, and finance. Examples include modeling mechanical vibration, heat, sound vibration, elasticity, and fluid dynamics [4]. Some applications in real life phenomena appear as nonlinear fractional order of PDEs. The fractional nonlinear PDE describes a variety of important physical phenomena, acoustics, elector-chemistry, materials science, fluid dynamics[2], optics, and viscoelasticity [1]. The other type of nonlinear PDEs is also Korteweg-de Vries Burgers' equation. Korteweg-de Vries Burgers' equation is a model mathematical statement for a wide class of nonlinear wave models of fluid in an elastic tube, liquid with small bubbles and turbulence [21]. This Burgers-Korteweg-de Vries (Burgers-KdV) equation has wide applications in physics, engineering, and fluid mechanics. The Poincaré phase plane analysis reveals that the Burgers-KdV equation has neither nontrivial bell-profile travelling solitary waves nor periodic waves [9].

The Burgers-KdV equation is the simplest form of the wave equation containing the nonlinearity, in which dispersion and the dissipation term occur [9]. It arises from many physical contexts, for example, the propagation of glandular bores in shallow water [5, 18], the flow of liquids containing gas bubbles [13], the propagation of waves in an elastic tube filled with a viscous fluid [36], weakly nonlinear plasma waves with certain dissipate effects [13, 16], and the cascading down the process of turbulence. It is also widely used as a nonlinear governing model in the crystal lattice theory, the nonlinear circuit hypothesis, and atmospheric dynamics [10].

Travelling waves or solitons as solutions to the Korteweg-de Vries equation (KdV), which is a nonlinear partial differential of mathematical statement four-third-order, have been of interest already for 150 years [37]. The Kdv equation was discovered in 1895 by Korteweg and de Vries [23], but this equation was forgotten for a long time. So that, the KdV equation is widely recognized as a paradigm for the description of weakly nonlinear long waves in many branches of physics and engineering. It describes how waves evolve under the competing but comparable effects of weak nonlinearity and dynamic dispersion [37]. Random waves are an important subject of haphazard PDEs [11]. The nonlinear problems are solved easily and elegantly without

converting to linear form the problem by using the Adomian decomposition method [22]. A feature common to all the traditional methods is that they are using the transformations to reduce the equation into a more simple mathematical statement and then solve it. Unlike classical techniques, the nonlinear equations are solved easily and elegantly without transforming the equation by using the Adomian decomposition method [21]. This wonderful technique has many advantages over classical techniques [22]. Mainly, it avoids the process of converting linear form and perturbation to find solutions to a given nonlinear equation. It provides an efficient explicit solution with high accuracy, minimal calculation, and avoidance of physically unrealistic assumptions [21].

Many authors studied the exact and numerical solutions of the KdV equation. Wadati [34] first introduced and studied the stochastic KdV equations and gave the diffusion of solutions to the KdV equation under Gaussian noise. Xie firstly studied research on Wick-type stochastic KdV equation on white noise spaces and showed the auto-Backlund transformation and the exact white noise functional solutions in [16]. Furthermore, Chen and Xie [6, 8] and Xie [7, 38] worked on some Wick-type stochastic wave equations using the white noise analysis method. Recently, Ũgurlu and Kaya [14] gave the tanh function method, Wazzan [35] showed the modified tanh-coth techniques, and these methods have been applied to derive nonlinear transformations and exact solutions of nonlinear PDEs in mathematical physics [18]. In those driving analytical methods, there is time consumed and very tired because it requires long computational time. To save our time and obtain an accurate approximate solution of the KdV equation, we use numerical methods.

Different numeral techniques give accurate denotative solutions for the KdV equation. To obtain the numerical solution of the KdV equations, the same researcher was used the traditionally, mesh-dependent methods such as the finite-difference method, finite element techniques, and boundary elements method. However, these methods have a slow rate of convergence, instability, low accuracy, and difficulty of implementation in complex geometries [4]. Even though the accuracy of the aforementioned methods is promising, they require large memory and long computational time. Besides funding, the methods are not suitable for higher-dimensional and problems involving complex geometries. Hence, the treatment of mesh-size presents several difficulties that have to be addressed to ensure the accuracy of the solution, and efficiency of the method applied [4]. Indeed scientists in the field of computational mathematics have been trying to develop numerical techniques by using computers for further application of different model problems. From those techniques, higher-order difference schemes and differential quadrature methods are more powerful techniques.

In many application areas, such as aeroacoustics [26] and electromagnetic [29], the propagation of acoustic and magnetic force waves needs to be accurately simulated over very long periods of time and far distances. Especially, the nonlinear partial differential equation is derived that plays a role in a

normal form, that is, in the first approximation, it determines the behavior of all solutions for the original boundary value problem with initial conditions from a sufficiently small neighborhood of equilibrium [20]. Therefore, in order to reduce the accumulation of errors during this process, the numerical algorithm must be highly accurate. To accomplish this goal, high-order compact finite difference schemes have been developed for simulating the solution of applications to wave propagation like [31, 32] and surface water modeling [27, 28]. The investigation of numerical solutions for nonlinear partial differential equations plays an important role in mathematics, physics, and other applied science areas.

High-order finite difference schemes can be classified into two main categories: explicit schemes and Pade-type or compact schemes. These hardcore techniques compute the numerical derivatives directly at each grid by using large stencils, while compact schemes obtain all the numerical derivatives along a grid line using smaller stencils for solving linear and nonlinear systems of equations. Experience has shown that compact schemes are much more accurate than the corresponding explicit scheme of the same order. Hence motivating this performance of implicit (Pade-type) scheme, this paper aims to construct an efficient and accurate sixth-order compact finite difference numerical method for solving a one-dimensional KdV-Burger equation in the application of wave propagation.

Statement of the problem

Consider a homogeneous KdV-Burger equations considered in [24] given by

$$U_t + aUU_x + vU_{xxx} = 0, \qquad (x,t) \in (0,b) \times (0,T), \tag{1}$$

with initial and boundary conditions, respectively,

$$U(x,0) = U_0(x), \qquad 0 \le x \le b, \tag{2}$$

$$U(0,t) = U_o(t), U(b,t) = U_b(t), \qquad 0 \le x \le T, \tag{3}$$

where $U_o(t)$ are $U_b(t)$ are periodic smooth functions and $0 \leqslant T < \infty$. Now we define a mesh size h and $\triangle t$ as constant grid points. They are defined by drawing distant horizontal and vertical lines of distance "h" and "$\triangle t$", respectively, in "x" and "t" directions. These lines are called grid lines, and the point at which they interact is known as the mesh point. This fabric topology point that lies at end of the domains, is called the boundary point. Points that lie inside the region are called interior points. The goal is to find the approximate solution of "$U(x,t)$" at the interior mesh points and to investigate how it converges too. Thus to do this, first, we discretized the solution domain concerning spatial direction as

$$0 = x_0 < x_1 < x_2 < \cdots < x_M = b,$$

where $x_{j+1} = x_j + h$ and $j = 1(1)M$. Moreover, M is the maximum number of grid points in the x-direction. Thus, to accomplish our work, the present paper is organized as follows. Section 2 is a description of numerical methods. Sections 3 is stability, consistence, and convergence analysis. Section 4 is the results of numerical experiments. Sections 5 is the discussion, and Section 6 is the Conclusion.

2 Description of proposed numerical method

2.1 Spatial discretization

In the high-order compact finite difference methods, the spatial derivatives in the governing PDEs are not approximated directly by some finite differences. They are evaluated by some compact difference schemes. We assume that $U(x,t)$ is a sufficiently smooth function and has continuous higher-order partial derivative on its domain. For the sake of simplicity, let $U(x_j,t) = U_j$ and let $\dfrac{\partial^\rho U(x_j,t)}{\partial x^\rho} = \partial_x^\rho U_j$ in which $\rho \geq 1$, be ρth-order partial derivative of $U(x,t)$ concerning with spatial variable x. By using the Taylor series expansion, we have

$$U_{j+1} = U_j + h\partial U_j + \frac{h^2}{2!}\partial_x^2 U_j + \frac{h^3}{3!}\partial_x^3 U_j + \frac{h^4}{4!}\partial_x^4 U_j + \cdots, \tag{4}$$

$$U_{j-1} = U_j - h\partial u_j + \frac{h^2}{2!}\partial_x^2 U_j - \frac{h^3}{3!}\partial_x^3 U_j + \frac{h^4}{4!}\partial_x^4 U_j - \cdots, \tag{5}$$

$$U_{j+2} = U_j + 2h\partial U_j + \frac{4h^2}{2!}\partial_x^2 U_j + \frac{8h^3}{3!}\partial_x^3 U_j + \frac{16h^4}{4!}\partial_x^4 U_j + \cdots, \tag{6}$$

$$U_{j-2} = U_j - 2h\partial U_j + \frac{4h^2}{2!}\partial_x^2 U_i - \frac{8h^3}{3!}\partial_x^3 U_j + \frac{16h^4}{4!}\partial_x^4 U_j - \cdots. \tag{7}$$

Now subtracting (7) from (11), we obtain

$$\partial U_j = \frac{U_{j+1} - U_{j-1}}{2h} - \frac{h^2}{6}\partial_x^3 U_j - \frac{h^4}{120}\partial_x^5 U_j - \cdots. \tag{8}$$

Hence from (10), we obtain the second-order central finite difference equation for the first-order partial derivative of $U(x,t)$ concerning spatial variable x given by

$$\delta_{cx}^{(1)} U_j = \frac{U_{j+1} - U_{j-1}}{2h} + T_1, \tag{9}$$

where $T_1 = -\frac{h^2}{6}\partial_x^3 U_j = \frac{h^2}{6}\partial_x^3 U(x_j,t)$ is its local truncation error. Again subtracting (9) from (8), we obtain

$$\frac{8h^3}{3}\partial_x^3 U_j = U_{j+2} - U_{j-2} - 4h\partial U_j - \frac{32h^5}{120}\partial_x^5 U_j - \frac{128h^7}{5040}\partial_x^7 U_j - \cdots . \quad (10)$$

Now substituting (10) into (10), we obtain

$$\frac{8h^3}{3}\partial_x^3 U_j = U_{j+2} - U_{j-2} - 4h\left(\frac{U_{j+1}-U_{j-1}}{2h} - \frac{h^2}{6}\partial_x^3 U_j - \cdots\right) - \frac{32h^5}{120}\partial_x^5 U_j - \cdots .$$

Collecting like terms in the above difference equation, we obtain

$$\partial_x^3 U_j = \frac{U_{j+2} - 2(U_{j+1}-U_{j-1}) - U_{j-2}}{2h^3} - \frac{7h^2}{60}\partial_x^5 U_j - \frac{31h^4}{2520}\partial_x^7 U_j - \cdots . \quad (11)$$

Hence from (14), we obtain the second-order central finite difference equation for third-order partial derivative of $U(x,t)$ concerning spatial variable x, given by

$$\partial_x^3 U_j = \frac{U_{j+2} - 2(U_{j+1}-U_{j-1}) - U_{j-2}}{2h^3} + T_2, \quad (12)$$

where $T_2 = -\frac{7h^2}{60}\partial_x^5 U_j$ is their local truncation error. Again substituting (11) and (7) into (9), we obtain

$$\begin{aligned}
\delta_{cx}^{(1)} U_j =& \frac{\left(U_j + h\partial_x U_j + \frac{h^2}{2!}\partial_x^2 U_j + \frac{h^3}{3!}\partial_x^3 U_j + \cdots\right)}{2h} \\
&- \frac{\left(U_j - h\partial U_j + \frac{h^2}{2!}\partial_x^2 U_j - \frac{h^3}{3!}\partial_x^3 U_j - \cdots\right)}{2h} + T_1, \\
\delta_{cx}^{(1)} U_j =& \partial_x U_j + \frac{h^2}{6}\partial_x^3 U_j + \frac{h^4}{120}\partial_x^5 U_j + \frac{h^6}{5040}\partial_x^7 U_j + \frac{h^8}{362880}\partial_x^8 U_j + \cdots , \\
\delta_{cx}^{(1)} U_j =& \partial_x U_j + \frac{h^2}{6}\partial_x^3 U_j + \frac{h^4}{120}\partial_x^5 U_j + T_3, \quad (13)
\end{aligned}$$

where $T_3 = \frac{h^6}{5040}\partial_x^7 U_j$ is their principal local traction error. Again substituting (11)-(9) into (15), we obtain

$$\begin{aligned}
\delta_{cx}^{(2)} U_j =& \frac{(U_j + 2h\partial_x U_j + \cdots) - 2\left((U_j + h\partial_x U_j + \cdots) - (U_j - h\partial U_j - \cdots)\right)}{2h^3} \\
&- \frac{(U_j + h\partial_x U_j + \cdots)}{2h^3} + T_2
\end{aligned}$$

$$\delta_{cx}^{(2)}U_j = \partial_x^3 U_j + \frac{31h^2}{120}\partial_x^5 U_j + \frac{8h^4}{315}\partial_x^7 U_j + \frac{51h^6}{72576}\partial_x^9 U_j + \cdots .$$

Thus from this, we obtain the second-order difference equation given by

$$\delta_{cx}^{(2)}U_j = \partial_x^3 U_j + \frac{31h^2}{120}\partial_x^5 U_j + \frac{8h^4}{315}\partial_x^7 U_j + T_4, \tag{14}$$

where $T_4 = \dfrac{51h^6}{72576}\partial_x^9 U_j$ is its principal local truncation error. Now using (2) and applying the successive derivative on it in terms of spatial variable x, we obtain the fifth- and sixth-order central difference scheme, and they are given by

$$\partial_x^5 U_j = \delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right), \tag{15}$$

$$\partial_x^7 U_j = \delta_{cx}^{(3)}\left(\delta_{cx}^{(3)}\left(\delta_{cx}^{(1)}U_j\right)\right). \tag{16}$$

Now substituting (15) into (13), we obtain

$$\delta_{cx}^{(1)}U_j = \partial_x U_j + \frac{h^2}{6}\partial_x^3 U_j + \frac{h^4}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) + T_3,$$

$$\delta_{cx}^{(1)}U_j = \partial_x U_j + \frac{h^2}{6}\delta_{cx}^{(3)}U_j + \frac{h^4}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) + T_3,$$

$$\frac{\partial U_j}{\partial x} = \delta_{cx}^{(1)}U_j - \frac{h^2}{6}\delta_{cx}^{(3)}U_j - \frac{h^4}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) - T_3. \tag{17}$$

Again substituting (15) and(16) into (14), we obtain

$$\delta_{cx}^{(2)}U_j = \partial_x^3 U_j + \frac{31h^2}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) + \frac{8h^4}{315}\delta_{cx}^{(3)}\left(\delta_{cx}^{(3)}\left(\delta_{cx}^{(1)}U_j\right)\right) + T_4,$$

$$\partial_x^3 U_j = \delta_{cx}^{(2)}U_j - \frac{31h^2}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) - \frac{8h^4}{315}\delta_{cx}^{(3)}\left(\delta_{cx}^{(3)}\left(\delta_{cx}^{(1)}U_j\right)\right) - T_4. \tag{18}$$

Now substituting (17) and (18) into (2), we obtain semi-discretization of the given governing PDE and the obtained nonlinear system of ordinary differential equation (ODE) is

$$\frac{dU_j}{dt} + aU_j\left(\delta_{cx}^{(1)}U_j - \frac{h^2}{6}\delta_{cx}^{(3)}U_j - \frac{h^4}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) - T_3\right)$$
$$+ v\left(\delta_{cx}^{(2)}U_j - \frac{31h^2}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) - \frac{8h^4}{315}\delta_{cx}^{(3)}\left(\delta_{cx}^{(3)}\left(\delta_{cx}^{(1)}U_j\right)\right) - T_4\right) = 0.$$

This implies that

$$\frac{dU_j}{dt} = aU_j\left(\frac{h^2}{6}\delta_{cx}^{(3)}U_j + \frac{h^4}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right) - \delta_{cx}^{(1)}U_j\right) \tag{19}$$

$$+v\left(\frac{31h^2}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right)+\frac{8h^4}{315}\delta_{cx}^{(3)}\left(\delta_{cx}^{(3)}\left(\delta_{cx}^{(1)}U_j\right)\right)-\delta_{cx}^{(2)}U_j\right)+T_j,$$

where $T_j=\left(\frac{aU_j}{5040}\partial_x^7+\frac{51v}{72576}\partial_x^9\right)h^6U_j$ is the jth terms of principal local truncation errors with order of accuracy $\mathbf{O}\left(h^6\right)$. Hence by truncating this local truncation error, our proposed scheme is

$$\begin{aligned}
\frac{dU_j}{dt}=&aU_j\left(\frac{h^2}{6}\delta_{cx}^{(3)}U_j+\frac{h^4}{120}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right)-\delta_{cx}^{(1)}U_j\right)\\
&+v\left(\frac{31h^2}{20}\delta_{cx}^{(3)}\left(\delta_{cx}^{(2)}U_j\right)+\frac{16h^4}{105}\delta_{cx}^{(3)}\left(\delta_{cx}^{(3)}\left(\delta_{cx}^{(1)}U_j\right)\right)-6\delta_{cx}^{(2)}U_j\right),\\
\frac{dU_j}{dt}=&\frac{ah^2U_j}{6}\left(\frac{U_{j+2}-2U_{j+1}+2U_{j-1}-U_{j-2}}{2h^3}\right)\\
&+\frac{vh^4}{120}\left(\frac{U_{j+3}-4U_{j+2}+5U_{j+1}-5U_{j-1}+4U_{j-2}-U_{j-3}}{2h^3}\right)\\
&-a\left(\frac{U_{j+1}-U_{j-1}}{2h}\right)\\
&+v\frac{31h^4}{20}\left(\frac{U_{j+3}-4U_{j+2}+5U_{j+1}-5U_{j-1}+4U_{j-2}-U_{j-3}}{2h^3}\right)\\
&-6v\left(\frac{U_{j+2}-2U_{j+1}+2U_{j-1}-U_{j-2}}{2h^3}\right)\\
&+\frac{128vh^4}{840}\Big(\frac{1}{8h^7}\left(U_{j+5}-4U_{j+4}+3U_{j+3}+8U_{j+2}-14U_{j+1}\right.\\
&\qquad\left.+14U_{j-1}-4U_{j-2}-3U_{j-3}+4U_{j-4}-U_{j+5}\right)\Big).
\end{aligned}$$

This implies that

$$\begin{aligned}
\frac{dU_j}{dt}=&\frac{aU_j}{480h}\left(U_{j+3}+36U_{j+2}-315U_{j+1}+315U_{j-1}-36U_{j-2}-U_{j-3}\right)\\
&+\frac{v}{6720h^3}\left(U_{j+5}-4(U_{j+4}-U_{j-4})+2607(U_{j+3}-U_{j-3})\right)\\
&+\left(-30568(U_{j+2}-U_{j-2})+53326(U_{j+1}-U_{j-1})-U_{j-5}\right),
\end{aligned}$$

which implies that

$$\begin{aligned}
\frac{dU_j}{dt}=&\frac{3aU_j}{160h}\left(0.1(U_{j+3}-U_{j-3})+36(U_{j+2}-U_{j-2})-315(U_{j+1}-U_{j-1})\right)\\
&+\frac{0.5v}{3360h^3}(U_{j+5}-U_{j+5})\\
&-\frac{v}{3360h^3}\left(2(U_{j+4}-U_{j-4})+1303.5(U_{j+3}-U_{j-3})\right)\\
&-\frac{v}{3360h^3}\left(15284(U_{j+2}-U_{j-2})+26663(U_{j+1}-U_{j-1})\right),
\end{aligned}\tag{20}$$

where $j=0,1,2,3,\ldots,M-1$. From (20), the nonlinear system of the ODE is given in the form of

$$\begin{aligned}
\frac{dU_0}{dt} =& \frac{3aU_0}{160h}\left(0.1(U_3 - U_{-3}) + 36(U_2 - U_{-2}) - 315(U_1 - 315U_{-1})\right) \\
&+ \frac{0.5v}{3360h^3}(U_5 - U_{-5}) \\
&- \frac{v}{3360h^3}(2(U_4 - U_{-4}) + 1303.5(U_3 - U_{-3}) - 15284(U_2 - U_{-2}) \\
&+ 26663(U_1 - U_{-1})) \\
\frac{dU_1}{dt} =& \frac{3aU_1}{160h}\left(0.1(U_4 - U_{-2}) + 36(U_3 - U_{-1}) - 315(U_2 - 315U_{-0})\right) \\
&+ \frac{0.5v}{3360h^3}(U_6 - U_{-4}) \\
&- \frac{v}{3360h^3}(2(U_5 - U_{-3}) + 1303.5(U_4 - U_{-2}) - 15284(U_3 - U_{-1}) \\
&+ 26663(U_1 - U_{-0})) \\
&\vdots \\
\frac{dU_{M-1}}{dt} =& \frac{3aU_{M-1}}{160h}(0.1(U_{M+4} - U_{M-6}) + 36(U_{M+3} - U_{M-1}) \\
&- 315(U_{M+2} - 315U_M)) \\
&+ \frac{0.5v}{3360h^3}(U_{M+6} - U_{M-4}) \\
&- \frac{1}{3360h^3}\left(2(U_{M+5} - U_{M-3}) + 1303.5(U_M + 4 - U_{M-2})\right) \\
&+ \frac{1}{3360h^3}\left(15284(U_{M+3} - U_{M-1}) - 26663(U_{M+1} - U_M)\right). \quad (21)
\end{aligned}$$

Now by introducing the vectors $U(t) = [U_1(t), U_2(t), U_3(t), \ldots, U_M(t)]^t$ in (21), we can rewrite the system of equation in the matrix form given by

$$\frac{dU(t)}{dt} = AH\left(U(t)\right) \quad (22)$$

subject to discretized initial and nondiscretized boundary conditions given by

$$\begin{aligned}
U(x_j, 0) = U_0(x_j), \qquad 0 \le x_j \le b, \\
U(0, t) = U_0(t), \quad U(b, 0) = U_b(t), \qquad 0 \le t \le T, \quad (23)
\end{aligned}$$

and where H is a nonlinear function of U which contains the entries h_j give by $H(U_j) = h_j(U_1, U_2, \ldots, U_{M-1})$.

2.2 Temporary discretization

Consider the discretization of time domain $[0, T]$ by using the equidistant mesh-size with time step $\triangle t$ given as

$$D_n = \{t_{n+1} = t_n + \triangle t\}, \tag{24}$$

where $\triangle t = T/N$ and $n = 1(1)N$, in which N is the Maximum number of grid points in temporary (time) direction. Then the resulting system of Odes in (22) can now be solved by using the classical fourth-order Runge–Kutta method. This Runge–Kutta techniques integrate the value of U from n to $n + 1$ by using the operator given by

$$\begin{aligned} U_0 &= U(x_j, t_0), \qquad K_1 = \triangle t AH(U_0), \\ U_1 &= U_0 + 1/2K_0, \qquad K_2 = \triangle t AH(U_1), \\ U_2 &= U_0 + 1/2K_1, \qquad K_3 = \triangle t AH(U_2), \\ U_3 &= U_0 + 1/2K_2, \qquad K_4 = \triangle t AH(U_2), \\ U_{n+1} &= U_n + 1/6\left(K_1 + 2K_2 + 2K_3 + K_4\right), \end{aligned} \tag{25}$$

with both discretized initial and boundary conditions given by

$$U(x_j, 0) = U_0(x_j), \qquad 0 \le x_j \le b,$$

$$U(0, t_n) = U_0(t_n), \quad U(b, 0) = U_b(t_n), \qquad 0 \le t_n \le T,$$

where $j = 1(1)M - 1$ and $n = 1(1)N - 1$.

3 Stability, consistency, and convergence method

3.1 Stability of scheme

The stability of the proposed numerical method is investigated by using von Neumann stability analysis. To do this, we assume that the nonlinear term UU_x of partial differential equation in (2) as linear by taking $\gamma = U_j$, where γ is constant. Then without loss of generality, we obtain the linear system of ODEs. Assume that $\gamma = \max_j(U_j)$ in (20), and we can inquire about the eigenvalues of the M by M system of ODEs in (20). To obtain this eigenvalue, as [3, 12, 22, 25] takes, we assume that a trial solution is

$$U(x, t) = \varphi(t)\phi(x). \tag{26}$$

Furthermore to investigate the stability of the proposed method by von Neumann techniques, we assume that the trial function is defined by

$$\phi(x) = e^{iK_P x_j} = e^{ijhK_P}, \tag{27}$$

where $i = \sqrt{-1}$ and $K_P = P\pi$, $P = 1, 2, 3, \ldots, M$. Moreover, K is a Fourier number or amplification factor. Now substituting (26) and (27) into (20), we obtain

$$\begin{aligned}
&\frac{d\varphi(t)e^{ijhK_P}}{dt} \\
&= \frac{3a\gamma\varphi(t)}{160h}\left(0.1(e^{ihK_p(j+3)} - e^{ihK_P(j-3)}) + 4(e^{ihk_P(j+2)} - e^{ihK_p(j-2)})\right) \\
&\quad - \frac{3a\gamma\varphi(t)}{160h}\left(15(e^{ihK_P(j+1)} - e^{ihK_P(j-1)})\right) \\
&\quad + \frac{0.5v}{3360h^3}(e^{ihK_p(j+5)} - e^{ihK_p(j-5)}) \\
&\quad - \frac{v\varphi(t)}{3360h^3}\left(2(e^{ihK_p(j+4)} - e^{ihK_p(j-4)}) + 1303.5(e^{ihK_p(j+3)} - e^{ihK_p(j-3)})\right) \\
&\quad - \frac{v\varphi(t)}{3360h^3}\left(15284(e^{ihK_p(j+2)} - e^{ihk_p(j-2)}) + 26663(e^{ihk_p(j+1)} - e^{ihk_p(j-1)})\right).
\end{aligned}$$

Dividing both sides by e^{ijhk_p}, this implies that

$$\begin{aligned}
\frac{d\varphi(t)}{dt} =& \frac{3a\gamma\varphi(t)}{160h}\left(0.1(e^{3ihK_p} - e^{-3ihK_p}) + 4(e^{2ihK_p} - e^{-2ihK_p})\right) \\
&- \frac{3a\gamma\varphi(t)}{160h}\left(15(e^{ihK_p} - e^{-ihK_p})\right) + \frac{0.5v}{3360h^3}(e^{5ihK_p} - e^{-5ihK_p}) \\
&- \frac{v\varphi(t)}{3360h^3}\left(2(e^{4ihK_p} - e^{-4ihK_a}) + 1303.5(e^{3ihK_p} - e^{-3ihK_p})\right) \\
&- \frac{v\varphi(t)}{3360h^3}\left(15284(e^{2ihK_p} - e^{-2ihK_p}) + 26663(e^{ihK_p} - e^{-ihK_p})\right)
\end{aligned} \tag{28}$$

Since for any differentiable function $\varphi(t)$, the Eigen-value problem of $\varphi(t)$ is

$$d\varphi(t)/dt = \lambda_p\varphi(t)$$

for $p = 1(1)M$. Hence substituting this into (28), we obtain

$$\begin{aligned}
\lambda_p\varphi(t) =& \frac{3a\gamma\varphi(t)}{160h}\left(0.1(e^{3ihK_p} - e^{-3ihK_p}) + 4(e^{2ihK_p} - e^{-2ihK_p})\right) \\
&- \frac{3a\gamma\varphi(t)}{160h}\left(15(e^{ihK_p} - e^{-ihK_p})\right) \\
&+ \frac{0.5v}{3360h^3}(e^{5ihK_p} - e^{-5ihK_p}) \\
&- \frac{v\varphi(t)}{3360h^3}\left(2(e^{4ihK_p} - e^{-4ihK_p}) + 1303.5(e^{3ihK_p} - e^{-3ihK_p})\right) \\
&- \frac{v\varphi(t)}{3360h^3}\left(15284(e^{2ihK_p} - e^{-2ihK_p}) + 26663(e^{ihK_p} - e^{-ihK_p})\right).
\end{aligned}$$

Therefore this implies that

$$\begin{aligned}\lambda_p =&\frac{3a\gamma\varphi(t)}{160h}\left(0.1(e^{3ihK_p}-e^{-3ihK_p})+4(e^{2ihK_p}-e^{-2ihK_p})\right)\\&-\frac{3a\gamma}{160h}\left(15(e^{ihK_p}-e^{-ihK_p})\right)+\frac{0.5v}{3360h^3}(e^{5ihK_p}-e^{-5ihK_p})\\&-\frac{v}{3360h^3}\left(2(e^{4ihK_p}-e^{-4ihK_a})+1303.5(e^{3ihK_p}-e^{-3ihK_p})\right)\\&-\frac{v}{3360h^3}\left(15284(e^{2ihK_p}-e^{-2ihK_p})+26663(e^{ihK_p}-e^{-ihK_p})\right).\end{aligned}\tag{29}$$

From (29), using the Euler expiration into trigonometric form, we obtain

$$\begin{aligned}\lambda_p =&\frac{ia}{1680h^3}\left(0.5\sin(5hK_p)-2\sin(4hK_p)+(6.3\gamma h^2+1303.5)\sin(3hK_p)\right)\\&+\frac{iv}{1680h^3}\left((252\gamma h^2-15284)\sin(2hK_p)+(945\gamma h^2-26663)\sin(hK_p)\right).\end{aligned}\tag{30}$$

Hence from (30), we obtain the required Eigen-value. Real part of all Eigenvalues is zero. This shows that strictly $Real(\lambda_p) \leq 0$ and so that, we have $|\lambda_p| < 1$ for all values of variable "p". Thus the required condition is satisfied. Therefore the obtained system of equation in (20) is stable.

Theorem 1. The difference mathematical statement given in (20) is stabilized equation if and only if all eigenvalues of coefficient matrix of obtained system of difference equation are simultaneously satisfy condition $Real(\lambda_a) \leq 0$.

Proof. The proof of this theorem is briefly given in [22]. □

3.2 Consistency of the scheme

Round-off errors and truncation errors occur when mathematical equations are solved numerically. Rounding errors originate from the fact that computers can only represent numbers (approximate solution) by using a fixed and limited number of significant figures. Therefore, such numbers cannot be represented exactly in different computer memories. The error introduced by this limitation is called a round-off error. Truncation errors in numerical investigation arise when approximations are used to estimate by using the finite term for infinite quantity by truncating certain terms from the boundless series term. Note that in numerical computation, if we work to minimize round-off error, then the truncation error is increased. Therefore it is difficult to avoid these errors in numerical computation. So that, in our work, we must find the balancing (equilibrium) points and minimize both of them simultaneously.

Generally, in all proposed numerical difference schemes, the accuracy of solution for a given mathematical problem obtained by those methods de-

pends on how small we make the step size, h, and time step $\triangle t$. Therefore, if the local truncation error produced in a numerical scheme is near to zero for which both h and $\triangle t$ simultaneously approach to zero, then this numerical method is called consistent.

Hence we have in-minding this, from our present proposed scheme in (20), that the principal part of local truncation error (the derived local shortening error) is

$$\lim_{j\to\infty} T_j = \lim_{j\to\infty} \left(\frac{U_j}{5040}\partial_x^7 + \frac{51}{12096}\partial_x^9 \right) h^6 U_j = 0.$$

Therefore from the above equation and the proposed theoretical-bounds of truncation error, we have $\max_j ||T_j|| \leq \mathbf{O}\ (h^6)$. Hence this implies that $\max_j ||T_j|| \longmapsto 0$ as $h \mapsto 0$. So that, our scheme is consistent with the order of accuracy $\mathbf{O}\ (h^6)$.

3.3 Convergence analysis

Stability plus consistency of the proposed scheme leads to the convergence of the method. Hence our present system is both consistent and stable. So that, the outline present method is convergent with six-order of convergence in the spatial direction. To test the performance of this convergent and accuracy of the proposed method, we use the maximum point-wise absolute error, L_2 and L_∞ norms. Theses norms are calculated by
$L_\infty = \max_j |U(x_j, t_n) - U_{j,n}|$ and $L_2 = \sqrt{1/M \sum_{j=0}^{M} |U(x_j, t_n) - U_{j,n}|}$,
where M is the maximum number of grid points, $U(t_j, t_n)$ is exact and $U_{j,n}$ is an approximation solution of the KdV-Burger equation in (2) at the grid point (x_j, t_n). If the exact solution of the model problem does not exist, then we take the approximate solution $U_{2j,2n} \cong U(x_j, t_n)$ obtained at (x_j, t_n) by taking double number of grid-point as an exact result, and we can compare the error between them at specified grid points.

4 Results of numerical experiments

To test the validity of proposed method, we have considered the following model problem.

Example 1. Consider that one-dimensional KdV equation given by

$$U_t + \alpha \left(U^2\right)_x + \beta U_{xxx} = 0$$

with periodic boundary conditions and the following initial condition:

$$(A) \quad U_0(x) = 3\ a\ \mathrm{sech}(G(x-c)) + 3\ b\ \mathrm{sech}^2(P(x-d))$$

$$a = 0.3,\ b = 0.1,\ c = 0.5,\ 0 \le x \le 2,\ 0 \le t \le 4,$$
$$\beta = 5.84 \times 10^{-4}, \alpha = 0.5, G = \alpha\sqrt{a/\beta}, P = \alpha\sqrt{b/\beta},$$
$$(B) \quad U_0(x) = 3\ a\ \mathrm{sech}(G(x-c)),$$
$$a = 0.3,\ c = 0.5,\ 0 \le x \le 1,\ 0 \le t \le 4,$$
$$\beta = 10^{-4},\ G = 1/2\sqrt{a/\beta},$$
$$(C) \quad U_0(x) = \frac{2}{3}\ a\ \mathrm{sech}^2(\frac{(x-1)}{\sqrt{108\beta}}),$$
$$a = 0.5,\ 0 \le x \le 3, 0 \le t \le 4,$$
$$\beta = 10^{-4},$$

Example 2. Consider the one-dimensional KdV equation considered in [37] given by

$$U_t + 6UU_x + U_{xxx} = 0.$$

With analytical solution obtained in [37] given by

$$U(x,t) = \frac{c}{2}\ \mathrm{sech}^2(\frac{\sqrt{c}}{2}(x - ct - D)),$$

where D is a constant of integration.

Example 3. Consider the KdV equation considered in [30] given by

$$U_t + (\alpha + \beta U)UU_x + \gamma U_{xx} - \delta U_{xxx} = 0,$$

where initial condition of this equation for $\gamma = 0$ and $\delta = -1$ is given by

$$U(x,0) = -\frac{\alpha}{\beta}\left(1 + \tanh\left(\frac{\alpha}{2\sqrt{-6\beta}}x\right)\right).$$

In [30], by letting $\beta = 0$, $\alpha = 2$, $\gamma = -5$, and $\delta = -3$, its analytical solution is given by

$$U(x,t) = \frac{1}{3}\left(\mathrm{sech}^2(\frac{\theta}{2}) + 2\tanh(\frac{\theta}{2}) + 2\right),$$

where $\theta = -\frac{1}{3}x + \frac{2}{3}t$.

Example 4. Consider that fractional KdV equation considered in [1] given by

$$\frac{\partial^\alpha U}{\partial t^\alpha} + U\frac{\partial U}{\partial x} + \frac{\partial^3 U}{\partial x^3} = F(x,t), \qquad (x,t) \in (0,1).$$

Analytical solution obtained in [1] is given by

$$U(x,t) = te^x,$$

by assuming $\alpha = 0.9$.

Table 1: Point-wise absolute error and root mean square error, for Example 1 (A) if $M = 61$

provided grid x	points t	estimated error at L_∞	provided grid point L_2
0.66667	3.9333	$4.7839E-03$	$1.1526E-04$
1.6667	3.9833	$4.4147E-05$	$3.1165E-06$
1.9667	3.9983	$3.5070E-05$	$2.300E-06$
2	4	$2.4568E-06$	$2.300E-06$

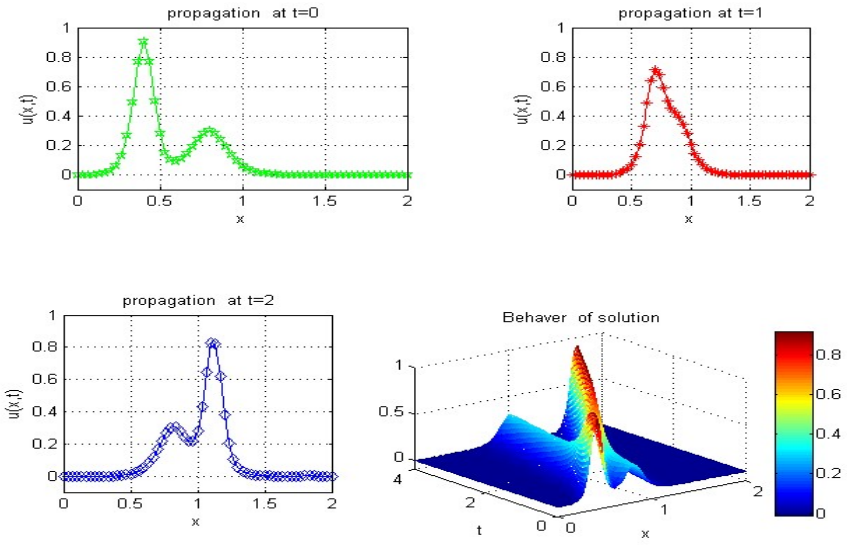


Figure 1: Graph of the physical behavior of the solution when $\beta = 4.25 \times 10^{-04}$ and $M = 61$ with high-frequency oscillations for Example 1 (A).

5 Discussions

In this paper, sixth-order compact finite difference method is presented for solving a one-dimensional KdV-Burger equation. At first stage, we discretized the solution interval for concerning spatial variable. Next by applying the Taylor series expansion, we discretization partial derivative of the model problem in (2) involving spatial variable, we obtain sixth-order finite difference

Table 2: Point-wise absolute error and root mean square error, for Example 1 (B) if $M = 61$

provided grid	points	estimated error at	provided grid point
x	t	L_∞	L_2
0.333	3.967	$4.7E-04$	$1.3838E-05$
0.8333	3.9917	$3.600E-05$	$32.2569E-06$
0.9833	3.9992	$3.532E-06$	$1.3216E-07$
1	4	$2.473E-07$	$3.2814E-08$

Table 3: Point-wise absolute error and root mean square error, for Example 1 (C) if $M = 61$

provided grid	points	estimated error at	provided grid point
x	t	L_∞	L_2
1.333	3.8667	$7.2E-04$	$3.6786E-05$
3.333	3.9677	$9.7E-05$	$4.0591E-06$
3.9333	3.9967	$6.25E-07$	$5.1082E-07$
4	4	$3.348E-07$	$3.126E-08$

Table 4: Point-wise absolute error and root mean square error obtained by the present method, for Example 2 using different step sizes h and time step

provided step size and	time step	Estimated Error at	provided grid point
h	$\triangle t$	L_∞	L_2
0.1	0.5	$8.128E-04$	$2.1526E-05$
0.05	0.5	$5.9148E-05$	$7.6374E-06$
0.02	0.1	$4.1139E-05$	$5.2931E-06$
0.01	0.1	$3.8425E-06$	$9.2631E-07$

Table 5: Point-wise absolute error and root mean square error obtained by the present method, for Example 3 with different step sizes h and time step

provided dance step size and	time step	Estimated Error at	provided grid point
h	$\triangle t$	L_∞	L_2
0.1	0.04	$1.3298E-06$	$6.3691E-07$
0.1	0.03	$5.1653E-07$	$4.6334E-07$
0.1	0.02	$4.623E-07$	$4.211E-07$
0.1	0.01	$1.975E-07$	$8.091E-08$

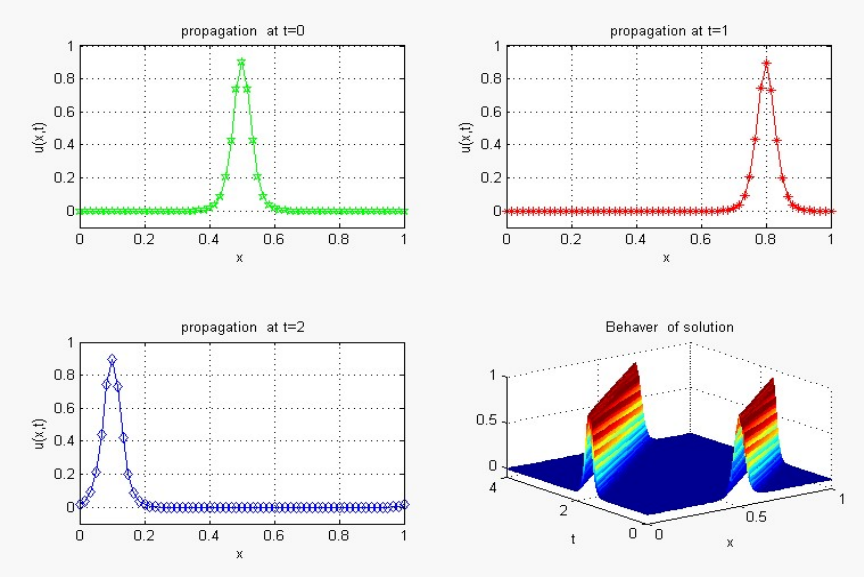


Figure 2: Graph of the physical behavior of the solution when $\beta = 4 \times 10^{-04}$ and $M = 61$ with high-frequency oscillations for Example 1 (B).

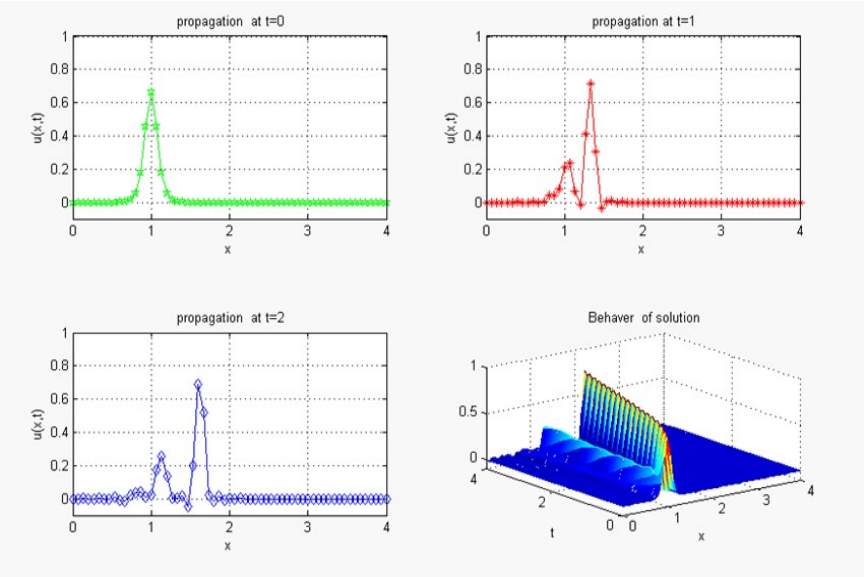
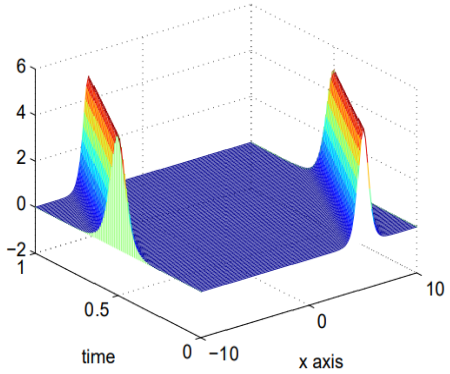


Figure 3: Graph of the physical behavior of the solution when $\beta = 10^{-04}$ and $M = 61$ with high-frequency oscillations for Example 1 (C).

a) Surface of exact solution



b) Above-ground of approximation result

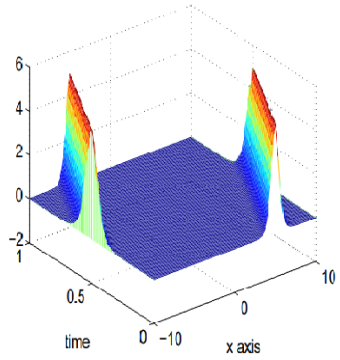


Figure 4: Graph of the physical behavior of the surface of the present numerical solution with mesh sizes $h = 0.01$ and $k = 0.01$ for Example 2 which we compare it with the surface of exact solution in [37]

scheme. Then, by rearranging these semi-discretized schemes and combining them into the remaining part of partial derivative in the model problem concerning temporal variable without linearizing it. We obtain a system of nonlinear ODEs. Then, we solve the obtained system by using the fourth-order Runge–Kutta method. To demonstrate the competence of playacting (to identify the applicability and validity of the present proposed scheme), four model examples are solved by taking different values for step sizes h and dispersion coefficients v, α, and β in the different solution domains. To further verify this validity of the proposed method, the norm of errors between the numerical results for model problem in Example 1 subjected to three different initial conditions associated with numerical results obtained by dabbling mesh size $(2M, 2N)$ taking as the exact solution, is summarized

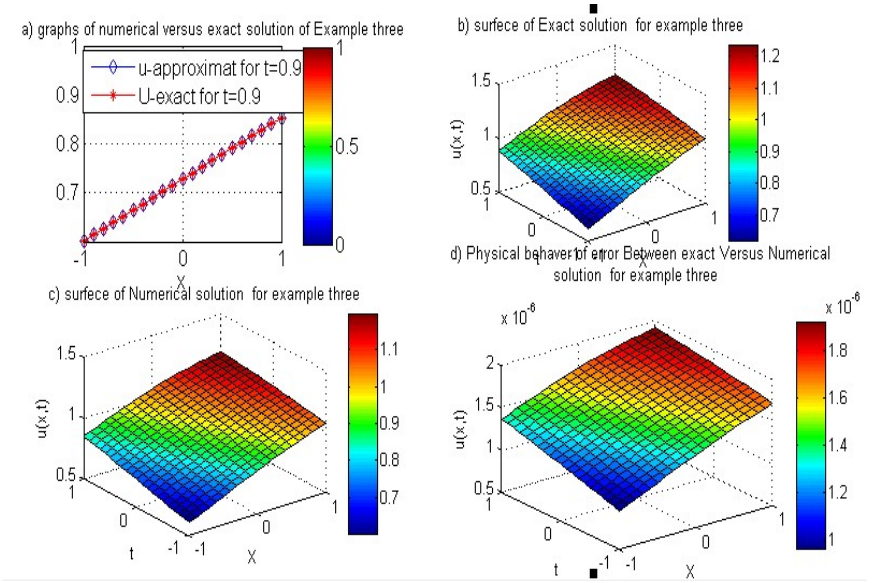


Figure 5: Graph of the physical behavior of the surface of the present numerical solution with mesh sizes $h = 0.05$ and $k = 0.01$ for Example 3, which we compare it with the solution in [30]

Table 6: Comparison of point-wise absolute error and root mean square error, for Example 4 using different step sizes h and time step $\triangle t$

provided step size and time step result by present method h	$\triangle t$	Estimated Error at L_∞	provided grid point L_2
1/10	1/20	$4.5349E-05$	$8.3691E-06$
1/30	1/20	$5.1653E-06$	$5.6334E-06$
1/50	1/20	$3.5283E-06$	$9.4184E-07$
Result by Method in [1]			
1/10	1/20	$1.5678E-04$	$6.1032E-04$
1/30	1/20	$7.8531E-05$	$3.9868E-05$
1/50	1/20	$1.6579E-05$	$7.1991E-06$

in Tables 1–3 and Figures 1–3 for different disparate parameters and its solution domain.

As it can be seen from these tables, the point-wise maximum absolute errors (L_∞) and root mean square error (L_2) are rapidly decreases as the value of x and t increase. This indicates that the vibration of the solution line has to become overlapping and that the dispersion of the travelling wave is equal. From those figures, also we can understand that, for different values of β and initial conditions, we obtain differences in diametrical travelling

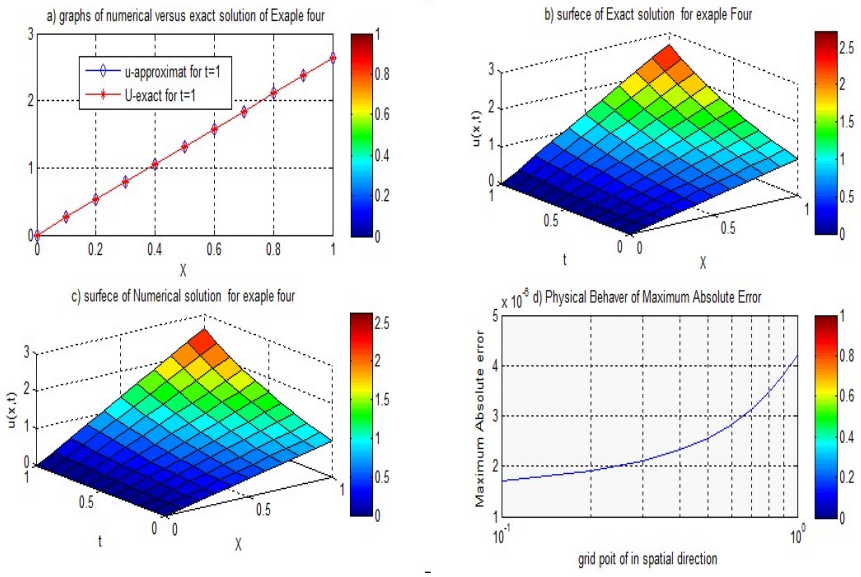


Figure 6: Graph of the physical behavior and surface of the present numerical solution with mesh sizes $h = 0.01$ and $\triangle t = 0.1$ for Example 4, which we compare it with the surface of exact solution in [1]

wave, and at a certain point in the assorted domain, the vibration line of the travelling wave is overlapping. Hence, in this case, a numerical approximation of Example 1 for different values of β, initial conditions, and the solution interval is very well, and it gives different behaviors of the solution. From Table 4 and Figure 4, we conclude that the present numerical resolution for Example 2 is accurate than the quantitative result in [37]. Thus this not only shows the solution given by the graphical form, but also as we saw from Table 4, if mesh-size decreases, both roots mean square error and point-wise error of the present numerical result are rapidly decreased. So that, this shows that the accuracy of the present acting method is increased and proves the preexisting accuracy. Hence the approximate solution by the present method for Example 2 is very well. Also, as we saw from Table 5 and Figure 5 that the present numerical solution of Example 3 is accurate than the numeric resolution in [30]. Because of, as it can be seen from the obtained error norm of numerical results specified in Table 5, both roots mean square error and maximum point-wise error are decreased. This shows the correcting and proving the preexisting results in [30]. Hence our result is more accurate than the event that exists in the literature. In addition to this, the present techniques are also used to solve a fractional order KdV-Burger equation. As it can be seen from the comparison of accuracy through the error norm for numerical results given in Table 6, the quality of the present method is more

superstar than the truth of solution in [1] for $\alpha = 0.9$. Because of both L_∞ and L_2 rapidly decrease as h decreases to zero with a fixed time step $\triangle t$.

Therefore, depending on all accuracy of the present numerical results, the proposed method validates and improves the accuracy of preexisting methods described in the literature. Also in this paper, both the theoretical and numerical error bounds have been established. The obtained quantitative results listed in the tables and graphs are also confirmed our established computational rate of convergence and theoretical estimates of the error bound.

6 Conclusion

In this study, a new approach, which was called sixth-order compact finite difference method, was presented to solve a one-dimensional KdV-Burger equation. The comparison of results obtained by the present method acting was more convenient, reliable, and effective than some methods listed in the literature. An error analysis based on the von Neumann stability investigating was also developed for the present study. So that, as it can be seen from both graphs and tables, the accuracy of the present method has improved the inaccuracy of preexisting techniques in literature by decreasing the value of h and $\triangle t$. As a summary, the accuracy of numerical results given in terms of tables and graphs indicated that as the values of h and $\triangle t$ decrease, the errors drop-off more rapidly. Generally, this technique is a reliable acting method and ables solving the one-dimensional KdV-Burger equation. Based on the findings, this method is well approximate and gives better accuracy of the numerical solution with both decreasing or some fixed step sizes h and time step $\triangle t$. Therefore this method is a suitable technique to approximate the exact solution of PDE very well.

Acknowledgments

Authors are grateful to their anonymous referees and editor for their constructive comments.

References

1. Ahmad, I., Ahmad, H., Inc, M., Rezazadeh, H., Akbar, M.A., Khater, M.M., Akinyemi, L. and Jhangeer, A. *Solution of fractional-order Korteweg-de Vries and Burgers' equations utilizing local meshless method*, J. Ocean Eng. Sci. (2021).

2. Ahmad, H., Khan, T.A., Stanimirovic, P.S. and Ahmad, I. *Modified variational iteration technique for the numerical solution of fifth order KdV-*

type equations, J. Appl. Comput. Mech. (2020).

3. Aliyi, K. and Muleta, H. *Numerical method of the line for solving one dimensional initial-boundary singularly perturbed Burger equation*, Indian J. Adv. Math. 1(2) (2021), 4–14.

4. Aliyi, K., Shiferaw, A. and Muleta, H. *Radial basis functions based differential quadrature method for one dimensional heat equation,* American Journal of Mathematical and Computer Modelling, 6 (2021), 35–42.

5. Benney, D.J. *Long waves on liquid films*, J. Math. Phys. 45 (1966), 150–155.

6. Chen, B. and Xie, Y.C. *Exact solutions for wick-type stochastic coupled Kadomtsev- Petviashili equations,* J. Phys. A, 38 (2005), 815–822.

7. Chen, B. and Xie, Y.C. *Exact solutions for generalized stochastic Wick-type KdV-mKdV equations* , Chaos Solitons Fractals, 23 (2005), 281–287.

8. Chen, B. and Xie, Y.C. *Periodic-like solutions of variable coefficient and Wick type stochastic NLS equations*, J. Comput. Appl. Math. 203 (2007), 249–263.

9. Feng, Z. and Qing-guo, M. *Burgers-Korteweg-de Vries equation and its travelling solitary waves* , Sci. China Ser. A, 50 (3) (2007), 412-422.

10. Gao, G. *A theory of interaction between dissipation and dispersion of turbulence* , Sci. Sinica Ser. A, 28 (1985), 616–627.

11. Ghany, H.A., and Fathallah, A. *Exact solutions for KDV-Burger equations with an application of white-noise analysis* , Int. J. Pure Appl. Math. 78(1) (2012), 17–27.

12. Gowrisankar, S. and Natesan, S. *Uniformly convergent numerical method for singularly perturbed parabolic initial-boundary-value problems with equidistributed grids,* Int. J. Comput. Math. 91 (2014), 553–577.

13. Grad, H. and Hu, P.N. *Unified shock profile in a plasma*, Phys Fluids, 10 (1967), 2596–2602.

14. Gurlu, Y.U. and Kaya, D. *Analytic method for solitary solutions of some partial differential equations*, Phys. Lett. A, 370 (2007), 251–259.

15. Hixon, R. and Turkel, E. *Compact implicit MacCormack-type schemes with high accuracy*, J. Comput. Phys. 158 (2000), 51–70.

16. Hu, P.N. *Collisional theory of shock and nonlinear waves in a plasma.* , Phys. Fluids. 15 (1972), 854–864.

17. Johnson, R.S. *A nonlinear equation incorporating damping and dispersion* , J. Fluid Mech. 42 (1970), 49–60.

18. Johnson, R.S. *Application of He's homotopy perturbation method to nonlinear integro-differential equations*, Appl. Math. Comput. 188(1) (2007), 538–548.

19. Johnson, R.S. *A modern introduction to the mathematical theory of water waves,* Cambridge, Cambridge University Press, 1997.

20. Kashchenko, S.A. *Normal form for the KdV-Burgers equation*, Dokl. Math. 93 (2016), 331–333.

21. Kaya, D. *An application of the decomposition method for the two-dimensional KdV-Burgers equation* , Comput. Math. Appl. 48 (2004), 1659–1665.

22. Koroche, A. K. *Numerical solution for one dimensional linear types of parabolic partial differential equation and application to heat equation*, Mathematics and Computer Science, 5 (2020), 76–85.

23. Korteweg, D.J. and de Vries, G. *On the change of form of long waves advancing in a Rectangular canal and on a new type of long stationary waves*, Lond. Edinb. Dublin philos. mag. j. sci. 39 (1895), 22–43.

24. Kudryashov, N.A. *On "new travelling wave solutions" of the KdV and the KdV-Burgers equations,* Commun. Nonlinear Sci. Numer. Simul. 14(5) (2009), 1891–1900.

25. Kutluay, S., Esen, A. and Dag, I. *Numerical solutions of the Burgers' equation by the least-squares quadratic B-spline finite element method*, J. Comput. Appl. Math. 167 (2004), 21–33.

26. Li, J. and Visbal, M.R. *High-order compact schemes for nonlinear dispersive waves*, J. Sci. Comput., 26 (2006), 1–23.

27. Navon, I.M. and Riphagen, H.A. *An implicit compact fourth order algorithm for solving the shallow-water equations in conservation-law form* , Mon. Weather Rev. 107 (1979), 1107–1127.

28. Navon, I.M. and Riphagen, H.A. *SHALL4 – An implicit compact fourth-order Fortran program for solving the shallow-water equations in conservation-law form* , Comput. Geosci. 12 (1986), 129–150.

29. Shang, J.S. *High-order compact-difference schemes for time-dependent Maxwell equations*, J. Comp. Phys. 153 (1999), 312–333.

30. Shi, Y.F. , Xu, B. and Guo, Y. *Numerical solution of Korteweg-de Vries-Burgers equation by the compact-type CIP method*, Adv. Differ. Equ. (2015), 1–9.

31. Spotz, W.F. and Carey, G.F. *Extension of high order compact schemes to time dependent problems* , Numer. Methods Partial Differential Equations 17 (2001), 657–672.

32. Visbal, M.R. and Gaitonde, D.V. *Very high-order spatially implicit schemes for computational acoustics on curvilinear meshes,* J. Comput. Acoust. 9 (2001), 1259–1286.

33. Wadati, M. *Deformation of solitons in random media*, J. Phys. Soc. Japan 59 59 (1990), 4201–4203.

34. Wadati, M. and Akutsu, Y. *Stochastic Korteweg de Vries equation,* J. Phys. Soc. Japan, 53 (1984), 3342–3350.

35. Wazzan, L. *A modified tanh-coth method for solving the KdV and the KdV–Burgers equations* , Commun. Nonlinear Sci. Numer. Simul. 14 (2009), 443–450.

36. van Wijngaarden, L. *On the motion of gas bubbles in a perfect fluid* , Arch. Mech. (Arch. Mech. Stos.) 34(3) (1982), 343–349 (1983).

37. Xiang, T. *A summary of the Korteweg-de Vries Equation*, (2015).

38. Xie, Y.C. *Exact solutions for stochastic KdV equations*, Phys. Lett. A, 310 (2003), 161–167.

How to cite this article
K. Aliyi Koroche and H. Muleta Chemeda Sixth-order compact finite difference method for solving KDV-Burger equation in the application of wave propagations. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 277-300. doi: 10.22067/ijnao.2021.72366.1058.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 301–314
DOI: 10.22067/ijnao.2021.70974.1043
https://ijnao.um.ac.ir/

A generalization of the ABS algorithms and its application to some special real and integer matrix factorizations

E. Golpar-Raboky* and N. Mahdavi-Amiri

Abstract

In 1984, Abaffy, Broyden, and Spediacto (ABS) introduced a class of the so-called ABS algorithms to solve systems of real linear equations. Later, the scaled ABS algorithm, the extended ABS algorithm, the block ABS algorithm, and the integer ABS algorithm were introduced leading to various well-known matrix factorizations. Here, we present a generalization of ABS algorithms containing all matrix factorizations such as triangular, WZ, and ZW. We present the octant interlocking factorization and show that the generalized ABS algorithm is more general to produce the octant interlocking factorization.

AMS subject classifications (2020): 15A03; 15A21; 15A23; 34C20; 49M27.

Keywords: ABS algorithms; Quadrant interlocking factorization; Octant interlocking factorization.

1 Introduction

The basic ABS class of algorithms was first introduced by Abaffy, Broyden, and Spedicato [1] for solving linear systems of equations. Let $\mathbb{R}$ and $\mathbb{R}^{m\times n}$ denote the set of real numbers and the set of $m\times n$ real matrices, respectively. Consider the system of linear equations

* Corresponding author

Received 14 June 2021; revised 9 November 2021; accepted 24 December 2021

Effat Golpar-Raboky
Department of Mathematics, University of Qom. Qom, Iran. Tel: +98-25-32103791 e-mail:g.raboky@qom.ac.ir

Nezam Mahdavi-Amiri
Faculty of Mathematical Sciences, Sharif University of Technology, Tehran, Iran. e-mail: nezamm@sharif.edu

$$Ax = b, \qquad x \in \mathbb{R}^n, \quad A \in \mathbb{R}^{m\times n}, \quad b \in \mathbb{R}^m, \ m \leq n, \tag{1}$$

where $rank(A)$ is arbitrary. With $A = [a_1, \ldots, a_m]^T$, the system is equivalently written as

$$a_i^T x = b_i, \quad i = 1, \ldots, m. \tag{2}$$

An ABS algorithm generates a sequence of approximations x_i such that x_{i+1} is a particular solution of the first i equations and leads to the general solution of a linear system by computing a particular solution and a matrix with rows producing the null space of the coefficient matrix.

An ABS method starts with an arbitrary initial vector $x_1 \in \mathbb{R}^n$ and a nonsingular matrix $H_1 \in \mathbb{R}^{n\times n}$. Given x_i , a solution of the first $i-1$ equations, and H_i a matrix with rows generating the null space of the first $i-1$ rows of the coefficient matrix, an ABS algorithm computes x_{i+1} as a solution of the first i equations and H_{i+1}, with rows generating the null space of the first i rows of the coefficient matrix. Below, we give the class of basic ABS algorithms for solving systems of linear equations (13).

Algorithm 1 (Basic ABS algorithm)**.**
(1) Give $x_1 \in \mathbb{R}^{n\times n}$, arbitrary, $H_1 \in \mathbb{R}^{n\times n}$, arbitrary and nonsingular. Set $i = 1$ and $r = 0$.
(2) Compute $\tau_i = a_i^T x_i - b_i$ and $s_i = H_i a_i$.
(3) **If** ($s_i = 0$ and $\tau_i = 0$), **then** let $x_{i+1} = x_i, H_{i+1} = H_i$ and **go to** (6) (the ith row of A is dependent on its first $i-1$ rows). **If** $s_i = 0$ and $\tau_i \neq 0$, **then stop** (the ith equation and hence the system is incompatible).
(4) Compute the search vector p_i by

$$p_i = H_i^T f_i, \tag{3}$$

where $f_i \in \mathbb{R}^n$ is an arbitrary vector satisfying $a_i^T H_i^T f_i \neq 0$. Compute

$$\alpha_i = \frac{\tau_i}{a_i^T p_i}$$

and

$$x_{i+1} = x_i - \alpha_i p_i.$$

(5) (Update the null space generator) Update H_i by

$$H_{i+1} = H_i - \frac{H_i a_i q_i^T H_i}{q_i^T H_i a_i}, \tag{4}$$

where $q_i \in R^n$ is an arbitrary vector satisfying $s_i^T q_i \neq 0$, and let $r = r + 1$.
(6) **If** $i = m$, **then Stop** (H_{m+1}^T generates the null space of A and r is its rank) **else** let $i = i + 1$ and **go to** (2).

Here, we recall some properties of ABS algorithms; for more details, see [3]. For simplicity, we assume that $A \in \mathbb{R}^{m\times n}$ has full row rank.

1. The system may be incompatible, which can be detected in step (3), when $s_i = 0$ and $\tau_i = b_i - a_i^T x_i \neq 0$.
2. $H_i a_i \neq 0$ if and only if a_i is linearly independent of $a_1, \ldots, a_{i-1}$.
3. If $a_1, \ldots, a_i$ are linearly independent, then the search vectors $p_1, \ldots, p_i$ are linearly independent.
4. If $a_1, \ldots, a_i$ are linearly independent, then with $P_i = (p_1, \ldots, p_i)$, the implicit factorization $AP_i = L_i$ holds, where L_i is nonsingular lower triangular. Different choices of the parameters H_1, f_i, and q_i lead to different matrix factorizations.

Theorem 1 (LX factorization)**.** Let $A \in \mathbb{R}^{n\times n}$ be a nonsingular matrix and let $H_1 = I$. Then, there exits an index set $k_1 < k_2 < \cdots < k_n$ such that $e_{k_i}^T H_i a_i \neq 0$, for $k = 1, \ldots, n$, and the parameter choices $f_i = q_i = e_{k_i}$ are well-defined. Let $[n] = \{1, \ldots, n\}$, $B_i = \{k_1, \ldots, k_i\}$, and $N_i = [n]\backslash B_i$. Then, we have the following properties:
(1) Every kth row of H_{i+1} with $k \in B_i$ is a null row.
(2) The vector p_i has $n - i$ zero components; its k_ith component is equal to one.
(3) For each $k \in N_i$, the kth column of H_{i+1} is the unit vector e_k, while for each $k \in B_i$, the kth column of H_{i+1} has zero components in the jth position, with $j \in B_i$, implying that only $i(n - i)$ elements need to be computed for H_{i+1}.

Proof. See [17]. □

Corollary 1 (LU factorization)**.** Let $A \in \mathbb{R}^{n\times n}$ be a strongly nonsingular matrix (that is, the determinants of all the principal submatrices are nonzero) and let $H_1 \in \mathbb{R}^{n\times n}$ be the identity matrix. Then
(1) the sequence $\{H_i\}$, where $q_i = \frac{e_i}{e_i^T H_i a_i}$, is well-defined.
(2) the first i rows of H_{i+1} are identically zero and the last $n - i$ columns of H_{i+1} are equal to the last $n - i$ columns of H_i.
(3) $P = (p_1, \ldots, p_n)$ is an upper triangular matrix.

Proof. See [3, Theorems 6.3 and 6.5]. □

The scaled ABS method produces a matrix factorization $V^T AP = L$, where L is a lower triangular matrix. Choices of the parameters H_1, f_i, and q_i determine particular methods within the class so that various matrix factorizations are derived; see [1, 3, 4, 18, 16, 17, 19].

Obviously, the original system (13) is equivalent to the following scaled system:

$$V^T Ax = V^T b, \tag{5}$$

where V, the scale matrix, is an arbitrary nonsingular m by m matrix. By replacing a_i with $A^T v_i$ in Algorithm 1, a scaled ABS algorithm is obtained. Chen and Zhou [5] proposed a generalization of the ABS algorithms, named as extended ABS (EABS) class of algorithms, which differs from the ABS

class of algorithms only in updating the Abaffian matrices H_i. The block ABS algorithm was developed by Abaffy and Galantai [2]. Let $n_1, \ldots, n_s$ be positive integer numbers such that $n_1 + \cdots + n_s = n$. Consider a block form of A as $A = (A_1^T, \ldots, A_s^T)^T$, where $A_i \in \mathbb{R}^{n_i \times n}$, for $i = 1, \ldots, s$. The Block ABS methods may be formulated as follows:
(1) Compute $S_i = H_i A_i^T$.
(2) Determine $F_i \in \mathbb{R}^{n \times n_i}$ such that $F_i^T S_i$ is nonsingular and set $P_i = H_i^T F_i$.
(3) Update the Abaffian matrix H_i by

$$H_{i+1} = H_i - H_i A_i^T (Q_i^T H_i A_i^T)^{-1} Q_i^T H_i, \tag{6}$$

where $Q_i \in \mathbb{R}^{n \times n_i}$ is an arbitrary matrix such that $S_i^T Q_i$ is nonsingular. Esmaeili, Mahdavi-Amiri, and Spedicato [7] presented the integer ABS class of algorithms for solving linear Diophantine equations, developed conditions for the existence of an integer solution, and determined all integer solutions [6]. An extension of the integer ABS algorithm using the scaled ABS algorithms was developed by Spedicato et al. [16]. A new class of extended integer ABS algorithms for solving linear Diophantine systems by computing an integer basis for the null space while controlling the growth of intermediate results was developed by Khorramizadeh and Mahdavi-Amiri [13]. Golpar-Raboky and Mahdavi-Amiri [10, 11, 12, 14] presented new ideas for updating the H_i leading to the development of a new class of extended integer ABS algorithms. They also showed how to compute the Smith normal form of an integer matrix using the scaled integer ABS algorithm [10].

For the (not necessarily independent) rows of H_{i+1} to be a generator of null space of the first i rows of A, the extended integer ABS algorithms can always be tuned to produce an integer basis for the integer null space of the coefficient matrix; see Esmaeili, Mahdavi-Amiri, and Spedicato [6].

2 Generalized ABS class of algorithms

The central problem of linear algebra is the solution of linear system of equations. Direct methods used to solve linear systems of equations are based on factorizations of the coefficient matrix into factors to be easy for use in solving the equations. The Gaussian elimination with the corresponding matrix decomposition, the LU decomposition, is the most useful method for solving linear equations. The method is well-defined if and only if A is strongly nonsingular, that is, all principal submatrices, $A(1:k, 1:k)$, for all k, are nonsingular. The parallel implicit elimination (PIE) method and the WZ factorization for solving large systems, suitable for parallel computers, have been introduced by Evans [15].

Definition 1. Let $[n] = \{1, \ldots, n\}$ and let $\alpha_i \subset [n]$, for $i = 1, \ldots, s$. We say $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ is an index set of $[n]$ if and only if $\alpha_i \bigcap \alpha_j = \emptyset$, whenever $i \neq j$, and $\cup_{i=1}^{s} \alpha_i = [n]$. We denote the cardinality of α by $|\alpha|$.

Let $|\alpha_i| = n_i$, for $i = 1, \ldots, s$. Then, an index set $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ is a block permutation vector, when the ith block size is equal to n_i and $n = \sum_{i=1}^{s} n_i$.

Let $A \in \mathbb{R}^{m\times n}$ and let α and β be two index sets of $[n]$. Let $A(:, b)$ denote the $m \times |b|$ submatrix of A containing the columns specified by b and let $A(\alpha_i, \beta_j)$, the (i, j)th block of A, denote the $|\alpha_i| \times |\beta_j|$ submatrix of A composed of the rows specified by α_i and the columns specified by β_j. If $\alpha_i = \beta_j$, then $A(\alpha_i, \beta_j)$ is a principal submatrix of A and if $\alpha_i = \beta_j = \{1, \ldots, k\}$, $1 \le k \le \min\{m, n\}$, then $A(\alpha_i, \beta_j)$ is a leading principal submatrix of A.

Definition 2. Let $A \in R^{n\times n}$, let $t = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$ be two index sets of n, and let $A(\alpha_i, \beta_j)$, $1 \le i, j \le s$, denote the (i, j)th block of A. Then,

$$A(\alpha_1, \beta_1) \subset \cdots \subset A(\cup_{i=1}^{s}\alpha_i, \cup_{j=1}^{s}\beta_j) = A \tag{7}$$

is a nested submatrix sequence of A. We say A is (α, β)-block strongly nonsingular if and only if $A(\cup_{i=1}^{k}\alpha_i, \cup_{i=1}^{k}\beta_i)$, for $k = 1, \ldots, s$, are nonsingular.

Let P_α and P_β denote the permutation matrices moving the rows and columns of A based on the index sets α and β, respectively, and let

$$A_{\alpha,\beta} = P_\alpha^T A P_\beta. \tag{8}$$

Corollary 2. Let $A \in \mathbb{R}^{n\times n}$. Then A is (α, β)-block strongly nonsingular if and only if $A_{\alpha,\beta}$ is strongly nonsingular.

Note that Theorem 1 provides a relationship between elimination methods and the ABS method. The search vectors p_i in step (4) of Algorithm 1 are aligned sequentially. The generalized elimination method and the ABS method do not produce the same matrix factorizations, generally. In the next section, we present a generalization of the ABS algorithms containing all matrix factorizations produced by the generalized elimination method such as triangular, WZ, and ZW. The generalized ABS method is more general to produce some new matrix factorizations such as the SO and OS factorizations which cannot be produced by the generalized elimination method.

Let $A \in \mathbb{R}^{n\times n}$, let $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$ be two index sets, and let $A_i \in \mathbb{R}^{|\alpha_i|\times n}$ be such that $A_i = A(\alpha_i, :)$.

Algorithm 2 (Generalized ABS algorithm)**.**
Input: $A \in \mathbb{R}^{m\times n}$, an arbitrary nonzero vector $x \in \mathbb{R}^{m\times n}$, an arbitrary nonsingular matrix $H_1 \in \mathbb{R}^{n\times n}$, and two index sets $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$.
For $i = 1, \ldots, s$ do
(1) Compute $S_i = H_i A(\alpha_i, :)$.
(2) Determine $F_i \in \mathbb{R}^{n\times\beta_i}$ such that $F_i^T S_i$ is nonsingular and set $P(:, \alpha_i) = H_i^T F_i$.
(3) Update the approximation for the solution by

$$x(\alpha_{i+1}) = x(\alpha_i) - P(:, \alpha_i)d_i,$$

where d_i is the unique solution of the following nonsingular system:

$$A(\alpha_i, :)P(:, \alpha_i)d_i = r_i,$$

and $r_i = A(\alpha_i, :)x_i - \beta_i$.
(3) Update the Abaffian matrix H_i by

$$H_{i+1} = H_i - H_i A(\alpha_i, :)(Q_i^T H_i A(\alpha_i, :))^{-1} Q_i^T H_i, \tag{9}$$

where $Q_i \in \mathbb{R}^{n \times \alpha_i}$ is an arbitrary matrix so that $S_i^T Q_i$ is nonsingular.
end for.
(4) Let $P = (p_1, \ldots, p_n)$ and compute $C = AP$.

Algorithm 2 provides a null space characterization for the matrix A. From Algorithm 2, we have $H_{i+1}a_j = 0$, for $j \in \cup_{k=1}^{i} \alpha_k$, where a_j^T is the jth row of A. According to (8), Theorem 1 and Corollary 1, we have the following results.

Theorem 2. Let $A \in \mathbb{R}^{n \times n}$, and let $\alpha = (\alpha_1, \ldots, \alpha_s)$ and $\beta = (\beta_1, \ldots, \beta_s)$ be two index sets. Then, A is (α, β)-block strongly nonsingular if and only if $I(:, b(i))^T H_i A(t(i), :)^T$, for $i = 1, \ldots, s$, are nonsingular.

Theorem 3. Let $Q_i = I(:, \beta_i)$, and let H_{i+1} be defined by (9). Then, the following properties hold:
(a) The jth row of H_{i+1} is zero, for $j \in \cup_{k=1}^{i} b(k)$.
(b) The jth column of H_{i+1} is equal to the jth column of H_1, for $j \notin \cup_{k=1}^{i} b(k)$.

Proof. See [3, Theorem 6.3]. □

Now, consider the following definition.

Definition 3. $A \in \mathbb{Z}^{n \times n}$ is a unimodular matrix if and only if $|det(A)| = 1$.

Note that, A is unimodular if and only if A^{-1} is unimodular.

Remark 1. Let $A \in \mathbb{Z}^{n \times n}$. If $A(\alpha_k, \beta_k)$, for $k = 1, \ldots, s$, are unimodular, then Algorithm 2 produces an integer matrix factorizatin (B and C are integer matrices).

Algorithm 2 produces a matrix factorization $AP = C$. Different choices of the parameters Q, F, t and b lead to different matrix factorizations. For $\alpha_i = \beta_i = i$, $Q_i = F_i = e_i$, $1 \leq i \leq n$, Algorithm 2 produces an LU factorization. Next, we show how to choose the parameters in Algorithm 2 to compute the WZ and ZW factorizations. We also discuss the octant interlocking factorization method and present two new factorizations named as the OS and SO factorizations.

3 Quadrant interlocking factorization

A direct method, called the WZ factorization, for solving linear systems of equations $Ax = b$ was introduced by Evans and Hotzopoulos [9]. Let A be an $n \times n$ nonsingular matrix. The WZ factorization of [8] expresses A as $A = WZ$, where W and Z have the following forms:

$$W = \begin{pmatrix} \bullet & \circ & \circ & \circ & \bullet \\ \bullet & \bullet & \circ & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \circ & \bullet & \bullet \\ \bullet & \circ & \circ & \circ & \bullet \end{pmatrix}, \; Z = \begin{pmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \circ & \bullet & \bullet & \bullet & \circ \\ \circ & \circ & \bullet & \circ & \circ \\ \circ & \bullet & \bullet & \bullet & \circ \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{pmatrix}, \; X = \begin{pmatrix} \bullet & \circ & \circ & \circ & \bullet \\ \circ & \bullet & \circ & \bullet & \circ \\ \circ & \circ & \bullet & \circ & \circ \\ \circ & \bullet & \circ & \bullet & \circ \\ \bullet & \circ & \circ & \circ & \bullet \end{pmatrix}, \tag{10}$$

where the empty bullets stand for zero and the other bullets stand for possible nonzeros.

The matrix W is called a unit W-matrix if in addition, $w_{ii} = 1$, for $i = 1, \ldots, n$, and $w_{i,n-i+1} = 0$, for $i \neq (n+1)/2$, when n is odd. The transpose of a (unit) *W-matrix* is called a (unit) *Z-matrix* and vise versa. Moreover, A matrix which is both a Z- and a *W-matrix* is called an *X-matrix*.

Note that, we assume that A is nonsingular and of an even size n (without loss of generality) and that $s = \frac{n}{2}$.

Theorem 4. Let A is an $n \times n$ matrix and let n be even. Then A has a WZ factorization if and only if the nested submatrices $A(1:k, n-k+1:n, 1:k, n-k+1:n)$ are invertible, for $k = 1, \ldots, n/2$.

Proof. See proof of of [15, Theorem 2]. □

Now, we show how to choose the parameters of the generalized ABS algorithm to compute the WZ factorization.
Consider two index sets $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$ such that $\alpha_k = \beta_k = \{k, n-k+1\}$ and $F_k = Q_k = [e_k, e_{n-k+1}]$, for $k = 1, \ldots, s$. Then, P is a Z-matrix, C is a W-matrix, and Algorithm 2 leads to a WZ factorization of A.

Remark 2. Let $A \in \mathbb{Z}^{n \times n}$. If the nested submatrices $A(1:k, n-k+1:n, 1:k, n-k+1:n)$ are unimodular, for $k = 1, \ldots, \frac{n}{2}$, then A has an integer WZ factorization.

Consider two equal index sets $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$ such that $\alpha_k = \beta_k = \{s-k+1, s+k\}$ and $F_k = Q_k = [e_{s-k+1}, e_{s+k}]$, for $k = 1, \ldots, s$. Then, P is a W-matrix, C is a Z-matrix, and Algorithm 2 leads to a WZ factorization of A.

Theorem 5. Let $A \in \mathbb{R}^{n \times n}$. Then A has a ZW factorization if and only if the nested submatrices $A(s-k+1:s+k, s-k+1:s+k)$ are invertible, for $k = 1, \ldots, s$.

Remark 3. Let $A \in \mathbb{Z}^{n\times n}$. If the nested submatrices $A(s-k+1:s+k, s-k+1:s+k)$ are unimodular, for $k=1,\ldots,s$, then A has an integer ZW factorization.

4 Octant interlocking factorization

Here, we first define octant matrices and then present the Octant Interlocking Factorization (OIF). We provide the necessary and sufficient conditions for the existence of OIF and show how to choose the parameters of the generalized ABS algorithm to compute the factorization.

Definition 4. We say $A \in \mathbb{R}^{n\times n}$ is an octant matrix, if it has one of the following structures:

$$S = \begin{pmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ \bullet & \bullet & \circ & \bullet & \bullet \\ \bullet & \circ & \circ & \circ & \bullet \\ \bullet & \bullet & \circ & \bullet & \bullet \\ \bullet & \bullet & \bullet & \bullet & \bullet \end{pmatrix}, O = \begin{pmatrix} \circ & \circ & \bullet & \circ & \circ \\ \circ & \bullet & \bullet & \bullet & \circ \\ \bullet & \bullet & \bullet & \bullet & \bullet \\ \circ & \bullet & \bullet & \bullet & \circ \\ \circ & \circ & \bullet & \circ & \circ \end{pmatrix}, \tag{11}$$

with the empty bullets standing for zero and the other bullets standing for possible nonzeros. The matrices in (11) are called the S-matrix and the O-matrix, respectively.

Definition 5. Let $A \in \mathbb{R}^{n\times n}$. We say that A has an octant interlocking factorization, if $A = BC$ such that B and C are octant matrices.

The transpose and the inverse of an S-matrix are an S-matrix and an O-matrix, respectively, as well as the transpose and the inverse of an O-matrix are an O-matrix and an S-matrix, respectively.

Note that, we assume that $A \in \mathbb{R}^{n\times n}$ is nonsingular, that n is an even number, and that $s = \frac{n}{2}$.
Consider two index sets $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$ such that $\alpha_k = \{k, n-k+1\}, \beta_k = \{s-k+1, s+k\}$, and $Q_k = F_k = [e_k, e_{n-k+1}]$. Then, C is an O-matrix and P is an S-matrix.

Now, let $\beta_k = \{k, n-k+1\}$, let $\alpha_k = \{s-k+1, s+k\}$, and let $Q_k = F_k = [e_{s-k+1}, e_{s+k}]$. Then, P is an O-matrix and C is an S-matrix.

Theorem 6. Let $A \in \mathbb{R}^{n\times n}$. Then A has an OS factorization if and only if the nested submatrices $A(s-k+1:s+k, 1:k, n-k+1:n)$ are invertible, for $k=1,\ldots,s$.

Remark 4. Let $A \in \mathbb{Z}^{n\times n}$. If the nested submatrices $A(s-k+1:s+k, 1:k, n-k+1:n)$, for $k=1,\ldots,s$, are unimodular, then A has an integer OS factorization.

5 Generalized ABS algorithm and matrix factorizations

An elimination method is a sequence of elementary row or column operations to divide a matrix into parts with partial zeroing of columns or rows of the matrix leading to a matrix factorization implicitly.

Let $A \in \mathbb{R}^{n\times n}$ and let $\alpha = \{\alpha_1, \ldots, \alpha_s\}$ and $\beta = \{\beta_1, \ldots, \beta_s\}$ be two index sets of n. Algorithm 2 computes a matrix factorization $AP = C$, with parameter choices $Q_i = F_i = I(:, \alpha_i)$ and $P(:, \beta_i) = H_i^T F_i$ as follows:

$$A(\alpha_i, :)H_k^T = 0, \quad i = 1, \ldots, k-1,$$

and

$$C(\alpha_i, \beta_k) = A(\alpha_i, :)P(:, \beta_k) = A(\alpha_i, :)H_k^T I(:, \alpha_i) = 0, \qquad i = 1, \ldots, k-1.$$

This means that in the kth step the generalized ABS algorithm performs a partial zeroing such that the elements of the submatrix C corresponding to the columns specified by β_k and the rows specified by α_i, for $i = 1, \ldots, k-1$, turn to zero. Here, we set the parameters in Algorithm 2 to present the associated matrix factorizations $C = AP$.

We consider different choices for α and β such that all the blocks $A(\alpha_i, \beta_j)$ turn to be 1×1 and present the associated matrix factorizations. First, we define two index sets. Let $J = \{j_1, \ldots, j_n\}$ with the J_i as follows:

$$j_i = \begin{cases} \frac{i+1}{2} & \text{if } i \text{ is odd,} \\ n - \frac{i}{2} + 1 & \text{if } i \text{ is even.} \end{cases} \tag{12}$$

We define the index set $K = \{k_1, \ldots, k_n\}$ as follows. If n is an even number, then define

$$k_i = \begin{cases} \frac{n}{2} - \frac{i+1}{2} + 1 & \text{if } i \text{ is odd,} \\ \frac{n}{2} + \frac{i}{2} & \text{if } i \text{ is even,} \end{cases} \tag{13}$$

if n is an odd number, then define

$$k_i = \begin{cases} \frac{n+1}{2} - \frac{i}{2} & \text{if } i \text{ is even,} \\ \frac{n+1}{2} + \frac{i-1}{2} & \text{if } i \text{ is odd.} \end{cases} \tag{14}$$

If $\alpha_i = \beta_i = i$, then C is a lower triangular and P is an upper triangular matrix. For $\alpha_i = \beta_i = n - i + 1$, C is an upper triangular and P is a lower triangular matrix. The different cases are noted in Figure 1

In Figure 2, we give a MATLAB code for Algorithm 2, where $A \in \mathbb{R}^{n\times n}$, $t = \{t_1, \ldots, t_n\}$ and $b = \{b_1, \ldots, b_n\}$ are two index sets, and all the blocks $A(\alpha_i, \beta_i)$ are 1×1, for $i = 1, \ldots, n$.

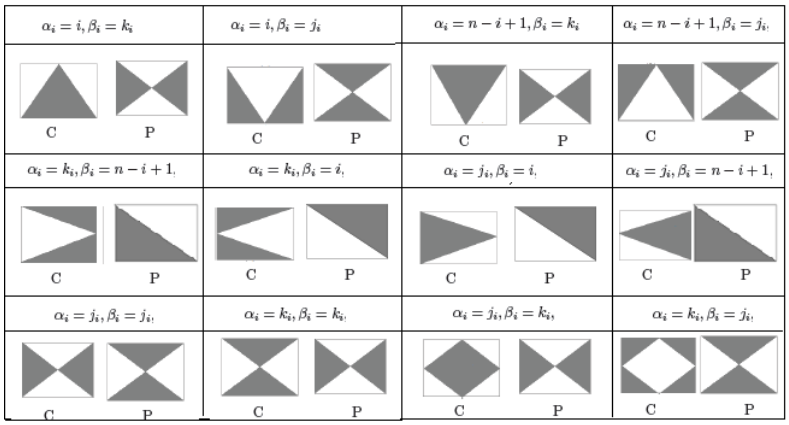


Figure 1: Matrix factorizations associated with different index sets

```
Function    [C,P]=GENABS(A,t,b)
clc
[n,n]=size(A);
E=eye(n);
P=zeros(n);
H=E;
for  i=1:n
       e=E(:,[t(i)]);
       q=E(:,[b(i)]);
       P(:,[b(i)])=H'*q;
       U=e'*A*H'*q;
       H=H-(H*A'*e*inv(U')*q'*H);
end;
C=A*P;
end
```

Figure 2: MATLAB code for Algorithm 2

6 Numerical illustrations

We give illustrations of our proposed algorithms to compute the QIF and OIF factorizations. The algorithms were implemented using $Matlab\ R2020a$.

Example 1. Consider the following matrix:

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 3 & 6 & 10 & 15 & 21 \\ 1 & 4 & 10 & 20 & 35 & 56 \\ 1 & 5 & 15 & 35 & 70 & 126 \\ 1 & 6 & 21 & 56 & 126 & 252 \end{pmatrix}.$$

Let $\alpha_k = \beta_k = \{s-k+1, s+k\}$, for $k = 1, 2, 3$ and $s = \frac{n}{2} = 3$. Then, the generalized elimination algorithm, Algorithm 1, produces P as a W-matrix and C as a Z-matrix, and we have a ZW factorization as follows:

$$P = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 & 0 & 3 \\ 2 & -1 & 1 & 0 & 2.5 & -8 \\ -1 & 0.3 & 0 & 1 & -3 & 9 \\ 0.2 & 0 & 0 & 0 & 1 & -4.8 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

and

$$C = \begin{pmatrix} 0.2 & 0.3 & 1 & 1 & 0.5 & 0.2 \\ 0 & 0.2 & 3 & 4 & 0.5 & 0 \\ 0 & 0 & 6 & 10 & 0 & 0 \\ 0 & 0 & 10 & 20 & 0 & 0 \\ 0 & 0.5 & 15 & 35 & 2.5 & 0 \\ 0.2 & 1.8 & 21 & 56 & 10.5 & 1.2 \end{pmatrix}.$$

Example 2. Let

$$A = \begin{pmatrix} 13.8966 & 15.3103 & 14.1873 & 2.3800 & 15.0253 & 10.9443 \\ 6.3420 & 15.9040 & 15.0937 & 9.9673 & 5.1019 & 2.7725 \\ 19.0044 & 3.7375 & 5.5205 & 19.1949 & 10.1191 & 2.9859 \\ 0.6889 & 9.7953 & 13.5941 & 6.8077 & 13.9815 & 5.1502 \\ 8.7749 & 8.9117 & 13.1020 & 11.7054 & 17.8181 & 16.8143 \\ 7.6312 & 12.9263 & 3.2522 & 4.4762 & 19.1858 & 5.0856 \end{pmatrix}.$$

Let $\alpha_k = \{s-k+1, s+k\}$ and $\beta_k = \{k, n-k+1\}$, for $k = 1, 2, 3$ and $s = 3$. By the generalized ABS algorithm, Algorithm 2, we have

$$P = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1.0000 & -3.3515 & -11.9955 & 0 & 0 \\ 1 & -0.7279 & 4.0998 & 13.5268 & -0.8931 & 0 \\ 0 & 0.0146 & -0.8436 & 1.3279 & -0.2703 & 1 \\ 0 & 0 & -1.2767 & -5.7630 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

and

$$C = \begin{pmatrix} 14.1873 & 5.0185 & -0.4413 & -64.2317 & 1.7108 & 2.3800 \\ 15.0937 & 5.0633 & 0 & 0 & -11.0731 & 9.9673 \\ 5.5205 & 0 & 0 & 0 & 0 & 19.1949 \\ 13.5941 & 0 & 0 & 0 & 0 & 6.8077 \\ 13.1020 & -0.4537 & 0 & 0 & 2.9522 & 11.7054 \\ 3.2522 & 10.6245 & -50.6286 & -210.6023 & 15.0712 & 4.4762 \end{pmatrix}.$$

We realize that P is an O-matrix, that C is an S-matrix, and that $AP = C$.

7 Concluding remarks

We presented a generalized elimination approach for solving linear systems. We established the necessary and sufficient conditions under which the proposed method is applicable. We showed that different matrix factorizations could be derived from the method such as the LU, WZ, and ZW factorizations. We also proposed the octant interlocking factorization to factorize a nonsingular matrix into octant matrices. We presented a generalized ABS algorithm and showed how to choose the parameters of the algorithm to compute the WZ and the ZW factorizations as well as the octant interlocking factorization of real and integer matrices.

Acknowledgements

The first author thanks the Research Council of Qom University and the second author thanks the Research Council of Sharif University of Technology for supporting this work.

References

1. Abaffy, J., Broyden, C.G. and Spedicato, E. *A class of direct methods for linear systems*, Numer. Math. 45 (1984) 361–376.
2. Abaffy, J. and Galantai, A. *Conjugate direction methods for linear and nonlinear systems of algebraic equations*, Colloquia Mathematica Societatis Janos Bolyai 50 (1986) 481–502.
3. Abaffy, J. and Spedicato, E. *ABS projection algorithms, mathematical techniques for linear and nonlinear equations*, Halsted Press, Chichester, 1989.
4. Adib, M., Mahdavi-Amiri, N. and Spedicato, E. *Broyden method as an ABS algorithm,* Publ. Univ. Miskolc Ser. D Nat. Sci. Math. 40 (1999), 3–13.

5. Chen, Y. and Zhou, B. *On g-inverses and nonsingularity of a bordered matrix* $\begin{pmatrix} A & B \\ C & O \end{pmatrix}$, Linear Algebra Appl. 133 (1990) 133–151.

6. Esmaeili, H., Mahdavi-Amiri, N. and Spedicato, E. *Generating the integer null space and conditions for determination of an integer basis using the ABS algorithms*, Bull. Iran. Math. Soc. 27(1) (2001) 1–18.

7. Esmaeili, H., Mahdavi-Amiri, N. and Spedicato, E. *A class of ABS algorithms for linear Diophantine systems*, Numer. Math. 90 (2001) 101–115.

8. Evans, D.J., Hadjidimos, A. and Noutsos, D. *The parallel solution of banded linear equation by the new quadrant interlocking factorization (Q.I.F.) method,* Internat. J. Comput. Math. 9(2) (1981) 151–161.

9. Evans, D.J. and Hatzopoulos, M. *A parallel linear system solver*, Int. J. Comput. Math. 7(3) (1979) 227–238.

10. Golpar-Raboky, E.and Mahdavi-Amiri, N. *Diophantine quadratic equation and Smith normal form using scaled extended integer ABS algorithms,* J. Optim. Theory Appl. 152(1) (2012) 75–96.

11. Golpar-Raboky, E. and Mahdavi-Amiri, N. *WZ factorization via Abaffy-Broyden-Spedicato algorithms*, Bull. Iran. Math. Soc. 40(2) (2014) 1–13.

12. Golpar-Raboky, E. and Mahdavi-Amiri, N. *A new interpretation of the integer and real WZ factorization using block scaled ABS algorithms*, Stat. Optim. Inf. Comput. 2 (2014) 243–256.

13. Khorramizadeh, M. and Mahdavi-Amiri, N. *Integer extended ABS algorithms and possible control of intermediate results for linear Diophantine systems*, 4OR 7 (2009) 145–167.

14. Mahdavi-Amiri, N. and Golpar-Raboky, E. *Real and integer Wedderburn rank reduction formulas for matrix decompositions*, Optim. Methods Softw. 30(4) (2015) 864–879.

15. Rao, S.C.S. *Existence and uniqueness of WZ factorization*, Parallel Comput. 23 (1997) 1129–1139.

16. Spedicato, E., Bodon, E. Del Popolo, A. and Mahdavi-Amiri, N. *ABS methods and ABSPACK for linear systems and optimization: A review*, 4OR 1 (2003) 51–66.

17. Spedicato, E., Bodon, E., Del Popolo, A. and Xia, Z. *ABS algorithms for linear systems and optimization: A review and a bibliography,* Ricerca Operativa 29 (2000) 39–88.

18. Spedicato, E., Bodon, E., Zunquan, X. and Mahdavi-Amiri, N. *ABS methods for continuous and integer linear equations and optimization*, CEJOR Cent. Eur. J. Oper. Res. 18 (2010) 73–95.

19. Spedicato, E., Xia, Z. and Zhang, L. *The implicit LX method of the ABS class,* Optim. Methods Softw. 8 (1997) 99–110.

How to cite this article
E. Golpar-Raboky and N. Mahdavi-Amiri A generalization of the ABS algorithms and its application to some special real and integer matrix factorizations. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 301-314. doi: 10.22067/ijnao.2021.70974.1043.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 315–333
DOI: DOI:10.22067/ijnao.2021.72492.1056
https://ijnao.um.ac.ir/

How to cite this article
Research Article

Hybrid of Block-pulse and orthonormal Bernstein functions for fractional differential equations

M. Entezari, S. Abbasbandy* and E. Babolian

Abstract

Differential equations of fractional order have been the focus of many studies due to their frequent appearance in various applications in fluid mechanics, biology, physics, and engineering. In general, it is not easy to derive the analytical solutions to most of these equations. Therefore, it is vital to develop some reliable and efficient techniques to solve fractional differential equations. A numerical method for solving fractional differential equations is proposed in this paper. The method is based on a hybrid of Block-pulse and orthonormal Bernstein functions. Convergence analysis is given, and numerical examples are introduced to illustrate the effectiveness and simplicity of the method.

AMS subject classifications (2020): 34A08; 34K28.

Keywords: Fractional Differential Equations; Hybrid Functions; Block-Pulse Functions; Bernstein Polynomials.

---

* Corresponding author

Received 13 September 2021; revised 5 December 2021; accepted 28 December 2021

Mahsa Entezari
Department of Mathematics, Science and Research Branch, Islamic Azad University, Tehran, Iran. email: mahsa.entezari8017@gmail.com

Saeid Abbasbandy
Department of Applied Mathematics, Faculty of Science, Imam Khomeini International University, Qazvin, Iran. email: abbasbandy@ikiu.ac.ir

Esmail Babolian
Department of Mathematics, Kharazmi University, Tehran, Iran. email: babolian@khu.ac.ir

1 Introduction

Fractional calculus is one of the most popular calculus types having a vast range of applications in many different scientific and engineering disciplines. The order of the derivatives in the fractional calculus might be any real number that separates the fractional calculus from the ordinary calculus where the order of derivatives are allowed to be only natural numbers. Fractional calculus is a highly efficient and useful tool in the modeling of many sorts of scientific phenomena including image processing, earthquake engineering, biomedical engineering, computational fluid mechanics, and physics. Fundamental concepts of fractional calculus and its applications to different research areas can be seen in the references [3, 4, 2, 7, 6, 5, 9, 12, 10, 16, 19, 20, 24, 29, 31, 32, 34], amongst many others. In recent years, many different basis functions have been used for solving fractional equations, such as operational matrix of Chebyshev polynomials, Block-pulse functions (BPFs), hybrid Bernoulli and Block-pulse functions, and hybrid Legendre [1, 8, 15, 23, 27, 26, 30, 33].

In this paper, we employ hybrid functions consisting of a combination of BPFs with orthonormal Bernstein polynomials (OBPs) to find the numerical solution of the fractional equation

$$F(x, y(x), D^{\alpha_1} y(x), \ldots, D^{\alpha_k} y(x)) = 0,\ x \in [0, X],\ \alpha_i \geq 0, \quad i = 1, \ldots, k, \tag{1}$$

with boundary or supplementary conditions

$$Z_i(y(\xi_i), y^{'}(\xi_i), \ldots, y^{(p)}(\xi_i)) = d_i, \quad i = 0, 1, \ldots, p, \tag{2}$$

where $0 \leq p < \max\{\alpha_i, i = 1, \ldots, k\} \leq p + 1,\ \xi_i \in [0, X],\ i = 0, \ldots, p$ and $Z_i, i = 0, \ldots, p$, are independent linear combinations of $y(\xi_i), y^{'}(\xi_i), \ldots, y^{(p)}(\xi_i)$ and $y \in L^2[0, X]$. It should be noted that F can be nonlinear in general.

The fractional equations are used in a variety of fields including continuum mechanics, potential theory, geophysics, electricity and magnetism, antenna synthesis, communication theory, mathematical economics, population genetics, the particle transport problem in astrophysics, and reactor theory. For the most part, it can be challenging to deduce the analytical solutions to most of these equations. As a result, it is crucial to develop some reliable and efficient ways to solve fractional differential equations. In this paper, we introduce a new method based on a hybrid of Block-pulse and orthonormal Bernstein functions. The present study's primary goal is to present and analyze a new stable algorithm for the numerical solution of the fractional differential equation based upon the hybrid of Block-pulse and orthonormal Bernstein functions (HBB method). To solve this problem, we propose a mathematical formulation that takes advantage of the orthogonality of BPFs and the OBPs properties to reduce the fractional differential equation to an algebraic system. Convergence analysis of the method is

discussed theoretically and numerically. Moreover, the applicability of the method is examined by solving some fractional differential equations and the Basset equation, which describes the unsteady motion of a sphere immersed in a Stokes fluid.

The outline of this paper is as follows: In section 2, we briefly overview some basic definitions for fractional calculus. In section 3, hybrid functions are introduced; therefore we approximate functions by using hybrid functions. In section 4, we present useful properties of hybrid functions such as coefficient matrix and operational matrix of derivative to solve fractional differential equations. In section 7, we present estimates for the error of the best approximation of smooth functions by hybrid functions. In section 6, we apply the proposed method to find an approximate solution to fractional differential equations. Finally, numerical examples are presented to show the effectiveness of the proposed method in section 7.

2 Fractional calculus

We give some basic definitions and properties of the fractional calculus theory, which are used further in this paper.

Definition 1 (see [32])**.** The Riemann–Liouville fractional integral operator of order $\alpha \geq 0$ is defined as

$$J^{\alpha} f(x) = \frac{1}{\Gamma(\alpha)} \int_0^x (x-t)^{\alpha-1} f(t) dt = \frac{1}{\Gamma(\alpha)} x^{\alpha-1} * f(x), \quad \alpha > 0, x > 0, \tag{3}$$

$$J^0 f(x) = f(x). \tag{4}$$

It has the following property:

$$J^{\alpha} x^{\gamma} = \frac{\Gamma(\gamma+1)}{\Gamma(\gamma+1+\alpha)} x^{\gamma+\alpha}, \quad \gamma > -1.$$

Definition 2 (see [32])**.** The Caputo definition of fractional derivative operator is given by

$$D^{\alpha} f(x) = J^{m-\alpha} D^m f(x) = \frac{1}{\Gamma(m-\alpha)} \int_0^x (x-t)^{m-\alpha-1} f^{(m)}(t) dt,$$

where $m-1 < \alpha \leq m,\ m \in \mathbb{N},\ x > 0$. For the Caputo derivative, we have

$$D^{\alpha} C = 0 \qquad (C \textit{ is a constant}),$$

$$D^{\alpha} x^j = \begin{cases} 0 & \text{for } j \in \mathbb{N} \cup 0 \textit{ and } j < \lceil \alpha \rceil, \\ \frac{\Gamma(j+1)}{\Gamma(j+1-\alpha)} x^{j-\alpha} & \text{for } j \in \mathbb{N} \cup 0 \textit{ and } j \geq \lceil \alpha \rceil \textit{ or } j \in \mathbb{N}. \end{cases} \tag{5}$$

3 Hybrid of Block-pulse and orthonormal Bernstein functions

The orthonormal set of hybrids $h_{ji}(x)$, $j = 1, 2, \ldots, m$ and $i = 0, 1, \ldots, n$, where j is the order for BPFs, i is the order for OBPs, and x is the normalized time, is defined on the interval $[0, X)$ as

$$h_{ji}(x) = \begin{cases} \sqrt{m}\, b_{in}(mx - \gamma_j) & \text{if } \frac{X(j-1)}{m} \le x < \frac{Xj}{m}, \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

where $\gamma_j = X(1-j)$ and $b_{in}, i = 0, \ldots, n$ are OBPs defined on $[0, X]$. Then

$$H(x) = [h_{10}(x), \ldots, h_{1n}(x), \ldots, h_{j0}(x), \ldots, h_{jn}(x), \ldots, h_{m0}(x), \ldots, h_{mn}(x)]^T$$
$$= [H_1(x), \ldots, H_j(x), \ldots, H_m(x)]^T,$$

is the $m(n+1)$ hybrid function vector. Here, the Bernstein polynomials (BPs) of the nth degree are defined on the interval $[0, X]$ as

$$B_{in}(x) = \frac{1}{X^n} \binom{n}{i} x^i (X - x)^{n-i}, \quad i = 0, 1, 2, \ldots, n,$$

by using expansion of $(X - x)^{n-i}$, we have

$$B_{in}(x) = \sum_{j=i}^{n} (-1)^{j-i} \left(\frac{1}{X^j}\right) \binom{n}{i} \binom{n-i}{j-i} x^j, \quad i = 0, 1, 2, \ldots, n,$$

where

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}.$$

According to [25], BPs form a complete basis over the interval $[0, X]$.

The explicit representation of the OBPs, denoted by $b_{in}(x)$ here, was recently discovered by analyzing the resulting orthonormal polynomials after applying the Gram–Schmidt process; see [13]

$$b_{in}(x) = (\sqrt{2(n-i)+1})(X-x)^{n-i} \sum_{k=0}^{i} (-1)^k \frac{1}{X^{n-k}} \binom{2n+1-k}{i-k} \binom{i}{k} x^{i-k}.$$

In addition, it can be written in terms of the non-orthonormal Bernstein basis functions as

$$b_{in}(x) = (\sqrt{2(n-i)+1}) \sum_{k=0}^{i} (-1)^k \frac{\binom{2n+1-k}{i-k}\binom{i}{k}}{\binom{n-k}{i-k}} B_{i-kn-k}(x),$$

for $i = 0, 1, \ldots, n$. By using the Taylor expansion, $b_{in}(x)$ can be represented as [21]

$$b_{in} = Z_i T_n(x), \quad i = 0, \ldots, n,$$

where $T_n(x) = [1, x, x^2, \ldots, x^n]^T$, Z_i is a row vector of Taylor coefficients as

$$(Z_i)_j = \sqrt{2(n-i)+1} \sum_{k=\max\{0, j-n+i\}}^{\min\{i,j\}} \rho_{ij-k}\beta_{ik}, \quad j = 0, \ldots, n,$$

where

$$\rho_{ir} = (-1)^r \binom{n-i}{r}, \quad r = 0, \ldots, n-i,$$

$$\beta_{ij} = (-1)^{i-j} \binom{2n+1-i+j}{j} \binom{i}{i-j}, \quad j = 0, \ldots, i.$$

Thus $b(x) = [b_{0n}(x), b_{1n}(x), \ldots, b_{nn}(x)]^T$ can be defined by

$$b(x) = ZT_n(x), \tag{7}$$

where Z is an $(n+1) \times (n+1)$ matrix whose ith row is $Z_i, i = 0, \ldots, n$.

Now, from (7) and (4), we have

$$H_j(x) = \sqrt{m} Z T_n(mx - \gamma_j), \tag{8}$$

where

$$T_n(mx - \gamma_j) = [1, mx - \gamma_j, (mx - \gamma_j)^2, \ldots, (mx - \gamma_j)^n]^T,$$

by using the binomial expansion of $(mx - \gamma_j)^k$, (8) can be expressed as

$$H_j(x) = Z\Lambda_j T_n(x),$$

where

$$\Lambda_j = \sqrt{m} \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ \gamma_j & m & 0 & 0 & \cdots & 0 & 0 \\ \binom{2}{0}(\gamma_j)^2 & \binom{2}{1}(\gamma_j)m & \binom{2}{2}m^2 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \cdots & \vdots & \vdots \\ \binom{n}{0}(\gamma_j)^n & \binom{n}{1}(\gamma_j)^{n-1}m & \binom{n}{2}(\gamma_j)^{n-2}m^2 & \binom{n}{3}(\gamma_j)^{n-3}m^3 & \cdots & \binom{n}{n-1}(\gamma_j)m^{n-1} & \binom{n}{n}m^n \end{pmatrix}.$$

Therefore the hybrid function vector $H(x)$ can be defined by

$$H(x) = \tilde{\mathbf{\Gamma}}\tilde{T}(x), \tag{9}$$

where $\tilde{T}(x) = [T_n(x), T_n(x), \ldots, T_n(x)]^T$ is an $m(n+1)$ vector, and $\tilde{\mathbf{\Gamma}} = diag[\Gamma_1, \Gamma_2, \ldots, \Gamma_m]$ is an $m(n+1) \times m(n+1)$ coefficient matrix of the $(n+1) \times (n+1)$ coefficient submatrix $\Gamma_j = Z\Lambda_j$.

3.1 Function approximation

Suppose that $\Omega = L^2[0, X]$ and

$$\{h_{10}, \ldots, h_{1n}, \ldots, h_{j0}, \ldots, h_{jn}, \ldots, h_{m0}, \ldots, h_{mn}\} \subset \Omega$$

are the set of hybrid functions and that

$$\Omega_{mn} = span\{h_{10}, \ldots, h_{1n}, \ldots, h_{j0}, \ldots, h_{jn}, \ldots, h_{m0}, \ldots, h_{mn}\}.$$

Let f_{mn} be the best approximation of an arbitrary function $f \in L^2(\Omega)$ out of Ω_{mn}, that is,

$$\|f - f_{mn}\|_2 \leq \| f - g \|_2 \qquad \text{for all } g \in \Omega_{mn}.$$

Moreover, since $f_{mn} \in \Omega_{mn}$, there exist unique coefficients $c_{10}, c_{11}, \ldots, c_{mn}$ such that

$$f(x) \simeq f_{mn}(x) = \sum_{j=1}^{m}\sum_{i=0}^{n} c_{ji}h_{ji}(x) = \sum_{j=1}^{m} C_j H_j(x) = C^T H(x) = H^T(x)C,$$

where C and $H(x)$ are $m(n+1)$ vectors as

$$C = [c_{10}, \ldots, c_{1n}, \ldots, c_{j0}, \ldots, c_{jn}, \ldots, c_{m0}, \ldots, c_{mn}]^T = [C_1, \ldots, C_j, \ldots, C_m]^T,$$

$$H = [h_{10}, \ldots, h_{1n}, \ldots, h_{j0}, \ldots, h_{jn}, \ldots, h_{m0}, \ldots, h_{mn}]^T \quad = [H_1, \ldots, H_j, \ldots, H_m]^T,$$

and the hybrid function coefficients c_{ji} are obtained by

$$c_{ji} = \frac{\langle f, h_{ji}\rangle}{\|h_{ji}\|_2^2}.$$

Since the set of hybrid functions is a complete orthonormal system in Ω, then

$$\|h_{ji}\|_2 = \int_0^X h_{ji}(x)h_{ji}(x)dx = 1.$$

Therefore

$$c_{ji} = \langle f, h_{ji} \rangle = \int_0^X f(x) h_{ji}(x) dx. \tag{10}$$

4 Operational matrix of differentiation

We construct the operational matrices of differentiation for $H(x)$. First, we consider integer order derivatives of $H(x)$. Hence

$$\frac{dH(x)}{dx} = \mathbf{D}^{(1)} H(x),$$

where $\mathbf{D}^{(1)}$ is the $m(n+1) \times m(n+1)$ operational matrix of first derivative of Hybrid function. From (9), we have

$$\frac{dH(x)}{dx} = \tilde{\mathbf{\Gamma}} \frac{d\tilde{T}(x)}{dx} = \tilde{\mathbf{\Gamma}} \tilde{\mathbf{Q}} \tilde{T}(x) = \tilde{\mathbf{\Gamma}} \tilde{\mathbf{Q}} \tilde{\mathbf{\Gamma}}^{-1} H(x),$$

where $\tilde{\mathbf{Q}} = diag[Q, Q, \ldots, Q]$ is an $m(n+1) \times m(n+1)$ matrix that consists the m submatrix $Q_{(n+1)\times(n+1)}$ as

$$Q = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 2 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & n & 0 \end{pmatrix}_{(n+1)\times(n+1)}.$$

Therefore we have

$$\mathbf{D}^{(1)} = \tilde{\mathbf{\Gamma}} \tilde{\mathbf{Q}} \tilde{\mathbf{\Gamma}}^{-1}. \tag{11}$$

In general, we obtain

$$\frac{d^n H(x)}{dx^n} = (\mathbf{D}^{(1)})^n H(x),$$

where $n \in \mathbb{N}$ and the superscript, in $\mathbf{D}^{(1)}$, denotes the matrix powers. Thus

$$\mathbf{D^{(n)}} = (\mathbf{D}^{(1)})^n, \qquad n = 1, 2, \ldots.$$

For the Caputo fractional derivative introduced in section 2, we denote fractional order derivative of $H(x)$ by

$$\frac{d^\alpha H(x)}{dx^\alpha} = \mathbf{D}^{(\alpha)} H(x),$$

where $\mathbf{D}^{(\alpha)}$ is an $m(n+1) \times m(n+1)$ operational matrix of fractional order derivative of Hybrid function. From (9), we have

$$\frac{d^\alpha H(x)}{dx^\alpha} = \tilde{\mathbf{\Gamma}} \frac{d^\alpha \tilde{T}(x)}{dx^\alpha},$$

where

$$\frac{d^\alpha \tilde{T}(x)}{dx^\alpha} = [\frac{d^\alpha T(x)}{dx^\alpha}, \frac{d^\alpha T(x)}{dx^\alpha}, \ldots, \frac{d^\alpha T(x)}{dx^\alpha}]^T,$$

and by using (3), we can specify the elements of $\frac{d^\alpha T(x)}{dx^\alpha}$ as

$$\frac{d^\alpha x^k}{dx^\alpha} = \begin{cases} 0, & 0 \le k < \lceil \alpha \rceil, \\ \frac{\Gamma(k+1)}{\Gamma(k+1-\alpha)} x^{k-\alpha}, & \lceil \alpha \rceil \le k < n. \end{cases}$$

We can also approximate $x^{k-\alpha}$ for $k = \lceil \alpha \rceil, \ldots, n$ by hybrid functions as

$$x^{k-\alpha} \simeq \sum_{j=1}^{m} \sum_{i=0}^{n} p_{kji} h_{ji}(x) = P_k^T H(x),$$

where P_k is an $m(n+1)$ vector that we can obtain by (10) and (9) as

$$P_{kji} = \int_0^X x^{k-\alpha} h_{ji}(x) dx = \tilde{\Gamma}_{ji} \int_{\frac{X(j-1)}{m}}^{\frac{X_j}{m}} x^{k-\alpha+i} dx.$$

Therefore

$$\mathbf{D}^{(\alpha)} = \tilde{\mathbf{\Gamma}} \tilde{\mathbf{\Psi}} \tilde{\mathbf{P}}, \tag{12}$$

where $\tilde{\mathbf{\Psi}} = diag[\Psi, \Psi, \ldots, \Psi]$ is an $m(n+1) \times m(n+1)$ matrix with m submatrix $\Psi_{(n+1)\times(n+1)}$ as

$$\Psi = diag[0, \ldots, 0, \frac{\Gamma(\lceil \alpha \rceil + 1)}{\Gamma(\lceil \alpha \rceil + 1 - \alpha)}, \ldots, \frac{\Gamma(n+1)}{\Gamma(n+1-\alpha)}],$$

and $\tilde{\mathbf{P}} = [P, P, \ldots, P]^T$ is the $m(n+1) \times m(n+1)$ matrix with the $(n+1) \times m(n+1)$ submatrix P, defined as

$$P = \begin{pmatrix} p_{\lceil \alpha \rceil 10} & p_{\lceil \alpha \rceil 11} & p_{\lceil \alpha \rceil 12} & \cdots & p_{\lceil \alpha \rceil m(n-1)} & p_{\lceil \alpha \rceil mn} \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots \\ p_{n10} & p_{n11} & p_{n12} & \cdots & p_{nm(n-1)} & p_{nmn} \end{pmatrix}.$$

5 Convergence analysis

The purpose of this section is to obtain an estimate of the error norm of the best approximation of a smooth function of two variables, on a certain domain $[0, X]$, by a bivariate polynomial. This estimate will be used for comparisons in section 6, when analyzing the error of the numerical results.

We assume that $f = \sum_{j=1}^{m} f_j$ is a sufficiently smooth function on $[0, X]$ and that $p \simeq \sum_{j=1}^{m} p_j$ is the interpolating polynomial to f at points $x_i, i = 0, 1, \ldots, n$, which are the roots of the $(n+1)$-degree shifted Chebyshev polynomial on $[0, X]$. Then we have [17]

$$f(x) - P(x) = \sum_{j=1}^{m}(f_j - P_j) = \sum_{j=1}^{m}(\frac{f_j^{(n+1)}(\xi)}{(n+1)!}\prod_{i=0}^{n}(x - x_i)),$$

where $\xi \in I_j = [\frac{(j-1)X}{m}, \frac{jX}{m}]$. Therefore

$$\mid f(x) - P(x) \mid \leq \sum_{j=1}^{m} \max_{x \in I_j} \mid f_j^{(n+1)}(x) \mid \frac{\prod_{i=0}^{n}(x - x_i)}{(n+1)!}. \tag{13}$$

We assume that there is a real number γ such that

$$\max_{x \in I_j} \mid f_j^{(n+1)}(x) \mid \leq \gamma. \tag{14}$$

By replacing (14) in (13) and taking into account the estimates for Chebyshev interpolation nodes [28], we obtain

$$\mid f(x) - P(x) \mid \leq \gamma \frac{X^{n+1}}{2^{2n+1} m^n (n+1)!}. \tag{15}$$

With the help of (15) we obtain the following result.

Theorem 1. Let $f_{mn}(x) = C^T H(x)$ be the hybrid functions expansion of the sufficiently smooth real function f in Ω, where

$$C = [c_{10}(x), \ldots, c_{1n}(x), \ldots, c_{j0}(x), \ldots, c_{jn}(x), \ldots, c_{m0}(x), \ldots, c_{mn}(x)]^T,$$

and

$$c_{ji} = \int_0^X f(x) h_{ji}(x) dx.$$

Then, there exists a real number γ' such that

$$\| f - f_{mn} \|_2 \leq \gamma' \frac{X^{n+1}}{2^{2n+1} m^n (n+1)!}. \tag{16}$$

Proof. Let Ω_{mn} be the space of bivariate polynomials of degree $\leq n$. As shown in section 3, f_{mn} is the best approximation of f in Ω_{mn}, that is,

$$\| f - f_{mn} \|_2 \leq \| f - g \|_2,$$

where g is any arbitrary polynomial in Ω_{mn}. In particular, we have

$$\|f-f_{mn}\|_2^2 dx \leq \int_0^X \mid f(x)-p_{mn}(x) \mid^2 dx, \tag{17}$$

where p_{mn} is the interpolating polynomial of f. From (15) and (17), we obtain

$$\|f-f_{mn}\|_2^2 \leq \int_0^X [\gamma \frac{X^{n+1}}{2^{2n+1}m^n(n+1)!}]^2 dx = X[\gamma \frac{X^{n+1}}{2^{2n+1}m^n(n+1)!}]^2. \tag{18}$$

From (18), we conclude that (16) is valid with

$$\gamma' = \gamma\sqrt{X}.$$

□

Theorem 2. Suppose $f \in L^2[0,1], n-1 < \alpha \leq n$. Then

$$\| D^\alpha f - D^\alpha f_{mn} \| \leq \frac{1}{\Gamma(n-\alpha+1)} \gamma' \frac{X^{n+1}}{2^{2n+1}m^n(n+1)!}. \tag{19}$$

Proof. By using (18) and [11], we have

$$\begin{aligned} \| D^\alpha f - D^\alpha f_{mn} \|^2 &= \| I^{n-\alpha}(D^n f - D^n f_{mn}) \|^2 \\ &= \| \frac{1}{x^{1+\alpha-n}\Gamma(n-\alpha)}(D^n f - D^n f_{mn}) \|^2 \\ &\leq (\frac{1}{(n-\alpha)\Gamma(n-\alpha)})^2 \| D^n f - D^n f_{mn} \|^2 \\ &\leq (\frac{1}{\Gamma(n-\alpha+1)})^2 \| f - f_{mn} \|^2 . \end{aligned}$$

From (18), we get (19). □

6 Solving fractional differential equations

In this section, in order to show the high importance of the proposed method, we apply it to solve fractional differential equation (1) and (2). In order to use hybrid functions for this problem, we approximate by hybrid functions as

$$y(x) \simeq \sum_{j=1}^{m}\sum_{i=0}^{n} c_{ji}h_{ji}(x) = C^T H(x), \tag{20}$$

where $C = [c_{10}, \ldots, c_{1n}, \ldots, c_{m0}, \ldots, c_{mn}]^T$ is an unknown vector. Using (11) and (12), we have

$$D^{\alpha_j} y(x) \simeq C^T D^{\alpha_j} H(x) \simeq C^T \mathbf{D}^{(\alpha_j)} H(x), \ j = 1, \ldots, k.$$

By substituting these equations in (1) and (2), we get

$$F(x, C^T H(x), C^T \mathbf{D}^{(\alpha_1)} H(x), \ldots, C^T \mathbf{D}^{(\alpha_k)} H(x)) = 0, \tag{21}$$

$$Z_i(C^T H(\xi_i), C^T \mathbf{D}^{(1)} H(\xi_i), \ldots, C^T \mathbf{D}^{(p)} H(\xi_i)) = d_i, i = 0, 1, \ldots, p. \tag{22}$$

To find the solution y, we first collocate (21) at $m(n+1) - (p+1)$ points. For suitable collocation points, we use

$$x_i = \frac{2i-1}{2m(n+1)}, i = 1, \ldots, m(n+1) - (p+1).$$

These equations together with (22) generate $m(n+1)$ algebraic equations, which can be solved to find $c_{ji}, j = 1, \ldots, m, i = 0, \ldots, n$. Consequently, the approximate solution of the unknown function y, given in (20), can be calculated.

7 Numerical examples

In this section, numerical results of some examples are presented. The absolute errors of this method are compared with those of the existing methods reported in [35, 14, 22]. The computations associated with these examples were performed using Mathematica 9.0.

Example 1. Consider the fractional equation

$$\sqrt{x} D^{\frac{1}{2}} y(x) + e^x y(x) = x(\frac{2}{\sqrt{\pi}} + e^x), \quad 0 < x \leq 1, \tag{23}$$

with the following initial conditions $y(0) = 0$. The exact solution of this problem is $y(x) = x$. The numerical method developed in section 6 is applied to (23) for $m = n = 2$, we approximate solution as

$$y(x) \simeq c_{10}h_{10} + c_{11}h_{11} + c_{12}h_{12} + c_{20}h_{20} + c_{21}h_{21} + c_{22}h_{22} = C^T H(x).$$

Here, we have

$$\tilde{\mathbf{\Gamma}} = \begin{pmatrix} 3.16228 & -12.6491 & 12.6491 & 0 & 0 & 0 \\ -2.44949 & 29.3939 & -48.9898 & 0 & 0 & 0 \\ 1.41421 & -22.6274 & 56.5685 & 0 & 0 & 0 \\ 0 & 0 & 0 & 12.6491 & -25.2982 & 12.6491 \\ 0 & 0 & 0 & -29.3939 & 78.3837 & -48.9898 \\ 0 & 0 & 0 & 26.8701 & -79.196 & 56.5685 \end{pmatrix},$$

$$\tilde{\mathbf{\Psi}} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1.12838 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.50451 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.12838 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.50451 \end{pmatrix},$$

$$\tilde{\mathbf{P}} = \begin{pmatrix} 2.38514 & 0. & 0.666667 & 0.672262 & 0.430222 & 0.236893 \\ 0.170367 & 0.263932 & 0.161905 & 0.415476 & 0.383286 & 0.235524 \\ 0.0283945 & 0.0879772 & 0.084127 & 0.262718 & 0.331263 & 0.235838 \\ 2.38514 & 0. & 0.666667 & 0.672262 & 0.430222 & 0.236893 \\ 0.170367 & 0.263932 & 0.161905 & 0.415476 & 0.383286 & 0.235524 \\ 0.0283945 & 0.0879772 & 0.084127 & 0.262718 & 0.331263 & 0.235838 \end{pmatrix},$$

$$\mathbf{D}^{\frac{1}{2}} = \begin{pmatrix} -1.89128 & -2.09283 & -0.709874 & -0.930387 & 0.8335 & 1.12652 \\ 3.55781 & 2.26954 & -0.830648 & -5.58347 & -11.7032 & -9.57079 \\ -1.93327 & 0.748753 & 3.02605 & 11.7513 & 18.4068 & 14.0581 \\ -4.32293 & -5.85992 & -3.02074 & -6.86047 & -4.63714 & -2.23512 \\ 12.9755 & 16.8594 & 8.1193 & 17.3836 & 9.48449 & 3.44875 \\ -12.8079 & -16.0982 & -7.30845 & -14.7689 & -6.05866 & -0.975538 \end{pmatrix}.$$

Finally, the obtained results are reported in Table 1. It shows that our method has better accuracy with the piecewise constant orthogonal function developed in [35].

Example 2. Consider the following fractional differential equation:

$$D^{\frac{1}{3}}y(x) + x^{\frac{1}{3}}y(x) = \frac{3}{2\Gamma(2.3)}x^{\frac{2}{3}} + x^{\frac{4}{3}}, \quad 0 < x \leq 1,$$

with the following initial conditions

$$y(0) = 0.$$

We know that the exact solution is $y(x) = x$. The maximum errors for different values of x are listed in Table 1. It is shown that, we obtain the error of order 10^{-16} for $m = 2, n = 2$. In the Haar wavelets method [14], the error is 0.0014 obtained for $m = 64$, where m is the order of Haar wavelets. Clearly, the results obtained by our method are better than those in [14] in terms of accuracy if the exact solution is sufficiently smooth. Therefore, the HBB method is a valid method in solving fractional differential equations.

Table 1: Numerical results for Examples 1 and 2 for $m = 2$ and $n = 2$.

x	0.1	0.3	0.5	0.7	0.9
Example 1	3.46945×10^{-17}	5.55112×10^{-17}	3.21965×10^{-15}	2.33147×10^{-15}	1.77636×10^{-15}
Example 2	1.38778×10^{-17}	5.55112×10^{-17}	1.5099×10^{-14}	1.55431×10^{-15}	1.27676×10^{-15}

Example 3. Consider the fractional equation

$$D^{\alpha}y(x) = -y(x) + x^2 + \frac{2x^{2-\alpha}}{\Gamma(3-\alpha)}, \quad 0 < x \leq 1, \tag{24}$$

with the following initial conditions $y(0) = 0$. The accurate solution, in this case, is given by $y(x) = x^2$. For $\alpha = 0.25, 0.75$ and $m = n = 2$, the approximation solution for (24) is obtained by using the HBB method. The absolute error of HBB method and fast wavelet collocation method (FWCM) reported in [22] is shown in Table 2 when $\alpha = 0.25$, $\alpha = 0.75$, and $m = n = 2$.

The logarithm of absolute error for different values of m and n is shown in Figures 1 and 2. Table 2 and Figures 1 and 2 confirm that the HBB method approximates the solution of fractional differential equation uniformly.

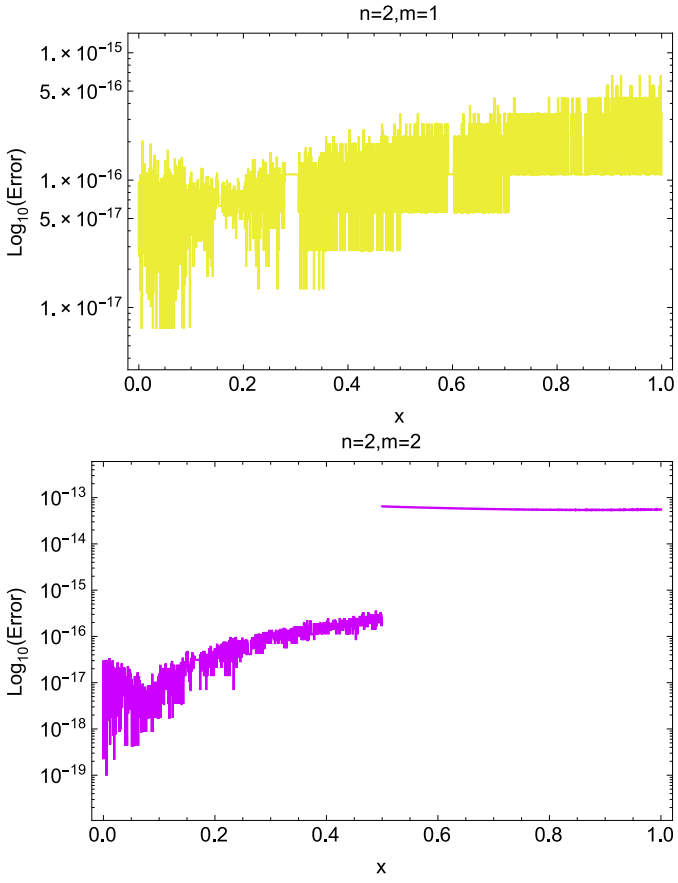


Figure 1: The curves of the logarithm of the absolute errors of Example 3 for $\alpha = 0.25$ and different value of m and n.

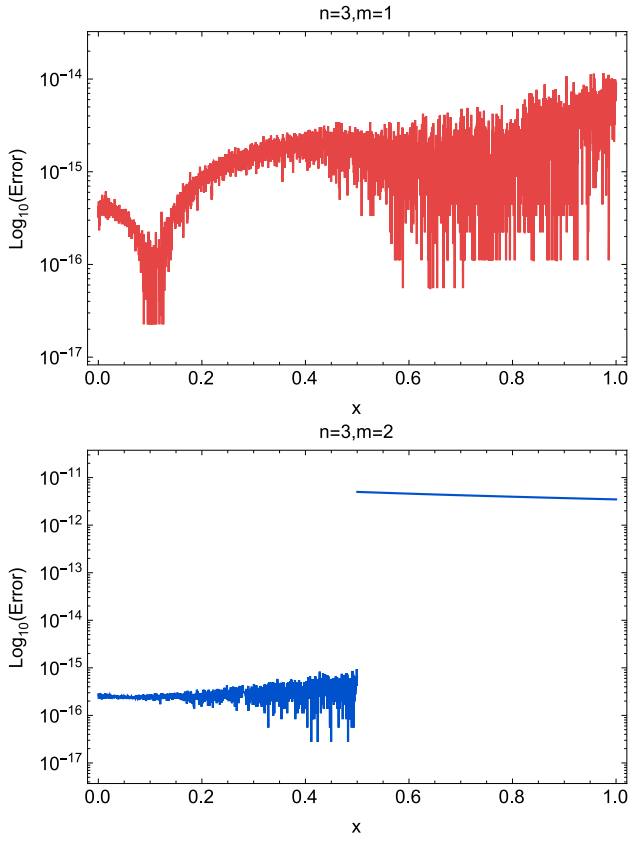


Figure 2: The curves of the logarithm of the absolute errors of Example 3 for $\alpha = 0.75$ and different value of m and n.

Table 2: The errors of Example 3 compared to [22]

x	$\alpha = 0.25$		$\alpha = 0.75$	
	Current Method	FWC method in [22]	Current Method	FWC method in [22]
0.03125	3.46945×10^{-18}	0.05958×10^{-12}	3.95517×10^{-16}	0.33012×10^{-13}
0.09375	2.77556×10^{-17}	0.04301×10^{-12}	4.02456×10^{-16}	0.34730×10^{-13}
0.18750	8.32667×10^{-17}	0.05390×10^{-12}	3.88578×10^{-16}	0.02296×10^{-13}
0.28125	1.38778×10^{-17}	0.02446×10^{-12}	3.88578×10^{-16}	0.19817×10^{-13}
0.37500	2.22045×10^{-16}	0.00394×10^{-12}	4.16334×10^{-16}	0.29393×10^{-13}
0.46875	6.15064×10^{-14}	0.06483×10^{-12}	5.55112×10^{-16}	0.22148×10^{-13}
0.56250	5.77316×10^{-14}	0.05401×10^{-12}	6.66134×10^{-15}	0.19428×10^{-13}
0.65625	5.54001×10^{-14}	0.00982×10^{-12}	8.88178×10^{-15}	0.17097×10^{-13}
0.75000	5.42899×10^{-14}	0.01798×10^{-12}	1.33227×10^{-15}	0.09103×10^{-13}
0.84375	5.45127×10^{-14}	0.03186×10^{-12}	1.77636×10^{-15}	0.11990×10^{-13}
0.93750	6.25614×10^{-14}	0.20128×10^{-12}	2.47315×10^{-15}	0.27533×10^{-13}
CPU time	$3.65s$		$2.57s$	

Example 4 (see [18]). Consider the following fractional Relaxation problem:

$$D^{\frac{1}{2}}y(x) = -y(x), \quad 0 < x \leq 1, \tag{25}$$

with initial condition $y(0) = 1$.

The exact solution of this equation is $y(x) = \mathbb{E}_\alpha(-x^\alpha)$. Here $\mathbb{E}_\alpha(x)$ is called the Mittag–Lefler function,

$$\mathbb{E}_\alpha(x) = \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(\alpha k + 1)}.$$

Table 3 shows a comparison of our method and the Taylor matrix method presented in [18]. Clearly, the results obtained by the HBB method are in agreement with other mentioned numerical method and in total this approach has high accuracy.

Table 3: Comparison of our method for $m = 3$ and $n = 8$, and the Taylor matrix method in [18] of Example 4

x	Current Method	Ref.[18]
0.0	3.4685×10^{-16}	0.0000
0.1	1.7745×10^{-10}	0.1000×10^{-9}
0.2	6.3421×10^{-10}	0.9000×10^{-9}
0.3	5.3461×10^{-9}	0.4000×10^{-9}
0.4	6.4326×10^{-10}	0.2000×10^{-9}
0.5	7.4902×10^{-9}	0.1000×10^{-8}
0.6	4.3721×10^{-8}	0.6000×10^{-8}
0.7	3.5684×10^{-7}	0.2400×10^{-7}
0.8	9.2341×10^{-7}	0.8400×10^{-7}
0.9	8.4563×10^{-7}	0.2480×10^{-6}
1.0	5.3475×10^{-7}	0.6740×10^{-6}

8 Conclusion

In this paper, we obtained operational matrices of the fractional derivatives of a combination of OBPs and BPFs. Then by using these matrices, we reduced the multi-order fractional differential equations to a system of algebraic equations that can be solved easily. The convergence analysis of the HBB bases is given in section 7. The numerical solutions obtained using the suggested method showed that numerical solutions are in very good coincidence with the exact solution. For future research, we will apply this numerical method for solving nonlinear fractional integro-differential equations, nonlinear Volterra–Fredholm–Hammerstein integral equations, and so on.

Acknowledgements

The authors would like to express deep gratitude to the editors and referees for their valuable suggestions which led us to a better presentation of this paper.

References

1. Ali, M.R. and Hadhoud, A.R. *Hybrid orthonormal Bernstein and Block-pulse functions wavelet scheme for solving the 2D Bratu problem*, Results in Physics, 12 (2019) 525–530.

2. Ali, M.R., Hadhoud, A.R. and Ma, W.X. *Evolutionary numerical approach for solving nonlinear singular periodic boundary value problems*, J. Intell. Fuzzy Syst. 39(5) (2020) 7723–7731.

3. Ali, M.R., Hadhoud, A.R. and Srivastava, H.M. *Solution of fractional Volterra–Fredholm integro–differential equations under mixed boundary conditions by using the HOBW method*, Adv. Differ. Equ. 2019, 115 (2019).

4. Ali, M.R. and Ma, W.X. *Detection of new multi-wave solutions in an unbounded domain*, Modern Physics Letters B, 33(34) (2019) 1950425.

5. Ali, M.R., Ma, W.X. and Sadat, R. *Lie symmetry analysis and invariant solutions for (2+1) dimensional Bogoyavlensky–Konopelchenko equation with variable–coefficient in wave propagation*, Journal of Ocean Engineering and Science (JOES), (2021).

6. Ali, M.R. and Sadat, R. *Construction of Lump and optical solitons solutions for (3+1) model for the propagation of nonlinear dispersive waves in inhomogeneous media*, Opt. Quantum Electron. 53 (5) (2021) 1–13.

7. Ali, M.R. and Sadat, R. *Lie symmetry analysis, new group invariant for the (3+1)-dimensional and variable coefficients for liquids with gas bubbles models*, Chin. J. Phys. 71 (2021) 539–547.

8. Alipour, M., Baleanu, D. and Babaei, F. *Hybrid Bernstein Block-pulse functions method for second kind integral equations with convergence analysis*, Abstract and Applied Analysis. Vol. 2014. Hindawi, 2014.

9. Ayub, A., Sabir, Z., Altamirano, G.C., Sadat, R. and Ali, M.R. *Characteristics of melting heat transport of blood with time-dependent cross–nanofluid model using Keller-Box and BVP4C method*, Eng. Comput. (2021) 1–15.

10. Baleanu, D., Sadat, R. and Ali, M.R. *The method of lines for solution of the carbon nanotubes engine oil nanofluid over an unsteady rotating disk*, Eur. Phys. J. Plus. 135(10) (2020) 1–13.

11. Bass, R.F. *Real analysis for graduate students*, Createspace Ind Pub, 2013.

12. Bayram, M. *Automatic analysis of the control of metabolicnetworks*, Comput. Biol. Med. 26 (1996) 401–408.

13. Bellucci, M.A. *On the explicit representation of orthonormal Bernstein Polynomials*, Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, U.S.A, 2014.

14. Chen, Y., Yi, M. and Yu, C. *Error analysis for numerical solution of fractional differential equation by Haar wavelets method*, J. Comput. Sci. 3 (2012) 367–373.

15. Doha, E.H., Bhrawy, A.H. and Ezz-Eldien, S.S. *A Chebyshev spectral method based on operational matrix for initial and boundary value problems of fractional order*, Comput. Math. Appl. 62 (2011) 2364–2373.

16. Feng, Y., Lin, X., Zhou, S. and Li, H. *Chaos in a fractional-order neutral differential system*, Appl. Math. Inf. Sci. (AMIS), 7 (2013) 233–238.

17. Gasca, M. and Sauer, T. *On the history of multivariate polynomial interpolation*, J. Comput. Appl. Math. 122 (2000) 23–35.

18. Gülsu, M., Öztürk, Y. and Anapalı, A. *Numerical approach for solving fractional relaxation-oscillation equation*, Appl. Math. Model. 37 (2013) 5927–5937.

19. Guzel, N., Emiroglu, I., Guler, C., Tasci F. and Sivri, M. *A solution proposal to the interval fractional transportation problem*, Appl. Math. Inf. Sci. (AMIS), 6 (2012) 567–571.

20. Hilfer, R. *Applications of fractional calculus in physics*, Academic Press, Orlando, 1999.

21. Javadi, Sh. and Taheri, Z. *Solving generalized pantograph equations by shifted orthonormal Bernstein polynomials*, Journal of Computational and Applied Mathematics, 2016.

22. Li, X. *Numerical solution of fractional differential equations using cubic-spline wavelet collocation method*, Commun. Nonlinear Sci. Numer. Simul. 17 (2012) 3934–3946.

23. Li, Y. and Sun, N. *Numerical solution of fractional differential equations using the generalized Block-pulse operational matrix*, Comput. Math. Appl. 62 (2011) 1046–1054.

24. Ma, W.X., Ali, M.R. and Sadat, R. *Analytical solutions for nonlinear dispersive physical model*, Complexity 2020 (2020).

25. Maleknejad, K., Hashemizadeh, E. and Basirat, B. *Computational method based on Bernstein operational matrices for nonlinear Volterra–Fredholm–Hammerstein integral equations*, Commun. Nonlinear Sci. Numer. Simul. 17 (2012) 52–61.

26. Marzban, H.R. and Malakoutikhah, F. *Solution of delay fractional optimal control problems using a hybrid of Block-pulse functions and orthonormal Taylor polynomials*, J. Franklin Inst. 356(15) (2019) 8182–8215.

27. Mashayekhi, S. and Razzaghi, M. *Numerical solution of nonlinear fractional integro–differential equations by hybrid functions*, Eng. Anal. Bound Elem. 56 (2015) 81–89.

28. Mason, J.C. and Handscomb, D.C. *Chebyshev polynomials*, CRC Press LLC, 2003.

29. Moghaddam, B.P. and Aghili, A. *A numerical method for solving linear non-homogenous fractional ordinary differential equation by using the operational matrices of the Bernstein polynomials*, Appl. Math. Inf. Sci. 6 (2012) 441–445.

30. Mollahasani, N., Moghadam, M.M. and Afrooz, K. *A new treatment based on hybrid functions to the solution of telegraph equations of fractional order*, Appl. Math. Model. 40 (2016), no. 4, 2804–2814.

31. Mousa, M.M., Ali, M.R. and Ma, W.X. *A combined method for simulating MHD convection in square cavities through localized heating by method of line and penalty–artificial compressibility*, J. Taibah Univ. Sci. 15(1) (2021) 208–217.

32. Podlubny, I. *Fractional differential equations: An introduction to fractional derivatives, fractional differential equations, some methods of their solution and some of their applications*. San Diego: Academic Press, 1999.

33. Ramadan, M. and Osheba, H.S. *A new hybrid orthonormal Bernstein and improved Block-pulse functions method for solving mathematical physics and engineering problems*, Alex. Eng. J. 59(5) (2020) 3643–3652.

34. Sabir, Z., Ali, M.R., Zahoor Raja, M.A., Shoaib, M., Sandoval Nunez, R.A. and Sadat, R. *Computational intelligence approach using Levenberg–Marquardt backpropagation neural networks to solve the fourth-order nonlinear system of Emden–Fowler model*, Eng. Comput. (2021) 1–17.

35. Salimi Shamloo, A. and Babolian, E. *Numerical solution of fractional differential, integral and integro–differential equations by using piecewise*

constant orthogonal functions, PAMM: Proceedings in Applied Mathematics and Mechanics. Vol. 7. No. 1. Berlin: WILEY-VCH Verlag, 2007.

How to cite this article
M. Entezari, S. Abbasbandy and E. Babolian Hybrid of Block-pulse and orthonormal Bernstein functions for fractional differential equations. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 315-333. doi: DOI:10.22067/ijnao.2021.72492.1056.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 334–354
DOI: 10.22067/ijnao.2021.72806.1060
https://ijnao.um.ac.ir/

Elzaki transform method for solving deterministic modeling of terrorism

G. Adamu* and M.O. Ibrahim

Abstract

A deterministic mathematical model of terrorism with government intervention was constructed from five compartments and subdivided into two core and non-core groups. A non-core group is a general group $G(t)$, while the core group is susceptible $S(t)$, moderate $I(t)$, terrorism $T(t)$, and recovered $R(t)$. The Elzaki transform method with differential transform to handle the nonlinear terms is employed to solve the model. The results show that government intervention on susceptible groups proved to be 90% effective in reducing terrorist threats since the group of susceptible in the population appears to be at risk of adopting the ideology through different means of contact. Also, due to the government intervention, the moderate group reduces gradually in time.

AMS subject classifications (2020): 93A30, 65C20, 39A39.

Keywords: Modeling; Terrorism; Elzaki transform; Differential transform and Intervention.

1 Introduction

Terrorism can be seen as extra-normal violence or a threat to intimidate innocent civilians through imposing their ideology, religiosity, and political objectives in a population. Global terrorism index 2020 defined terrorism as a threatened, or actual use of illegal force and violence by a non-state

* Corresponding author

Received 1 October 2021; revised 18 December 2021; accepted 29 December 2021

Gambo Adamu
Department of Mathematics, University of Ilorin, Nigeria.
e-mail: ahlulbaity002@gmail.com

Mohammed Olarenwaju Ibrahim
Department of Mathematics, University of Ilorin, Nigeria.
e-mail: moibraheem@unilorin.edu.ng

actor to attain political, economic, religious, or social goals through fear, oppression or intimidation. This definition deduces that terrorism is not only the physical act of an attack but also the mental blow it had on humanity for many years; see [18]. Government intervention strategies consist of law enforcement. When violent extremists are investigated, prosecuted, and imprisoned, rehabilitation involves psychological counseling and correcting of their extremist views through religious education followed by vocational training. The military strategy consists of when violent extremists are killed or captured. Counter-terrorism, however, agrees that these strategies cannot reduce the threat of terrorism solitarily in [26, 8].

Similarly, the authors of [24, 25] stated that governments intervention could use an additional set of strategies collectively known as countering violent extremism and classified it into three programs: 1) Prevention program which seeks to prevent the terrorist process from occurring, 2) disengagement program, which attempts to stop or control terrorist from occurring, and 3) terrorist recovery program, which attempts to change an individual extremist belief to re-integrate them into the society. These programs frequently aim to target convicted terrorist foot soldiers. Martcheva [22] discussed a mathematical model as a system using mathematical tools and language. Castillo-Chavez and Song [10] first developed transmission dynamics and spread of terrorism via mathematical modeling, in which they incorporated the spread of radical ideologies, fanaticism, recruitment, and terrorist activities. The models of terrorism were first developed in [10]. A mathematical model of the dynamic behavior of terrorism was developed in [1], and the authors controlled to check the recruitment pool. The model is developed to control the spread of terrorist ideologies in society. Furthermore, Santoprete [26] built a model of countering violent extremism (CVE) for prevention programs. It seeks to stop the radicalization process from occurring. According to [13], it was stated that Elzaki strictly connected with the Laplace transform has some advantages such as Elzaki over Laplace converges faster to the exact solution.

Moreover, Akinola, Akinpelu, and Oladejo [6] applied the Elzaki decomposition method for solving the epidemic model. The obtained result agreed with the Adomian decomposition method, homotopy perturbation method, variational iteration method, and differential transform method (DTM). Suleman et al. [31] used the Elzaki projected differential transform method to investigate the numerical solution of the fractional-order of the HIV model. According to [14], the Elzaki transform method (ETM) is a new integral transform, which is based on the Fourier series to handle only linear differential equations. Also, in [15, 23, 17, 16, 12, 9, 33], the Elzaki method transforms the fundamental properties and their applications to differential equations for the exact solution and analyzes boundary value problems via the Elzaki transform. The ETM was introduced to facilitate the process of solving ordinary and partial differential equations in the time domain. This

method is incapable of handling nonlinear equations, because of the difficulties that are caused by the nonlinear terms in [15, 23].

Furthermore, Kalavathi, Kohila, and Upadhyaya [21] presented the degenerate Elzaki transform to examine some of its properties and relations. Also, they investigated a scale preserving theorem for the degenerate Elzaki transform and theoretical dual transform to the degenerate Laplace transform. Aggarwal, Mishra, and Kumar [3] showed a comparative study of two new integral transforms Mohand and Elzaki transforms, respectively. Also, Anjum et al. [7] applied the ETM on the numerical solution for nonlinear oscillators. Bhadane and Ghadle [9] solved a system of linear differential equations using the ETM. Although, in [27], it was proposed a numerical solution for a third-order ordinary differential equation, using the DTM and ETM, then compared the results to see which method converges quickly to the exact solution. Some new applications of the ETM were defined in [29] for the solution to linear Volterra type integral equations. The results obtained were compared with solutions of the Adomian decomposition method, variational iteration method, method of successive approximation, Galerkin method, and Laplace transform method. The authors developed dualities in [2, 4] between the Elzaki transform and some useful integral transforms and then developed equations with decay problems. Their results showed that the Elzaki transform is in complete agreement with some useful methods. Therefore, in [31, 20, 35, 28, 32], it was worked a new technique of the ETM for solving differential equations to an exact solution. Their obtained results showed that Elzaki converges to the exact solution faster than any other method. The Ezaki transform is a more powerful tool than the Laplace transform, which can still serve as an auxiliary method to the Laplace transform.

This study aims to modify the existing models in [1, 26, 24] by incorporating terrorist recovered groups and government intervention. Then we apply the ETM to solve the model equations together with DTM, to linearize the nonlinear term in the equations for numerical/exact solutions and compare the result of the methods on a graph.

2 Materials and method

Mathematical modeling of terrorism with government intervention is built, and then we apply numerical methods called ETM and DTM, respectively, to solve the model equations.

2.1 Basic functions of ETM

A set of function $f(t)$ of exponential order is defined by A as follows:

$$A = \left\{ f(t) : \exists M, k_1, k_2 > 0, |f(t)| < M \exp(\frac{|t|}{k_j}), \text{ if } t \in (-1)^j [0, \infty) \right\}, \quad (1)$$

where M is a real number and k_1 and k_2 are integer. Then the Elzaki transform denoted by the operator E as the new integral equation is defined by

$$E\left[f(t)\right] = T(u) = u \int_0^\infty f(t) \exp\left(\frac{-t}{u}\right) dt, \qquad t \geq 0, k_1 \leq u \leq k_2, 0 \leq t \leq \infty. \quad (2)$$

Theorem 1. Let $T(u)$ be the Elzaki transform. Then $f(t)$ is a function taking the Elzaki $f(t)$ as

$E\left[f^{'}(t)\right] = \frac{T(u)}{u} - uf(0).$

$E\left[f^{''}(t)\right] = \frac{T(u)}{u^2} - f(0) - uf'(0).$

$E\left[f^{(n)}(t)\right] = \frac{T(u)}{u^n} - \sum_{k=0}^{n-1} u^{2-n+k} f_{(0)}^{(k)}$

Proof. Suppose that $E\left[f(t)\right] = u \int_0^\infty f'(t) \exp\left(-\frac{t}{u}\right) dt$. Using integration by part gives $E\left[f'(t)\right] = \frac{T(u)}{u} - uf(0)$ and $E\left[f''(t)\right] = \frac{T(u)}{u^2} - f(0) - uf'(0)$. The last expression of the theorem can be proved by mathematical induction. □

Therefore, Elzaki, Elzaki, and Elnour [17] stated the properties of Elzaki transform of a function below:

Let $T_1(u)$ and $T_2(u)$ be the Elzaki transform of $F_1(t)$ and $F_2(t)$, respectively. Then the Elzaki transform of $[aF_1(t) + bF_2(t)]$ is given by $[aT_1(u) + bT_2(u)]$, where a and b are arbitrary constants, and the inverse of Elzaki transform can be defined as $f(t) = E^{-1}(T(u))$.

The Elzaki transform of some functions is useful when applying on differential equations as follows:

(1) If $f(t) = 1$, then $E(1) = u \int_0^\infty \exp\left(-\frac{t}{u}\right) dt = u^2$.

(2) If $f(t) = t$, then $E(t) = u \int_0^\infty t \exp\left(-\frac{t}{u}\right) dt = u^3$.

(3) If $f(t) = t^2$, then $E(t^2) = u \int_0^\infty t^2 \exp\left(-\frac{t}{u}\right) dt = 2!u^4$.

(4) If $f(t) = t^n, n \in \mathbb{N}$, then $E(t^n) = u \int_0^\infty t^n \exp\left(-\frac{t}{u}\right) dt = n!u^{n+2}$.

(5) If $f(t) = \exp(at)$, then $E(\exp(at)) = u \int_0^\infty \exp(at) \exp\left(-\frac{t}{u}\right) dt = \frac{u^2}{1-au}$.

(6) If $f(t) = \cos(at)$, then $E(\cos at) = u \int_0^\infty \cos at \exp\left(-\frac{t}{u}\right) dt = \frac{u^2}{1-a^2u^2}$.

(7) If $f(t) = \sin(at)$, then $E(\sin at) = u \int_0^\infty \sin at \exp\left(-\frac{t}{u}\right) dt = \frac{2au^4}{1+a^2u^2}$.

(8) If $f(t) = t\cos(at)$, then $E(t\cos at) = u \int_0^\infty t\cos at \exp\left(-\frac{t}{u}\right) dt = \frac{u(1-a^2u^2)}{(1+a^2u^2)^2}$.

2.2 Advantages of Elzaki transform in comparison with Laplace transform

In this subsection, the advantages of ETM, in comparison with the Laplace transform method are as follows:

(a) The ETM has the capability of combining two powerful methods at the same time, compared to the Laplace transform method.

(b) The Ezaki transform method is a powerful tool compared to the Laplace transform, which can still serve as an auxiliary method to the Laplace transform method.

(c) The ETM converges to exact solutions faster than the Laplace transform method.

(d) The ETM is a new integral transform that does not require perturbation.

2.3 Differential transform method (DTM)

There are many methods for solving differential equations. One of them is the Taylor series. The new form of the Taylor series called DTM was proposed by [34], and they applied it to solve mathematical problems in an electrical circuit. The idea of DTM is to determine the root of the Taylor series of a function by solving the induced recursive equation. In [19], it was developed a mathematical model for the transmission dynamics of the Syphilis disease and employed the DTM to handle the nonlinear systems. The DTM is a semi-analytic technique that depends on the Taylor series. The Taylor polynomial of degree n is defined by

$$P_n(x) = \sum_{k=0}^{n} \frac{1}{k!}[f^{(k)}(c)(x-c)^k]. \tag{3}$$

Suppose that the function f has $(n+1)$ continuous derivative on the interval $(c-r, c+r)$, for all $r > o$, and that $\lim_{n\to\infty} R_n(x) = 0$, where $R_n(x)$ is the error between $P_n(x)$ and the function $f(x)$. Then the Taylor series expanded about $x = c$ converges to $f(x)$; that is,

$$f(x) = \sum_{k=0}^{\infty} \tfrac{1}{k!}[f^{(k)}(c)(x-c)^k], \qquad \text{for all} \quad x \in (c-r, c+r). \tag{4}$$

Differential transform of the function $f(x)$ for the kth derivative is given in [5, 19] as

$$F(k) = \tfrac{1}{k!}[\tfrac{d^k f(x)}{dx^k}]_{x=x_0}, \tag{5}$$

where $f(x)$ is the original function and $F(k)$ is the transformed function. Using the ideas of [5, 19], the inverse differential transform of $F(k)$ is defined as

$$f(x) = \sum_{k=0}^{\infty} (x-x_0)^k F(k). \tag{6}$$

Substituting (7) into (8) yields

$$f(x) = \sum_{k=0}^{n} (x-x_0)^k \tfrac{1}{k!} \left[\tfrac{d^k f(x)}{dx^k}\right]_{x=x_0}. \tag{7}$$

The basic idea of DTM is considered in Table 1.

Table 1: The fundamental operations of DTM

Original Function	**Transform Function**
$y(x) = u(x) \pm m(x)$	$Y(k) = U(k) \pm M(k)$
$y(x) = \alpha m(x)$	$Y(k) = \alpha M(k)$
$y(x) = \frac{du(x)}{dx}$	$Y(k) = (k+1)U(k+1)$
$y(x) = \frac{d^2u(x)}{dx^2}$	$Y(k) = (k+1)(k+2)U(k+2)$
$y(x) = \frac{d^m u(x)}{dx^n}$	$Y(k) = (k+1)(k+2)k(k+n)U(k+n)$
$y(x) = 1$	$Y(k) = \delta(k)$
$y(x) = x$	$Y(k) = \delta(k-l)$
$y(x) = x^m$	$Y(k) = \delta(k-m)$
$y(x) = g(x)h(x)$	$Y(k) = \sum_{m=0}^{k} H(m)G(k-m)$
$y(x) = \exp(\lambda x)$	$Y(k) = \frac{\lambda^k}{k!}$
$y(x) = (1+x)^m$	$Y(k) = \frac{m(m-1)\cdots(m-k+1)}{k!}$

3 Formulation of the model

The flow diagram of terrorist network within a host community can be deduced in Figure 1.

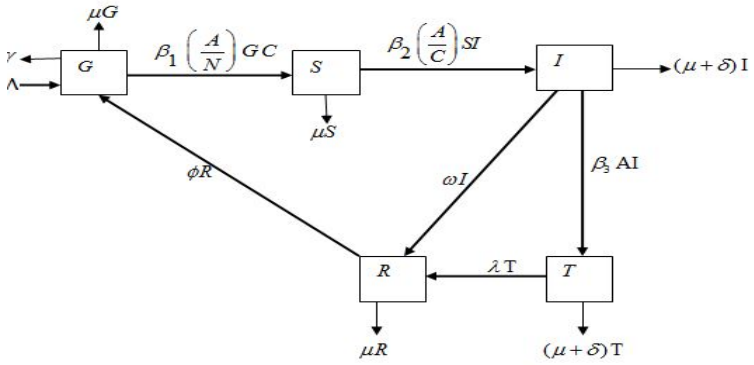


Figure 1: Flow diagram of terrorist with government intervention

.

The model of terrorism with government intervention corresponding to the flow diagram in Figure 1 is given by the deterministic system of nonlinear differential equations as

$$\left.\begin{aligned}
\frac{dG}{dt} &= \Lambda - \gamma + \phi R(t) - \mu G(t) - \beta_1 \left(\frac{A}{N}\right) G(t)C(t) \\
\frac{dS}{dt} &= \beta_1 \left(\frac{A}{N(t)}\right) G(t)C(t) - \mu S(t) - \beta_2 \left(\frac{A}{C(t)}\right) S(t)I(t) \\
\frac{dI}{dt} &= \beta_2 \left(\frac{A}{C(t)}\right) S(t)I(t) - (\omega + \mu + \delta)I(t) - \beta_3(A)I(t) \\
\frac{dT}{dt} &= \beta_3 AI(t) - (\mu + \delta)T(t) - \lambda T(t) \\
\frac{dR}{dt} &= \lambda T(t) + \omega I(t) - \mu R(t) - \phi R(t)
\end{aligned}\right\}. \tag{8}$$

The model focuses on a population of interest $T(t)$, which is divided into noncore group $G(t)$ and core group $C(t)$ in [11]. The noncore group $G(t)$ is the general population, which the individuals in the group are at risk of adopting the ideology. The noncore group is usually the source of the recruitment pool, where the entire core group builds their members from. The core group $C(t)$ consists of susceptible group $S(t)$, the moderate group $I(t)$, Terrorist group $T(t)$, and recovered group $R(t)$. The susceptible group $S(t)$ is individuals who have not yet been converted into the ideology but have begun to walk around and fall into terrorist beliefs. Moderate group $I(t)$ includes individuals who have been transformed with beliefs to reinforce their

terrorist views (Training stage). The terrorist groups $T(t)$ are individuals who have completely converted and have received training to become either leaders or foot soldiers of the terrorist respectively. The recovered group $R(t)$ is individuals who are neutralized due to counter-terrorist/NGOs activities applying to each group.

Table 2: Description of parameters of terrorism model equation (10)

Parameters	**Description**
Λ	Recruitment per-capital (birth/migration rate)
δ	Death due to suicide attack or arrest by counter-terror activities
β_1	Rate at which terrorists recruit their members from general to moderate group
β_2	Contact rate between susceptible and moderate groups
μ	Natural death rates
β_3	Contact rate between moderate and terrorist groups
A	Probability the government measure intervention on each groups
γ	Immigration rate
λ	Rate at which individuals recover from the ideology
ω	Probability that the moderate terrorists move to $R(t)$ due to government intervention
ϕ	Rate at which terrorist individuals progress from $R(t)$ to $G(t)$ after being certified by the counter-terrorist/NGO's

Table 3: Description of variables of terrorism model equation (10)

Variables	**Description**
$G(t)$	Number of general group at time t
$S(t)$	Number of partial-terrorist at time t
$I(t)$	Number of moderate terrorist (indoctrinization) at time t
$T(t)$	Number of terrorist at time t
$R(t)$	Number of terrorist recovered at time t

We have

$$C(t) = S(t) + I(t) + T(t) + R(t). \tag{9}$$

The total population is

$$N(t) = G(t) + C(t). \tag{10}$$

Then $\beta_1, \beta_2, \beta_3$ are parameters, which define the recruitment pool of each subgroup with each other. Similarly, $\beta_1 \left(\frac{A}{N(t)}\right) G(t)C(t)$ is the transition rate from $G(t)$-$S(t)$. The moderate group has more chances of recruiting, since the group builds its members from $S(t)$ and transfers them into the terrorist group individuals. There will accept their duties and execute the

mission given to them (to become complete terrorist at $T(t)$), in [11]. The dynamics between $S(t)$-$I(t)$ and the transfer rate from $I(t)$-$T(t)$ within the core are $\beta_2\left(\frac{A}{C(t)}\right)S(t)I(t)$ and $\beta_3 AI(t)$, respectively.

The intervention rate A is bounded in an interval $0 \leq A < 1$, which is the measure of intervention on each compartment. The intervention rate A is the military/NGO's activities, which aimed at reducing the recruitment pool/threats, respectively. Due to the intervention, some individuals will move to a recovered group. After some time, individuals who received either jail time or certified repentance and others will fall back to $G(t)$. It is assumed that recovery from terrorism is not permanent, since it deals with ideology, marginalization and poverty. Moreover, Λ is the birth/migration rate, γ is the immigration rate, μ is the natural death from each compartment, δ is terrorist induced death rate due to military intervention or by suicide, while λ is the rate at which individuals recover from the ideology due to intervention and ϕ is the rate at which terrorist individuals progress from $R(t)$-$G(t)$ after being certified by the counter-terrorist/NGO's.

3.1 Assumptions of the model

From the model formulated, it is assumed that A is the intervention rate, while $C(t)$ is equal to 1, $(C(t) = 1)$. Similarly, it is assumed that $N(t) = 1$. The recovered rate of individual terrorists in this model is not permanent since their behavior is dynamics. This means individual terrorists can change after being integrated into society can be re-engaged. Also, A is assumed to be a probability that not all individuals terrorists the intervention will capture (some will hide). The rate at which an individual leaves the terrorist group to enter the recovered group is λ, where ϕ is the fraction at which terrorists successfully move to the general group due to the intervention by the government. Susceptible, moderates and terrorists are assumed to have frequent contacts with the general group.

Finally, the age limit of an individual ranges from five years and above while everybody in the population has the same average natural death rate μ.

3.2 Solution of the model using ETM and DTM

Consider the nonlinear differential equation corresponding to the model diagram as

$$\frac{dG}{dt} = \Lambda - \gamma + \phi R - (\mu + \beta_1 A)G, \tag{11}$$

$$\frac{dS}{dt} = \beta_1 AG - \mu S - \beta_2(A)SI, \tag{12}$$

$$\frac{dI}{dt} = \beta_2(A)SI - (\omega + \mu + \delta + \beta_3 A)I, \tag{13}$$

$$\frac{dT}{dt} = \beta_3 AI - (\mu + \delta + \lambda)T, \tag{14}$$

$$\frac{dR}{dt} = \lambda T + \omega I - (\mu + \phi)R. \tag{15}$$

So, using (14)-(15) gives the following equations:

$$\frac{dG}{dt} = L_1 + \phi R - L_2 G, \tag{16}$$

$$\frac{dS}{dt} = L_3 G - \mu S - L_4 IS, \tag{17}$$

$$\frac{dI}{dt} = L_4 IS - L_5 I, \tag{18}$$

$$\frac{dT}{dt} = L_6 I - L_7 T, \tag{19}$$

$$\frac{dR}{dt} = \lambda T + \omega I - L_8 R, \tag{20}$$

where

$$\begin{gathered} L_1 = \Lambda - \gamma, \\ L_2 = \mu + \beta_1 A, \\ L_3 = \beta_1 A, \\ L_4 = \beta_2 A, \\ L_5 = \omega + \mu + \delta + \beta_3 A, \\ L_6 = \beta_3 A, \\ L_7 = \mu + \delta + \lambda, \\ L_8 = \mu + \phi. \end{gathered}$$

Applying the ETM on both sides of equations (16)–(20) gives

$$E\left(\frac{dG}{dt}\right) = E\{L_1 + \phi R - L_2 G\}, \tag{21}$$

$$E\left(\frac{dS}{dt}\right) = E\{L_3 G - \mu S - L_4 IS\}, \tag{22}$$

$$E\left(\frac{dI}{dt}\right) = E\{L_4 IS - L_5 I\}, \tag{23}$$

$$E\left(\frac{dT}{dt}\right) = E\{L_6 I - L_7 T\}, \tag{24}$$

$$E\left(\frac{dR}{dt}\right) = E\{\lambda T + \omega I - L_8 R\}. \tag{25}$$

Suppose that $E\left[G(t)\right] = T_1(u)$, that $E\left[S(t)\right] = T_2(u)$, that $E\left[I(t)\right] = T_3(u)$, that $E\left[T(t)\right] = T_4(u)$, that $E\left[R(t)\right] = T_5(u)$ and that L_1 in (21) is a constant. Then using the ETM properties of constant on L_1 yields $E(L_1) = u\int_0^\infty L_1 \exp(\frac{-t}{u})dt = u\left\{\lim_{t\to\infty}\left(L_1\exp(\frac{-t}{u}) + uL_1\right)\right\} = u^2L_1$ as $E(L_1) = u^2L_1$.

Applying the DTM on the nonlinear part in (22) and (23) implies

$$E\left(\frac{dS}{dt}\right) = E\{L_3G - \mu S - L_4A_n\}, \tag{26}$$

$$E\left(\frac{dI}{dt}\right) = E\{L_4A_n - L_5I\}, \tag{27}$$

where $A_o = \sum_{m=0}^{k} S(m)I(k-m)$, which is defined by the DTM and the polynomial as

$$\left.\begin{aligned} A_0 &= S_0I_0, \\ A_1 &= S_0I_1 + S_1I_0, \\ A_2 &= S_0I_2 + S_1I_1 + S_2I_0, \\ A_3 &= S_0I_3 + S_1I_2 + S_2I_0, \\ A_4 &= S_0I_4 + S_1I_3 + S_2I_2 + S_3I_2 + S_4I_0, \end{aligned}\right\} \tag{28}$$

Subsequently

$$\left[\frac{T_1(u)}{u} - uG_0\right] = L_1[u^2] + \phi[T_5(u)] - L_2[T_1(u)], \tag{29}$$

$$\left[\frac{T_2(u)}{u} - uS_0\right] = L_3[T_1(u)] - \mu[T_2(u)] - E[L_4A_n], \tag{30}$$

$$\left[\frac{T_3(u)}{u} - uI_0(0)\right] = E[L_4A_n] - L_5[T_3(u)], \tag{31}$$

$$\left[\frac{T_4(u)}{u} - uT_0(0)\right] = L_6[T_3(u)] - L_7[T_4(u)], \tag{32}$$

$$\left[\frac{T_5(u)}{u} - uR_0(0)\right] = \lambda[T_4(u)] + \omega[T_3(u)] - L_8[T_5(u)], \tag{33}$$

with the initial conditions

$$\left.\begin{aligned} G(0) &= G_0(t) = G_0 = 100, \\ S(0) &= S_0(t) = S_0 = 50, \\ I(0) &= I_0(t) = I_0 = 15, \\ T(0) &= T_0(t) = T_0 = 25, \\ R(0) &= R_0(t) = R_0 = 10, \end{aligned}\right\} \text{for } t \geq 0. \tag{34}$$

By simplifying and collecting the same terms of (29)–(33), we have

$$\left[\frac{1}{u}+L_2\right]T_1(u)-\phi T_5(u)=u^2L_1+uG_0, \tag{35}$$

$$\left[\frac{1}{u}+\mu\right]T_2(u)-L_3T_1(u)=E[L_4A_n]+uS_0, \tag{36}$$

$$\left[\frac{1}{u}+L_5\right]T_3(u)=E[L_4A_n]+uI_0, \tag{37}$$

$$\left[\frac{1}{u}+L_7\right]T_4(u)-L_6T_3(u)=uT_0, \tag{38}$$

$$\left[\frac{1}{u}+L_8\right]T_5(u)-\omega T_3(u)-\lambda T_4(u)=uR_0. \tag{39}$$

Solve equations (35)–(39), for $T_1(u)$, $T_2(u)$, $T_3(u)$, $T_4(u)$, and $T_5(u)$. Then truncate A_n at S_0, I_0, using (28) and applying the ETM. Subsequently, the DTM is employed to handle the nonlinear terms in (22) and (23), and we solved the system with the help of Maple 17 software (Intel (R) Celeron (R) CPU@ 1.10GH$_Z$, (RAM) 4.00GB), which gives the following results:

$$\begin{aligned}T_1(u)=&\frac{1}{(1+uL_8)(uL_2+1)(1+uL_7)(uL_5+1)}(((u\lambda L_6+u\omega L_7)+\omega\phi EuL_4(A_n)\\&+u^2L_8L_1L_5L_7+(((L_1+R_0\phi+G_0L_8)L_5+L_1L_8+\phi\omega I_0)L_7\\&+(\phi\lambda T_0+L_1L_8)L_5+\phi\lambda L_6I_0)u^3\\&+((R_0\phi+L_5G_0+G_0L_8+L_1)L_7+(L_1+R_0\phi+G_0L_8)L_5\\&+(\lambda T_0+\omega I_0)\phi+L_1L_8)u^2\\&+(L_1+G_0L_7+R_0\phi+L_5G_0+G_0L_8)u+G_0)u^2)\end{aligned} \tag{40}$$

$$\begin{aligned}T_2(u)=&(((1+((L_3\phi\omega+L_2L_8L_5)L_7+L_3\phi\lambda L_6)u^4+(((L_8+L_2)L_5+L_2L_8)L_7\\&+L_3\phi\omega+L_2L_8L_5)u^3+((L_8+L_5+L_2)L_7+(L_8+L_2)L_5+L_2L_8)u^2\\&+(L_5+L_2+L_7+L_8)u)EL_4(A_n)\\&u(u^5L_3L_8L_1L_5L_7+(((L_1+R_0\phi+G_0L_8)L_3+L_8L_2S_0)L_5\\&+L_3(L_1L_8+\phi\omega I_0))L_7\\&+L_3((\phi\lambda T_0+L_1L_8)L_5+\phi\lambda L_6I_0))u^4+(((L_3G_0+S_0(L_8+L_2))L_5\\&+(L_1+R_0\phi+G_0L_8)L_3+L_8L_2S_0)L_7\\&+((L_1+R_0\phi+G_0L_8)L_3+L_8L_2S_0)L_5+L_3(L_1L_8+(\lambda T_0+\omega I_0)\phi))u^3\\&+((S_0L_5+L_3G_0+S_0(L_8+L_2))L_7\\&+(L_3G_0+S_0(L_8+L_2))L_5+(L_1+R_0\phi+G_0L_8)L_3+L_8L_2S_0)u^2\\&+(S_0L_7+S_0L_5+L_3G_0+S_0(L_8+L_2))u+S_0))u)/\end{aligned}$$

$$((1+uL_8)(uL_2+1)(1+uL_7)(uL_5+1)) \tag{41}$$

$$T_3(u) = \frac{(EL_4(A_n)+uI_0)u}{uL_5+1} \tag{42}$$

$$T_4(u) = \frac{u^2(EL_4(A_n)L_6+(L_5T_0+L_6I_0)u+T_0)}{(1+uL_7)(uL_5+1)} \tag{43}$$

$$\begin{aligned} T_5(u) =& \frac{1}{(1+uL_7)(uL_5+1)(1+uL_8)}(u^2(\lambda L_6+\omega L_7)u+\omega)EL_4(A_n) \\ &+ ((L_5R_0+\omega I_0)L_7+\lambda(L_5T_0+L_6I_0))u^2 \\ &+ (R_0L_7+\lambda T_0+L_5R_0+\omega I_0)u+R_0)). \end{aligned} \tag{44}$$

Thus the initial condition and parameters are
$G(0) = 100$, $S(0) = 50$, $I(0) = 15$, $T(0) = 25$, $R(0) = 10$, $\Lambda = 600$, $\phi =$ 0,0022, $\omega = 0.015$, $\lambda = 0.008$, $A = 0.7$, $\beta_1 = 0.0035$, $\beta_2 = 0.000005$, $\beta_3 =$ 0.00045, $\delta = 0.0036$, $\mu = 0.000034247$ and $\gamma = 0.7$.

Applying the conditions above on (40)–(44) gives

$$\begin{aligned} T_1(u) =& 100u^2 + 599.0035754u^3 - 1.90652029u^4 + 0.00605328738u^5 \\ &- 0.0000189756893u^6 + \cdots, \\ T_2(u) =& 50u^2 + 0.4708656101u^3 + 1.886410919u^4 - 0.00606738403u^5 \\ &+ 0.0000192353168u^6 + \cdots, \\ T_3(u) =& 14u^2 - 0.2800526593u^3 + 0.005228652322u^4 \\ &- 0.00009762023033u^5 + 0.000001822593812u^6 + \cdots, \\ T_4(u) =& 25u^2 - 0.2847812124u^3 + 0.003197551982u^4 \\ &- 0.00003499897343u^5 + 3.65259688510^{-7}u^6 + \cdots, \\ T_5(u) =& 10u^2 + 0.3886565195u^3 - 0.007974537753u^4 \\ &+ 0.0001361224536u^5 - 0.000002327546774u^6 + \cdots. \end{aligned}$$

Take the inverse of the Elzaki transform to solve the induced recursive equation of $G(t) = E^{-1}(T_1(u)), S(t) = E^{-1}(T_2(u)), I(t) = E^{-1}(T_3(u)), T(t) =$ $E^{-1}(T_4(u))$, and $R(t) = E^{-1}(T_5(u))$.
Then using Maple 17 software to solve the above expression, the closed form of the solution when $k = 4$ can be written as

$$\begin{aligned} G(t) =& 100 + 599.0035t - 0.95326t^2 + 0.001008t^3 + 7.9065 \times 10^{-7}t^4 + \cdots, \\ S(t) =& 50 + 0.02628t + 0.0839t^2 - 0.000009706t^3 + 5.667 \times 10^{-10}t^4 + \cdots, \\ I(t) =& 15 - 0.28005t + 0.00261t^2 - 0.00001626t^3 + 7.594 \times 10^{-8}t^4 + \cdots, \\ T(t) =& 25 - 0.2847t + 0.00159t^2 - 0.0000058t^3 + 1.5219 \times 10^{-8}t^4 + \cdots, \end{aligned}$$

$$R(t) = 10 + 0.3886t - 0.00398t^2 + 0.0000226t^3 - 9.698 \times 10^{-8}t^4 + \cdots,$$

Furthermore, the DTM is also employed to solve (16)–(20) numerically and compare the result on the graph with that of Elzaki transform. The initial condition and the parameter values are
$G(0) = 100$, $S(0) = 50$, $I(0) = 15$, $T(0) = 25$, $R(0) = 10$, $\Lambda = 600$, $\phi =$ 0,0022, $\omega = 0.015$, $\lambda = 0.008$, $A = 0.7$, $\beta_1 = 0.0035$, $\beta_2 = 0.000005$, $\beta_3 =$ 0.00045, $\delta = 0.0036$, $\mu = 0.000034247$ and $\gamma = 0.7$.
Also, using the transformation in Table 1, for (16)–(20), gives

$$G(k+1) = \frac{L_1(k) + \phi R(k) - L_2 G(k)}{k+1}, \tag{45}$$

$$S(k+1) = \frac{L_4 \sum_{m=0}^{k} S(m) I(k-m) + L_3 G(k) - \mu S(k)}{k+1}, \tag{46}$$

$$I(k+1) = \frac{L_4 \sum_{m=0}^{k} S(m) I(k-m) - L_5 I(k)}{k+1}, \tag{47}$$

$$T(k+1) = \frac{L_6 I(k) - L_7 T(k)}{k+1}, \tag{48}$$

$$R(k+1) = \frac{\lambda T(k) + \omega I(k) - L_8 R(k)}{k+1}. \tag{49}$$

Applying the initial condition in (45) to (49) and using Maple 17 software, we have

$$\begin{aligned}
G(t) &= \sum_{m=0}^{4} G(k) t^k \\
&= 100 + 599.2905t - 0.09373t^2 + 0.0000069105t^3 + 1.1743 \times 10^{-8}t^4 + \cdots, \\
S(t) &= \sum_{m=0}^{4} S(k) t^k \\
&= 50 + 0.02658t + 0.08389t^2 - 0.000009522t^3 - 1.8694 \times 10^{-9}t^4 + \cdots, \\
I(t) &= \sum_{m=0}^{4} I(k) t^k \\
&= 15 - 0.27975t + 0.002608t^2 - 0.00001605t^3 + 7.24869 \times 10^{-8}t^4 + \cdots, \\
T(t) &= \sum_{m=0}^{4} T(k) t^k \\
&= 25 - 0.29031t + 0.0016837t^2 - 0.000006498t^3 + 1.87567 \times 10^{-8}t^4 + \cdots, \\
R(t) &= \sum_{m=0}^{4} R(k) t^k \\
&= 10 + 0.3886t - 0.003965t^2 + 0.000022338t^3 - 9.3485 \times 10^{-8}t^4 + \cdots.
\end{aligned}$$

4 Numerical simulation

The simulations were carried out using the following variables and parameters for initial conditions. The final time was $t = 5$ years. Computations were run in Maple 17 software (Intel (R) Celeron (R) CPU@ 1.10GH$_Z$, (RAM) 4.00GB) for analyzing. Therefore, the variable and parameter values of source are
$G(t) = 100$, $S(t) = 50$, $I(t) = 15$, $T(t) = 25$, $R(t) = 10$, $\Lambda = 600$ [26], $\delta = 0.0036$ [24], $\beta_1 = 0.0035$ [24], $\beta_2 = 0.000005$ [24], $\beta_3 = 0.00045$ [24], $\mu = 0.000034247$ [24], $\lambda = 0.008$ [26], $A = 0.1$, 0.7,0.9 varies, $\gamma = 0.7$ [26], $\omega = 0.015$ and $\phi = 0.0022$.

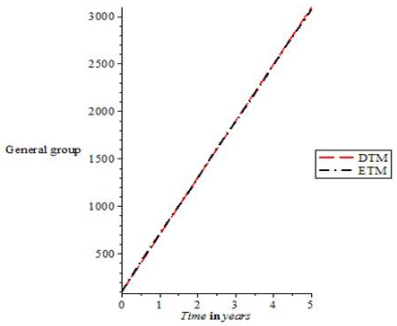


Figure 2: Comparison graph between ETM and DTM on general group in time.

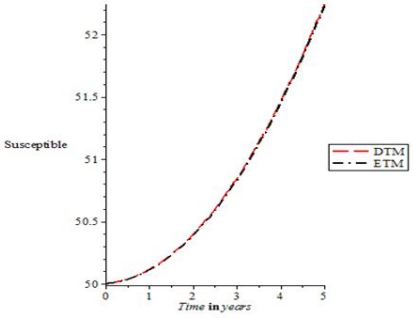


Figure 3: Comparison graph between ETM and DTM on susceptible in time.

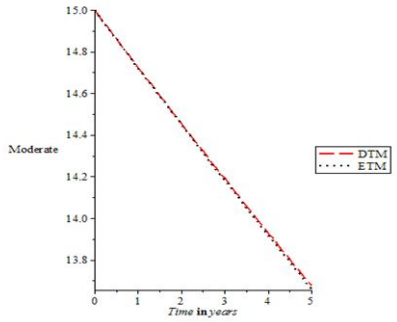


Figure 4: Comparison graph between ETM and DTM on moderate group in time.

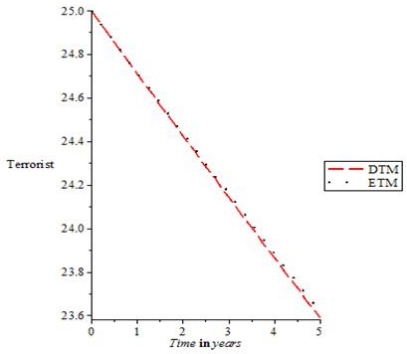


Figure 5: Comparison graph between ETM and DTM on terrorist group in time.

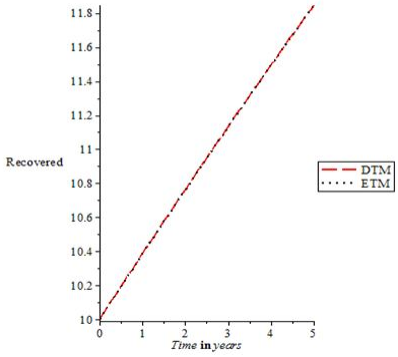


Figure 6: Comparison graph between ETM and DTM on recovered group in time.

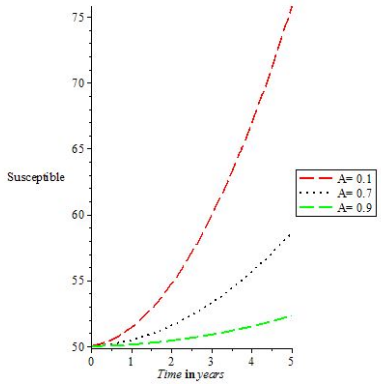


Figure 7: Graph of susceptible group in time with different rate of government intervention A.

5 Discussion of results

The ETM was introduced to determine the approximate numerical solution of a system of ordinary differential equations. Maple 17 software was used for simulations. Government intervention on susceptible groups proved to be effective in reducing terrorist threats since the susceptible group in this research is seen to be at risk of adopting the ideology through contact. similarly, this can be deduced from Figure 3 with different rates of intervention A.

The graph in Figure 2 for five years, showed that ETM and DTM agreed with each other without any significant difference in comparison. Similarly, this can be deduced from Figure 3 where the two methods also agreed on the comparison. Figure 4 shows the comparison graph between ETM and DTM on the moderate group against time, which is gradually reduced due to government intervention. From the graph, this can be seen when time duration continues after 5 years, the methods for analytic solution may not agree with each other due to control on moderate terrorist since it is dynamics. Furthermore, a similar scenario can be deduced from Figure 5 on the comparison between ETM and DTM. This can also be observed from the graph in Figure 6, where the effect of the comparison showed no significant difference between ETM and DTM on the recovered group. Finally, Figures 2–3 showed that both methods are more effective, more powerful, and accurate for solving a system of differential equations for an approximate solution, but Elzaki transform converges faster than DTM.

6 Conclusion

A mathematical model of terrorism with government intervention was developed to reduce the recruitment pools within the five compartments. The ETM and DTM were tested and solved the governing differential equations compared to the results on the graph. Government intervention on susceptible groups proved to be 90% effective as well as a moderate group. The susceptible and moderate group in this study appears to be exposed to adopting the ideology through contact. The ETM was employed to solve the model with Maple 17 software and applied DTM to decompose the nonlinear term since some differential equations have difficulty finding analytic/exact solutions. Finally, a plan to incorporate other issues would be addressed in the future study.

Acknowledgements

Special thanks go to my Ph.D. supervisor Prof. M.O Ibrahim of the University of Ilorin, Nigeria, and Prof. Mohammed Jiya of the Federal University of Technology Minna, Nigeria, for their help and introducing the Elzaki transform method to me for solving differential equations.

References

1. Adamu, G. and Ibrahim, M.O. *Mathematical modelling of dynamics behaviour of terrorism and control*, Caspian. J. Math. Sci., 9(1) (2020), 68–85.

2. Aggarwal, S., Bhatnagar, K. and Dua, A. *Dualities between Elzaki transform and some useful integral transforms*. Int. J. Inno. Tech. Expl. Eng., 8(12)(2019), 4312–4318.

3. Aggarwal, S., Mishra, R. and Kumar, A. *A comparative study of Mohand and Elzaki Transforms*, Global. J. Eng. Sci. Res., 6(2) (2019), 203–213.

4. Aggarwal, S., Singh, D.P., Asthana, N. and Gupta, A.R. *Application of Elzaki transform for solving population growth and decay problems*, J. Emer. Tech. Inno Res., 5(9) (2018), 281–284.

5. Ahmad, M.Z., Alsarayreh, D., Alsarayreh, A. and Qaralleh, I. *Differential transformation method (DTM) for solving SIS and SI epidemic models*, Sains. Malay., 46(10) (2017), 2007–2017.

6. Akinola, E.I., Akinpelu, F.O. and Oladejo, A.A.J. *Elzaki decomposition method for solving epidemic model*, Int. j. Chem. Math. Phys., 1(1) (2017), 68–72.

7. Anjum, N., Suleman, M., Lu, D., He, J.H. and Ramzan, M. *Numerical iteration for nonlinear oscillators by Elzaki transform*, J. Freq. Nois. Vibr. Cont., 0(0) (2019), 1–6.

8. Basit, A. *Countering violent extremism: evaluating Pakistan's counter-radicalization and de-radicalization initiatives*, Islamabad. Pol. Res. Inst. J., 15(2) (2015), 44–68.

9. Bhadane, P. and Ghadle, K. *Application of Elzaki transform to system of linear differential equations*, Int. J. Res. Pub., 6(2) (2016), 36–43.

10. Castillo-Chavez, C. and Song, B. *Models for the transmission dynamics of fanatic behaviours*, In Bioterrorism: Math. Modell. Appl. Homel. Secur. Soci. Indust. Appl. Math., 28 (2003), 155–172.

11. Cherif, A., Yoshioka, H., Ni, W. and Bose, P. *Terrorism mechanism of radicalization process, control of contagion and counter-terrorist measures*, Santa Fe. Insti., 910 (2010), 1–14.

12. Duz, M. and Elzaki, T. M. *Solution Of constant coefficients partial derivative equations with Elzaki transform method*, TWMS J. Appl. Eng. Mathe., 9(3) (2019), 563–570.

13. Elzaki. T.M. *On the connections between Laplace and Elzaki transforms*, Adv. Theory.Appl. Math., 6(1) (2011), 1–11.

14. Elzaki, T.M. *The new integral transform Elzaki transform*, Global J. Pure Appl. Math., 7(1) (2011), 57–64.

15. Elzaki, T.M. *Solution of nonlinear differential equations using mixture of Elzaki transform and differential transform method*, In Int. Math. Forum., 7(13) (2012), 631–638.

16. Elzaki, T.M., Alamri, B.A.S. *Combination of transform methods for solving linear and nonlinear differential equations with proportional delay*, Int. Math. Forum., 10(10) (2015), 467–476.

17. Elzaki, T.M., Elzaki, S.M. and Elnour, E.A. *On the new integral transform "ELzaki Transform" fundamental properties investigations and applications*, Global J. Math. Sci. Theory Pract., 4(1) (2012), 1–13.

18. Global Terrorism Index 2020. *Measuring the impact of terrorism, institute for economics and peace*, Sydney, November 2020.

19. Ifeyinwa, M.H. *Mathematical modeling of the transmission dynamics of syphilis disease using differential transformation method*, Math. Model. Appl., 5(2) (2020), 47–54.

20. Ige, O.E. and Oderinu, R.A. *Adomian polynomial and Elzaki transform method for solving Sine-Gordon equations*, IAENG Int. J. Appl. Math., 49(3) (2019), 1–7.

21. Kalavathi, A., Kohila, T. and Upadhyaya, L.M. *On the degenerate Elzaki transform*, Bull. Pure Appl. Sci. Sect. E Math. Stat. 40E (2021). 99–107.

22. Martcheva, M. *An introduction to mathematical epidemiology*, New York, Springer, 2015.

23. Mohamed, M.Z. and Elzaki, T.M. *Applications of new integral transform for linear and nonlinear fractional partial differential equations*, J. King Saud Uni. Sci., 32(1) (2020), 544–549.

24. Nathan, O., Lawi, G. and Nthiiri, J. *Modelling the dynamics of radicalization with government intervention*, Neural Parallel. Sci. Comput., 26 (2018), 211–224.

25. Orakzai, S.B. *Pakistan's approach to countering violent extremism (CVE), reframing the policy framework for peace building and development strategies*. Study. Conf. Terro., 42(8) (2019), 755–770.

26. Santoprete, M. *Countering violent extremism: A mathematical model*, Appl. Math. Comput., 358 (2019), 314–329.

27. Saqlain, M., Irshad, W., Bashir, U. and Murtaza, G.*Numerical solutions for third order ordinary differential equation by differential transform method & Elzaki transform method*, Sci. Inquiry Review, 2(4) (2018), 01–12.

28. Shah, N.A., Dassios, I. and Chung, J.D. *A decomposition method for a fractional-order multi-dimensional telegraph equation via the Elzaki transform*, Symt., 13(1) (2021), 1–8.

29. Sharjeel, S. and Barakzai, M.A.K. *Some new applications of Elzaki transform for solution of linear Volterra type integral equations*, J. Appl. Math. Phys., 7(8) (2019), 1877–1892.

30. Suleman, M., Elzaki, T., Wu, Q., Anjum, N. and Rahman, J.U. *New application of Elzaki projected differential transform method*, J. Comput. Theoretical Nanosci., 14(1) (2017), 631–639.

31. Suleman, M., Lu, D., He, J.H., Farooq, U., Hui, Y.S. and Rahman, J.U. *Numerical investigation of fractional HIV model using Elzaki projected differential transform method*, Fract, 26(5) (2018), 1850062.

32. Verma, D. and Alam, A. *Analysis of simultaneous differential equations By Elzaki transform approach*, Sci. Tech. Devel., 9(1) (2020), 364–367.

33. Verma, D. and Gupta, R. *Analyzing Boundary Value Problem in Physical Sciences Via Elzaki Transform*, ASIO J. Chem. Phys. Math. Appl. Sci., 4(1) (2020), 17–20.

34. Zhou, J.K., *Differential transformation and its applications for electrical circuits*, Huazhong University Press, Wuhan, China, 1986.

35. Ziane, D., Elzaki, T.M. and Cherif, M.H. *Elzaki transform combined with variational iteration method for partial differential equations of fractional order*, Fundamental J. Math. Appl., 1(1) (2018), 102–108.

How to cite this article
G. Adamu and M.O. Ibrahim Elzaki transform method for solving deterministic modeling of terrorism. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 334-354. doi: 10.22067/ijnao.2021.72806.1060.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 355–370
DOI: 10.22067/ijnao.2021.73813.1076
https://ijnao.um.ac.ir/

How to cite this article
Research Article

A numerical approach for singular perturbation problems with an interior layer using an adaptive spline

E. Srinivas, M. Lalu and K. Phaneendra*

Abstract

An adaptive spline is used in this work to deal with singularly perturbed boundary value problems with layers in the interior region. To evaluate the layer behavior in the solution, a different technique on a uniform mesh is designed by replacing the first-order derivatives with nonstandard differences in the adaptive cubic spline. A tridiagonal solver is used to solve the tridiagonal system of the difference scheme. The fourth-order convergence of the approach is established. The validity of the suggested computational method is demonstrated through numerical experiments, which are compared to other methods in the literature. Layer profile is depicted in graphs.

AMS subject classifications (2020): 65L10; 65L1; 65L12.

Keywords: Singular perturbation problem; Interior layer; Adaptive spline; Tridiagonal system.

---

* Corresponding author

Received 23 November 2021; revised 29 December 2021; accepted 29 December 2021

Erla Srinivas
Department of Mathematics, University College of Engineering, Osmania University, Hyderabad. email: srinivas.erla@gmail.com

Mudavath Lalu
Department of Mathematics, University College of Engineering, Osmania University, Hyderabad. email: lalunaikmudathou@gmail.com

Kolloju Phaneendra
Department of Mathematics, University College of Engineering, Osmania University, Hyderabad. email: kollojuphaneendra@yahoo.co.in

1 Introduction

A small parameter leads to singular perturbation problems (SPPs) in a variety of science and engineering problems in fluid mechanics, elasticity, aerodynamics, magneto-hydrodynamics, optimal control, and other domains of fluid motion [1]. WKB difficulties, the modeling of steady and unsteady viscous flow problems having big Reynolds numbers, magneto-hydrodynamics duct problems with high Hartman numbers, and so on are few notable examples. The numerical solutions to SPPs are often nontrivial because of the rapid development or decay (boundary/interior layer behavior) of solutions to dissipative problems. Here, we include a few references related to the interior layer problems. The authors in [2, 3, 5, 11, 16, 24, 25] provided a full theoretical, analytical, and numerical discussion of this topic. The authors of [8, 9] provided an informative overview of SPPs on layers in their survey publications. In [10], for the Emden–Fowler type equations, the author proposed a finite element collocation technique based on cubic B-splines. Farrell [4] provides a uniformly convergent scheme for the SPP with a turning point problem. Geng, Qian, and Li [6] suggested an approach based on the reproducing kernel method and the asymptotic expansion technique. Miller, O'Riordan, and Shishkin [12] explained how to solve convection-diffusion and reaction-diffusion problems using conventional techniques on Shishkin meshes. Natesan, Vigo-Aguiar and Ramanujam [13] split the region into two subdomains, the layer domain and the regular domain, and then resolved the layer problem with the fitted operator method and the regular problem with the finite difference technique. Navnit [14] proposed a fourth-order numerical approach for solving an SPP using an adaptive cubic spline. The same author in [15] proposed a general scheme for the solution of nonlinear SPP using an adaptive cubic spline. The author in [19] developed a nonuniform mesh optimal B-spline collocation method for the numerical solution of a singular two-point boundary value problem describing electro hydrodynamic flow of a fluid in a circular cylindrical conduit. In [20], two B-spline collocation methods were proposed to solve a class of nonlinear derivative dependent singular boundary value problems. Ramos [22] presented a locally analytical technique for solving SPPs with internal and boundary layers, as well as turning points.

With this motivation, in this paper, a higher order finite difference method on a uniform mesh for solving SPP turning point problems with an interior layer is proposed.

2 Statement of the problem

Consider a second-order SPP of the form

$$\varepsilon w''(t) + P(t)w'(t) + Q(t)w(t) = f(t), \qquad -1 \le t \le 1, \tag{1}$$

with boundary conditions

$$w(-1) = \alpha \qquad \text{and} \qquad w(1) = \beta, \tag{2}$$

where $0 < \varepsilon \ll 1$ is a positive perturbation parameter, α and β are finite constants, and $P(t)$, $Q(t)$, and $f(t)$ are considered to be suitably smooth functions. Based on the coefficient $P(t)$, the solution to (2) has a layer or turning point behavior. The presence of a turning point in the solution to the problem makes it substantially more difficult to handle a boundary or inner layer. In this paper, we examine at the situation where a problem's turning point results in a solution with an interior layer. Under assumptions

$$P(0) = 0, P'(0) \geq 0, \quad Q(t) \leq Q_0 < 0, \qquad \text{for all } t \in D = [-1,\ 1],$$

$$|P'(t)| \geq \frac{|P'(0)|}{2}, \qquad \text{for all } t \in D = [-1,\ 1],$$

the given turning point problem has a solution with interior layers at $t = 0$.

3 Adaptive spline

With grid points t_i in $[a, b]$, consider a mesh such that $\{\Omega : a = t_0 < t_1 < t_2 < \cdots < t_N = b$, where $h = t_i - t_{i-1}$ for $i = 1, 2, \ldots, N\}$. A function $\psi(t, \tau)$ interpolates $w(t)$ at the grid points t_i, depends on a variable, and leads to a cubic spline $\psi(t)$ in $[a, b]$ as $\tau \to 0$, named as an adaptive spline; see [7, 23]. The function $\psi(t, \tau)$ satisfies the equation

$$\varepsilon\psi''(t,\tau) - p\psi'(t,\tau) = \frac{t - t_{i-1}}{h}\left(\varepsilon\mathcal{M}_i - pm_i\right) + \frac{t_i - t}{h}\left(\varepsilon\mathcal{M}_{i-1} - pm_{i-1}\right), \tag{3}$$

where $t_{i-1} \leq t \leq t_i$, $\psi'(t, \tau) = m_i$, and $\psi''(t, \tau) = \mathcal{M}_i$. Solving (3) and using the interpolatory conditions $\psi(t_{i-1}, \tau) = w_{i-1}$ and $\psi(t_i, \tau) = w_i$, we have $\psi(s, \tau) = \mathcal{A}_i + \mathcal{B}_i e^{\tau z} - \frac{h^2}{\tau^3}\left[\frac{1}{2}\tau^2 z^2 + \tau z + 1\right]\left(\mathcal{M}_i - \frac{\tau}{h}m_i\right)$

$$+\frac{h^2}{\tau^3}\left[\frac{1}{2}\tau^2(1-z)^2 + \tau(1-z) + 1\right]\left(\mathcal{M}_{i-1} - \frac{\tau}{h}m_{i-1}\right), \tag{4}$$

where

$$\mathcal{A}_i\left(e^{\tau} - 1\right) = -x_i + x_{i-1}e^{\tau} - \frac{h^2}{\tau^3}\left[\left(\frac{\tau^2}{2} + \tau + 1\right) - \tau e^{\tau}\right]\left(\mathcal{M}_i - \frac{\tau}{h}m_i\right)$$

$$-\frac{h^2}{\tau^3}\left[\left(\frac{\tau^2}{2} - \tau + 1\right) - \tau\right]\left(\mathcal{M}_{i-1} - \frac{\tau}{h}m_{i-1}\right),$$

$$\mathcal{B}_i\left(e^{\tau}-1\right)=x_i-x_{i-1}e^{\tau}+\frac{h^2}{\tau^3}\left[(\frac{\tau}{2}+1)-\tau e^{\tau}\right](\mathcal{M}_i-\frac{\tau}{h}m_i)$$
$$+\left[(\frac{\tau}{2}-1)-\tau\right](\mathcal{M}_{i-1}-\frac{\tau}{h}m_{i-1}),$$

$\tau=\frac{Ph}{\varepsilon}$, and $z=\frac{s-s_{i-1}}{h}$. The spline function $\psi(t,\tau)$ on $[t_i,t_{i+1}]$ is acquired with replacing i by $(i+1)$ in (11). Utilizing the first or second derivative continuity conditions of $\psi(t,\tau)$ at $t=t_i$, we get the following relationship:

$$\begin{aligned}&\left(\mathcal{M}_{i+1}-\frac{\tau}{h}m_{i+1}\right)\left[e^{-\tau}(\frac{\tau^2}{2}+\tau+1)-1\right]\\&+\left(\mathcal{M}_i-\frac{\tau}{h}m_i\right)\left[e^{-\tau}\left(\frac{\tau^2}{2}-\tau-2\right)+\left(-\frac{\tau^2}{2}-\tau+2\right)\right]\\&+\left(\mathcal{M}_{i-1}-\frac{\tau}{h}m_{i-1}\right)\left[e^{-\tau}-1+\tau-\frac{\tau^2}{2}\right]\\&\quad=-\frac{\tau^2}{h^3}\left[e^{-\tau}w_{i+1}-\left(1+e^{-\tau}\right)w_i+w_{i-1}\right].\end{aligned}\tag{5}$$

Furthermore, the below relations are given for the adaptive splines:

$$(i)\qquad m_{i-1}=-h\left(\mathcal{A}_1\mathcal{M}_{i-1}+\mathcal{A}_2\mathcal{M}_i\right)+\frac{1}{h}\left(w_i-w_{i-1}\right),$$

$$(ii)\qquad m_i=h\left(\mathcal{A}_3\mathcal{M}_{i-1}+\mathcal{A}_4\mathcal{M}_i\right)+\frac{1}{h}\left(w_i-w_{i-1}\right),$$

$$(iii)\qquad \frac{\theta\ h}{2\tau}\mathcal{M}_{i-1}=-\left(\mathcal{A}_4m_{i-1}+\mathcal{A}_2m_i\right)+B_1\frac{(w_i-w_{i-1})}{h},$$

$$(iv)\qquad \frac{\theta\ h}{2\tau}\mathcal{M}_i=\left(\mathcal{A}_3m_{i-1}+\mathcal{A}_1m_i\right)+B_2\frac{(w_i-w_{i-1})}{h},$$

where

$$\mathcal{A}_1=\frac{1}{4}\left(1+\theta\right)+\frac{\theta}{20},\quad \mathcal{A}_2=\frac{1}{4}\left(1-\theta\right)-\frac{\theta}{20},$$
$$\mathcal{A}_3=\frac{1}{4}\left(1+\theta\right)-\frac{\theta}{20},\quad \mathcal{A}_4=\frac{1}{4}\left(1-\theta\right)+\frac{\theta}{20},$$
$$\mathcal{B}_1=\frac{1}{4}\left(1-\theta\right),\quad \mathcal{B}_2=-\frac{1}{2}\left(1+\theta\right),\quad \text{and}\quad \theta=\coth\left(\frac{\tau}{2}\right)-\frac{2}{\tau}.$$

We also obtain

$$\mathcal{A}_2\mathcal{M}_{i+1}+\left(\mathcal{A}_1+\mathcal{A}_4\right)\mathcal{M}_i+\mathcal{A}_3\mathcal{M}_{i-1}=\frac{1}{h^2}\left[w_{i+1}-2w_i+w_{i-1}\right].\tag{6}$$

Remark: In the limiting case when $\tau\to 0$, we have

$$\mathcal{A}_1=\mathcal{A}_4=\frac{1}{3},\quad \mathcal{A}_2=\mathcal{A}_3=\frac{1}{6},\quad \mathcal{B}_1=\frac{1}{2},\quad \mathcal{B}_2=-\frac{1}{2},\quad \theta=0,\quad \frac{\theta}{\tau}=\frac{1}{6},$$

and (8) reduces to the ordinary cubic spline scheme.

4 Description of the procedure

At the mesh point t_i, the suggested approach can be discretized by the convection-diffusion equation (2) as

$$\varepsilon\ \mathcal{M}_i = f(t_i) - P(t_i){w_i}'(t) - Q(t_i)w(t_i). \tag{7}$$

The above equations shall be replaced by (8), and by approximating the first order derivatives of t at the mesh points $t_1, t_2, \ldots, t_{N-1}$ as

$$\begin{aligned} w'_{i-1} &\approx \frac{-w_{i+1} + 4w_i - 3w_{i-1}}{2h}, \qquad w'_{i+1} \approx \frac{3w_{i+1} - 4w_i + w_{i-1}}{2h}, \\ w'_i &\approx \left(\frac{1 + 2\eta h^2 Q_{i+1} + \eta h\,[3P_{i+1} + P_{i-1}]}{2h}\right) w_{i+1} - 2\eta\ \ [P_{i+1} + P_{i-1}]\, w_i \\ &\quad - \left(\frac{1 + 2\eta h^2 Q_{i-1} - \eta h\,[P_{i+1} + 3P_{i-1}]}{2h}\right) w_{i-1} + \eta\ h\,[f_{i+1} - f_{i-1}], \end{aligned}$$

we get the tridiagonal system

$$L_i w_{i-1} + C_i\ w_i + U_i w_{i+1} = W_i \qquad \text{for}\ \ i = 1, 2, \ldots,\ N-1, \tag{8}$$

where

$$\begin{aligned} L_i =& -\ \ \varepsilon - \frac{3}{2}\mathcal{A}_3\ P_{i-1}\ h \\ & - (\mathcal{A}_1 + \mathcal{A}_4)\, P_i h\left[1 + 2\eta h^2 Q_{i-1} - \eta h\,(P_{i+1} + 3P_{i-1})\right] \\ & + \frac{A_2}{2} P_{i+1} + \mathcal{A}_3\ Q_{i-1} h^2 \end{aligned}$$

$$\begin{aligned} C_i =& 2\varepsilon + 2\mathcal{A}_3\ P_{i-1}\ h - 4\,(\mathcal{A}_1 + \mathcal{A}_4)\ \ P_i h_i^2 \eta [P_{i+1} + P_{i-1}] \\ & - 2\mathcal{A}_2\ P_{i+1} h + 2\,(\mathcal{A}_1 + \mathcal{A}_4)\ \ Q_i h^2 \end{aligned}$$

$$\begin{aligned} U_i =& -\varepsilon - \frac{\mathcal{A}_3}{2}\ P_{i-1}\ h\ \ + (\mathcal{A}_1 + \mathcal{A}_4)\, P_i h\left[1 + 2\eta h^2 Q_{i-1} + \eta h\,(3P_{i+1} + P_{i-1})\right] \\ & + \frac{3}{2}\mathcal{A}_2 \eta P_{i+1} h + \mathcal{A}_2 Q_{i+1} h^2 \end{aligned}$$

$$\begin{aligned} W_i =& \ h^2\left[(\mathcal{A}_2 - 2\eta\,(\mathcal{A}_1 + \mathcal{A}_4)\ \ P_i h)\, f_{i+1} + 2\,(\mathcal{A}_1 + \mathcal{A}_4)\ \ f_i\right] \\ & + h^2\left[(\mathcal{A}_3 + 2\eta\,(\mathcal{A}_1 + \mathcal{A}_4)\ \ P_i h)\, f_{i-1}\right]. \end{aligned}$$

The tridiagonal system (10) is solved for $i = 1, 2, \ldots, N-1$, to obtain the approximations $w_{1,} w_2, \ldots, w_{N-1}$ of the solution $w(t)$ at $t_1, t_2, \ldots, t_{N-1}$.

5 Truncation error

Developed local truncation error associated with the scheme in (10) is

$$\begin{aligned}T_i(h) =& \varepsilon\left[1-\left(2\left(\mathcal{A}_1+\mathcal{A}_4\right)+\mathcal{A}_2+\mathcal{A}_3\right)\right]h^2w''+\varepsilon\left(\mathcal{A}_3-\mathcal{A}_2\right)h^3w'''\\ &+\left[\frac{\mathcal{A}_2+\mathcal{A}_3}{2}-4\eta\varepsilon\left(\mathcal{A}_1+\mathcal{A}_4\right)-\frac{1}{6}\left[2\left(\mathcal{A}_1+\mathcal{A}_4\right)+\mathcal{A}_2+\mathcal{A}_3\right]\right]P_ih^4w'''\\ &+\frac{\varepsilon}{12}\left[1-6\left(\mathcal{A}_2+\mathcal{A}_3\right)\right]h^4w^{iv}\\ &-\frac{1}{12}\left(\mathcal{A}_3-\mathcal{A}_2\right)\left[P_iw^{iv}+2\left(P_i'+Q_i\right)w'''+6\left(P_i''+Q_i'\right)w''\right.\\ &\left.+2\left(P_i'''+3Q_i'''\right)w'\right]+\left[2Q_i'''\ w-2f'''\right]h^5+O\left(h^6\right).\end{aligned}$$

Thus for different values of A_2, A_3, A_1+A_4 in the scheme, (10) indicates different orders.
Remarks:
(i) If $\mathcal{A}_2=\mathcal{A}_3$, for any choice of arbitrary $\mathcal{A}_2, \mathcal{A}_1+\mathcal{A}_4$ with $(\mathcal{A}_1+\mathcal{A}_4)+\mathcal{A}_2=\frac{1}{2}$ and for any value of η , then the method is obtained for second order.
(ii) For $\mathcal{A}_2=\mathcal{A}_3=\frac{1}{12}, (\mathcal{A}_1+\mathcal{A}_4)=\frac{5}{12}$, and $\eta=-\frac{1}{20\varepsilon}$, the fourth-order method is derived.

6 Convergence

The convergence analysis of the proposed method is discussed in this section. The system of equations (10) in the matrix form with the boundary conditions is

$$(D+F)W+G+T(h)=0, \tag{9}$$

where

$$D=[-\varepsilon,\quad 2\varepsilon,\quad -\varepsilon]=\begin{bmatrix}2\varepsilon & -\varepsilon & 0 & 0 & \dots & 0\\ -\varepsilon & 2\varepsilon & -\varepsilon & 0 & \dots & 0\\ 0 & -\varepsilon & 2\varepsilon & -\varepsilon & \dots & 0\\ \dots & \dots & \dots & \dots & \dots & \dots\\ \dots & \dots & \dots & \dots & \dots & \dots\\ 0 & \dots & \dots & 0 & -\varepsilon & 2\varepsilon\end{bmatrix}$$

and

$$F=[\widetilde{z}_i,\widetilde{v}_i,\widetilde{w}_i]=\begin{bmatrix}\widetilde{v}_1 & \widetilde{w}_1 & 0 & 0 & \dots & 0\\ \widetilde{z}_2 & \widetilde{v}_2 & \widetilde{w}_2 & 0 & \dots & 0\\ 0 & \widetilde{z}_3 & \widetilde{v}_3 & \widetilde{w}_3 & \dots & 0\\ \dots & \dots & \dots & \dots & \dots & \dots\\ \dots & \dots & \dots & \dots & \dots & \dots\\ 0 & \dots & \dots & 0 & \widetilde{z}_{N-1} & \widetilde{v}_{N-1}\end{bmatrix}$$

in which

$$\widetilde{z}_i = -\frac{3}{2}\mathcal{A}_3\ P_{i-1}\ h\ -(\mathcal{A}_1+\mathcal{A}_4)\,P_i h\left[1+2\eta h^2 Q_{i-1}-\eta h\,(P_{i+1}+3P_{i-1})\right]$$
$$+\frac{\mathcal{A}_2}{2}P_{i+1}h+\mathcal{A}_3\ Q_{i-1}h^2$$
$$\widetilde{v}_i = 2\mathcal{A}_3\ P_{i-1}\ h-4\,(\mathcal{A}_1+\mathcal{A}_4)\ \ P_i h^2\eta\,[P_{i+1}+P_{i-1}]\ -2\mathcal{A}_2\ P_{i+1}h$$
$$+2\,(\mathcal{A}_1+\mathcal{A}_4)\ \ Q_i h^2$$
$$\widetilde{w}_i = -\frac{\mathcal{A}_3}{2}\ P_{i-1}\ h\ +(\mathcal{A}_1+\mathcal{A}_4)\,P_i h\left[1+2\eta h^2 Q_{i-1}+\eta h\,(3P_{i+1}+P_{i-1})\right]P_i$$
$$+\frac{3}{2}\mathcal{A}_2 P_{i+1}h+\mathcal{A}_2 Q_{i+1}h^2 \qquad \text{for all} \quad i=1,2,3,4,\ldots,N-1,$$

and

$$G=\left[q_1-\widetilde{z}_1\alpha, q_2, q_3, \ldots, q_{N-1}-\widetilde{w}_{N-1}\beta\right],$$

in which

$$q_i = h^2\left[(\mathcal{A}_2-2\eta\,(\mathcal{A}_1+\mathcal{A}_4)\ \ P_i h)\,f_{i+1}+2\,(\mathcal{A}_1+\mathcal{A}_4)\ \ f_i\right]$$
$$+h^2\left[(\mathcal{A}_3+2\eta\,(\mathcal{A}_1+\mathcal{A}_4)\ \ P_i h)\,f_{i-1}\right] \qquad \text{for all } i=2,\ldots,N-1,$$
$$T(h) = 0(h^6),\ \mathcal{A}_2=\mathcal{A}_3=\frac{1}{12},\quad (\mathcal{A}_1+\mathcal{A}_4)=\frac{5}{12},\quad \eta=-\frac{1}{20\varepsilon},$$
$$W=[W_1,W_2,W_3,\ldots,W_{N-1}]^T,\quad T(h)=[T_1,T_2,\ldots,T_{N-1}]^T,$$
$$O=[0,0,\ldots,0]^T.$$

Let $w=[w_1,w_2,\ldots,w_{N-1}]^T\cong W$ satisfy the equation

$$(D+F)\,w+G=0. \tag{10}$$

Let $E=[e_1,e_2,\ldots,e_{N-1}]^T=w-W$, where $e_i=w_i-W_i,\ i=1,2,\ldots,N-1$, is the discretization error. Using (9) and (10), we obtain

$$(D+F)\,E=T(h). \tag{11}$$

Let $|Q(s)|\le\xi_1$ and $|P(s)|\le\xi_2$, where ξ_1 and ξ_2 are positive constants. If $\zeta_{i,j}$ is the (i,j)th element of F, then

$$|\zeta_{i,i+1}|=|\widetilde{w}_i|\le\varepsilon\left(h\,(\mathcal{A}_2+(\mathcal{A}_1+\mathcal{A}_4))\,\xi_1+h^2\mathcal{A}_2\xi_2+4\,(\mathcal{A}_1+\mathcal{A}_4)\,\eta h^2\xi_1^2\right)$$
$$+\left(2h^3\,(\mathcal{A}_1+\mathcal{A}_4)\,\eta\xi_1\xi_2\right) \qquad \text{for all } i=1,2,\ldots,N-2,$$
$$|\zeta_{i,i-1}|=|\widetilde{z}_i|\le\varepsilon\left(h(\mathcal{A}_2+(\mathcal{A}_1+\mathcal{A}_4))\eta_1+h^2\mathcal{A}_2\eta_2+4\,(\mathcal{A}_1+\mathcal{A}_4)\,\eta h^2\xi_1^2\right)$$
$$+\left(2h^3\,(\mathcal{A}_1+\mathcal{A}_4)\,\eta\xi_1\xi_2\right) \qquad \text{for all } i=2,3,\ldots,N-1.$$

Hence, for sufficiently small h, we have

$$|\zeta_{i,i+1}|\le\varepsilon \qquad \text{for all } i=1,2,\ldots,N-2, \tag{12}$$

$$|\zeta_{i,i-1}| \leq \varepsilon \qquad \text{for all } i = 2, 3, \ldots, N-1. \tag{13}$$

Therefore, $(D+F)$ is irreducible (see [27]).
Let the sum of the values of the ith row of $(D+F)$ be S_i. Then

$$S_i = \varepsilon - \frac{\mathcal{A}_2 h}{2}\left(P_{i+1} - 3P_{i-1}\right) + h\left(\mathcal{A}_1 + \mathcal{A}_4\right)P_i + h^2\left(\mathcal{A}_2 Q_{i-1} + 2\left(\mathcal{A}_1 + \mathcal{A}_4\right)Q_i\right)$$
$$+ h^2\left(\mathcal{A}_1 + \mathcal{A}_4\right)\eta P_i\left(3P_{i-1} + P_{i+1}\right) - 2h^3\left(\mathcal{A}_1 + \mathcal{A}_4\right)\eta P_i Q_{i-1} \qquad \text{for } i = 1,$$

$$S_i = h^2\left(Q_{i-1} + 2\left(\mathcal{A}_1 + \mathcal{A}_4\right)Q_i + \mathcal{A}_2 Q_{i+1}\right) + 2h^3\left(\mathcal{A}_1 + \mathcal{A}_4\right)P_i\eta\left(Q_{i+1} - Q_{i-1}\right)$$
$$\text{for all } i = 2, 3, \ldots, N-2,$$

$$S_i = \varepsilon + \frac{\mathcal{A}_2 h}{2}\left(P_{i-1} - 3P_{i+1}\right) - h\left(\mathcal{A}_1 + \mathcal{A}_4\right)P_i + h^2\left(\mathcal{A}_2 Q_{i-1} + 2\left(\mathcal{A}_1 + \mathcal{A}_4\right)Q_i\right)$$
$$- h^2\left(\mathcal{A}_1 + \mathcal{A}_4\right)\eta P_i\left(3P_{i+1} + P_{i-1}\right) - 2h^3\left(\mathcal{A}_1 + \mathcal{A}_4\right)\eta P_i Q_{i-1}$$
$$\text{for } i = N-1.$$

Let $\xi_{1^*} = \min\limits_{1\leq i\leq N}|P(t_i)|$, let $\xi_1^* = \max\limits_{1\leq i\leq N}|P(t_i)|$, let $\xi_{2^*} = \min\limits_{1\leq i\leq N}|Q(t_i)|$, and let $\xi_2^* = \max\limits_{1\leq i\leq N}|Q(t_i)|$.

Since ε is very small and $\varepsilon \propto o(h)$, for suitable small h, $(D+F)$ is monotone (see [26, 27]). Hence, $(D+F)^{-1}$ exists and $(D+F)^{-1} \geq 0$. Thus from (14), we get

$$||E|| \leq ||(D+F)^{-1}||||T||. \tag{14}$$

Let (i,k)th element of $(D+F)^{-1}$ be $(D+F)^{-1}_{i,k}$, and define

$$||(D+F)^{-1}|| = \max_{1\leq i\leq N-1}\sum_{k=1}^{N-1}(D+F)^{-1}_{i,k}, \text{and } ||T(h)|| = \max_{1\leq i\leq N-1}|T(h)|. \tag{15}$$

Since $(D+F)^{-1}_{i,k} \geq 0$ and $\sum_{k=1}^{N-1}(D+F)^{-1}_{i,k}\,\bar{S}_k = 1$, for all $i = 1, 2, \ldots, N-1$, hence

$$(D+F)^{-1}_{i,1} \leq \frac{1}{\bar{S}_1} < \frac{1}{h^2\left[\left(\mathcal{A}_2 + 2(\mathcal{A}_1 + \mathcal{A}_4)\right)\xi_{2^*} - 4(\mathcal{A}_1 + \mathcal{A}_4)\psi\xi_1^2\right]}, \tag{16}$$

$$(D+F)^{-1}_{i,N-1} \leq \frac{1}{\bar{S}_N - 1} < \frac{1}{h^2\left[\left(\mathcal{A}_2 + 2(\mathcal{A}_1 + \mathcal{A}_4)\right)\xi_{2^*} - 4(\mathcal{A}_1 + \mathcal{A}_4)\psi\xi_1^2\right]}. \tag{17}$$

Furthermore, for all $i = 2, 3, \ldots, N-2$,

$$\sum_{k=2}^{N-2}(D+F)^{-1}_{i,k} \leq \frac{1}{\min\limits_{2\leq k\leq N-2}\bar{S}_k} < \frac{1}{h^2\left[2\left(\mathcal{A}_2 + \left(\mathcal{A}_1 + \mathcal{A}_4\right)\right)\xi_{2^*}\right]}. \tag{18}$$

By utilizing (15)–(18) and using (14), we have

$$||E|| \leq O(h^4). \tag{19}$$

Hence, the method given in (10) is fourth-order convergent for

$$\mathcal{A}_2 = \mathcal{A}_3 = \frac{1}{12}, \qquad (\mathcal{A}_1 + \mathcal{A}_4) = \frac{5}{12}, \quad \text{and} \quad \eta = -\frac{1}{20\varepsilon}.$$

7 Numerical examples

Three examples with internal layer behavior are examined in this part to explain the concept computationally. These problems were chosen because they have received considerable attention in the literature. The numerical rate of convergence is computed using the formula given by Doolan, Miller, and Schilders [3] as $R^N = \frac{\log\left(E^N\right)-\log\left(E^{2N}\right)}{\log(2)}$.

Example 1. [17] Consider

$$\varepsilon w''(t) + 2tw'(t) = 0, \qquad t \in (-1, 1),$$

with $w(-1) = -1$ and $w(1) = 1$.
Thus $w(t) = erf\left(\frac{t}{\sqrt{\varepsilon}}\right)$ is the exact solution to the problem.
Table 1 shows the computed solution's maximum absolute errors (MAEs). Figure 1 graphically depicts the numerical and exact solutions. The error plot in the solution of this example is shown in Figure 4.

Example 2. [21] Consider

$$\varepsilon w''(t) + 2(2t-1)w'(t) - 4w(t) = 0, \qquad t \in (0, 1),$$

with $w(0) = 1$ and $w(1) = 1$. Hence

$$w(t) = -\frac{e^{\frac{1}{2\varepsilon}-\frac{(1-2t)^2}{2\varepsilon}}\left(2e^{\frac{(1-2t)^2}{2\varepsilon}}\sqrt{2\pi}terf\left(\frac{1-2t}{\sqrt{2\varepsilon}}\right) - e^{\frac{(1-2t)^2}{2\varepsilon}}\sqrt{2\pi}erf\left(\frac{1-2t}{\sqrt{2\varepsilon}}\right) - 2\sqrt{\varepsilon}\right)}{e^{\frac{1}{2\varepsilon}}\sqrt{2\pi}erf\left(\frac{1}{\sqrt{2\varepsilon}}\right) + 2\sqrt{\varepsilon}}$$

is the exact solution to the problem.

The MAE of the solution is shown in Tables 2 and 3. Figure 2 shows the layer profile of the numerical and exact solutions. The error plot in the solution of this example is shown in Figure 5.

Example 3. [17] Consider

$$\varepsilon w'' + tw' - w = 0, \qquad -1 \leq t \leq 1,$$

with $w(-1) = 1$ and $w(1) = 2$.
The exact solution is

$$w(t) = \frac{2\sqrt{\varepsilon}\left(t + 3e^{-\frac{t^2-1}{2\varepsilon}}\right) + \left(e^{\frac{1}{2\varepsilon}}\sqrt{2\pi}erf\left(\frac{1}{\sqrt{2\varepsilon}}\right) + 3e^{\frac{1}{2\varepsilon}}\sqrt{2\pi}erf\left(\frac{t}{\sqrt{2\varepsilon}}\right)\right)}{2e^{\frac{1}{2\varepsilon}}\sqrt{2\pi}erf\left(\frac{1}{\sqrt{2\varepsilon}}\right) + 4\sqrt{\varepsilon}}.$$

The MAE is shown in Table 4. Figure 3 shows the layer profile of the numerical and exact solutions. The error plot in the solution of this example is shown in Figure 6.

Table 1: MAE of the solution of Example 1

$\varepsilon\downarrow$	$N=2^5$	$N=2^6$	$N=2^7$	$N=2^8$	$N=2^9$	$N=2^{10}$
present method						
2^{-5}	1.1881(-4)	7.3536(-6)	4.5818(-7)	2.8613(-8)	1.7889(-9)	1.1198(-10)
2^{-6}	4.8395(-4)	2.9414(-5)	1.8231(-6)	1.1455(-7)	7.1558(-9)	4.4732(-10)
2^{-7}	0.0019(-0)	1.1881(-4)	7.3536(-6)	4.5818(-7)	2.8613(-8)	1.7890(-9)
2^{-8}	0.0065(-0)	4.8395(-4)	2.9414(-5)	1.8231(-6)	1.1455(-7)	7.1558(-9)
2^{-9}	0.0203(-0)	0.0019(-0)	1.1881(-4)	7.3536(-6)	4.5818(-7)	2.8613(-8)
2^{-10}	0.0957(-0)	0.0065(-0)	4.8395(-4)	2.9414(-5)	1.8231(-6)	1.1455(-7)
R^N	3.8757	3.7514	4.0403	4.0120	3.9923	
Results in [17]						
2^{-5}	3.3962(-1)	4.9291(-1)	6.1199(-1)	7.0727(-1)	7.8166(-1)	8.3818(-1)
2^{-6}	2.6635(-1)	4.2332(-1)	5.5588(-1)	6.6268(-1)	7.4724(-1)	8.1197(-1)
2^{-7}	1.7078(-1)	3.3962(-1)	4.9291(-1)	6.1199(-1)	7.0727(-1)	7.8166(-1)
2^{-8}	9.5121(-2)	2.6635(-1)	4.2332(-1)	5.5588(-1)	6.6268(-1)	7.4724(-1)
2^{-9}	2.9280(-2)	1.7078(-1)	3.3962(-1)	4.9291(-1)	6.1199(-1)	7.0727(-1)
2^{-10}	1.1065(-1)	9.5121(-2)	2.6635(-1)	4.2332(-1)	5.5588(-1)	6.6268(-1)

Total elapsed time with maximum number of subintervals 2^{10} is 0.086497 sec.

Table 2: MAE of the solution of Example 2

$\varepsilon\downarrow$	$N=2^5$	$N=2^6$	$N=2^7$	$N=2^8$	$N=2^9$	$N=2^{10}$
present method						
2^{-5}	8.6799(-6)	5.4123(-7)	3.3837(-8)	2.1151(-9)	1.3232(-10)	8.7138(-12)
2^{-6}	2.4578(-5)	1.5331(-6)	9.5748(-8)	5.9831(-9)	3.7396(-10)	2.3510(-11)
2^{-7}	6.7327(-5)	4.3399(-6)	2.7061(-7)	1.6919(-8)	1.0576(-9)	6.6184(-11)
2^{-8}	1.9559(-4)	1.2289(-5)	7.6654(-7)	4.7874(-8)	2.9916(-9)	1.8699(-10)
2^{-9}	5.6901(-4)	3.3663(-5)	2.1700(-6)	1.3531(-7)	8.4593(-9)	5.2878(-10)
2^{-10}	0.0015(-0)	9.7794(-5)	6.1444(-6)	3.8327(-7)	2.3937(-8)	1.4958(-9)
R^{-N}	3.9490	3.9924	4.0028	4.0010	4.0003	
Results in [18]						
2^{-5}	5.9701(-3)	3.3654(-3)	1.7391(-3)	8.7449(-4)	4.3697(-4)	2.1822(-4)
2^{-6}	5.3525(-3)	3.2322(-3)	1.7219(-3)	8.7336(-4)	4.3719(-4)	2.1834(-4)
2^{-7}	1.1177(-2)	2.9851(-3)	1.6827(-3)	8.6953(-4)	4.3725(-4)	2.1848(-4)
2^{-8}	2.5867(-2)	2.6763(-3)	1.6161(-3)	8.6093(-4)	4.3668(-4)	2.1860(-4)
2^{-9}	4.7842(-2)	5.5886(-3)	1.4925(-3)	8.4134(-4)	4.3477(-4)	2.1862(-4)
2^{-10}	7.5829(-2)	1.2934(-2)	1.3381(-3)	8.0805(-4)	4.3046(-4)	2.1834(-4)

Total elapsed time with maximum number of subintervals 2^{10} is 0.089720 sec.

Table 3: MAE of the solution of Example 2

$\varepsilon\downarrow$	$N=2^5$	$N=2^6$	$N=2^7$	$N=2^8$	$N=2^9$	$N=2^{10}$
Present method						
2^{-6}	2.4578(-5)	1.5331(-6)	9.5748(-8)	5.9831(-9)	3.7396(-10)	2.3510(-11)
2^{-8}	1.9559(-4)	1.2289(-5)	7.6654(-7)	4.7874(-8)	2.9916(-9)	1.8699(-10)
2^{-10}	0.0015(-3)	9.7794(-5)	6.1444(-6)	3.8327(-7)	2.3937(-8)	1.4958(-9)
2^{-12}	4.4958(-3)	7.5528(-4)	4.8897(-5)	3.0722(-6)	1.9163(-7)	1.1969(-8)
2^{-14}	3.4263(-3)	1.7659(-3)	3.7764(-4)	2.4449(-5)	1.5361(-6)	9.5817(-8)
R^N	0.9562	2.2253	3.9492	3.9924	4.0028	
Results in [21]						
2^{-6}	3.630(- 3)	1.475(-3)	5.047(- 4)	1.508(- 4)	4.146(- 4)	1.089(- 4)
2^{-8}	3.987(- 3)	1.952(- 3)	9.168(- 4)	3.733(- 4)	1.272(- 4)	3.789(- 5)
2^{-10}	7.147(- 3)	1.979(- 3)	9.814(- 4)	4.866(- 4)	2.297(- 4)	9.359(- 4)
2^{-12}	5.562(- 3)	3.583(- 3)	9.837(- 4)	4.898(- 4)	2.444(- 4)	1.215(- 4)
2^{-14}	4.045(- 3)	2.778(- 3)	1.794(- 3)	4.908(- 4)	2.446(- 4)	1.222(- 4)

Table 4: MAE of the solution of Example 3

$\varepsilon\downarrow$	$N=2^5$	$N=2^6$	$N=2^7$	$N=2^8$	$N=2^9$	$N=2^{10}$
present method						
2^{-5}	1.3020(-5)	8.1184(-7)	5.0756(-8)	3.1727(-9)	1.9854(-10)	1.3311(-11)
2^{-6}	3.6866(-5)	2.2996(-6)	1.4362(-7)	8.9747(-9)	5.6092(-10)	3.5138(-11)
2^{-7}	1.0099(-4)	6.5099(-6)	4.0592(-7)	2.5378(-8)	1.5863(-09)	9.9321(-11)
2^{-8}	2.9338(-4)	1.8433(-5)	1.1498(-6)	7.1811(-8)	4.4873(-09)	2.8046(-10)
2^{-9}	8.5352(-4)	5.0495(-5)	3.2550(-6)	2.0296(-7)	1.2689(-08)	7.9318(-10)
2^{-10}	2.2657(-3)	1.4669(-4)	9.2166(-6)	5.7490(-7)	3.5906(-08)	2.2437(-09)
R^N	3.9491	3.9924	4.0029	4.0010	4.0003	
Results in [18]						
2^{-5}	1.0203(-2)	5.5017(-3)	2.7453(-3)	1.3449(-3)	6.6344(-4)	3.2928(-4)
2^{-6}	9.1104(-3)	5.3503(-3)	2.7665(-3)	1.3581(-3)	6.6733(-4)	3.3030(-4)
2^{-7}	7.9110(-3)	5.1014(-3)	2.7508(-3)	1.3726(-3)	6.7246(-4)	3.3172(-4)
2^{-8}	2.2330(-2)	4.5552(-3)	2.6751(-3)	1.3832(-3)	6.7905(-4)	3.3366(-4)
2^{-9}	4.6794(-2)	3.9555(-3)	2.5507(-3)	1.3754(-3)	6.8632(-4)	3.3623(-4)
2^{-10}	7.7601(-2)	1.1165(-2)	2.2776(-3)	1.3376(-3)	6.9162(-4)	3.3953(-4)
Results in [17]						
2^{-5}	1.0111(-1)	1.3778(-1)	1.6169(-1)	1.7746(-1)	1.8802(-1)	1.9520(-1)
2^{-6}	5.3820(-2)	8.5843(-2)	1.0677(-1)	1.2048(-1)	1.2958(-1)	1.3573(-1)
2^{-7}	2.2863(-2)	5.0556(-2)	6.8889(-2)	8.0847(-2)	8.8730(-2)	9.4011(-2)
2^{-8}	5.6850(-3)	2.6910(-2)	4.2921(-2)	5.3386(-2)	6.0238(-2)	6.4792(-2)
2^{-9}	2.0288(-2)	1.1431(-2)	2.5278(-2)	3.4445(-2)	4.0424(-2)	4.4365(-2)
2^{-10}	2.9876(-2)	2.8425(-3)	1.3455(-2)	2.1461(-2)	2.6693(-2)	3.0119(-2)

Total elapsed time with maximum number of subintervals 2^{10} is 0.090109 sec.

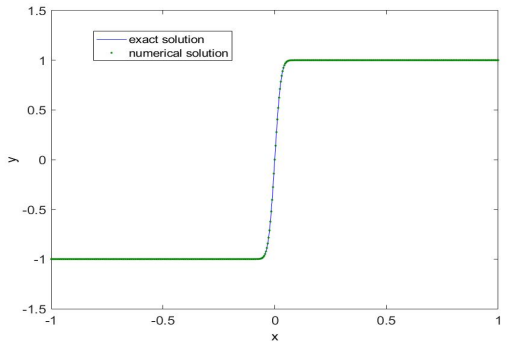


Figure 1: Numerical and exact solution of Example 1 for $\varepsilon = 2^{-10},\ h = 2^{-7}$.

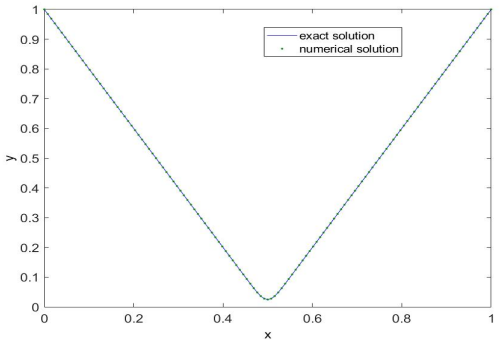


Figure 2: Numerical and exact solution of Example 2 for $\varepsilon = 2^{-10},\ h = 2^{-7}$.

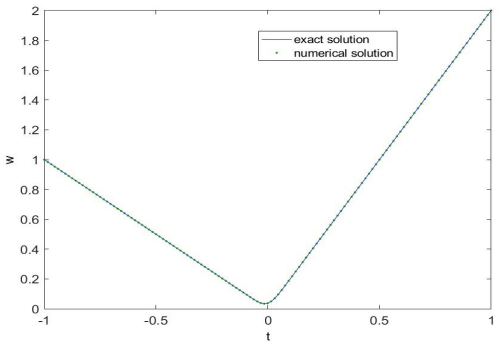


Figure 3: Numerical and exact solution of Example 3 for $\varepsilon = 2^{-10},\ h = 2^{-7}$.

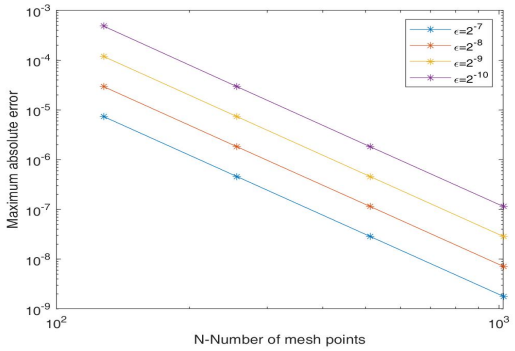


Figure 4: Log-Log scale for Example 1

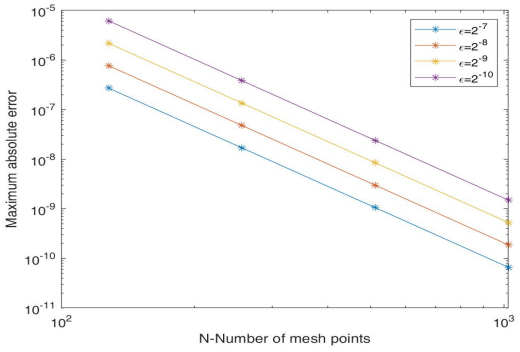


Figure 5: Log-Log scale for Example 2

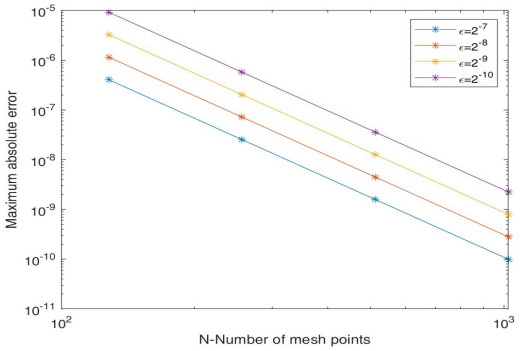


Figure 6: Log-Log scale for Example 3

8 Conclusion

In this paper, for solving singularly perturbed two-point boundary value problems with an interior layer, a higher-order finite difference approach was proposed. To derive the discretization equation, an adaptive cubic spline approach has been extended for a singularly perturbed boundary value problem with interior layers. It was produced by substituting nonstandard finite differences in the first-order derivatives of $w(t)$. The tridiagonal solver discrete consistent embedding was used to solve the discretization equation. Convergence was evaluated in the proposed method. To demonstrate the method, numerical problems have been solved, when ε is either small or large as compared to the mesh size h. To justify the method, numerical results were compared with the results taken by the methods given in [17, 18, 21]. For the layer behavior, the solutions were graphically displayed, and we discovered that the numerical solution closely matches the exact solution. This technique is simple to compute and requires little computational effort.

References

1. Bender, C.M. and Orszag, S.A. *Advanced mathematical methods for scientists and engineers*, International Series in Pure and Applied Mathematics. McGraw-Hill Book Co., New York, 1978.

2. Brauner, C.M., Gay, B.B. and Mathieu, J. *Singular perturbations and boundary layer theory*, Lecture Notes in Mathematics, Vol. 594. Springer-Verlag, Berlin-New York, 1977.

3. Doolan, E.P., Miller, J.J.H. and Schilders, W.H.A. *Uniform numerical methods for problems with initial and boundary layers*, Boole Press, Dún Laoghaire, 1980.

4. Farrel, P.A., Hegarty, A.F., Miller, J.J.H., O'Riordan,E. and Shiskin, G.I. *Singular perturbed convection-diffusion problems with boundary and weak interior layers*, J. Comput. Appl. Math. 166 (2004) 133–151.

5. Gartland, Jr. E.C. *Uniform high-order difference schemes for a singularly perturbed two point boundary value problem*, Math. Comput. 48 (178) (1987) 551–564.

6. Geng, F.Z., Qian, S.P. and Li, S. *A numerical method for singularly perturbed turning point problems with an interior layer*, J. Comput. Appl. Math. 225 (2014) 97–105.

7. Jain, M.K. *Numerical solution of differential equations, finite difference and finite element methods*, 4th Edition, New Age International Publishers, 2018.

8. Kadalbajoo, M.K. and Reddy, Y.N. *Asymptotic and numerical analysis of singular perturbation problems*, Appl. Math. Comput. 30 (1989) 223–259.

9. Kadalbajoo, M.K. and Patidar, K.C. *A survey of numerical techniques for solving singularly perturbed ordinary differential equations*, Appl. Math. Comput. 30 (2002) 457–510.

10. Khuri, S.A. and Sayfy, A. *Numerical solutions for the nonlinear Emden-Fowler type equations by a fourth-order adaptive method*, Int. J. Comput. Methods. 11 (1) (2014), 1350052–1350072.

11. Lin, P. *A class of variational difference schemes for a singular perturbation problem*, Appl. Math. Mech. 10 (4) (1989) 353–359.

12. Miller, J.J.H., O'Riordan, E. and 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 Publishing Co., Inc., River Edge, NJ, 1996.

13. Natesan, S., Vigo-Aguiar, J. and Ramanujam, N. *A numerical algorithm for singular perturbation problems exhibiting weak boundary layers*, Comput. Math. Appl. 45 (2003) 469–479.

14. Navnit, Jha. *Computational method for nonlinear singularly perturbed singular boundary value problems using nonpolynomial spline*, J. Inf. Comput. Sci. 7(2) (2012) 019–096.

15. Navnit, Jha. *Nonpolynomial spline finite difference scheme for nonlinear singular boundary value problems with singular perturbation and its mechanization*, Discrete Contin. Dyn. Syst. 2013, Dynamical systems, differential equations and applications. 9th AIMS Conference. Suppl., 355–363.

16. OMalley, R.E. *Singular perturbation methods for ordinary differential equations*, Applied Mathematical Sciences, 89. Springer-Verlag, New York, 1991.

17. Phaneendra, K., Reddy, Y.N. and Soujanya, G.B.S.L. *noniterative numerical integration method for singular perturbation problems exhibiting internal and twin boundary layers*, Int. J. Appl. Math. Comput. 3 (2011) 9–20.

18. Phaneendra, K. and Lalu, M. *Gaussian quadrature for two-point singularly perturbed boundary value problems with exponential fitting* Comm. App. Math. Comp. Sci. 10 3(2019) 447–467.

19. Pradip, R. *A fourth-order non-uniform mesh optimal B-spline collocation method for solving a strongly nonlinear singular boundary value problem describing electrohydrodynamic flow of a fluid*, Appl. Numer. Math. 153 (2020) 558–574.

20. Pradip, R. and Prasad Goura, V.M.K. *B-spline collocation methods and their convergence for a class of nonlinear derivative dependent singular boundary value problems*, Appl. Math. Comput. 341(2019) 428–450.

21. Rai, P. and Shama, K.K. *Numerical method for singularly perturbed differential-difference equations with turning point*, Int. J. Pure Appl. Math. 73(4) (2011) 451–470.

22. Ramos, J.I. *A smooth locally-analytical technique for singularly perturbed two-point boundary-value problems*, Appl. Math. Comput. 163 (2005), 1123–1142.

23. Rashidinia, J. *Applications of spline to numerical solution of differential equations*, M.phil dissertation, A.M.U.India, 1990.

24. Roos, H.G., Stynes, M. and Tobiska, L. *Robust numerical methods for singularly perturbed differential equations. Convection-diffusion-reaction and flow problems,* Second edition. Springer Series in Computational Mathematics, 24. Springer-Verlag, Berlin, 2008.

25. Smith, D.R. *Singular Perturbation Theory – An Introduction with applications*, Cambridge University Press, Cambridge , 1985.

26. Varga, R.S. *Matrix iterative analysis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1962.

27. Young, D.M. *Iterative solutions of large linear systems.* Academic press, New York, 1971.

How to cite this article
E. Srinivas, M. Lalu and K. Phaneendra A numerical approach for singular perturbation problems with an interior layer using an adaptive spline. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 355-370. doi: 10.22067/ijnao.2021.73813.1076.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 371–396
DOI: 10.22067/ijnao.2022.71857.1052
https://ijnao.um.ac.ir/

Image denoising via a new hybrid TGV model based on Shannon interpolation

E. Tavakkol, S.M. Hosseini* and A. Hosseini

Abstract

A new hybrid variational model is presented for image denoising, which incorporates the merits of Shannon interpolation, total generalized variation (TGV) regularization, and a symmetrized derivative regularization term based on l^1-norm. In this model, the regularization term is a combination of a TGV functional and the symmetrized derivative regularization term, while the data fidelity term is characterized by the l^2-norm. Unlike most variational models that are discretized using a finite-difference scheme, our approach in structure is based on Shannon interpolation. Quantitative and qualitative assessments of the new model indicate its effectiveness in restoration accuracy and staircase effect suppression. Numerical experiments are carried out using the primal-dual algorithm. Numerous real-world examples are conducted to confirm that the newly proposed method outperforms several current state-of-the-art numerical methods in terms of the peak signal to noise ratio and the structural similarity (SSIM) index.

AMS subject classifications (2020): 68Q25, 68R10, 68U05.

Keywords: Variational model; Total generalized variation regularization; Staircasing effect; Primal-dual algorithm.

* Corresponding author

Received 6 August 2021; revised 8 January 2022; accepted 16 January 2022

Elaheh Tavakkol
Department of Applied Mathematics, Tarbiat Modares University, P.O. Box 14115-175, Tehran, Iran. e-mail: e.tavakkol@modares.ac.ir

Seyed Mohammad Hosseini
Department of Applied Mathematics, Tarbiat Modares University, P.O. Box 14115-175, Tehran, Iran. e-mail: hossei_m@modares.ac.ir

Alireza Hosseini
School of Mathematics, Statistics and Computer Science, College of Science, University of Tehran, P. O. Box 14155-6455, Tehran, Iran. e-mail: hosseini.alireza@ut.ac.ir

1 Introduction

Image processing is an extensively utilized and rapidly growing area in computer science enriched with various applications, including image restoration [4], medical visualization [7], industrial inspection [27], law enforcement [50], and so on. Recently, various techniques have been proposed in image processing, which aims to recover the underlying true image from its degraded observation, including image denoising models [19, 49, 40, 47], image deblurring models [22, 17], image inpainting models [38, 15], image deconvolution models [28, 2], and so on. Image denoising is one of the most fundamental problems in image processing and computer vision. In the past few decades, many image denoising models have been introduced in the literature, such as variational models [39, 45], wavelet transform based models [32], nonlocal means filters [48], curvelet transform based models [29], bilateral filters [3], and so on. Variational models have been widely used in image denoising due to their capability in image feature preservation [33, 21, 43, 42, 44]. TV regularization proposed by Rudin, Osher, and Fatemi [30] (the ROF model) is a successful example of variational models. Many different papers have reported the efficiency of TV regularization [37, 13, 8, 9, 36, 18, 46]. Assume that $\Gamma \subset \mathbb{R}^d$ is a continuous domain and that $s : \Gamma \to \mathbb{R}$ is a d-dimensional image in continuous domain Γ. The continuous TV semi-norm of s is defined as

$$\mathrm{TV}(s) = \sup \left\{ \int_\Gamma s \ \mathrm{div} v \ d\mathbf{x} \ \middle| \ v \in C_c^1(\Gamma, \mathbb{R}^d), \ ||v||_\infty \leq 1 \right\} = \int_\Gamma |Ds| \ d\mathbf{x},$$

where $|\cdot|$ is the Euclidean norm and $|Ds|$ denotes the variation-measure of the distributional derivative Ds, which is a vector-valued Radon measure. For smooth function s, we have $Ds = \nabla s$. Therefore, TV is the integral of its gradient magnitude:

$$\mathrm{TV}(s) = \int_\Gamma |\nabla s| \ d\mathbf{x}.$$

In TV regularization, the space of functions of bounded variation (BV) is considered as an appropriate class for image restoration. The TV semi-norm defines the norm $||s||_{\mathrm{BV}} := ||s||_{L^1} + \mathrm{TV}(s)$ on the space of BV functions. Although TV regularization-based techniques have become one of the most widely used regularizers due to their edge-preserving ability, they leave some form of staircase artefacts in the smooth regions of the image. To alleviate the staircase effect, several high-order methods have been proposed in the literature [12, 34, 14, 35, 25]. One technique to overcome this drawback is total generalized variation (TGV) regularization [5]. In most algorithms, the TV and TGV regularizations are discretized using a finite-differences scheme. For this reason, denoised images produced by these approaches cannot be efficiently interpolated using standard interpolation models. Several types of

research have been conducted as a remedy to this drawback [24, 26]. Shannon interpolation has been recently utilized in [1] for total variation-based image restoration leading to image improvements in terms of restoration accuracy.

In the present work, a new hybrid total generalized variation model is studied for image denoising in which the regularization term is considered to be a combination of TGV regularization and a symmetrized derivative regularization term based on l^1-norm, and the data fidelity term is characterized by the l^2-norm. Furthermore, in our approach, Shannon interpolation is used rather than a finite-differences scheme yielding efficient images in terms of restoration accuracy. Quantitative and qualitative evaluations indicate that the proposed method significantly outperforms several current state-of-the-art methods in staircase effect suppression and restoration accuracy. We employ the primal-dual algorithm presented in [11] to provide a global minimizer to the new proposed variational model.

The outline of the paper is organized as follows. In Section 2, we present the necessary definitions and basic properties of TGV model and Shannon interpolation. In Section 3, our proposed method and some necessary analytical discussions are presented. Section 4 contains the numerical algorithm. In Section 5, we present the convergence analysis for the proposed model. Section 4 is devoted to numerical experiments. Finally, Section 7 contains some concluding remarks.

2 Preliminaries

In this section, we first introduce some notations used throughout this paper, and then we briefly review the concepts of TGV regularization [5] and Shannon interpolation [1].

2.1 Notations

1. $\Omega = I_{N_1} \times I_{N_2}$ is a discrete domain of size $N_1 \times N_2$, where

$$I_{N_1} = \{0, 1, 2, \ldots, N_1 - 1\}, \quad I_{N_2} = \{0, 1, 2, \ldots, N_2 - 1\}.$$

2. $u \in \mathbb{R}^{\Omega}$ is a discrete gray-level image with size $N_1 \times N_2$.
3. For $d \geq 1$, $\Gamma \subset \mathbb{R}^d$ is a continuous domain.
4. $L^1_{loc}(\Gamma)$ is the set of locally Lipschitz integrable functions $s : \Gamma \to \mathbb{R}$.
5. λ_0 and λ_1 are fixed positive parameters.
6. $\mathrm{Sym}^2(\mathbb{R}^d)$ is the vector space of symmetric 2-tensors defined by

$$\mathrm{Sym}^2(\mathbb{R}^d) = \Big\{ \eta : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R} \;\Big|\; \eta \textit{ is 2-linear and symmetric} \Big\}.$$

7. $C_c^2(\Gamma, \mathrm{Sym}^2(\mathbb{R}^d))$ is the space of 2-times continuous differentiable functions with compact support from Γ to $\mathrm{Sym}^2(\mathbb{R}^d)$.
8. $C_c^2([0, N_1] \times [0, N_2], \mathrm{Sym}^2(\mathbb{R}^2))$ is the space of 2-times continuously differentiable functions with compact support from $[0, N_1] \times [0, N_2]$ to $\mathrm{Sym}^2(\mathbb{R}^2)$.

2.2 TGV regularization and Shannon interpolation

For $s \in L^1_{loc}(\Gamma)$, the TGV functional of order 2 with weight λ is defined as [5]

$$\mathrm{TGV}^{2,\lambda}(s) = \sup \Big\{ \int_\Gamma s \ \mathrm{div}^2 \upsilon \ d\mathbf{x} \ \Big| \ \upsilon \in C_c^2(\Gamma, \mathrm{Sym}^2(\mathbb{R}^d)), \ ||\upsilon||_\infty \le \lambda_0, \ ||\mathrm{div}\upsilon||_\infty \le \lambda_1 \Big\}, \quad (1)$$

where $\lambda = (\lambda_0, \lambda_1)$, $(\mathrm{div}\upsilon)_i = \sum_{j=1}^{d} \frac{\partial \upsilon_{ij}}{\partial x_j}$, and $\mathrm{div}^2\upsilon = \sum_{i=1}^{d} \frac{\partial^2 \upsilon_{ii}}{\partial x_i^2} + \sum_{i<j} 2\frac{\partial^2 \upsilon_{ij}}{\partial x_i \partial x_j}$. Moreover, the infinity norms of υ and the vector field w, where $w = \mathrm{div}\upsilon$, are defined as

$$||\upsilon||_\infty = \sup \Big\{ \Big(\sum_{i=1}^{d} |\upsilon_{ii}(\mathbf{x})|^2 + 2 \sum_{i<j} |\upsilon_{ij}(\mathbf{x})|^2 \Big)^{1/2} \ \Big| \ \mathbf{x} \in \Gamma \Big\},$$

$$||w||_\infty = \sup \Big\{ \Big(\sum_{i=1}^{d} |w_i(\mathbf{x})|^2 \Big)^{1/2} \ \Big| \ \mathbf{x} \in \Gamma \Big\}.$$

In second-order TGV regularization model, the space of bounded generalized variation functions of order 2, defined as

$$\mathrm{BGV}^{2,\lambda}(\Gamma) = \Big\{ s \in L^1(\Gamma) \ \Big| \ \mathrm{TGV}^{2,\lambda}(s) < \infty \Big\},$$

is considered as an appropriate class for image restoration. The Fenchel dual formulation of (2) for $d = 2$ and $u \in \mathbb{R}^{N_1 \times N_2}$ can be written as [6, 20]

$$\mathrm{TGV}^{2,\lambda}(u) = \min_{p \in (\mathbb{R}^2)^{N_1 \times N_2}} \lambda_0 ||\xi(p)||_1 + \lambda_1 ||\nabla u - p||_1, \quad (2)$$

where ∇u and $\xi(p)$ stand for the gradient and symmetrized derivative operators, respectively, which can be formulated as

$$\nabla u = [\partial_x^+ u \ \ \partial_y^+ u]^T, \quad \xi(p) = \begin{bmatrix} \partial_x^- p_1 & \frac{1}{2}(\partial_x^- p_2 + \partial_y^- p_1) \\ \frac{1}{2}(\partial_x^- p_2 + \partial_y^- p_1) & \partial_y^- p_2 \end{bmatrix},$$

where $(\partial_x^+ u)(i,j) = u(i+1,j) - u(i,j)$, $(\partial_y^+ u)(i,j) = u(i,j+1) - u(i,j)$,

$(\partial_x^- p_1)(i,j) = p_1(i,j) - p_1(i-1,j)$, and $(\partial_y^- p_1)(i,j) = p_1(i,j) - p_1(i,j-1)$ ($(\partial_x^- p_2)(i,j)$ and $(\partial_y^- p_2)(i,j)$ can be formulated similarly) are discrete first-order derivatives [5].

The Shannon interpolate $U : \mathbb{R}^2 \to \mathbb{R}$ of the discrete image u is defined as [1]

$$U(x,y) = \frac{1}{N_1 N_2} \sum_{\alpha=-\frac{N_1}{2}}^{\frac{N_1}{2}} \sum_{\beta=-\frac{N_2}{2}}^{\frac{N_2}{2}} \varepsilon_{N_1}(\alpha)\ \varepsilon_{N_2}(\beta)\ \widehat{u}(\alpha,\beta)\ e^{2\pi i(\frac{\alpha x}{N_1}+\frac{\beta y}{N_2})}, \quad (3)$$

where $\widehat{u}(\alpha,\beta) = \sum_{k=0}^{N_1-1} \sum_{l=0}^{N_2-1} u(k,l)\ e^{-2\pi i(\frac{\alpha k}{N_1}+\frac{\beta l}{N_2})}$ is the 2-dimensional discrete Fourier transform (DFT) of u, and $\varepsilon_{N_1}(\alpha)$ and $\varepsilon_{N_2}(\beta)$ are the weighting terms [1] formulated as

$$\varepsilon_{N_1}(\alpha) = \begin{cases} 1/2 & \text{if } |\alpha| = \frac{N_1}{2}, \\ 1 & \text{otherwise,} \end{cases} \qquad \varepsilon_{N_2}(\beta) = \begin{cases} 1/2 & \text{if } |\beta| = \frac{N_2}{2}, \\ 1 & \text{otherwise.} \end{cases}$$

3 Proposed model

Let $U : \mathbb{R}^2 \to \mathbb{R}$ be the Shannon interpolate of $u \in \mathbb{R}^\Omega$ defined in (7). For the case $d = 2$ in (2), we define the second-order Shannon interpolation based TGV regularization with weight λ as

$$\mathrm{STGV}^{2,\lambda}_\infty(u) = \sup \Big\{ \int_{[0,N_1]\times[0,N_2]} U(x,y)\ \mathrm{div}^2 V(x,y)\ dxdy \ \Big|\ V \in C_c^2([0,N_1]\times[0,N_2], \mathrm{Sym}^2(\mathbb{R}^2)),\ ||V||_\infty \le \lambda_0,\ ||\mathrm{div}V||_\infty \le \lambda_1 \Big\}, \quad (4)$$

where $\lambda = (\lambda_0, \lambda_1)$. The integral in (4) is generally difficult to compute but it can be easily approximated using the Riemann sum as

$$\mathrm{STGV}^{2,\lambda}_n(u) = \max \Big\{ \frac{1}{n^2} \langle U, \mathrm{div}_n^2 \nu \rangle_{\mathbb{R}^{N_1\times N_2}} \ \Big|\ \nu \in (\mathbb{R}^4)^{N_1\times N_2}, ||\nu||_\infty \le \lambda_0, ||\mathrm{div}_n \nu||_\infty \le \lambda_1 \Big\}, \quad (5)$$

where $n \ge 2$ in practice, div_n^2 is the discrete second-order divergence operator corresponding to the continuous operator div^2, and

$$\frac{1}{n^2} \langle U, \mathrm{div}_n^2 \nu \rangle_{\mathbb{R}^{N_1\times N_2}} = \frac{1}{n^2} \sum_{(k,l)\in\Omega_n} U(\frac{k}{n}, \frac{l}{n})\ \mathrm{div}_n^2 \nu(k,l),$$

in which $\mathrm{div}_n^2\nu(k,l) = \mathrm{div}^2 V(\frac{k}{n}, \frac{l}{n})$ and

$$\Omega_n = I_{nN_1} \times I_{nN_2} = \{0,1,2,\ldots,nN_1-1\} \times \{0,1,2,\ldots,nN_2-1\}.$$

In order to compute $\mathrm{STGV}_n^{2,\lambda}(u)$, we need to obtain its Fenchel dual formulation. For this purpose, we first introduce the discrete gradient operator ∇_n, the discrete symmetrized gradient operator ξ_n, the discrete first-order divergence operator div_n, and the discrete second-order divergence operator div_n^2 in the following. To distinguish between the two cases $\mathrm{div}_n p$ and $\mathrm{div}_n \nu$, where $p \in (\mathbb{R}^2)^{nN_1 \times nN_2}$ and $\nu \in (\mathbb{R}^4)^{N_1 \times N_2}$, we use the notations $\nabla_n^* p := -\mathrm{div}_{1n} p$ and $\xi_n^* \nu := -\mathrm{div}_{2n} \nu$ ($*$ denotes the adjoint operator). As a result, div_n^2 in (3) will be denoted as $\mathrm{div}_{1n}(\mathrm{div}_{2n}(\nu))$ from now on.

Let $U : \mathbb{R}^2 \to \mathbb{R}$ be the Shannon interpolate of $u \in \mathbb{R}^\Omega$ defined in (7). The continuous gradient operator ∇U is formulated as $\nabla U(x,y) = [D_x(U(x,y)) \;\; D_y(U(x,y))]^T$ [5, 1] in which

$$D_x(U(x,y)) = \frac{2\pi i}{N_1 N_2} \sum_{\alpha=-\frac{N_1}{2}}^{\frac{N_1}{2}} \sum_{\beta=-\frac{N_2}{2}}^{\frac{N_2}{2}} \frac{\alpha}{N_1}\, \varepsilon_{N_1}(\alpha)\, \varepsilon_{N_2}(\beta)\, \widehat{u}(\alpha,\beta)\, e^{2\pi i(\frac{\alpha x}{N_1} + \frac{\beta y}{N_2})}, \tag{6}$$

$$D_y(U(x,y)) = \frac{2\pi i}{N_1 N_2} \sum_{\alpha=-\frac{N_1}{2}}^{\frac{N_1}{2}} \sum_{\beta=-\frac{N_2}{2}}^{\frac{N_2}{2}} \frac{\beta}{N_2}\, \varepsilon_{N_1}(\alpha)\, \varepsilon_{N_2}(\beta)\, \widehat{u}(\alpha,\beta)\, e^{2\pi i(\frac{\alpha x}{N_1} + \frac{\beta y}{N_2})}. \tag{7}$$

For $(k,l) \in \Omega_n$, the discrete gradient operator $\nabla_n : \mathbb{R}^{N_1 \times N_2} \longrightarrow (\mathbb{R}^2)^{nN_1 \times nN_2}$ can be derived as [1]

$$\nabla_n u(k,l) = \nabla U(\frac{k}{n}, \frac{l}{n}). \tag{8}$$

For $p = [p_1 \;\; p_2]^T \in (\mathbb{R}^2)^{nN_1 \times nN_2}$ and $(k,l) \in \Omega_n$, the discrete first-order divergence operator $\mathrm{div}_{1n} : (\mathbb{R}^2)^{nN_1 \times nN_2} \longrightarrow \mathbb{R}^{N_1 \times N_2}$ is defined as

$$\mathrm{div}_{1n}(p(k,l)) = \mathrm{div}_{1n_1}(p(k,l)) + \mathrm{div}_{1n_2}(p(k,l)) \tag{9}$$

[5], where $\widehat{\mathrm{div}_{1n_1}(p)}(\alpha,\beta) = \widehat{D_x^*(p_1)}(\alpha,\beta)$ and $\widehat{\mathrm{div}_{1n_2}(p)}(\alpha,\beta) = \widehat{D_y^*(p_2)}(\alpha,\beta)$, for $(\alpha,\beta) \in \widehat{\Omega} = \widehat{I_{N_1}} \times \widehat{I_{N_2}}$, are formulated as [1]

$$\widehat{D_x^*(p_1)}(\alpha,\beta)=2\pi i\frac{\alpha}{N_1}\begin{cases}\widehat{p_1}(\alpha,\beta) & \text{if } |\alpha|<\frac{N_1}{2},|\beta|<\frac{N_2}{2},\\ \frac{1}{2}(\widehat{p_1}(\alpha,\beta)-\widehat{p_1}(-\alpha,\beta)) & \text{if } \alpha=-\frac{N_1}{2},|\beta|<\frac{N_2}{2},\\ \frac{1}{2}(\widehat{p_1}(\alpha,\beta)+\widehat{p_1}(\alpha,-\beta)) & \text{if } |\alpha|<\frac{N_1}{2},\beta=-\frac{N_2}{2},\\ \frac{1}{4}\sum\limits_{s_1=\pm1}\sum\limits_{s_2=\pm1}s_1\widehat{p_1}(s_1\alpha,s_2\beta) & \text{if } (\alpha,\beta)=(-\frac{N_1}{2},-\frac{N_2}{2}),\end{cases} \tag{10}$$

$$\widehat{D_y^*(p_2)}(\alpha,\beta)=2\pi i\frac{\beta}{N_2}\begin{cases}\widehat{p_2}(\alpha,\beta) & \text{if } |\alpha|<\frac{N_1}{2},|\beta|<\frac{N_2}{2},\\ \frac{1}{2}(\widehat{p_2}(\alpha,\beta)+\widehat{p_2}(-\alpha,\beta)) & \text{if } \alpha=-\frac{N_1}{2},\ |\beta|<\frac{N_2}{2},\\ \frac{1}{2}(\widehat{p_2}(\alpha,\beta)-\widehat{p_2}(\alpha,-\beta)) & \text{if } |\alpha|<\frac{N_1}{2},\beta=-\frac{N_2}{2},\\ \frac{1}{4}\sum\limits_{s_1=\pm1}\sum\limits_{s_2=\pm1}s_2\widehat{p_2}(s_1\alpha,s_2\beta) & \text{if } (\alpha,\beta)=(-\frac{N_1}{2},-\frac{N_2}{2}).\end{cases} \tag{11}$$

For $V=\begin{bmatrix}V_{11} & V_{12}\\ V_{12} & V_{22}\end{bmatrix}\in C_c^2([0,N_1]\times[0,N_2],\mathrm{Sym}^2(\mathbb{R}^2))$, the first-order continuous divergence operator div_2 can be written as [5]

$$\mathrm{div}_2V(x,y)=\begin{bmatrix}D_x(V_{11}(x,y))+D_y(V_{12}(x,y))\\ D_x(V_{12}(x,y))+D_y(V_{22}(x,y))\end{bmatrix},$$

where D_x and D_y are defined as (6) and (7). Now, the discrete first-order divergence operator $\mathrm{div}_{2n}:(\mathbb{R}^4)^{N_1\times N_2}\longrightarrow(\mathbb{R}^2)^{nN_1\times nN_2}$ can be derived as

$$\mathrm{div}_{2n}\nu(k,l)=\mathrm{div}_2V(\frac{k}{n},\frac{l}{n}), \tag{12}$$

for $\nu\in(\mathbb{R}^4)^{N_1\times N_2}$ and $(k,l)\in\Omega_n$. For $p\in(\mathbb{R}^2)^{nN_1\times nN_2}$ and $(\alpha,\beta)\in\widehat{\Omega}=\widehat{I_{N_1}}\times\widehat{I_{N_2}}$, the symmetrized gradient operator $\xi_n:(\mathbb{R}^2)^{nN_1\times nN_2}\longrightarrow(\mathbb{R}^4)^{N_1\times N_2}$ is given by [5]

$$\widehat{\xi_n(p)}(\alpha,\beta)=\begin{bmatrix}\widehat{D_x^*(p_1)}(\alpha,\beta) & \frac{1}{2}(\widehat{D_y^*(p_1)}(\alpha,\beta)+\widehat{D_x^*(p_2)}(\alpha,\beta))\\ \frac{1}{2}(\widehat{D_y^*(p_1)}(\alpha,\beta)+\widehat{D_x^*(p_2)}(\alpha,\beta)) & \widehat{D_y^*(p_2)}(\alpha,\beta)\end{bmatrix},$$

where $\widehat{D_x^*(p_1)}$, $\widehat{D_x^*(p_2)}$, $\widehat{D_y^*(p_1)}$, and $\widehat{D_y^*(p_2)}$ are defined similar to (10) and (11). For $(k,l)\in\Omega_n$, $\xi_np(k,l)$ is obtained by taking the inverse Fourier transform as

$$\xi_np(k,l)=\begin{bmatrix}\xi_{n_1}p(k,l) & \xi_{n_3}p(k,l)\\ \xi_{n_3}p(k,l) & \xi_{n_2}p(k,l)\end{bmatrix}, \tag{13}$$

where $\xi_{n_1}p(k,l)$ is achieved as

$$\xi_{n_1}p(k,l) = \frac{1}{N_1N_2}\sum_{\alpha=-\frac{N_1}{2}}^{\frac{N_1}{2}}\sum_{\beta=-\frac{N_2}{2}}^{\frac{N_2}{2}}\widehat{D_x^*(p_1)}(\alpha,\beta)\ e^{2\pi i(\frac{\alpha k}{N_1}+\frac{\beta l}{N_2})},$$

and $\xi_{n_2}p(k,l)$, $\xi_{n_3}p(k,l)$ can be formulated similarly.

The discrete functional $\mathrm{STGV}_n^{2,\lambda}$ in (3) is equivalent to Fenchel dual formulation

$$\mathrm{STGV}_n^{2,\lambda}(u) = \min_{p\in(\mathbb{R}^2)^{nN_1\times nN_2}} \frac{\lambda_0}{n^2}||\xi_n(p)||_1 + \frac{\lambda_1}{n^2}||\nabla_n u - p||_1, \tag{14}$$

where ∇_n and ξ_n are defined as (8) and (13), respectively. The analytical process for establishing this Fenchel dual formulation of (3) is exactly based on the lines of [5] and [20] (except the factor $\frac{1}{n^2}$ in (3) and the change on the operators). For this reason, we do not bring the proof here and we refer the reader to the mentioned references. Now, we concentrate on the new Shannon interpolation based the hybrid TGV (SHTGV) model defined as

$$\mathrm{SHTGV}_n^{2,\lambda}(u) = \mathrm{STGV}_n^{2,\lambda}(u) + \gamma\ ||\xi_n(\nabla_n(u))||_1,$$

where $\gamma > 0$ is a fixed positive parameter, $\mathrm{STGV}_n^{2,\lambda}(u)$ is defined as (14), and $\xi_n(\nabla_n(u))$ can be easily formulated by substituting $\nabla_n(u)$ with p in the procedure of deriving ξ_n in (13). The concerned variational problem is established in the form

$$\min_u\ ||u-u_0||_2^2 + \mathrm{SHTGV}_n^{2,\lambda}(u), \tag{15}$$

where u is the desired restored image and u_0 is the observed data. According to our experience, the fixed value $\gamma = 0.01$ is suitable and does not need to be tuned. Therefore, one can say that no additional parameter is introduced with the new model.

4 Numerical algorithm

We use the primal-dual algorithm (Algorithm 1) presented in [11] to provide a minimizer of the variational problem (9). This primal-dual algorithm is particularly efficient in finding a solution to the saddle-point problem

$$\min_{x\in X}\ \max_{y\in Y}\ \langle Kx,y\rangle_Y + F(x) - G(y), \tag{16}$$

where X and Y are Hilbert spaces, $K : X \longrightarrow Y$ is a linear and continuous mapping, and $F : X \longrightarrow (-\infty,\infty]$ and $G : Y \longrightarrow (-\infty,\infty]$ are proper, convex, and lower semi-continuous functionals. In order to adapt the optimization problem (9) to Algorithm 1, we need the corresponding saddle-point

Algorithm 1 The first-order primal-dual algorithm presented in [11].

1. Set $k = 0$, choose the initial estimates $x^0 \in X$, $y^0 \in Y$, $\bar{x}^0 = x^0$, and positive parameters τ and σ such that $\sigma\tau|||K|||^2 < 1$ ($|||\cdot|||$ denotes the induced l^2-norm).
2. Calculate x^{k+1}, y^{k+1}, and $\bar{x}^{k+1}$ using the following equations:
   $y^{k+1} = (I + \sigma\ \partial G)^{-1}(y^k + \sigma\ K\ \bar{x}^k)$,
   $x^{k+1} = (I + \tau\ \partial F)^{-1}(x^k - \tau\ K^*\ y^{k+1})$,
   $\bar{x}^{k+1} = 2x^{k+1} - x^k$.
3. Stop or set $k = k + 1$ and go back to step 2.

structure of this optimization problem, which is formulated as (see for details)

$$\min_{\substack{u\in\mathbb{R}^{N_1\times N_2}\\ p\in(\mathbb{R}^2)^{nN_1\times nN_2}}} \max_{\substack{w\in(\mathbb{R}^2)^{nN_1\times nN_2}\\ \nu\in(\mathbb{R}^4)^{N_1\times N_2}\\ q\in(\mathbb{R}^4)^{N_1\times N_2}}} ||u-u_0||_2^2 + \frac{\lambda_1}{n^2}\langle\nabla_n u - p, w\rangle + \frac{\lambda_0}{n^2}\langle\xi_n(p),\nu\rangle$$

$$+ \gamma\langle\xi_n(\nabla_n u), q\rangle - I_{\{||\cdot||_\infty\le 1\}}(w) - I_{\{||\cdot||_\infty\le 1\}}(\nu) - I_{\{||\cdot||_\infty\le 1\}}(q). \tag{17}$$

Using the saddle-point structure (17), x, y, the functionals F and G, and the mapping K in saddle-point problem (16) are obtained as

$$x = (u,p), \quad y = (w,\nu,q), \quad F(x) = F(u,p) = ||u-u_0||_2^2,$$

$$G(y) = G(w,\nu,q) = I_{\{||\cdot||_\infty\le 1\}}(w) + I_{\{||\cdot||_\infty\le 1\}}(\nu) + I_{\{||\cdot||_\infty\le 1\}}(q),$$

$$K = \begin{bmatrix} \frac{\lambda_1}{n^2}\nabla_n & -\frac{\lambda_1}{n^2}I \\ 0 & \frac{\lambda_0}{n^2}\xi_n \\ \gamma\xi_n\nabla_n & 0 \end{bmatrix}$$

From [1, Proposition 7], we have $|||\nabla_n||| \le n\pi\sqrt{2}$. Moreover, some computations yield $|||\xi_n||| \le n\pi\sqrt{2}$. Therefore, an upper bound for the induced l^2-norm of K can be easily achieved as

$$\begin{aligned}|||K|||^2 < {} & \frac{2\pi^2((\frac{\lambda_1}{n^2})^2 + (\frac{\lambda_0}{n^2})^2)n^2 + 4\pi^4\gamma^2n^4 + (\frac{\lambda_1}{n^2})^2}{2} \\ & + \Big(\frac{(2\pi^2((\frac{\lambda_1}{n^2})^2 + (\frac{\lambda_0}{n^2})^2)n^2 + 4\pi^4\gamma^2n^4 + (\frac{\lambda_1}{n^2})^2)^2}{4} \\ & + \frac{-16\pi^4n^4((\frac{\lambda_1}{n^2})^2(\frac{\lambda_0}{n^2})^2 + (\frac{\lambda_1}{n^2})^2\gamma^2) - 32\pi^6\gamma^2(\frac{\lambda_0}{n^2})^2n^6}{4}\Big)^{1/2}.\end{aligned}$$

Now the primal-dual algorithm for solving (9) can be formulated as (see [11] for more details),

$$u^{k+1} = \arg\min_u \{||u-u_0||_2^2 + \frac{\lambda_1}{n^2}\langle \nabla_n u - p, w\rangle + \gamma\langle \xi_n(\nabla_n u), q\rangle + \frac{1}{2\tau}\|u-u^k\|_2^2\}$$

$$= \frac{u^k + \tau\frac{\lambda_1}{n^2}\mathrm{div}_{1n}(w^{k+1}) - \tau\gamma\mathrm{div}_{1n}(\mathrm{div}_{2n}(q^{k+1})) + 2\tau u^0}{1+2\tau}, \tag{18}$$

$$p^{k+1} = \arg\min_p \{\frac{\lambda_1}{n^2}\langle \nabla_n u - p, w\rangle + \frac{\lambda_0}{n^2}\langle \xi_n(p), \nu\rangle + \frac{1}{2\tau}\|p-p^k\|_2^2\}$$

$$= p^k + \tau\ (\frac{\lambda_1}{n^2}\ w^{k+1} + \frac{\lambda_0}{n^2}\mathrm{div}_{2n}(\nu^{k+1})), \tag{19}$$

$$w^{k+1} = \arg\max_w \{\frac{\lambda_1}{n^2}\langle \nabla_n u - p, w\rangle - I_{\{||\cdot||_\infty \le 1\}}(w) - \frac{1}{2\sigma}\|w-w^k\|_2^2\}$$

$$= P_{\lambda_1}(w^k + \sigma\frac{\lambda_1}{n^2}\ (\nabla_n \bar{u}^k - \bar{p}^k)), \tag{20}$$

$$\nu^{k+1} = \arg\max_\nu \{\frac{\lambda_0}{n^2}\langle \xi_n(p), \nu\rangle - I_{\{||\cdot||_\infty \le 1\}}(\nu) - \frac{1}{2\sigma}\|\nu-\nu^k\|_2^2\}$$

$$= P_{\lambda_0}(\nu^k + \sigma\frac{\lambda_0}{n^2}\ (\xi_n(\bar{p}^k))), \tag{21}$$

$$q^{k+1} = \arg\max_q \{\gamma\langle \xi_n(\nabla_n u), q\rangle - I_{\{||\cdot||_\infty \le 1\}}(q) - \frac{1}{2\sigma}\|q-q^k\|_2^2\}$$

$$= P_\gamma(q^k + \sigma\gamma\ (\xi_n(\nabla_n \bar{u}^k))), \tag{22}$$

where

$$P_{\lambda_1}(\bar{w}) = \frac{\bar{w}}{\max(1, |\bar{w}|)}, \quad P_{\lambda_0}(\bar{\nu}) = \frac{\bar{\nu}}{\max(1, |\bar{\nu}|)}, \quad P_\gamma(\bar{q}) = \frac{\bar{q}}{\max(1, |\bar{q}|)},$$

and τ and σ are positive constants. Algorithms 2 is the adapted algorithm to solve (9) (see [11] for details).

Algorithm 2 The adapted algorithm for solving the SHTGV model.

1. Set $k = 0$, choose the initial estimates $u^0 \in \mathbb{R}^{N_1 \times N_2}$, $p^0 \in (\mathbb{R}^2)^{nN_1 \times nN_2}$, $w^0 \in (\mathbb{R}^2)^{nN_1 \times nN_2}$, $\nu^0 \in (\mathbb{R}^4)^{N_1 \times N_2}$, $q^0 \in (\mathbb{R}^4)^{N_1 \times N_2}$, $\bar{u}^0 = u^0$, $\bar{p}^0 = p^0$, and positive parameters τ, σ such that $\sigma\tau|||K|||^2 < 1$, where

$$|||K|||^2 < \frac{2\pi^2((\frac{\lambda_1}{n^2})^2+(\frac{\lambda_0}{n^2})^2)n^2+4\pi^4\gamma^2n^4+(\frac{\lambda_1}{n^2})^2}{2} + \Big(\frac{(2\pi^2((\frac{\lambda_1}{n^2})^2+(\frac{\lambda_0}{n^2})^2)n^2+4\pi^4\gamma^2n^4+(\frac{\lambda_1}{n^2})^2)^2}{4} + \frac{-16\pi^4n^4((\frac{\lambda_1}{n^2})^2(\frac{\lambda_0}{n^2})^2+(\frac{\lambda_1}{n^2})^2\gamma^2)-32\pi^6\gamma^2(\frac{\lambda_0}{n^2})^2n^6}{4}\Big)^{1/2}.$$

2. Calculate u^{k+1}, p^{k+1}, w^{k+1}, ν^{k+1}, q^{k+1}, $\bar{u}^{k+1}$, and $\bar{p}^{k+1}$ using the following equations:

$$w^{k+1} = P_{\lambda_1}(w^k + \sigma\frac{\lambda_1}{n^2}(\nabla_n\bar{u}^k - \bar{p}^k)), \text{ where } P_{\lambda_1}(\bar{w}) = \frac{\bar{w}}{\max(1,|\bar{w}|)},$$

$$\nu^{k+1} = P_{\lambda_0}(\nu^k + \sigma\frac{\lambda_0}{n^2}(\xi_n(\bar{p}^k))), \text{ where } P_{\lambda_0}(\bar{\nu}) = \frac{\bar{\nu}}{\max(1,|\bar{\nu}|)},$$

$$q^{k+1} = P_{\gamma}(q^k + \sigma\gamma(\xi_n(\nabla_n\bar{u}^k))), \text{ where } P_{\gamma}(\bar{q}) = \frac{\bar{q}}{\max(1,|\bar{q}|)},$$

$$u^{k+1} = \frac{u^k + \tau\frac{\lambda_1}{n^2}\mathrm{div}_{1n}(w^{k+1}) - \tau\gamma\mathrm{div}_{1n}(\mathrm{div}_{2n}(q^{k+1})) + 2\tau u^0}{1+2\tau},$$

$$p^{k+1} = p^k + \tau(\frac{\lambda_1}{n^2}w^{k+1} + \frac{\lambda_0}{n^2}\mathrm{div}_{2n}(\nu^{k+1})),$$

$$\bar{u}^{k+1} = 2u^{k+1} - u^k,\ \bar{p}^{k+1} = 2p^{k+1} - p^k.$$

3. Stop or set $k = k+1$ and go back to step 2.

5 Convergence analysis

In this section, we establish the convergence of the primal-dual algorithm used for solving (9). Here, we only give the basic proof frameworks and do not repeat the lengthy proving process in [11]. For this purpose, we first prove the existence of the solution to the variational problem (9) through the following theorem, which proof is similar to the proof of Theorem 1 in [23].

Theorem 1. There exists a solution to optimization problem (9).

Proof. Using the compactness property of $\mathrm{BV}(\Omega)$, for every bounded and minimizing sequence $\{u_k\}_{k\in\mathbb{N}} \subset \mathrm{BV}(\Omega)$, there exists a subsequence $\{u_{k_i}\}_{i\in\mathbb{N}} \subset \{u_k\}_{k\in\mathbb{N}}$ converging to some $u^* \in \mathrm{BV}(\Omega)$. According to [5], the functions $||\nabla_n u - p||_1$, $||\xi_n(p)||_1$, and $||\xi_n(\nabla_n(u))||_1$ are proper, convex, and lower semi-continuous, and the same conclusion holds for their weighted sum $\mathrm{SHTGV}_n^{2,\lambda}(u)$. Moreover, $\{||Au_{k_i} - u_0||_2^2\}_{i\in\mathbb{N}}$ is bounded. Therefore, we yield

$$||Au^* - u_0||_2^2 + \mathrm{SHTGV}_n^{2,\lambda}(u^*) \leq \lim_{i\to+\infty}\inf\ (||Au_{k_i} - u_0||_2^2 + \mathrm{SHTGV}_n^{2,\lambda}(u_{k_i})).$$

As a result, we can conclude the existence of the solution to optimization problem (9). □

Now the convergence theorem is exhibited in the following result.

Theorem 2. For positive parameters σ and τ, where $\sigma\tau|||K|||^2 < 1$, the sequence $\{u^k, p^k, w^k, \nu^k, q^k\}$ generated by Algorithm 2 converges to a solution of (9).

Proof. Based on Theorem 1, there exists a solution to optimization problem (9). For this reason, the set of saddle-points of (17) is non-empty. According to [5], the functions $||\nabla_n u - p||_1$, $||\xi_n(p)||_1$, and $||\xi_n(\nabla_n(u))||_1$ are proper, convex, and lower semi-continuous functions. Moreover, σ and τ in Algorithm 2 satisfy $\sigma\tau|||K|||^2 < 1$. As a result, conditions (a), (b), and (c) of Theorem 1 in [11] hold. Therefore, we can conclude that Algorithm 2 is an application of the original primal-dual algorithm in [11] and converges to a solution of (9). □

6 Experimental results

We compare the experimental results of our model with the current state-of-the-art models: isotropic TV (ITV), upwind TV [10] (UTV), Condat TV [16] (CTV), TGV [5], Shannon TV [1] (STV), and the Huber variant of Shannon TV [1] (HSTV). The experiments are conducted on test images illustrated in Figure 1. All test images are corrupted by additive white Gaussian noise of standard deviations 0.18 and 0.23. All the reconstruction processes are carried out using MATLAB R2015b on a PC with AMD A10-4600M APU with Radeon(tm) HD Graphics, 2.3-GHz Inter Core processor, and 4GB of RAM under Windows 8.

As discussed in [1], in practice, choosing the oversampling factor as $n = 2$ or $n = 3$ is enough. We set the oversampling factor $n = 2$ in STV, HSTV, and the new proposed SHTGV model. For the case $n = 3$ in STV, HSTV, and the new proposed SHTGV model, we observed that the convergence speed is decreased while no remarkable change is made in comparison with the case $n = 2$.

The restoration quality is measured by the peak signal to noise ratio (PSNR) [31] and the structural similarity index measure index [41]. The best result for each comparison item is highlighted in bold type font (see Table 2). The restored results are illustrated in Figures 2, 3, 4, 5, 6, and 7. For each model and test image, the results corresponding to parameters that provide the highest PSNR value are presented here.

The results of the ITV, UTV, CTV, STV, and HSTV models obviously illustrate staircasing effect in the affine regions. Although the TGV model is capable of suppressing the staircasing artifacts, the denoised results using this approach indicate that it sometimes leads to undesired edge blurring.

Moreover, the performance of the TGV model reveals that some regions contain an obvious staircasing effect. Our proposed model not only substantially alleviates the staircasing artifacts, but also sharply preserves important image features. Moreover, the reconstructed edges and details by our model are more distinct in comparison with other variational models. For example, zoomed-in regions in Figures 2 and 3 illustrate that the UTV model leaves some small white noise particles in denoised results. The ITV model performs to some extent better than the UTV model in noise removal, but some details and edges are not well reconstructed during the denoising process. The CTV model outperforms both the UTV and ITV models in restoration accuracy and noise removal. However, the results of the ITV, UTV, and CTV models obviously suffer from the staircasing artifacts. Although the STV and HSTV models have better performance in preserving edges and details in comparison with the ITV, UTV, and CTV models, they achieve results with obvious staircasing effect, while the staircasing effect is less sharp compared with the ITV, UTV, and CTV models. The denoised results by the TGV model indicate that this regularization scheme sometimes produces denoised results with blurred edges. The denoised results using the proposed model illustrate that this new scheme is capable of eliminating the staircasing effect and preserving the edges at the same time. In fact, the ITV, UTV, CTV, STV, and HSTV models cannot compete with our model in staircasing effect suppression, as observed in the denoised results of Figure 2. TGV model is capable of eliminating the staircasing effect to some extent, but it yields denoised results with blurred edges. As a result, we can conclude that TGV model cannot compete with our model in edge preservation. The same conclusion holds for denoised results in Figures 4, 5, and 6.

The ITV, UTV, CTV, and TGV regularization models are discretized using a finite-difference scheme, while the STV, HSTV, and SHTGV models are based on the concept of Shannon interpolation. For this reason, the images generated by means of ITV, UTV, CTV, and TGV schemes cannot be efficiently interpolated using standard interpolation models. To support this claim, croppings of the restored results are magnified with factor 3 using bicubic interpolation method, which reveals that interpolating the restored results of the ITV, UTV, CTV, and TGV models yield images with unwanted artifacts, while the results of the STV, HSTV, and SHTGV models are interpolated without creating artifacts. Since the proposed model outperforms both STV and HSTV models in staircase artifacts suppression, the staircasing artifacts are remarkably alleviated in interpolated results of our model. Moreover, the edges and details in interpolated results of our model are more distinct in comparison with the STV and HSTV models.

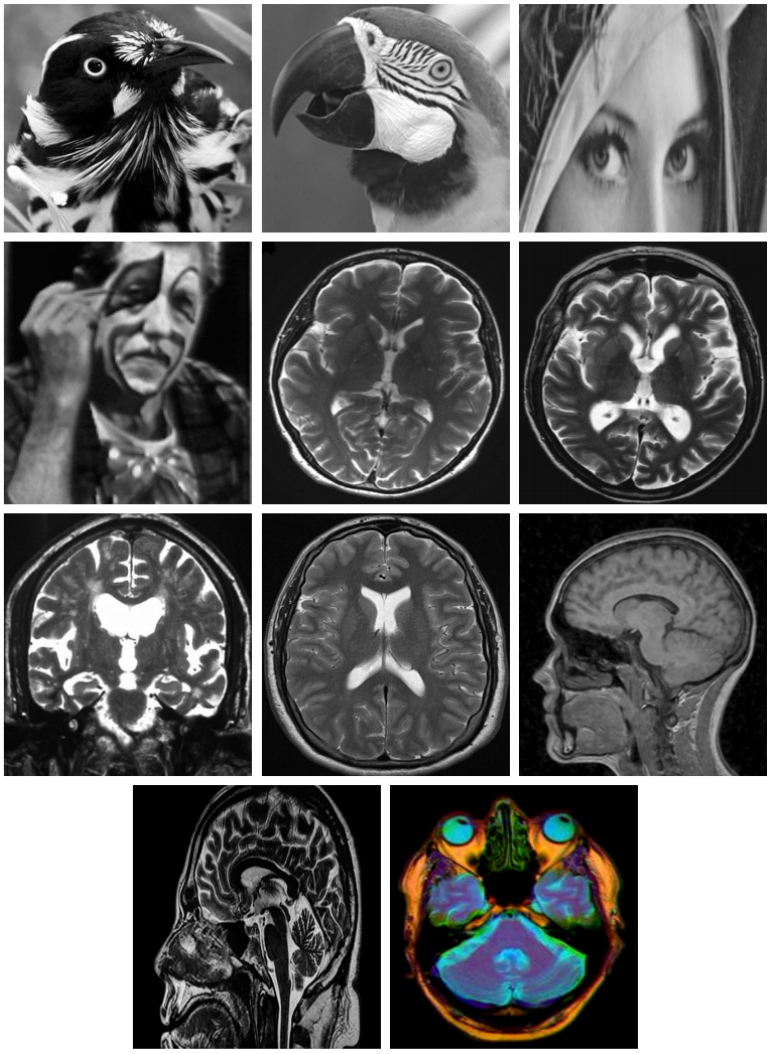


Figure 1: The test images used in our numerical experiments. First row, from left to right: Bird, Parrot, and Lena. Second row, from left to right: Clown, MRI1, and MRI2. Third row, from left to right: MRI3, MRI4, and MRI5. Forth row, from left to right: MRI6 and MRI7.

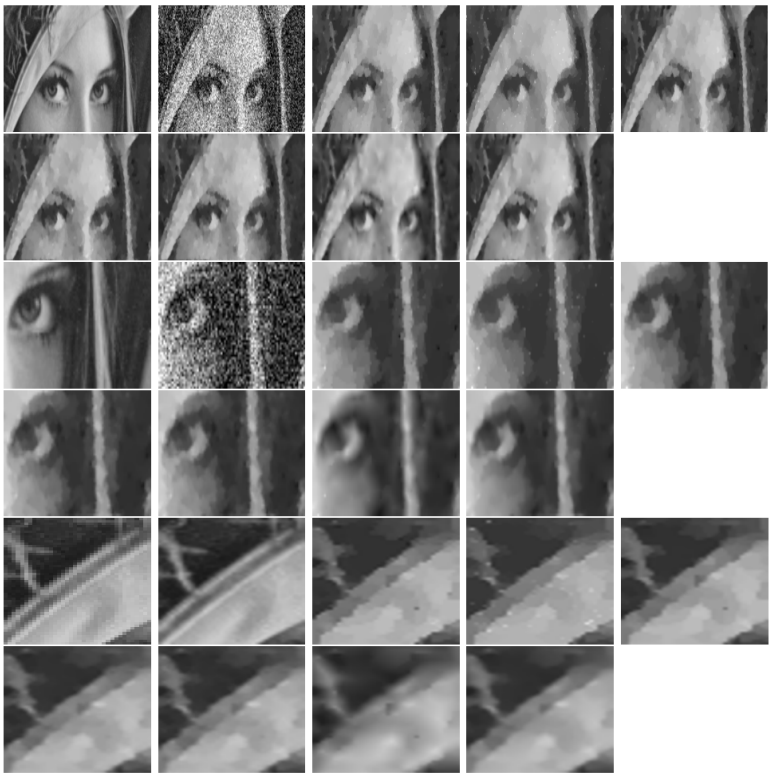


Figure 2: Performance comparison of Lena test image with additive white Gaussian noise of standard deviation 0.18. First row, from left to right: reference image, noisy image, restored results by ITV, UTV, and CTV. Second row, from left to right: restored results by STV, HSTV, TGV, and SHTGV. Images in the third and fourth rows are the corresponding zoomed-in regions of images in the first and second rows, respectively. Images in the fifth and sixth rows are croppings of the restored results of the Lena test image in the first and second rows magnified with factor 3 using the bicubic interpolation method. Fifth row, from left to right: reference image, resampling of the reference image, ITV, UTV, and CTV. Sixth row, from left to right: resampling of and STV, HSTV, TGV, and SHTGV.

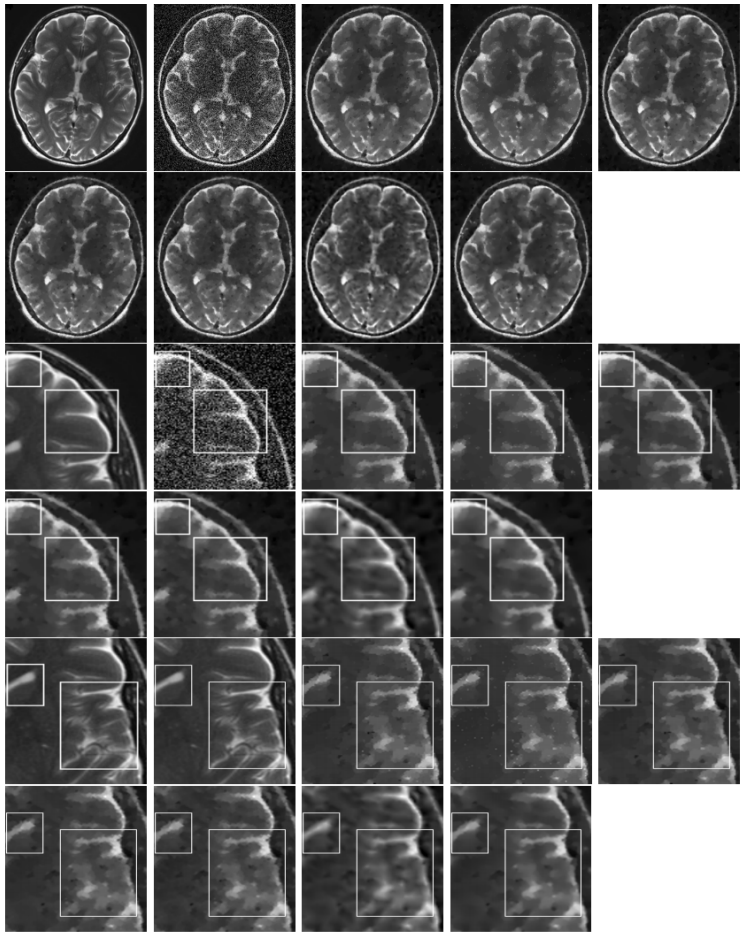


Figure 3: Performance comparison of MRI1 test image with additive white Gaussian noise of standard deviation 0.18. First row, from left to right: reference image, noisy image, restored results by ITV, UTV, and CTV. Second row, from left to right: restored results by STV, HSTV, TGV, and SHTGV. Images in the third and fourth rows are the corresponding zoomed-in regions of images in the first and second rows, respectively. Images in the fifth and sixth rows are croppings of the restored results of MRI1 test image in the first and second rows magnified with factor 3 using bicubic interpolation method. Fifth row, from left to right: reference image, resampling of the reference image, ITV, UTV, and CTV. Sixth row, from left to right: resampling of and STV, HSTV, TGV, and SHTGV.

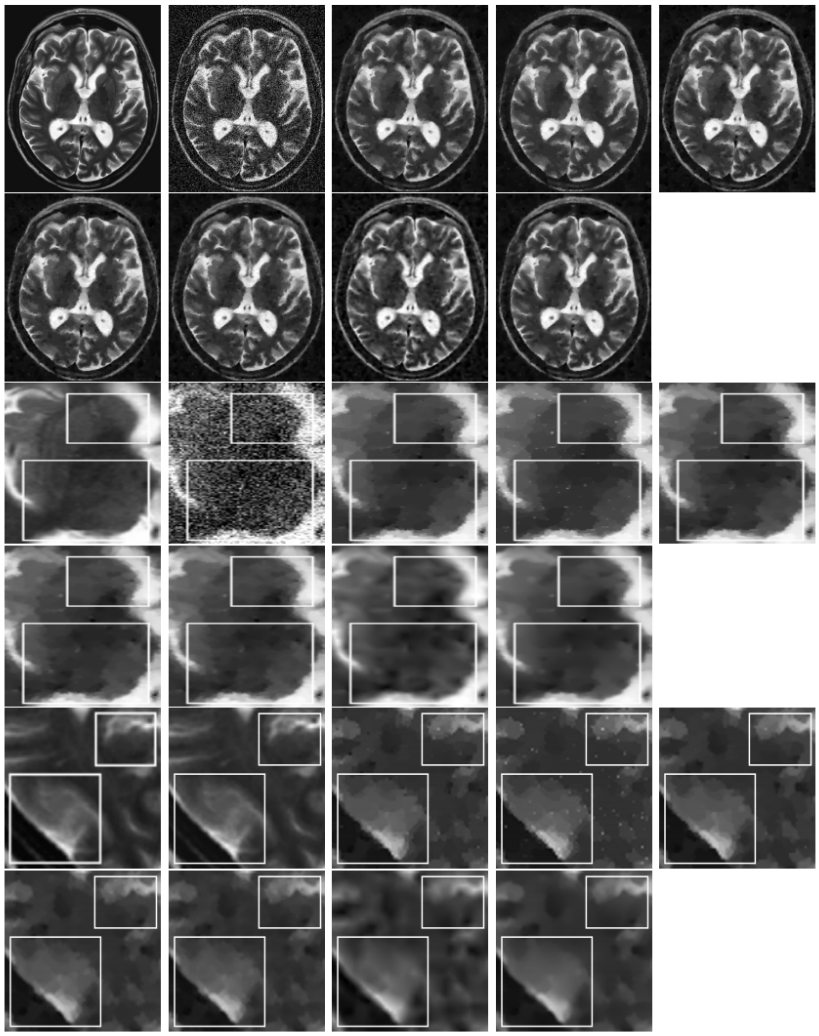


Figure 4: Performance comparison of MRI2 test image with additive white Gaussian noise of standard deviation 0.18. First row, from left to right: reference image, noisy image, restored results by ITV, UTV, and CTV. Second row, from left to right: restored results by STV, HSTV, TGV, and SHTGV. Images in the third and fourth rows are the corresponding zoomed-in regions of images in the first and second rows, respectively. Images in the fifth and sixth rows are croppings of the restored results of MRI2 test image in the first and second rows magnified with factor 3 using bicubic interpolation method. Fifth row, from left to right: reference image, resampling of the reference image, ITV, UTV, and CTV. Sixth row, from left to right: resampling of and STV, HSTV, TGV, and SHTGV.

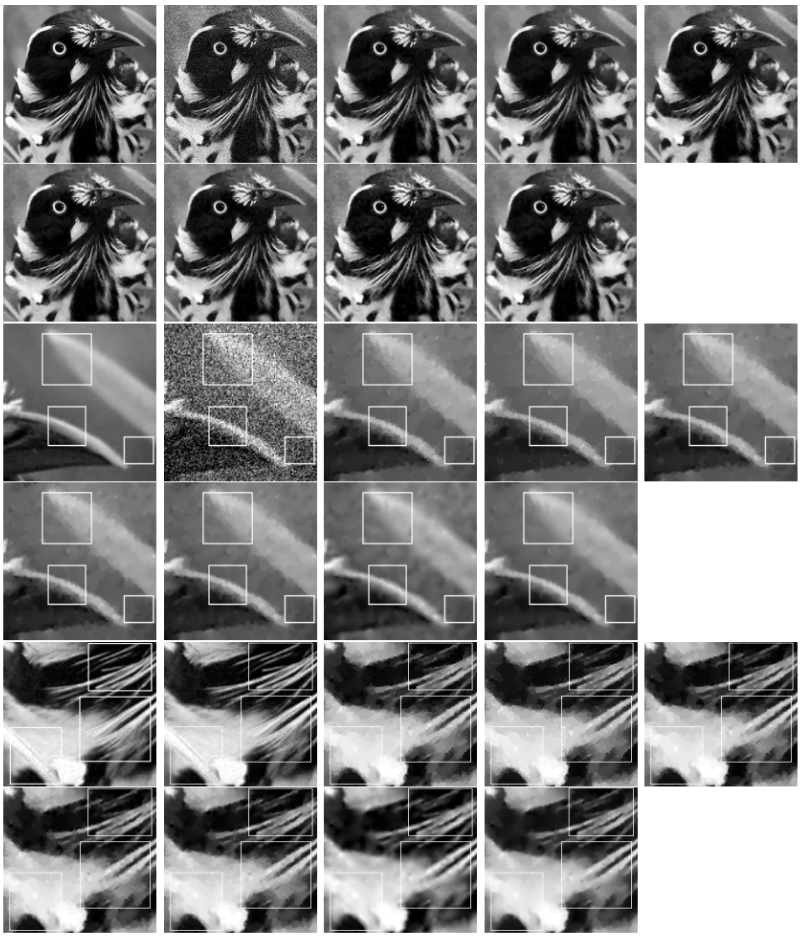


Figure 5: Performance comparison of Bird test image with additive white Gaussian noise of standard deviation 0.18. First row, from left to right: reference image, noisy image, restored results by ITV, UTV, and CTV. Second row, from left to right: restored results by STV, HSTV, TGV, and SHTGV. Images in the third and fourth rows are the corresponding zoomed-in regions of images in the first and second rows, respectively. Images in the fifth and sixth rows are croppings of the restored results of Bird test image in the first and second rows magnified with factor 3 using bicubic interpolation method. Fifth row, from left to right: reference image, resampling of the reference image, ITV, UTV, and CTV. Sixth row, from left to right: resampling of and STV, HSTV, TGV, and SHTGV.

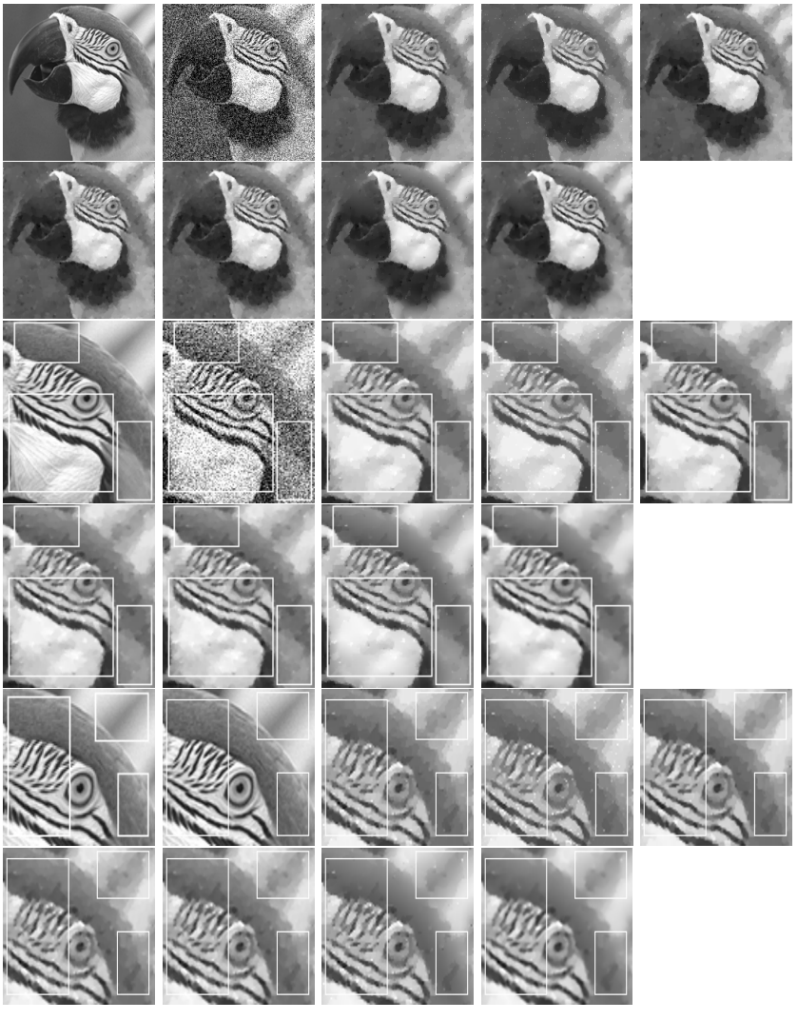


Figure 6: Performance comparison of Parrot test image with additive white Gaussian noise of standard deviation 0.18. First row, from left to right: reference image, noisy image, restored results by ITV, UTV, and CTV. Second row, from left to right: restored results by STV, HSTV, TGV, and SHTGV. Images in the third and fourth rows are the corresponding zoomed-in regions of images in the first and second rows, respectively. Images in the fifth and sixth rows are croppings of the restored results of Parrot test image in the first and second rows magnified with factor 3 using bicubic interpolation method. Fifth row, from left to right: reference image, resampling of the reference image, ITV, UTV, and CTV. Sixth row, from left to right: resampling of and STV, HSTV, TGV, and SHTGV.

Table 1: Comparison of different models in terms of PSNR and SSIM with additive white Gaussian noise of standard deviation 0.18.

Image	MRI1	MRI2
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	25.30 / 0.7810 / 0.15	25.33 / 0.7687 / 0.15
UTV	24.76 / 0.7566 / 0.18	24.80 / 0.7463 / 0.18
CTV	25.64 / 0.7898 / 0.14	25.64 / 0.7792 / 0.14
STV	26.14 / 0.8113 / 0.25	26.06 / 0.7996 / 0.25
HSTV	26.07 / 0.7965 / 0.25	26.00 / 0.7936 / 0.26
TGV	26.01 / 0.7824 / $(0.4, 0.1)$	26.10 / 0.7759 / $(0.5, 0.1)$
SHTGV	**26.39** / **0.8210** / $(0.45, 0.28)$	**26.34** / **0.8139** / $(0.45, 0.28)$
Image	Bird	Parrot
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	25.50 / 0.7348 / 0.15	25.33 / 0.7428 / 0.15
UTV	25.03 / 0.7109 / 0.18	24.80 / 0.7127 / 0.18
CTV	25.78 / 0.7438 / 0.14	25.63 / 0.7522 / 0.14
STV	26.09 / 0.7563 / 0.25	25.73 / 0.7464 / 0.24
HSTV	26.04 / 0.7509 / 0.26	25.68 / 0.7349 / 0.24
TGV	26.03 / 0.7416 / $(0.17, 0.11)$	25.46 / 0.7584 / $(0.15, 0.29)$
SHTGV	**26.29** / **0.7693** / $(0.45, 0.25)$	**25.84** / **0.7620** / $(0.42, 0.21)$

Table 2: Comparison of different models in terms of PSNR and SSIM with additive white Gaussian noise of standard deviation 0.23.

Image	MRI1	Parrot
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	24.10 / 0.7480 / 0.2	24.17 / 0.7111 / 0.2
UTV	23.55 / 0.7053 / 0.23	23.63 / 0.6603 / 0.23
CTV	24.42 / 0.7454 / 0.18	24.46 / 0.7086 / 0.18
STV	24.90 / **0.7762** / 0.33	24.56 / 0.7146 / 0.32
HSTV	24.84 / 0.7608 / 0.33	24.51 / 0.7020 / 0.32
TGV	24.75 / 0.7439 / $(0.26, 0.14)$	24.30 / 0.7117 / $(0.19, 0.36)$
SHTGV	**25.03** / 0.7551 / $(0.5, 0.5)$	**24.62** / **0.7178** / $(0.4, 0.28)$
Image	Lena	Clown
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	24.49 / 0.6701 / 0.21	22.42 / 0.6308 / 0.19
UTV	24.03 / 0.6393 / 0.25	21.97 / 0.6035 / 0.22
CTV	24.67 / 0.6759 / 0.19	22.67 / 0.6443 / 0.18
STV	24.69 / 0.6818 / 0.33	22.89 / 0.6563 / 0.31
HSTV	24.68 / 0.6814 / 0.34	22.89 / 0.6533 / 0.31
TGV	24.77 / 0.6916 / $(0.21, 0.17)$	23.00 / 0.6471 / $(0.23, 0.13)$
SHTGV	**24.83** / **0.6926** / $(0.5, 0.3)$	**23.10** / **0.6645** / $(0.6, 0.37)$

Table 3: Comparison of different models in terms of PSNR and SSIM with additive white Gaussian noise of standard deviation 0.18.

Image	MRI3	MRI4
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	24.57 / 0.7567 / 0.15	25.60 / 0.6891 / 0.15
UTV	24.06 / 0.7354 / 0.18	25.17 / 0.6676 / 0.18
CTV	24.91 / 0.7671 / 0.14	25.83 / 0.6966 / 0.14
STV	25.28 / 0.7810 / 0.25	26.16 / 0.7161 / 0.25
HSTV	25.25 / 0.7739 / 0.25	26.11 / 0.7096 / 0.26
TGV	24.99 / 0.7470 / $(0.29, 0.09)$	26.11 / 0.6988 / $(0.16, 0.12)$
SHTGV	**25.44** / **0.7893** / $(0.4, 0.2)$	**26.38** / **0.7231** / $(0.4, 0.2)$
Image	MRI5	MRI6
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	25.96 / 0.6351 / 0.16	23.30 / 0.6245 / 0.14
UTV	25.50 / 0.6173 / 0.19	22.86 / 0.6057 / 0.16
CTV	26.19 / 0.6413 / 0.15	23.58 / 0.6288 / 0.13
STV	26.43 / **0.6541** / 0.26	23.89 / 0.6285 / 0.22
HSTV	26.39 / 0.6429 / 0.26	23.86 / 0.6178 / 0.22
TGV	26.22 / 0.6174 / $(0.17, 0.11)$	23.67 / 0.5766 / $(0.24, 0.08)$
SHTGV	**26.53** / 0.6444 / $(0.4, 0.2)$	**24.06** / **0.6291** / $(0.4, 0.2)$
Image	Lena	Clown
Model	PSNR / SSIM / Optimal λ	PSNR / SSIM / Optimal λ
ITV	25.45 / 0.7028 / 0.15	23.55 / 0.6866 / 0.14
UTV	25.02 / 0.6800 / 0.19	23.08 / 0.6587 / 0.16
CTV	25.67 / 0.7133 / 0.14	23.82 / 0.6979 / 0.13
STV	25.70 / 0.7189 / 0.25	23.99 / 0.7083 / 0.23
HSTV	25.69 / 0.7160 / 0.25	23.99 / 0.7053 / 0.23
TGV	25.77 / 0.7277 / $(0.17, 0.11)$	24.16 / 0.7028 / $(0.18, 0.09)$
SHTGV	**25.84** / **0.7283** / $((0.42, 0.23)$	**24.24** / **0.7149** / $(0.44, 0.25)$

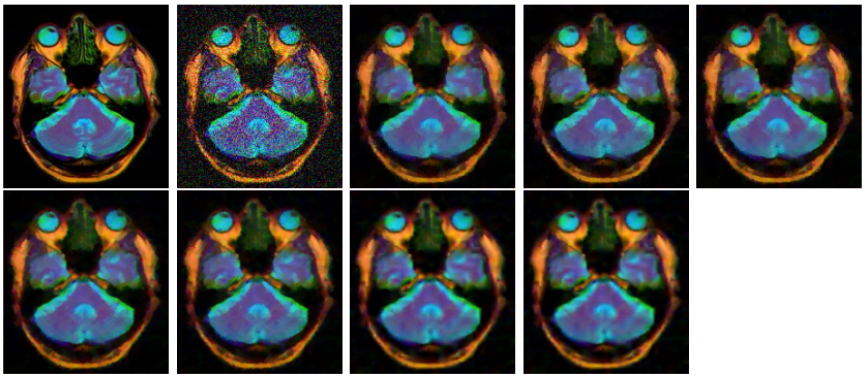


Figure 7: Performance comparison of denoised results of MRI7 with additive white Gaussian noise of standard deviation 0.18. First row, from left to right: reference image, noisy image, restored results by ITV (optimal $\lambda = 0.17$, PSNR = 25.63, SSIM = 0.7555), UTV (optimal $\lambda = 0.16$, PSNR = 25.79, SSIM = 0.7639), and CTV (optimal $\lambda = 0.17$, PSNR = 25.63, SSIM = 0.7555). Second row, from left to right: restored results by STV (optimal $\lambda = 0.26$, PSNR = 25.87, SSIM = 0.7774), HSTV (optimal $\lambda = 0.26$, PSNR = 25.81, SSIM = 0.7507), TGV (optimal $\lambda = (0.17, 0.11)$, PSNR = 25.84, SSIM = 0.7113), and SHTGV (optimal $\lambda = (0.46, 0.22)$, PSNR = 25.99, SSIM = 0.7525).

7 Conclusion

In this paper, a new hybrid TGV model was presented for image denoising. This model incorporated the advantages of Shannon interpolation, TGV regularization, and a symmetrized derivative-based l^1-norm regularized term. In comparison with some state-of-the-art techniques, experimental results illustrated that the proposed model substantially alleviates the staircasing effect and sharply preserves important image features. In fact, the reconstructed edges and details by our model were more distinct in comparison with several current state-of-the-art variational models. As we mentioned in the paper, most variational models were discretized using a finite-difference scheme, which cannot be efficiently interpolated using standard interpolation models. In this paper, we observed the efficiency of applying Shannon interpolation instead of the finite-differences schemes in the structure of variational models. Shannon interpolation applied in this paper was based on the Fourier transform. Designing a more powerful structure for Shannon interpolation based on the curvelet transform instead of Fourier transform will be considered as our future research.

References

1. Abergel, R. and Moisan, L. *The Shannon total variation*, J. Math. Imaging Vis., 59(2) (2017), 341–370.

2. Asim, M., Shamshad, F. and Ahmed, A. *Blind image deconvolution using deep generative priors*, IEEE Trans. Comput. Imaging, 6 (2020), 1493–1506.

3. Bhargava, G.U. and Gangadharan, S.V. *FPGA implementation of modified recursive box filter-based fast bilateral filter for image denoising*, Circuits, Syst. Signal Process., 40(3) (2021), 1438–1457.

4. Bioucas-Dias, J.M. and Figueiredo, M.A. *A new TwIST: Two-step iterative shrinkage/thresholding algorithms for image restoration*, IEEE Trans. Image Process., 16(12) (2007), 2992–3004.

5. Bredies, K., Kunisch, K. and Pock, T. *Total generalized variation*, SIAM J. Imaging Sci. , 3(3) (2010), 492–526.

6. Bredies, K. and Valkonen, T. *Inverse problems with second-order total generalized variation constraints*, Proceedings of SampTA, 201 (2011).

7. Burns, M., Haidacher, M., Wein, W., Viola, I. and Groller, M. E. *Feature emphasis and contextual cutaways for multimodal medical visualization*, EuroVis, 7 (2007), 275–282.

8. Caselles, V., Chambolle, A. and Novaga, M. *The discontinuity set of solutions of the TV denoising problem and some extensions*, Multiscale Model Simul. 6(3) (2007), 879–894.

9. Caselles, V., Chambolle, A. and Novaga M., *Regularity for solutions of the total variation denoising problem*, Rev. Mat. Iberoam., 27(1) (2011), 233–252.

10. Chambolle, A., Levine, S.E. and Lucier, B.J. *An upwind finite-difference method for total variation-based image smoothing*, SIAM J. Imaging Sci. 4(1) (2011), 277–299.

11. Chambolle, A. and Pock, T. *A first-order primal-dual algorithm for convex problems with applications to imaging*, J. Math. Imaging Vis., 40(1) (2011), 120–145.

12. Chan, T.F., Esedoglu, S. and Park, F.E. *Image decomposition combining staircase reduction and texture extraction*, J. Vis. Commun. Image Represent., 18(6) (2007), 464–486.

13. Chan, T.F. and Wong, C.K. *Total variation blind deconvolution*, IEEE Trans. Image Process., 7(3) (1998), 370–375.

14. Chan, T., Marquina, A. and Mulet, P. *High-order total variation-based image restoration*, SIAM J. Sci. Comput. 22(2) (2000), 503–516.

15. Chen, Y., Liu, L., Tao, J., Xia, R., Zhang, Q., Yang, K., Xiong, J. and Chen, X. *The improved image inpainting algorithm via encoder and similarity constraint*, Vis. Comput. 37(7) (2021), 1691–1705.

16. Condat, L. *Discrete total variation: New definition and minimization*, SIAM J. Imaging Sci. , 10(3) (2017), 1258–1290.

17. Dong, J. and Pan, J. *Deep outlier handling for image deblurring*, IEEE Trans. Image Process., 30 (2021), 1799–1811.

18. El Hamidi, A., Menard, M., Lugiez, M. and Ghannam, C. *Weighted and extended total variation for image restoration and decomposition*, Pattern Recognit. 43(4) (2010), 1564–1576.

19. Fang, F., Li, J., Yuan, Y., Zeng, T., Zhang, G. *Multi-level edge features guided network for image denoising*, IEEE Trans. Neural Netw. Learn. Syst., 32(9) (2021), 3956–3970.

20. Guo, W., Qin, J. and Yin, W. *A new detail-preserving regularization scheme*, SIAM J. Imaging Sci. , 7(2) (2014), 1309–1334.

21. Kang, S.H. and March, R. *Variational models for image colorization via chromaticity and brightness decomposition*, IEEE Trans. Image Process., 16(9) (2007), 2251–2261.

22. Liu, J., Yan, M. and Zeng, T. *Surface-aware blind image deblurring*, IEEE Trans. Pattern Anal. Mach. Intell., 43(3) (2021), 1041–1055.

23. Liu, X. *Total generalized variation and wavelet frame-based adaptive image restoration algorithm*, Vis. Comput. 35(12) (2019), 1883–1894.

24. Malgouyres, F. and Guichard, F. *Edge direction preserving image zooming: a mathematical and numerical analysis*, SIAM J. Numer. Anal., 39(1) (2001), 1–37.

25. Maso, G.D., Fonseca, I., Leoni, G. and Morini, M. *A higher order model for image restoration: the one-dimensional case*, SIAM J. Math. Anal., 40(6) (2009), 2351–2391.

26. Moisan, L. *How to discretize the total variation of an image?*, in the 6th International Congress on Industrial Applied Mathematics, Proceedings in Applied Mathematics and Mechanics, 7(1) (2007), 1041907–1041908.

27. Nacereddine, N., Zelmat, M., Belaifa, S.S. and Tridi, M. *Weld defect detection in industrial radiography based digital image processing*, Trans. Eng. Technol. Educ., 2 (2005), 145–148.

28. Rajora, S., Butola, M. and Khare, K. *Regularization-parameter-free optimization approach for image deconvolution*, Applied Optics. 60(19) (2021), 5669–5677.

29. Reddy, P.L. and Pawar, S. *Multispectral Image Denoising Using Curvelet Transform and Kriging Interpolation Based Wiener Filter*, Design Engineering, (2021), 838–849.

30. Rudin, L.I., Osher S. and Fatemi, E. *Nonlinear total variation based noise removal algorithms*, 'Phys. D: Nonlinear Phenom., 60(1-4) (1992), 259–268.

31. Russ JC. *The image processing handbook*, 6th ed. USA: CRC Press, 2011.

32. Sakthidasan alias Sankaran, K. and Nagarajan, V. *Noise removal through the exploration of subjective and apparent denoised patches using discrete wavelet transform*, IETE J. Res., 67(6) (2021), 843–852.

33. Samson, C., Blanc-Féraud, L., Aubert, G. and Zerubia, J. *A variational model for image classification and restoration*, IEEE Trans. Pattern Anal. Mach. Intell., 22(5) (2000), 460–472.

34. Scherzer, O. *Denoising with higher order derivatives of bounded variation and an application to parameter estimation*, Computing, 60(1) (1998), 1–27.

35. Setzer, S.I. and Steidl, G.A. *Variational methods with higher order derivatives in image processing*, Approximation, 12 (2008), 360–386.

36. Valsesia, D. and Boufounos, P. T. *Multispectral image compression using universal vector quantization*, IEEE Information Theory Workshop (ITW), (2016), 151–155.

37. Vogel, C.R. and Oman, M.E. *Fast, robust total variation-based reconstruction of noisy, blurred images*, IEEE Trans. Image Process., 7(6) (1998), 813–824.

38. Wang, N., Zhang, Y. and Zhang, L. *Dynamic selection network for image inpainting*, IEEE Trans. Image Process., 30 (2021), 1784–1798.

39. Wang, W., Zhang, C. and Ng, M.K. *Variational model for simultaneously image denoising and contrast enhancement*, Opt. Express, 28(13) (2020), 18751–18777.

40. Wang, Y., Song, X. and Chen, K. *Channel and space attention neural network for image denoising*, IEEE Signal Process. Lett., 28 (2021), 424–428.

41. Wang, Z., Bovik, A.C., Sheikh, H.R. and Simoncelli, E.P. *Image quality assessment: from error visibility to structural similarity*, IEEE Trans. Image Process., 13(4) (2004), 600–612.

42. Wu, T., Gu, X., Wang, Y. and Zeng, T. *Adaptive total variation based image segmentation with semi-proximal alternating minimization*, Signal Process. 183 (2021), 108017.

43. Wu, T. and Shao, J. *Non-convex and convex coupling image segmentation via tgpv regularization and thresholding*, Adv. Appl. Math. Mech., 12 (2020), 849–878.

44. Wu, T., Shao, J., Gu, X., Ng, M.K. and Zeng, T. *Two-stage image segmentation based on nonconvex $l_2 - l_p$ approximation and thresholding*, Appl. Math. Comput., 403 (2021), 126168.

45. Yuan, J. *An improved variational model for denoising magnetic resonance images*,Comput. Math. Appl. , 76(9) (2018), 2212–2222.

46. Yuan, Q., Zhang, L. and Shen, H. *Multiframe super-resolution employing a spatially weighted total variation model*, IEEE Trans. Circuits Syst. Video Technol., 22(3) (2011), 379–392.

47. Zhang, H., Chen, H., Yang, G. and Zhang, L. *LR-Net: Low-rank spatial-spectral network for hyperspectral image denoising*, IEEE Trans. Image Process., 30 (2021), 8743–8758.

48. Zhang, X. *Two-step non-local means method for image denoising*, Multidimens. Syst. Signal Process., (2021), 1–26.

49. Zhang, Y., Li, K., Li, K., Sun, G., Kong, Y. and Fu, Y. *Accurate and fast image denoising via attention guided scaling*, IEEE Trans. Image Process., 30 (2021), 6255–6265.

50. Zhang, Z. and Blum, R.S. *Region-based image fusion scheme for concealed weapon detection*, in Proceedings of the 31st Annual Conference on Information Sciences and Systems, (1997), 168–173.

How to cite this article
E. Tavakkol, S.M. Hosseini and A. Hosseini Image denoising via a new hybrid TGV model based on Shannon interpolation. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 371-396. doi: 10.22067/ijnao.2022.71857.1052.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 397–425
DOI: DOI:10.22067/ijnao.2022.71418.1049
https://ijnao.um.ac.ir/

Crocodile Hunting Strategy (CHS): A comparative study using benchmark functions

A.R. Balavand

Abstract

The crocodiles have a good strategy for hunting the fishes in nature. These creatures are divided into two groups of chasers and ambushers when hunting. The chasers direct prey toward shallow water with a powerful splash of its tail without catching them, and the ambushers wait in the shallow and try to snatch the fishes. Such behavior inspires the development of a new population-based optimization algorithm called the crocodile hunting strategy (CHS). In order to verify the performance of the CHS, several classical benchmark functions and four constrained engineering design optimization problems are used. In the classical benchmark function, the comparisons are performed using ant colony optimization, differential evolution, genetic algorithm, and particle swarm optimization. Constrained engineering design problems are compared with firefly algorithm, harmony search, shuffled frog-leaping algorithm, and teaching-learning-based optimization. The results of the comparison show that different operators designed in the CHS algorithm lead to fast algorithm convergence and show better results compared to other algorithms.

AMS subject classifications (2020): 45D05; 42C10; 65G99.

Keywords: Crocodile hunting strategy; Optimization algorithms; Numerical optimization; Classical benchmark functions; Constrained engineering design problem.

* Corresponding author

Received 11 July 2021; revised 18 January 2022; accepted 5 February 2022

Alireza Balavand
Department of Industrial Engineering, Science and Research Branch, Islamic Azad University. e-mail: a.balavand@srbiau.ac.ir

1 Introduction

Optimization is an art that finds the best solution among the set of solutions, and the optimization techniques are tools that converge to the optimal solutions by searching the solution space. Finding the best solution to mathematical problems has been a challenge for researchers, especially in complicated problems in the field of engineering. Therefore, convergence to the best solution in the least time with high accuracy has always been interesting to researchers in the field of optimization. Among the optimization tools, metaheuristic algorithms are widely being used for a large number of real-world engineering problems due to their acceptable accuracy and low time consumption. Their popularity derives from the following aspects. Firstly, all of these optimization techniques have some fundamental theories and mathematical models proven to be reasonable, which come from the real world and are inspired by all kinds of physical phenomena or biological behaviours [26, 7]. Secondly, metaheuristics are usually able to solve NP-hard problems with a high number of decision variables. They are very flexible and versatile since one can change the structures and parameters of algorithms to obtain better solutions. Thirdly, metaheuristic algorithms can effectively escape local optima, which is very valuable for multimodal engineering problems that have many local optima in their structures. In addition, their variants can be developed by absorbing the advantages of other algorithms to improve the accuracy of solutions in a reasonable time. Most metaheuristic algorithms can solve different types of problems. Some of them are inherently powerful because of efficient operators and other types of them have been well-hybridized by the classical algorithms [35]. These problems are divided into various categories, which are included constrained and unconstrained in terms of problem structure, discrete and continuous in terms of solutions, and single or multiobjective in terms of objective function [27]. Generally, metaheuristics are divided into two categories such as single-solution-based and population-based algorithms. In single-solution-based algorithms, the searching process starts with one candidate solution, whereas in population-based algorithms, the optimization process performs using a set of solutions (i.e., population). Population-based metaheuristics have advantages over single-solution. These are as follows:

1. The searching process starts with several random solutions as the initial population.

2. population-based metaheuristics can share the information with each other around the search space that, most of the time, leads to escape from local optimal.

3. The exploration capability of population-based metaheuristics has had better than single-solution-based techniques.

In a general view, population-based metaheuristics are classified into seven categories, including biology-based algorithms, physics-based algorithms, social-based algorithms, chemistry-based algorithms, math-based algorithms, sport-based algorithms, and music-based algorithms [1]. Biology-based algorithms are divided into three categories of evolution-inspired algorithms, swarm intelligence algorithms, and artificial immune systems [21]. The evolution-inspired algorithm is a generic population-based metaheuristic that is inspired by biological evolution that includes crossover, mutation, and selection. The second category is swarm-intelligence-based algorithms. These algorithms are based on the interaction of swarm with each other. Swarm-based algorithms are easier to implement than evolutionary-based algorithms due to including a less number of operators (i.e., selection, crossover, mutation). Apart from this, there are some advantages of swarm-based algorithms, which are as follows:

Swarm-based algorithms use the flow of information among solutions during iterations, whereas evolutionary-based algorithms do not use the information of the previous generations.

Swarm-based algorithms have few input parameters as compared to evolutionary techniques.

Swarm-based algorithms utilize less memory space for reaching the best optimal solutions [3, 36].

The artificial immune systems algorithm was introduced in [11]. These algorithms mimic the principle of the biological immune system, including the negative selection algorithm, clonal selection algorithm, the immune network model, and danger theory.

Solving the test functions has been used for comparing metaheuristics. Test functions include classical test functions, constrained engineering design problems, and CEC functions. Classical test functions have been solved in several dimensions using metaheuristics that are the most practical and simple type of test function. Constrained engineering design problems are real-world engineering problems that are the proper criterion for comparing metaheuristics. CEC test functions are classified into complicated test functions that have always been a challenge to metaheuristics.

In the remainder of this study, in Section 2, the literature review will be explained. The concepts and operators of crocodile hunting strategy (CHS) are explained in Section 3. Then in Section 4, the CHS is investigated in various aspects. In that section, the classical benchmark functions and constrained engineering design problems are solved using the CHS algorithm and compared with other well-known metaheuristics. Finally, in Section 5, the conclusion and discussion will be presented.

2 Literature review

Swarm intelligence of nature-inspired algorithms includes intensive techniques that mimic the social behavior of groups of animals. The power of swarm intelligence algorithms was appeared by Kennedy and Eberhart by developing the particle swarm optimization (PSO) algorithm [24]. The main idea of the PSO algorithm is inspired by the social behavior of the bird population. Ant colony optimization (ACO) [13] is inspired by the collective behavior of the ants, namely finding the shortest path from nest to food sources. Termite algorithm [29] is based on the behavior of termites. The shuffled fog-leaping algorithm [16] stems from the search for food by the frog group. Monkey search simulates a monkey that climbs trees and looks for food [34]. Cuckoo search [42] simulates the forced child parasitic behavior in combination with the levy flying behavior of some birds and fruit flies. The main idea of the firefly algorithm (FA) [41] is inspired by the optical connection between fireflies. Migrating birds optimization [14] simulates the process of the social evolution of birds. The fruit fly optimization algorithm [37] is inspired by the foraging behavior of fruit flies. Artificial bee colony [2] simulates the strategy of bees to find food from the nest to source food. Dolphin echolocation (DE) [23] is based on the behavior of dolphins in hunting, in which the dolphin finds its prey using the distance between its click and prey. The main idea of the bat algorithm [43] is based on the distance between the reflection of the bat's sound to source food. In social spider optimization [10], the spider receives and analyzes vibrations published on the web to determine the potential direction of a food source. Grey wolf optimizer [18, 33] is a nature-inspired metaheuristic algorithm that simulates the behavior of gray wolfs and the hierarchy of leadership in their hunting. Bird mating optimizer [4] is inspired by the mating strategies of bird species during the mating season. The tree-seed algorithm [25] is based on the relation between trees and their seeds. Moth-flame optimization [31] simulates the spiral motion of butterflies around the flame. Whale optimization algorithm [32] simulates the bubble net hunting method that is performed from humpback whales. Crow search algorithm [5] is modeled based on finding the food of other birds and stealing it by crows. The grasshopper optimization algorithm [40] is a nature-inspired metaheuristic algorithm that mimics the movement of Grasshoppers groups to find food sources. Queuing search algorithm [45] is inspired by human activities in queues. Pathfinder algorithm [44] is modeled by the collective movement of the animal group and mimics the leadership hierarchy of swarms to find the best food area or prey. The main inspiration of Harris hawks optimization [19] is the cooperative behavior and chasing style of Harris' hawks in nature called surprise pounce. The squirrel search algorithm [20] imitates the dynamic foraging behavior of southern flying squirrels and their efficient way of locomotion known as gliding. Marine predators algorithm [17] is inspired by widespread foraging strategies, namely Lévy and Brownian of predators in the ocean.

In this study, the new algorithm called CHS (crocodile optimization algorithm) is investigated using two kinds of benchmark functions. This algorithm was first proposed by Balavand [6] to find the optimum binary solution of a feature clustering method. The CHS is based on the swarm intelligence of crocodiles in hunting that simulates the behavior of crocodiles in the hunting of fishes— dividing crocodiles into different groups and creating the right strategy while hunting is the main idea of this algorithm. These creatures are divided into two groups when they are hunting fish. These groups include the chasers and the ambushers. These groups try to direct the fish to the attacking area by using the information flow and proper cooperation. After arriving prey in the attacking area, all members of the groups attack the fish and catch them. The details of how to encircle and attack by two groups will be explained in the next sections.

3 Crocodile optimization algorithm (CHS)

In this section, the optimization algorithm called CHS optimization algorithm is investigated, which was first used as one of the steps of feature clustering in [6]. The CHS is placed in the population-based algorithm that can solve optimization problems using swarm intelligence of crocodiles in hunting. Like other algorithms in this field, the CHS algorithm with exchanging information and creating an information flow through the collaboration of the members, converges to the optimum solution. This algorithm is based on the behavior of crocodiles that simulates the hunting strategy of crocodiles. Dinets [12] divided crocodiles into two kinds of chasers and ambushers when hunting. The chasers, which are bigger than other crocodiles, follow and chase fishes toward the shore with a powerful splash of their tail without catching them. Ambushers, which are smaller and more agile, wait in the shallow and try to snatch the fish. According to [12], crocodiles have a complicated strategy to snatch prey. According to the bottom section of Figure 1, the ambushers swim in a circular pattern around prey and take turns cutting through the center of the gradually shrinking circle, snatching the prey. According to the top section of Figure 1, chasers direct prey toward ambusher and attacking area [12]. In the following, the behaviors of the two groups will be simulated in mathematical equations.

3.1 Initializing

Like other metaheuristic algorithms, before entering the main phases, the initializing phase is done. In the initializing phase, several initial solutions are randomly generated. Indeed, these randomized solutions constitute the initial population of crocodiles. These solutions are generated with uniform

random distribution between the lower and upper bounds. These solutions are generated according to the following equation:

$$x = LB + r \times (UB - LB)\,. \tag{1}$$

After determining initial parameters such as the number of population, the number of max iterations, the lower bound of variables, and the upper bounds of variables, random solutions (x) are generated based on (2), where LB and UB are lower and upper bounds of the problem, respectively. Also, r is a uniform random variable that is generated between zero and one. Then these solutions are evaluated based on the objective function. In fact, in operators of CHS, the solutions are evaluated based on the objective function. If a better solution is found, then it is replaced instead of the best solution. The better solution has the minimum objective function value, which is known as the best solution (x_{prey}).

Figure 1: Crocodiles hunting strategy

3.2 Chasing the prey

In this section, the behavior of the chasers is simulated. As mentioned before, the population is divided into two sections. Hence, each section makes up 50 percent of the population. The group of chasers includes the first 50 percent of the solutions. Accordingly, the second part of 50 percent of the population includes ambushers. These two groups are randomly divided into two groups of chasers and ambushers. The idea behind simulating the behavior of chasers is based on the distance between prey and crocodiles that simulates the behavior of chasers. As previously mentioned, the chasers are another group of hunters that follow the prey without catching it and direct the prey to the shore and shallow area. The following equations are proposed to simulate this behavior:

$$d^{i,t} = \left| x^{t}_{prey} - x^{i,t}_{chaser} \right|, \qquad \text{for all } i, \tag{2}$$

$$\begin{cases} x^{i,t+1}_{chaser} = \left(x^{i,t}_{chaser} - \beta.d \right) & \text{for all } i \ (\frac{\ln(it)}{\ln(maxit)} \times r) \geq 0.5, \quad (a) \\ x^{i,t+1}_{chaser} = |\beta - (\beta.d) \ \ | & \text{for all } i \ (\frac{\ln(it)}{\ln(maxit)} \times r) < 0.5. \quad (b) \end{cases} \tag{3}$$

Here x^{t}_{prey} denotes the prey position at iteration t and $x^{i,t}_{chaser}$ indicates the position of the chaser ith at iteration t. The coefficient β has a uniform distribution that changes between $[-3,\ 3]$. This r is a random number between zero and one. Also, it is the global iteration number and i is local iteration number in inner loops. These intervals have been obtained experimentally after successive iterations of the algorithm. According to (3), d is the distance between the prey and ith chaser. According to (3), two states occur: First, if $\frac{it}{maxit} \times r \geq 0.5$, then $(3)(a)$ is implemented, which means that each chaser in the first group moved toward the prey with a positive coefficient, and if $\frac{it}{maxit} \times r < 0.5$, then a random search is done. In brief, (3) shows that by an intelligent approach, in the initial iterations, $(3)(b)$ is more implemented, and a random search is done. In the last iterations, $(3)(a)$ is more implemented, and the chasers close to prey by random approach.

3.3 Attacking to prey

The final position of the prey is where the ambushers are waiting to catch the prey. Indeed, the ambushers camouflage in the final position and the chasers try to direct the prey to this place or the attacking area. In order to simulate the attacking phase, it is assumed that attackers are forced to update their position according to the following equations:

$$d^{i,t} = \left| x^{t}_{prey} - x^{i,t}_{ambusher} \right|, \quad \text{for all } i, \tag{4}$$

$$A = \frac{apc + apa + x_{prey}}{3}, \tag{5}$$

$$\begin{cases} x_{ambusher}^{i,t+1} = \left(d.\cos\left(2r\pi\right) \ + x_{prey}^{t}\right), & \text{for all } i \ \ (p) \geq 0.5, \quad (a) \\ x_{ambusher}^{i,t+1} = \left(x_{ambusher}^{i,t} - \beta(A - x_{ambusher}^{i,t})\right), & \text{for all } i \ \ (p) < 0.5, \quad (b) \end{cases} \tag{6}$$

in which $p = \frac{\ln(it)}{\ln(maxit)} \times r$ Here $x_{ambusher}^{i,t}$ shows the position of ith ambusher at iteration t and d denotes the distance between the ambushers and the prey. Moreover, *apc* is the average position of chasers, *apa* is the average position of ambushers, and x_{prey} is best position or prey position. Crocodiles update their position according to the positions of the prey or average of the position of all groups. In this way, if $(\frac{\ln(it)}{\ln(maxit)} \times r) < 0.5$, then $(8)(a)$ is implemented that means that each ambusher swims around the prey with a rotational motion, and if $(\frac{\ln(it)}{\ln(maxit)} \times r) \geq 0.5$, then ambushers update their position according to the average of the position of all groups of chasers, ambushers, and the position of prey. How to calculate the average position of all groups is shown in (7).

3.4 Flowchart and pseudocode of the CHS

Given that the CHS algorithm is implemented in MATLAB software 2019b, in order to provide more explanation of the structure of the programming of the CHS algorithm, the pseudocode is presented as Algorithm 2, and the flowchart is shown in Figure 3. According to the pseudocode, after determining the initial parameters, such as the number of population, the maximum number of the iteration, the number of population of each group of the crocodiles, the lower and upper bound of the decision variable, and so on, the initializing phase is done. In the initializing phase, to create the number of the population, the random solutions are generated based on the uniform distribution that all of the solutions are between the lower and upper bounds of the objective function. These random solutions are evaluated based on the objective function. Before entering the main phases, the population is divided into two groups. Fifty percent of the population includes chasers, and the other fifty percent includes ambushers' population. It is assumed that the solution with the minimum objective value is the best solution, and other members of the population are considered crocodiles. Therefore, the population number is equal to $npop = nc + ng$, which is always an even number. Because the number of the population is halved, the number of all members of the population is always an even number.

In the following pseudocode, the first phase is performed based on (3) for each solution of the chasers population. Then each solution is evaluated based on the objective function. In this phase, if a better solution is found, then it will be replaced by the best solution. In the second phase, for each

solution of the ambusher's population, (8) is performed and each solution is evaluated. In this phase, a better solution will be replaced instead of the best solution. According to the flowchart, after passing all phases of the algorithm, the stopping Criteria's are checked, if the stopping Criteria's are satisfied, then the best solutions, along with the objective value, will be reported. Otherwise, the number of iterations is added to a unit, and from the first phase, all steps are repeated.

3.5 Exploration and exploitation

In designing a metaheuristic algorithm, the balancing of the two criteria of exploration and exploitation increases the efficiency of the algorithm [3, 36, 28]. The experiences show that in the initial iterations, the ability of exploration should be increased, and in the final iterations, the power of exploitation should be increased. This means that in the initial iterations, the different parts of the solution space should be searched, and in the last iterations, the optimal solution should be searched more accurately. This rule has been observed in the CHS algorithm. In this way, in the exploitation phases, small mutations occur, and in the exploration phases, longer mutations occur. The CHS is divided into two phases, and in each phase, there is the main equation. Equation (3) is the main equation of the first phase and (8) is the second one. In the first iterations, part (b) of the main equations is more implemented because, in most solutions, the value of $\left(\frac{\ln(it)}{\ln(maxit)} \times r\right)$ is less than 0.5 in most cases. Therefore (b) is implemented in most times that (b)'s increase the power of exploration. Whereas in the last iterations, part (a) of (3) and (8) is implemented because in most cases the value of $\left(\frac{\ln(it)}{\ln(maxit)} \times r\right)$ is greater than 0.5 that increase the power of exploitation.

4 Evaluation of the CHS algorithm

In this section, two types of test functions are used to evaluate the efficiency of the CHS algorithm. The first category includes classical benchmark functions, and second category includes constrained engineering design problems. In addition, several well-known metaheuristics are used to compare with the CHS algorithm. In the following, classical test functions will be explained and metaheuristics will be compared using test functions. Then, the comparative results of metaheuristics will be shown in the constrained engineering design problems.

Algorithm 1 The pseudocode of the CHS

set initial parameters
best solution=$+\infty$
Initializing
For *each population*
Generate random position between lower and upper bounds
Evaluation of random position
Update the best solution and current solution
End for
update the best solution
dividing the population into the chasers and ambushers
Main loop
While nfe $\leq$ ***maxnfe***
-First phase
For *each population of chasers*
If $\left(\frac{\ln(it)}{\ln(maxit)} \times r\right) > 0.5$
Implementing $(3)(a)$
Else if $\left(\frac{\ln(it)}{\ln(maxit)} \times r\right) \leq 0.5$
Implementing $(3)(b)$
end if
Update the best solution and current solution
End for
-Second phase
For *each population of ambushers*
If $\left(\frac{\ln(it)}{\ln(maxit)} \times r\right) > 0.5$
Implementing $(8)(a)$
Else if $\left(\frac{\ln(it)}{\ln(maxit)} \times r\right) \leq 0.5$
Implementing $(8)(b)$
end if
Update the best solution and current solution
End for
It=it+1
End while (main loop)
Report the best solution

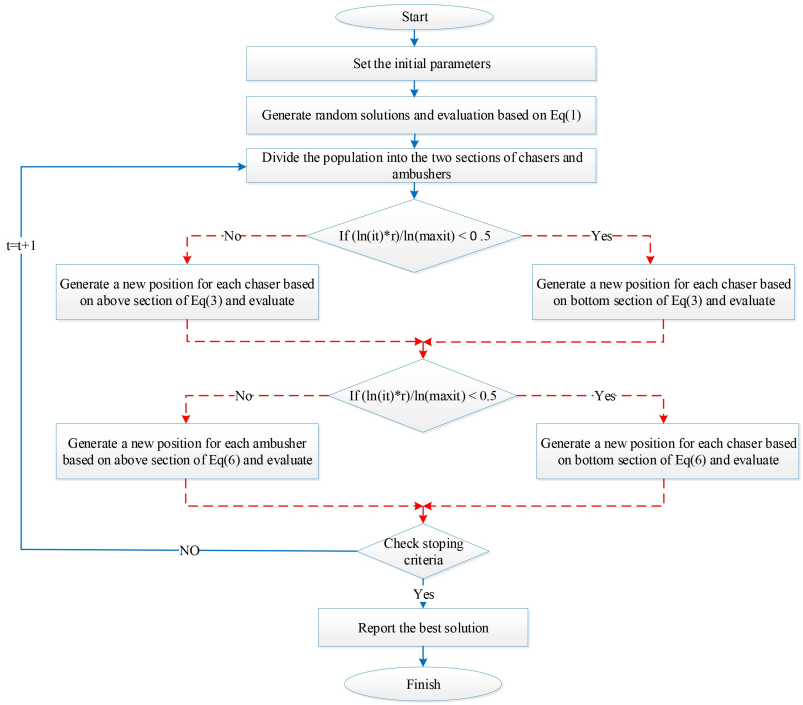


Figure 2: The flowchart of the CHS

4.1 Classic test functions

In Table 1, 20 well-known test functions are shown, which are used to evaluate the CHS algorithm. In this table, the code of each test function, the name of each function, the lower and upper bounds of each function, the dimensions investigated, the formula, the minimum value of each function, and the type of each function in terms of unimodal and multimodal are shown, respectively. According to Table 1, there are six unimodal and 14 multimodal functions. In particular, unimodal functions are useful tools for benchmarking exploitation. In contrast, multimodal functions have many local optima and are suitable tools to evaluate the ability to the exploration of metaheuristics.

In Table 4, perspective views of the classic test functions are displayed. In this study, the number of function evaluations (NFE) is used for the stopping criteria. The value of NFE is equal to $NFE = maxit \times npop$ that $maxit$ denotes a maximum of iteration, and $npop$ denotes the number of population of the algorithm. For the functions, f_1–f_{12}, the value of NFE is 50,000, and for f_{13}–f_{20}, the value of NFE is 180,000. Because the functions f_1–f_{12} are investigated in two dimensions, and the functions f_{13}–f_{20} are investigated in thirty dimensions. Given that as the dimensions increase, it becomes harder

Table 1: Benchmark functions (NF= number of functions, D= dimension, U=unimodal, M = multimodal)

NF	Function	Range	D	Formulation	Min	type
(f_1)	Beale	$[-4.5, 4.5]$	2	$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	0	U
(f_2)	Bird	$[-2\pi, 2\pi]$	2	$f(x) = sin(x_1).\left[\exp\left[1-\cos(x_2)\right]^2\right] + \cos(x_2)\ ].exp\left[1 - sin(x_1)^2\right] + (x_1 - x_2)^2$	-106.76	M
(f_3)	Booth	$[-10, 10]$	2	$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	0	M
(f_4)	Carrom- Table	$[-10, 10]$	2	$f(x) = \left[\frac{\left[-\cos^2(x_1).\cos^2(x_2)\ ].\exp\left[\left\lvert 1-\frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert\right]^2}{30}\right]$	-24.157	M
(f_5)	Cross-in-tray	$[-10, 10]$	2	$f(x) = -0.0001.\left[\left\lvert\sin(x_1).\sin(x_2).\exp\left[\left\lvert 100 - \frac{\left(\sqrt{x_1^2+x_2^2}\right)}{\pi}\right\rvert\right] + 1\right\rvert\right]^{0.1}$	-2.0626	M
(f_6)	Cross-Leg -Table	$[-10, 10]$	2	$f(x) = -\frac{1}{\left[\left\lvert\sin(x_1).\sin(x_2).\exp\left[\left\lvert 100 - \frac{\left(\sqrt{x_1^2+x_2^2}\right)}{\pi}\right\rvert\right] +1\right\rvert\right]^{0.1}}$	-1	M
(f_7)	Crowned cross	$[-10, 10]$	2	$f(x) = 0.0001\left[\left\lvert\sin(x_1).\sin(x_2).\exp\left[\left\lvert 100 - \frac{\left(\sqrt{x_1^2+x_2^2}\right)}{\pi}\right\rvert\right] + 1\right\rvert\right]^{0.1}$	0.0001	M
(f_8)	Easom	$[-100, 100]$	2	$f(x) = -\cos(x_1).\cos(x_2).\exp\left(-(x_1 - \pi)^2 - (x_2 - \pi)^2\right]$	-1	U
(f_9)	Himmelblau	$[-5, 5]$	2	$f(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$	0	M
(f_{10})	Matyas	$[-10, 10]$	2	$f(x) = 0.26\left(x_1^2 + x_2^2\right) - 0.48x_1x_2$	0	U
(f_{11})	Pen-holder	$[-11, 11]$	2	$f(x) = -\exp\left[-\left\lvert\cos(x_1).\cos(x_2).\exp\left[\left\lvert 1 - \frac{\sqrt{x_1^2+x_2^2}}{\pi}\right\rvert\right]\right\rvert\right]^{-1}$	-0.9635	M
(f_{12})	Six-hump-camel	$[-5, 5]$	2	$f(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + \left(4x_2^2 - 4\right)x_2^2$	-1.0316	M
(f_{13})	Ackley	$[-35, 35]$	30	$f(x) = -20\exp\left[-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right] - \exp\left[\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\ \right] + 20 + \exp(1)$	0	M
(f_{14})	Griewank	$[-600, 600]$	30	$f(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0	M
(f_{15})	Michalewicz	$[0, \pi]$	30	$f(x) = -\sum_{i=1}^{n}sin(x_i)sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$	-29.630	M
(f_{16})	Rastrigin	$[-5.12, 5.12]$	30	$f(x) = 10n + \sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i)\right]$	0	M
(f_{17})	Rosenbrock	$[-100, 100]$	30	$f(x) = \sum_{i=1}^{n-1}100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$	0	M
(f_{18})	Sphere	$[-100, 100]$	30	$f(x) = \sum_{i=1}^{n}x_i^2$	0	U
(f_{19})	Step	$[-5.12, 5.12]$	30	$f(x) = \sum_{i=1}^{n}(x_i + 0.5)^2$	0	U
(f_{20})	Quartic	$[-1.28, 1.28]$	30	$f(x) = \sum_{i=1}^{n}ix_i^4 + rand(0, 1)$	0	U

to reach the optimal solution; accordingly, it should give more time to the algorithms for reaching better solutions. Therefore, the value of NFE in the functions of f_{13}–f_{20} is greater than f_1–f_{12}. In order to validate the results, the CHS algorithm is compared with four well-known high-performance algorithms that are ACO, DE, genetic algorithm (GA), and PSO. Table 2 shows the results of the study on classical functions in two dimensions. Among these functions, there are three unimodal and nine multimodal functions. In addition, in Tables 2 and 3, three indicators of the best cost, the average of the best costs, and the standard deviation of the best costs have been used to investigate the performance of the CHS algorithm in the functions f_1–f_{12}. The average of the best cost is calculated based on $A = \frac{\sum_{i=1}^{maxit} bestcost_i}{maxit}$ that $bestcost_i$ is the value of the best cost in ith iteration. Also, the standard deviation of the best cost is calculated based on $std = std(bestcosts)$, in which $bestcosts$ is the vector of the best costs. In Table 3, the best cost, the average of the best costs, and the standard deviation of the best costs are calculated. In this table, the functions of Ackley, Griewank, Michalewicz, Rastrigin, Rosenbrock, Sphere, Step, and Quartic are investigated in the thirty dimensions. The Ackley function has plenty of local minimal, and a large hole is located in the middle of this function. The Griewank function has a lot of local minimal. The specific structure of the Michalewicz function makes that this function becomes a hard function. The Rastrigin function is a high multimodal function, but its local minima are distributed regularly. The well-known function of Rosenbrock is a hard function that converging

Table 2: Comparative results of the CHS with ACO, DE, GA, and PSO in two-dimensional benchmark functions.

Function	parameters	ACO	DE	GA	CHS	PSO
f_1	Best	3.7285E-15	**0.0000E+00**	5.3573E-09	**0.0000E+00**	**0.0000E+00**
	Mean	3.0446E-04	1.7499E-04	1.4618E-04	2.8217E-03	6.1959E-04
	Variance	3.6041E-03	1.4767E-03	2.3619E-03	3.9028E-02	9.8299E-03
f_2	Best	**-1.067E+02**	**-1.067E+02**	**-1.067E+02**	**-1.067E+02**	**-1.067E+02**
	Mean	-1.0657E+02	-1.0670E+02	-1.0671E+02	-1.0670E+02	-1.0675E+02
	Variance	2.3277E+00	6.2301E-01	6.1592E-01	8.0510E-01	1.4927E-01
f_3	Best	**0.0000E+00**	**0.0000E+00**	6.6867E-05	**0.0000E+00**	**0.0000E+00**
	Mean	1.1495E-03	1.3656E-02	2.1538E-04	6.3135E-03	1.9529E-03
	Variance	2.0576E-02	1.8990E-01	1.7110E-03	1.0428E-01	5.6387E-02
f_4	Best	**-2.4157E+01**	**-2.4157E+01**	**-2.4157E+01**	**-2.4157E+01**	**-2.4157E+01**
	Mean	-2.4152E+01	-2.4154E+01	-2.4154E+01	-2.4146E+01	-2.4154E+01
	Variance	9.7867E-02	6.5639E-02	5.5639E-02	2.5004E-01	7.9855E-02
f_5	Best	**-2.0626E+00**	**-2.0626E+00**	**-2.0626E+00**	**-2.0626E+00**	**-2.0626E+00**
	Mean	-2.0625E+00	-2.0624E+00	-2.0625E+00	-2.0626E+00	-2.0626E+00
	Variance	2.2683E-03	2.9736E-03	3.3138E-03	5.8323E-04	1.0088E-03
f_6	Best	-8.4778E-02	-8.4778E-02	-8.4778E-02	**-1.0000E+00**	-8.4778E-02
	Mean	-7.4230E-02	-7.3789E-02	-7.3789E-02	-1.0000E+00	-5.7640E-02
	Variance	2.7300E-02	2.7295E-02	2.7295E-02	0.0000E+00	3.6339E-02
f_7	Best	2.3790E-01	1.1795E-03	1.7427E-02	**1.0000E-04**	**1.0000E-04**
	Mean	4.4454E-01	3.2020E-02	2.8825E-02	1.0092E-03	6.1287E-02
	Variance	9.1072E-02	1.0853E-01	6.8593E-02	2.8750E-02	1.6701E-01
f_8	Best	**-1.0000E+00**	**-1.0000E+00**	**-1.0000E+00**	**-1.0000E+00**	**-1.0000E+00**
	Mean	-9.9294E-01	-9.4630E-01	-9.4630E-01	-9.8772E-01	-9.8969E-01
	Variance	7.2850E-02	2.1878E-01	2.1878E-01	8.9564E-02	9.2793E-02
f_9	Best	**0.0000E+00**	7.8886E-31	7.8886E-31	7.8886E-31	7.8886E-31
	Mean	5.5989E-03	1.2939E-02	1.3149E-02	1.4854E-02	5.6025E-03
	Variance	7.2871E-02	1.6169E-01	1.7269E-01	1.4631E-01	1.2813E-01
f_{10}	Best	3.6536E-14	1.0988E-70	1.0988E-70	**0.0000E+00**	9.3840E-83
	Mean	2.9606E-04	8.5572E-04	8.5572E-04	6.5336E-06	6.4553E-04
	Variance	3.8089E-03	1.1494E-02	1.1494E-02	2.0661E-04	1.4383E-02
f_{11}	Best	**-9.6353E-01**	**-9.6353E-01**	**-9.6353E-01**	**-9.6353E-01**	**-9.6353E-01**
	Mean	-9.6345E-01	-9.6352E-01	-9.6352E-01	-9.6347E-01	-9.6353E-01
	Variance	1.0913E-03	1.1988E-04	1.2088E-03	5.8280E-04	2.5529E-05
f_{12}	Best	**-1.0316E+00**	**-1.0316E+00**	**-1.0316E+00**	**-1.0316E+00**	**-1.0316E+00**
	Mean	-1.0303E+00	-1.0311E+00	-1.0307E+00	-1.0301E+00	-1.0308E+00
	Variance	3.5421E-02	5.7551E-03	1.7266E-02	3.3696E-02	2.3828E-02

to its optimal global solution has always been a challenge for metaheuristics. The sphere function is unimodal and convex. The Step function is the same as the Sphere function, with the difference that its center is moved. The Quartic function is a unimodal function that is always a hard function among convex functions.

The search history of the CHS algorithm in Ackley, Michalewicz, Rastrigin, and Quartic is shown in Table 5. This table shows that the CHS algorithm uses exploration to search almost all local optimizations, and after approaching the global solution, it has searched for better solutions around it. The results obtained in Table 2 show that, in all test functions, the CHS algorithm has a superior performance and reaches the optimal global solution. According to this table, except for f_8, the CHS has had a good performance, and in functions $f_6,\ f_7,\ f_{10}$, the CHS has had the best performance compared to other metaheuristics in terms of best costs. The results in Table 3 show the efficiency of the algorithms in the test functions f_{13} to f_{20}. Based on this table, the performance of the CHS algorithm has been better than other algorithms

Table 3: Comparative results of the CHS with ACO, DE, GA, and PSO in three-dimensional benchmark functions.

Function	parameters	ACO	DE	GA	CHS	PSO
f_{13}	Best	4.4409E-15	1.5099E-14	4.0861E-02	**8.8818E-16**	1.5099E-14
	Mean	3.4812E-01	1.3290E+00	8.8208E-01	4.3144E-03	3.1990E-01
	Variance	2.0814E+00	4.0717E+00	1.9128E+00	1.3531E-01	1.4827E+00
f_{14}	Best	9.8573E-03	**0.0000E+00**	5.9285E-02	**0.0000E+00**	1.2321E-02
	Mean	2.2289E+00	7.7930E+00	1.5710E+00	3.6101E-03	6.6917E-01
	Variance	2.5471E+01	4.4016E+01	1.1665E+01	8.9211E-02	1.0457E+01
f_{15}	Best	-1.5282E+01	-2.8588E+01	**-2.9588E+01**	-2.4338E+01	-2.5882E+01
	Mean	-1.4375E+01	-2.5639E+01	-2.8840E+01	-2.3855E+01	-2.5667E+01
	Variance	1.0662E+00	3.5141E+00	2.2322E+00	1.8568E+00	1.5003E+00
f_{16}	Best	8.9546E+00	3.5499E+01	2.5984E-02	**0.0000E+00**	3.6813E+01
	Mean	3.5185E+01	7.8142E+01	1.2294E+01	1.2679E+00	4.1114E+01
	Variance	6.3628E+01	5.0830E+01	3.1570E+01	1.7385E+01	2.4295E+01
f_{17}	Best	2.3585E+01	2.7599E+01	1.8210E+02	**1.5166E+00**	2.3434E+01
	Mean	7.7561E+07	2.2186E+08	9.5396E+06	3.7934E+06	8.7383E+06
	Variance	1.1298E+09	1.6832E+09	2.0580E+08	1.5632E+08	2.8500E+08
f_{18}	Best	5.9710E-107	8.2297E-28	1.5372E-02	**0.0000E+00**	3.2359E-40
	Mean	2.3492E+02	8.6412E+02	1.5473E+02	1.6111E+01	6.0851E+01
	Variance	2.7076E+03	5.0574E+03	1.2821E+03	5.5154E+02	1.1066E+03
f_{19}	Best	2.7733E-32	1.7133E-30	6.2775E-05	2.9041E-07	**0.0000E+00**
	Mean	5.8458E-01	2.0791E+00	2.4766E-01	9.7130E-02	1.8939E-01
	Variance	6.9647E+00	1.2470E+01	2.4888E+00	1.4607E+00	2.9788E+00
f_{20}	Best	1.6655E-03	9.6106E-03	1.9030E-02	**3.5024E-05**	4.3593E-03
	Mean	2.6485E-01	9.6546E-01	7.6241E-02	2.1654E-03	6.1895E-02
	Variance	3.8147E+00	7.1113E+00	7.0836E-01	6.8662E-02	1.3975E+00

in terms of the best solution in the functions f_{13}, f_{14}, f_{16}, f_{17}, f_{18}, f_{20}. Furthermore, the CHS algorithm has reached the optimal solution in f_{14}, f_{16}, f_{18}, f_{20}, and the difference between the optimal solution and the obtained result in the functions $f_{13}, f_{15}, f_{17}, f_{19}$ is low. The extreme convergence of the CHS algorithm in multimodal functions $f_{13}, f_{14}, f_{16}, f_{17}$ and the unimodal functions f_{18}, f_{19} , f_{20} are quite evident in Table 6. In total, among the 20 classic functions, the CHS algorithm has reached the optimal solution in 17 functions, where among 17 functions, 12 functions are multimodal and five of them are unimodal. Also, in the six functions, the CHS algorithm has had better results compared to the other algorithms. In all of the classical test functions except for f_9, in terms of the best solution, none of the algorithms is able to provide a better solution than the CHS. These results indicate a good balance between the exploration and exploitation in the CHS algorithm.

4.2 Constrained engineering design problems

In recent years, constrained engineering design problems have been used in various papers to examine the performance of metaheuristic algorithms. For example, six constrained engineering design problems to evaluate the crow search algorithm were used in [5], five constrained engineering design problems were used to evaluate the league championship algorithm in [22], eight constrained engineering design problems to examine the mine blast algorithm

Table 4: Perspective view for benchmark functions

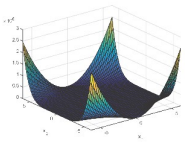
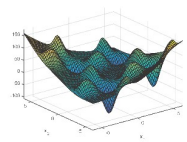
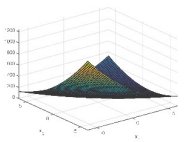
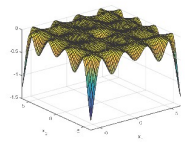
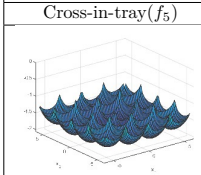
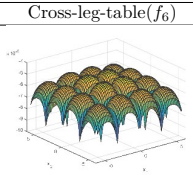
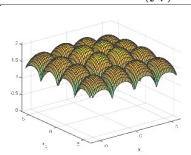
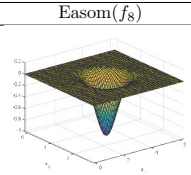
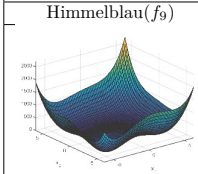
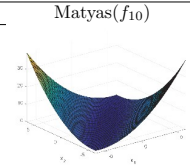
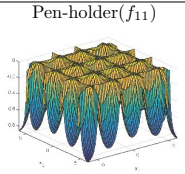
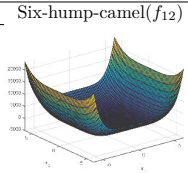
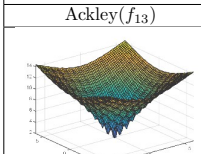
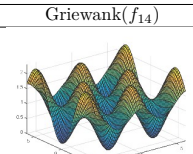
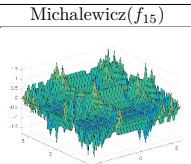
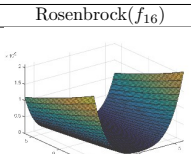
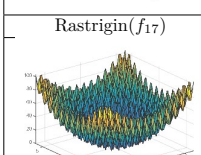
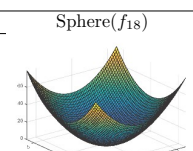
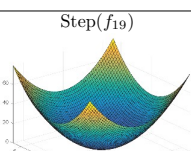
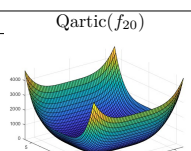
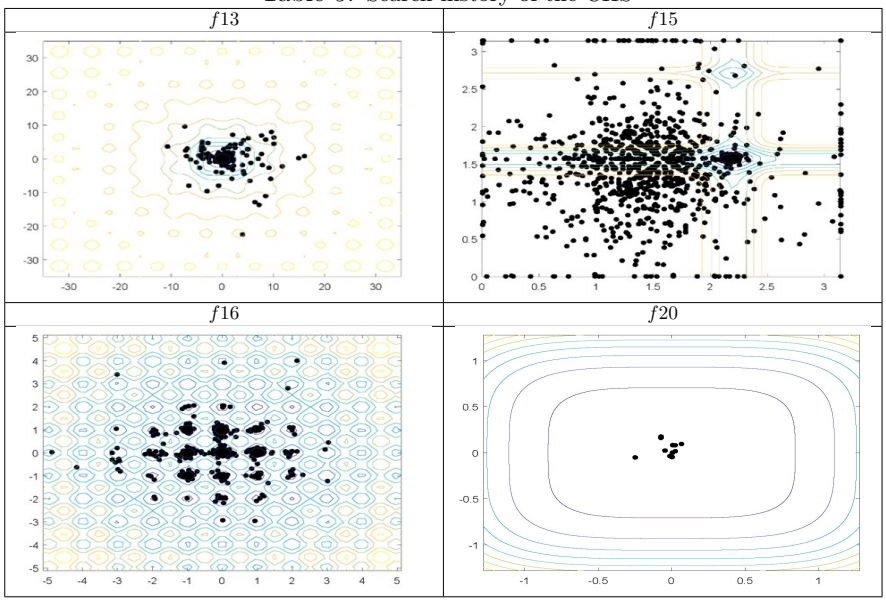
Beale(f_1) 	Bird(f_2) 	Booth(f_3) 	Carrom-table(f_4) 
Cross-in-tray(f_5) 	Cross-leg-table(f_6) 	Crowned cross (f_7) 	Easom(f_8) 
Himmelblau(f_9) 	Matyas(f_{10}) 	Pen-holder(f_{11}) 	Six-hump-camel(f_{12}) 
Ackley(f_{13}) 	Griewank(f_{14}) 	Michalewicz(f_{15}) 	Rosenbrock(f_{16}) 
Rastrigin(f_{17}) 	Sphere(f_{18}) 	Step(f_{19}) 	Qartic(f_{20}) 

Table 5: Search history of the CHS



were used in [39], and seven constrained engineering design problems were solved with water cycle algorithm in [15]. In this section, four constrained engineering design problems, including a three-bar design problem, pressure vessel design problem, Tension/compression spring design, and welded beam design problem, are used to study the performance of the CHS algorithm. Creating an effective balance in searching between acceptable and unacceptable solutions is one of the challenges in solving engineering design problems. Various strategies have emerged to handle unacceptable areas in evolutionary computing. Transferring constraints to the objective function with a penalty factor and transforming the optimization problem into an unstrained problem [9] is the basic idea of the penalty function. Additive penalty functions [9, 30] will be used to solve this problem in this paper, which is one of the famous methods. In order to investigate the performance of the CHS algorithm, several indicators are used, including the values of the decision variables, the constraint values, and the best solution. In order to compare the performance, the following four optimization algorithms were used:

1. FA (firefly algorithm)
2. Harmony Search (HS)
3. SFLA (shuffled frog-leaping algorithm)
4. Teaching-learning-based optimization (TLBO)

Table 6: Compare performance algorithms

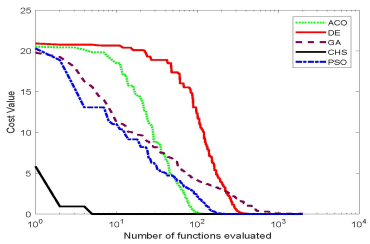
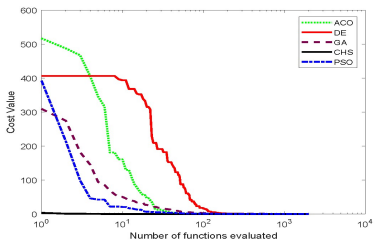
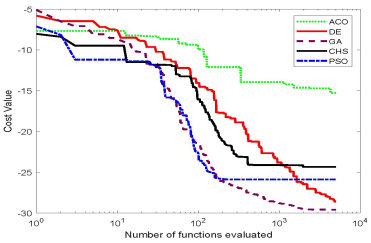
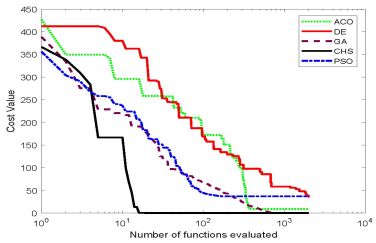
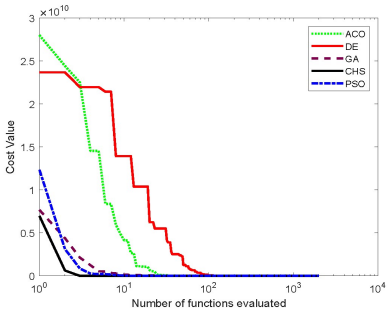
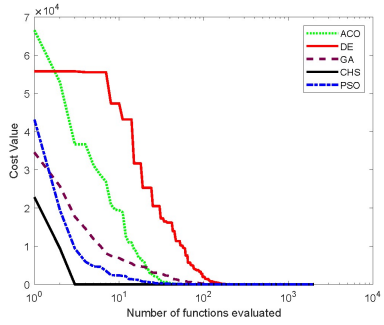
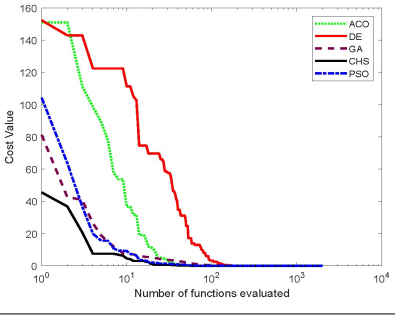
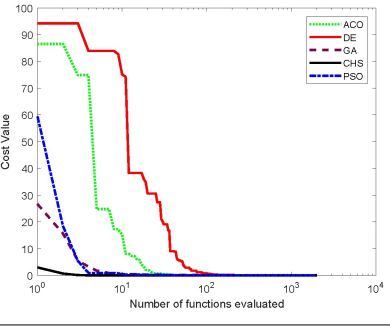
f_{13}	f_{14}
	
f_{15}	f_{16}
	
f_{17}	f_{18}
	
f_{19}	f_{20}
	

Table 7: Comparative results of the CHS with FA, HS, SFLA, and TLBO in three-bar truss design problem.

Parameters	FA	CHS	HS	SFLA	TLBO
x_1	7.886774424E-01	7.886783330E-01	7.880650859E-01	7.865946407E-01	7.886790873E-01
x_2	4.082417630E-01	4.082392461E-01	4.099765093E-01	4.141648543E-01	4.082371275E-01
g_1	-1.114419668E-11	-1.532325600E-09	4.653166741E-12	6.646239115E-12	-1.276911088E-08
g_2	-1.464109036	-1.464111898	-1.462138552	-1.457394322	-1.464114312
g_3	-5.358909643E-01	-5.358881039E-01	-5.378614480E-01	-5.426056785E-01	-5.358857010E-01
Best Value	**2.638958434E+02**	**2.638958436E+02**	**2.638961174E+02**	**2.638990472E+02**	**2.638958451E+02**

The three-bar truss design problem Minimizing the volume of a statistically loaded three-bar truss with respect to stress limits (σ) [38] is the main objective in the three-bar truss design problem. The mathematical model of the problem is based on the following equation:

$$\begin{aligned}
&\text{Min} f(x) = (2\sqrt{2}x_1 + x_2) \times l \\
&s.t. \\
&g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} p - \sigma \leq 0, \\
&g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} p - \sigma \leq 0, \\
&g_3(x) = \frac{1}{\sqrt{2}x_2 + x_1} p - \sigma \leq 0, \\
&0 \leq x_i \leq 1, \qquad i = 1, 2, \\
&l = 100\ cm, \quad p = 2\frac{kN}{cm^2}, \quad \sigma = 2\frac{kN}{cm^2}.
\end{aligned} \tag{7}$$

This problem has two continuous decision variables and three unequal constraints. In Figure 3, the three-bar truss design is represented schematically. Figure 4 shows the convergence rate of the CHS algorithm in finding an optimal solution to the three-bar truss design problem.

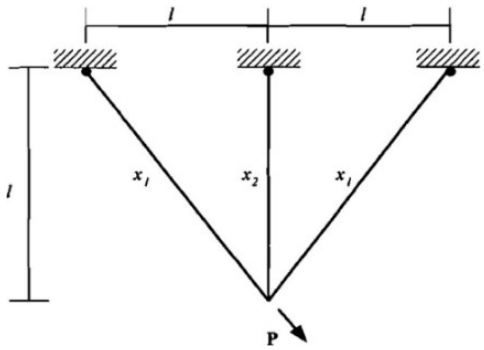


Figure 3: The three-bar truss design problem

The results of Table 4 show that there are no significant difference between decision variables, constraints values, and the best solutions. The best result is related to the FA algorithm, and then the CHS algorithm outperforms HS, SFLA, and TLBO. The amount of NFE for solving this problem is assumed 24,000, and the results of the decision variables and the best solution are equal to

$$X^* = (0.7886783, 0.4082392),$$
$$f(X^*) = 263.8958436.$$

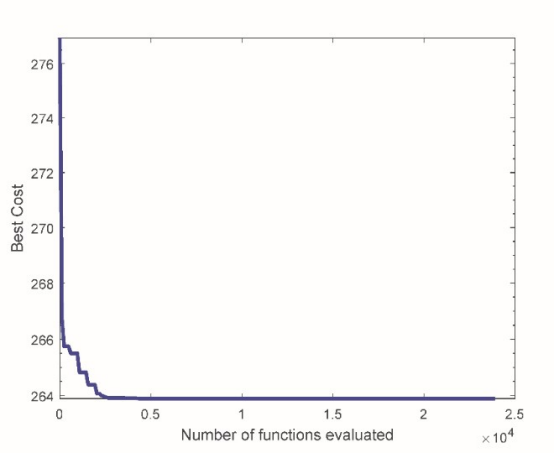


Figure 4: Convergence rate of the CHS for finding best solution of three-bar truss design problem

The Pressure vessel design problem As shown in Figure 5, in the design of a pressure vessel, a cylindrical pressure tank is used that has two joints in two cylindrical heads. The purpose of this problem is to minimize the weight of the pressure vessel, including the weight of the shell and the welding lines. The mathematical model of this problem is shown as follows:

$$\begin{aligned}
\text{Min} f(x) \; &= 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
&s.t. \\
&g_1(x) = -x_1 + 0.0193x_3 \le 0, \\
&g_2(x) = -x_2 + 0.00954x_3 \le 0, \\
&g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1,296,000 \le 0 \\
&g_4(x) = x_4 - 240 \le 0, \\
&0 \le x_i \le 100, \quad i = 1, 2, \\
&10 \le x_i \le 200, \quad i = 3, 4.
\end{aligned} \tag{8}$$

Table 8: Comparative results of the CHS with FA, HS, SFLA, and TLBO in Pressure vessel design problem.

Parameters	FA	CHS	HS	SFLA	TLBO
x_1	0.8750(14 × 0.0625)	0.8125(13 × 0.0625)	1(16 × 0.0625)	1.1875(19 × 0.0625)	0.8125(13 × 0.0625)
x_2	0.4375(7 × 0.0625)	0.4375(7 × 0.0625)	0.5000(8 × 0.0625)	0.5625(9 × 0.0625)	0.4375(7 × 0.0625)
x_3	4.533678756E+01	4.209844560E+01	5.121941472E+01	5.896226415E+01	4.209844560E+01
x_4	1.402538467E+02	1.766365958E+02	8.895572796E+01	4.004432558E+01	1.766365958E+02
g_1	3.330669074E-16	2.220446049E-16	-1.146529598E-02	-4.952830189E-02	-1.744160372E-13
g_2	-4.987046632E-03	-3.588082902E-02	-1.136678361E-02	3.330669074E-16	-3.588082902E-02
g_3	0	0	-1.304185484E-03	0	-3.213062882E-08
g_4	-9.974615329E+01	-6.336340416E+01	-1.510442720E+02	-1.999556744E+02	-6.336340416E+01
Best Value	**6.090526202E+03**	**6.059714335E+03**	**6.466011401E+03**	**7.050673234E+03**	**6.059714335E+03**

Based on this equation, (x_1 or T_s) and (x_2 or T_h) show the thickness of the shell and the thickness of the head, respectively. (x_3 or R) represents the inner radius, and (x_4 or L) shows the cylindrical section of the vessel. In order to solve this mixed-integer problem (MLP), the variables T_s and T_h are rounded, which are a coefficient of 0.0625, and R and L are continuous variables. Figure 6 shows the convergence rate of the CHS algorithm to find the best solution to this problem.

In order to solve this problem, the algorithms are implemented with NFE = 180,000. Based on Table 5, the CHS and TLBO algorithms have had better performance than the FA, HS, and SFLA algorithms. The values of decision variables and the value of the objective function at the best solution are

$$X* = (0.8125, 0.4375, 42.0984456, 176.6365958) \ ,$$

$$f(X*) = 6059.7143350.$$

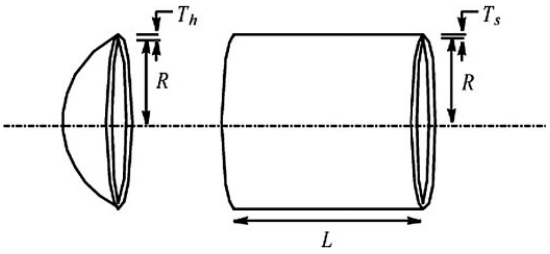


Figure 5: Pressure vessel design problem

Tension/compression spring design optimization problem The objective of this problem is to minimize tension/spring weight with a linear constraint and three nonlinear constraints [8]. Based on the following equation, the diameter of the wire (x_1), the mean diameter of the coil (x_3), and the number of active coils (x_2), are the decision variables of this problem:

$$\text{Min} f\,(x) \ = (x_3 + 2)\, x_2 x_1^2$$

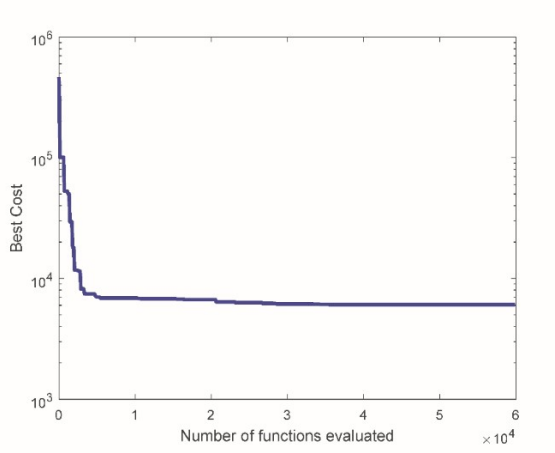


Figure 6: Convergence rate of the CHS for finding the best solution of pressure vessel design problem

$$
\begin{aligned}
& s.t. \\
& g_1(x) = 1 - \frac{x_2^3 x_3}{71,785 x_1^4} \leq 0, \\
& g_2(x) = \frac{4x_2^2 - x_1 x_2}{12,566\,(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \leq 0, \\
& g_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0, \\
& g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0, \\
& 0.05 \leq x_1 \leq 2, \qquad 0.25 \leq x_2 \leq 1.3, \qquad 2 \leq x_3 \leq 15.
\end{aligned} \tag{9}
$$

The schematic representation of the problem is shown in Figure 7. Figure 8 shows the convergence rate of the CHS algorithm in finding the optimal solution to the three-tension/compression spring design optimization problem.

Figure 7: Tension/compression spring structure

In order to evaluate the performance of the algorithms in this problem, the value of NFE is considered 60,000. The comparative results in Table 6 show that the best performance is related to the CHS algorithm. The FA and

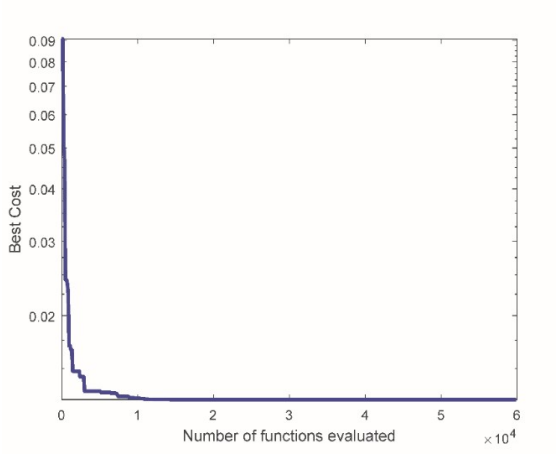


Figure 8: Convergence rate of the CHS for finding best solution of Tension/compression spring design optimization problem

Table 9: Comparative results of THE CHS with FA, HS, SFLA, and TLBO in Tension/compression spring design optimization problem.

Parameters	FA	CHS	HS	SFLA	TLBO
x_1	5.184988445E-02	5.184055530E-02	5.978313881E-02	5.743674882E-02	5.196149787E-02
x_2	3.605990697E-01	3.603732438E-01	5.845192287E-01	5.114076090E-01	3.633072944E-01
x_3	1.106499114E+01	1.107782927E+01	4.591474470	5.841058341	1.091284592E+01
g_1	5.184988445E-02	-2.220446049E-16	-5.553424742E-08	-2.220446049E-16	-5.536883352E-08
g_2	0	-2.220446049E-16	-5.749327642E-07	-3.330669074E-16	-1.695190543E-08
g_3	-4.061383701	-4.060945295	-4.352426075	-4.280629258	-4.066606162
g_4	-7.250340306E-01	-7.251908006E-01	-5.704650883E-01	-6.207704281E-01	-7.231541385E-01
Best Value	**1.266570321E-02**	**1.266565028E-02**	**1.377015419E-02**	**1.322883405E-02**	**1.266658115E-02**

TLBO algorithms have comparative results that show a better performance than HS and TLBO. The decision variables and the value of the objective function at the best solution are

$$X^* = (0.051840, 0.360373, 11.077829)\,,$$

$$f(X^*) = 0.0126657.$$

The welded beam design problem The aim of this problem is to minimize the cost of the welded beam based on the shear stress τ, bending stress in the beam σ, buckling load on the bar $\mathrm{p_c}$, and end deflection on the beam δ. Based on (10), the decision variables are the weld thickness (x_1), the weld length (x_2), the height of the bar (x_3), and the thickness of the bar (x_4). In Figure 9, the welded beam design problem is shown schematically. Figure 10 shows the convergence rate of the CHS algorithm in finding the optimal solution to this problem. Equation (10) shows that this problem has two linear and five nonlinear unequal constraints as follows:

$$
\begin{aligned}
\text{Min} f(x) \; =& 1.10471 x_1^2 x_2 + 0.004811 x_3 x_4 (14 + x_2) \\
& s.t. \\
g_1(x) =& \tau_x - \tau_{max} \leq 0, \\
g_2(x) =& \sigma_x - \sigma_{max} \leq 0, \\
g_3(x) =& x_1 - x_4 \leq 0, \\
g_4(x) =& 1.10471 x_1^2 + 0.04811 x_3 x_4 (14 + x_2) - 5 \leq 0, \\
g_5(x) =& 0.125 - x_1 \leq 0, \\
g_6(x) =& \delta_x - \delta_{max} \leq 0, \\
g_7(x) =& p - p_c(x) \leq 0,
\end{aligned} \tag{10}
$$

where

$$
\begin{aligned}
\tau(x) =& \sqrt{(\tau)^2 + 2\tau\tau \frac{x^2}{2R} + (\tau)^2}, \\
\tau =& \frac{p}{\sqrt{2} x_1 x_2}, \quad \tau = \frac{MR}{j}, \quad M = P\left(L + \frac{x_2}{2}\right), \\
R =& \sqrt{\frac{{x_2}^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \quad \delta(x) = \frac{4PL^3}{E {x_3}^3 x_4},
\end{aligned}
$$

$$
J = 2\left[\sqrt{2} x_1 x_2 \left\{\frac{{x_2}^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right\}\right], \quad \sigma(x) = \frac{6PL}{x_4 x_3^2},
$$

$$
\begin{aligned}
P_c(x) =& \frac{4.013E\sqrt{\frac{x_3^2 x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \\
P =& 6000\ lb, \quad L = 14\ in, \quad E = 30e6\ psi, \\
G =& 12e6\ psi, \quad \sigma_{max} = 30,000\ psi, \\
\tau_{max} =& 13,600\ psi, \\
\delta_{max} =& 0.25\ in, \quad 0.1 \leq x_1 \leq 2, \quad 0.1 \leq x_2 \leq 10, \\
0.1 \leq& x_3 \leq 10, \quad 0.1 \leq x_4 \leq 2.
\end{aligned}
$$

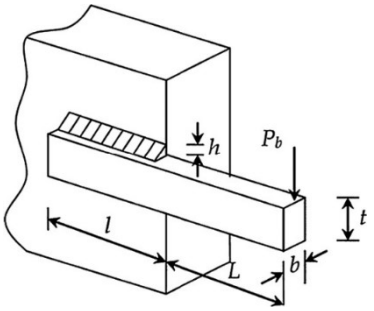


Figure 9: The welded beam design problem

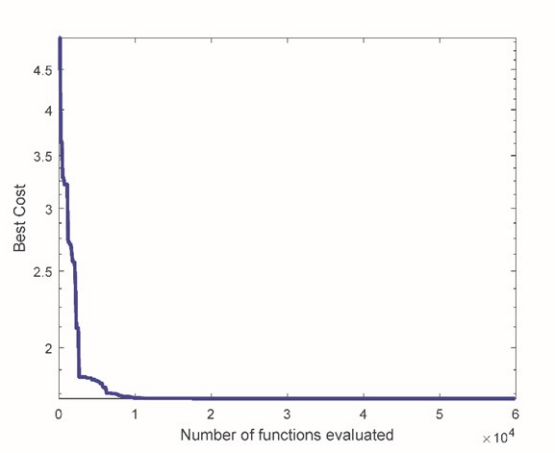


Figure 10: Convergence rate of the CHS for finding best solution of Welded beam design problem

Based on Table 7, with $NFE = 60,000$, each of the three algorithms, the FA, CHS, and TLBO algorithms, have had the same results in the values of decision variables, constraint values, and best solutions that have outperformed HS and SFLA. The decision variables and the value of the objective function at the best solution are

$$X^* = (0.2057296, 3.4704887, 9.0366239, 0.2057296)\,,$$

$$f(X^*) = 1.7248523.$$

Table 10: Comparative results of The CHS with FA, HS, SFLA, and TLBO in Welded beam design problem.

Parameters	**FA**	**CHS**	**HS**	**SFLA**	**TLBO**
x_1	2.057296398E-01	2.057296398E-01	2.350586911E-01	2.058573368E-01	2.057296398E-01
x_2	3.470488665	3.470488665	3.166035310	3.468820796	3.470488665
x_3	9.036623910	9.036623910	8.363477126	9.033820685	9.036623910
x_4	2.057296398E-01	2.057296398E-01	2.401792830E-01	2.058573368E-01	2.057296398E-01
g_1	0	0	-1.177816867E-02	-1.818989404E-12	0
g_2	-7.275957614E-12	-7.275957614E-12	-6.404832093E-04	0	-2.182787284E-11
g_3	6.695199950E-12	6.659200968E-12	-5.120591985E-03	8.307909916E-12	6.767586491E-12
g_4	-3.432983785	-3.432983785	-3.335285590	-3.432642630	-3.432983785
g_5	-8.072963979E-02	-8.072963979E-02	-1.100586911E-01	-8.085733680E-02	-8.072963979E-02
g_6	-2.355403226E-01	-2.355403226E-01	-2.343765145E-01	-2.355358357E-01	-2.355403226E-01
g_7	0	0	-3.061301599E+03	-9.953428131	0
Best Value	**1.724852309**	**1.724852309**	**1.852177649**	**1.725311383**	**1.724852309**

5 Conclusion

In this study, the CHS (crocodile optimization algorithm) was investigated as a swarm intelligence algorithm. The CHS algorithm was classified as the population-based algorithm that simulates the behavior of the crocodiles in finding food. Crocodiles have proper strategies while hunting. In this way, when hunting, the population is divided into two groups: the chasers and the ambushers. The chasers direct the prey to the attacking area without catching it, and the ambushers hide in the attacking area and attack fishes in the attacking area. Using this strategy, the prey was directed to the attacking area and eventually was hunted by the ambushers and chasers. In designing the CHS, two main equations were proposed for each group. The structure of the algorithm was designed in such a way that explorations are done in the first iterations, and exploitations are done in the last iterations. In order to evaluate the efficiency of the CHS algorithm, 20 classical test functions and four constrained engineering design problems were used. In classical benchmarks, 12 classic test functions were in two dimensions and eight classic test functions were investigated in 30 dimensions. In sum, it can be said that in the classical test functions, the CHS algorithm shows a better performance than the other algorithms in terms of convergence rate in less iteration and the exploration and exploitation capability. In addition, in constrained engineering design problems, the CHS algorithm was outperformed and able to produce good results. This performance showed that there is a good balance between exploration and exploitation in the CHS algorithm and that the equations are correctly designed.

References

1. Abdel-Basset, M., Abdel-Fatah, L. and Sangaiah, A.K. *Metaheuristic algorithms: A comprehensive review*, Computational intelligence for multimedia big data on the cloud with engineering applications, 2018, Elsevier,

185–231.

2. Akay, B. and Karaboga, D. *A modified artificial bee colony algorithm for real-parameter optimization,* Inf. Sci. 192 (2012), 120–142.

3. Alba, E. and Dorronsoro, B. *The exploration/exploitation tradeoff in dynamic cellular genetic algorithms,* IEEE Trans. Evol. Comput. 9(2) (2005), 126–142.

4. Askarzadeh, A. *Bird mating optimizer: an optimization algorithm inspired by bird mating strategies,* Commun. Nonlinear Sci. Numer. Simul. 19(4) (2014), 1213–1228.

5. Askarzadeh, A. *A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm,* Comput. Struct. 169 (2016), 1–12.

6. Balavand, A. *A new feature clustering method based on crocodiles hunting strategy optimization algorithm for classification of MRI images,* Vis. Comput. (2021) 1–30.

7. Clerc, M., *Particle swarm optimization*, Vol. 93. John Wiley & Sons, 2010.

8. Coello, C.A.C., *Use of a self-adaptive penalty approach for engineering optimization problems,* Comput. Ind. 41(2) (2000), 113–127.

9. Coello, C.A.C., *Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art,* Comput. Methods Appl. Mech. Eng. 191(11-12) (2002), 1245–1287.

10. Cuevas, E., Cienfuegos, M., Zaldívar, D., and Pérez-Cisneros, M.*A swarm optimization algorithm inspired in the behavior of the social-spider,* Expert Syst. Appl. 40 (16) (2013), 6374–6384.

11. Dasgupta, D. *An overview of artificial immune systems and their applications, in Artificial immune systems and their applications,* Artificial immune systems and their applications (1993), 3–21.

12. Dinets, V. *Apparent coordination and collaboration in cooperatively hunting crocodilians,* Ethol. Ecol. Evol. 27(2) (2015), 244–250.

13. Dorigo, M. and Gambardella, L.M. *Ant colony system: a cooperative learning approach to the traveling salesman problem,* IEEE Trans. Evol. Comput. 1(1) (1997), 53–66.

14. Duman, E., Uysal, M. and Alkaya, A.F. *Migrating birds optimization: A new metaheuristic approach and its performance on quadratic assignment problem,* Inform. Sci. 217 (2012), 65–77.

15. Eskandar, H., Sadollah, A., Bahreininejad, A., and Hamdi, M. *Water cycle algorithm-A novel metaheuristic optimization method for solving constrained engineering optimization problems,* Comput. Struct. 110 (2012), 151–166.

16. Eusuff, M., Lansey, K. and Pasha, F. *Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization,* Eng. Optim. 38(2) (2006), 129–154.

17. Faramarzi, A., Heidarinejad, M., Mirjalili, S., and Gandomi, A.H. *Marine predators algorithm: A nature-inspired Metaheuristic,* Expert Syst. Appl. 152 (2020) 113377.

18. Faris, H., Aljarah, I., Al-Betar, M. A., and Mirjalili, S. *Grey wolf optimizer: a review of recent variants and applications,* Neural Comput. Appl. 30(2) (2018), 413-435.

19. Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., and Chen, H. *Harris hawks optimization: Algorithm and applications,* Future Gener. Comput. Syst. 97 (2019), 849–872.

20. Jain, M., Singh, V. and Rani, A. *A novel nature-inspired algorithm for optimization: Squirrel search algorithm,* Swarm Evol. Comput. 44 (2019), 148–175.

21. José-García, A. and Gómez-Flores, W. *Automatic clustering using nature-inspired metaheuristics: A survey,* Appl. Soft Comput. 41 (2016), 192–213.

22. Kashan, A.H. *An efficient algorithm for constrained global optimization and application to mechanical engineering design: League championship algorithm (LCA),* Comput Aided Des. 43(12) (2011), 1769–1792.

23. Kaveh, A. and Farhoudi, N. *A new optimization method: Dolphin echolocation,*Adv. Eng. Softw. 59 (2013), 53–70.

24. Kennedy, J. and Eberhart, R. *Particle swarm optimization,* in Proceedings of ICNN'95-international conference on neural networks. 1995.

25. Kiran, M.S., *TSA: Tree-seed algorithm for continuous optimization,* Expert Syst. Appl. 42(19) (2015), 6686–6698.

26. Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. *Optimization by simulated annealing*, Science 220(4598) (1983), 671–680.

27. Li, J.Q., Sang, H.Y., Han, Y.Y., Wang, C.G. and Gao, K.Z. *Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions,* J. Clean. Prod. 181 (2018), 584–598.

28. Lin, L. and Gen, M. *Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation,* Soft Comput. 13, (2) (2009), 157–168.

29. Martin, R. and Stephen, W. *Termite: A swarm intelligent routing algorithm for mobilewireless Ad-Hoc networks,* Stigmergic optimization. Springer, Berlin, Heidelberg, 2006. 155–184.

30. Michalewicz, Z. *A Survey of constraint handling techniques in evolutionary computation methods,* Evol Comput. 4 (1995), 135–155.

31. Mirjalili, S. *Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm,* Knowl. Based Syst. 89 (2015), 228–249.

32. Mirjalili, S. and Lewis, A. *The whale optimization algorithm,* Adv. Eng. Softw., 95 (2016), 51–67.

33. Mirjalili, S., Mirjalili, S.M. and Lewis, A. *Grey wolf optimizer,* Adv. Eng. Softw. 69 (2014), 46–61.

34. Mucherino, A. and Seref. O. *Monkey search: A novel metaheuristic search for global optimization*, AIP conference proceedings. Vol. 953. No. 1. American Institute of Physics, 2007.

35. Nezhad, A.M., Shandiz, R.A. and Jahromi, A.E. *A particle swarm-BFGS algorithm for nonlinear programming problems,* Comput. Oper. Res. 40(4) (2013), 963–972.

36. Olorunda, O. and Engelbrecht. A.P. *Measuring exploration/exploitation in particle swarms using swarm diversity,*In 2008 IEEE congress on evolutionary computation (IEEE world congress on computational intelligence), pp. 1128–1134. IEEE, 2008.

37. Pan, W.-T. *A new fruit fly optimization algorithm: taking the financial distress model as an example,* Knowl. Based Syst. 26 (2012), 69–74.

38. Ray, T. and Liew, K.M. *Society and civilization: An optimization algorithm based on the simulation of social behavior,* IEEE Trans. Evol. Comput. 7(4) (2003), 386–396.

39. Sadollah, A., Bahreininejad, A., Eskandar, H., and Hamdi, M. *Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems,* Appl. Soft Comput. 13(5) (2013), 2592–2612.

40. Saremi, S., Mirjalili, S. and Lewis, A. *Grasshopper optimisation algorithm: theory and application,* Adv. Eng. Softw. 105 (2017), 30–47.

41. Yang, X.-S., *Firefly algorithm, stochastic test functions and design optimisation,*Int. J. Bio-Inspired Comput. 2(2) (2010), 78–84.

42. Yang, X.-S. and Deb. S. *Cuckoo search via Lévy flights,* In 2009 World congress on nature & biologically inspired computing (NaBIC), pp. 210–214. IEEE, 2009.

43. Yang, X.-S. and He, X. *Bat algorithm: literature review and applications,* Int. J. Bio-Inspired Comput. 5(3) (2013), 141–149.

44. Yapici, H. and Cetinkaya, N. *A new meta-heuristic optimizer: Pathfinder algorithm,* Appl. Soft Comput. 78 (2019), 545–568.

45. Zhang, J., Xiao, M., Gao, L. and Pan, Q. *Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems,* Appl. Math. Model. 63 (2018), 464–490.

How to cite this article
A.R. Balavand Crocodile Hunting Strategy (CHS): A comparative study using benchmark functions. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 397-425. doi: DOI:10.22067/ijnao.2022.71418.1049.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 426–448
DOI: 10.22067/IJNAO.2022.74201.1083
https://ijnao.um.ac.ir/

How to cite this article
Research Article

Modification of the double direction approach for solving systems of nonlinear equations with application to Chandrasekhar's Integral equation

A.I. Kiri, M.Y. Waziri and A.S. Halilu*

Abstract

This study aims to present an accelerated derivative-free method for solving systems of nonlinear equations using a double direction approach. The approach approximates the Jacobian using a suitably formed diagonal matrix by applying the acceleration parameter. Moreover, a norm descent line search is employed in the scheme to compute the optimal step length. Under the primary conditions, the proposed method's global convergence is proved. Numerical results are recorded in this paper using a set of large-scale test problems. Moreover, the new method is successfully used to address the problem of Chandrasekhar's integral equation problem appearing in radiative heat transfer. This method outperforms the existing Newton and inexact double step length methods.

AMS subject classifications (2020): 65K05, 90C53, 65D32, 34G20.

Keywords: Credit default swap (CDS); Acceleration parameter; Matrix-free, Inexact line search; Jacobian matrix.

---

* Corresponding author

Received 15 December 2021; revised 8 February 2022; accepted 28 February 2022

Aliyu Ibrahim Kiri
Department of Mathematics, Department of Mathematical Sciences, Bayero University, Kano, Nigeria. e-mails: aikiri.mth@buk.edu.ng

Mohammed Yusuf Waziri
Numerical Optimization Group, Department of Mathematical Sciences, Bayero University, Kano, Nigeria. e-mails: mywaziri.mth@gmail.com

Abubakar Sani Halilu
Numerical Optimization Group, Bayero University, Kano, Nigeria,
Department of Mathematics, Sule Lamido University, Kafin Hausa, Nigeria. e-mail: abubakars.halilu@slu.edu.ng

1 Introduction

Scientists are interested in nonlinear problems because most engineering, biology, mathematics, physics, and other science problems are naturally nonlinear. The standard nonlinear equation system is represented by

$$F(x) = 0, \tag{1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is a nonlinear map. The space $\mathbb{R}^n$ denotes the n-dimensional real space, $\|\cdot\|$ is the Euclidean norm, and $F_k = F(x_k)$ is used throughout this paper. Further applications of problem (1) can be found in chemical equilibrium systems [16] and signal and image processing [25]. The Chandrasekhar H-equation that arises in the theory of radioactive heat transfer is a nonlinear integral equation that can be discretized into nonlinear equations [20]. Iterative methods for solving these problems include the Newton and quasi-Newton methods [4, 21, 26, 14], Levenberg–Marquardt methods [15, 13, 12], matrix-free methods [17, 1, 8], and tensor methods [2]. Typically, the iterative formula for solving these methods is given by

$$x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, \ldots, \tag{2}$$

where x_{k+1} represents a current iterate, x_k is the previous iterate, α_k is a step length, and d_k is the search direction can be calculated by solving system of linear equations as follows:

$$F_k + F'_k d_k = 0, \tag{3}$$

where F'_k is the Jacobian matrix of F_k at x_k. One of the most important requirements of the line search is to reduce the function values sufficiently [9, 11], as shown below:

$$\|F_{k+1}\| \leq \|F_k\|. \tag{4}$$

Irrespective of how appealing the Newton and quasi-Newton approaches are, the Jacobian matrix or its approximation can be calculated at each iteration, making them unsuitable for solving large-scale problems. Due to the drawbacks of these methods, the double direction technique has been proposed [6], with the following iterate:

$$x_{k+1} = x_k + \alpha_k d_k + \alpha_k^2 b_k, \tag{5}$$

where d_k and b_k are search directions, respectively.

Suppose that f is a merit function defined by

$$f(x) = \frac{1}{2}\|F(x)\|^2. \tag{6}$$

Then problem (1) is analogous to the unconstrained optimization problem described below:

$$\min f(x), \quad x \in \mathbb{R}^n, \tag{7}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, and condition (4) is equivalent to

$$f(x_k + \alpha_k d_k) \leq f(x_k). \tag{8}$$

The iterative method generating the sequence $\{x_k\}$ that satisfies (4) is called the norm descent method [9]. If d_k is a descent direction of f at x_k, then condition (8) holds for all $\alpha_k > 0$ small enough. The Newton method (NM) with line search is norm descent. Nonetheless, d_k might not be a descent direction of f at x_k for quasi-Newton methods, even if the approximation of the Jacobian matrix B_k is positive definite and symmetric. Li and Fukushima [14] proposed an approximately norm-descent line search approach. However, the proposed method is not norm descent, but they established a global convergence theorem under the assumption that Jacobian is uniformly nonsingular.

The concept of the double direction approach was suggested by Duranović-Miličić [6] by using a multi-step iterative scheme and curve search to generate new iterates. However, in [7], another double direction algorithm was also presented to minimize nondifferentiable functions. Motivated by the work presented in [7], Petrović and Stanimirović [19] suggested a double direction model for solving unconstrained optimization problems. They used the acceleration parameter γ_k to approximate the Hessian matrix, that is,

$$\bigtriangledown^2 f(x_k) \approx \gamma_k I, \tag{9}$$

where I is the identity matrix, and the sequence of iterates $\{x_k\}$ is generated using (5). The attractive feature of the scheme in [19] is that the two directions presented in their work are derivative-free. Therefore, it enables their method to solve large-scale problems. However, the literature is infrequent to study derivative-free double direction methods for solving nonlinear equations. Based upon the idea presented in [19], Halilu and Waziri used the scheme in (5) to propose a method for solving a system of nonlinear equations using a double direction approach. They used the acceleration parameter $\gamma_k > 0$ in their work to approximate the Jacobian matrix, that is,

$$F'_k \approx \gamma_k I, \tag{10}$$

where I is an identity matrix and the acceleration parameter is derived as

$$\gamma_{k+1} = \frac{y_k^T y_k}{(\alpha_k + \alpha_k^2 \gamma_k) y_k^T d_k}. \tag{11}$$

The method's global convergent is proved by assuming that the Jacobian of F is positive definite and bounded. The double direction scheme is justified

by the fact that scheme (5) contains two corrections. If one of the iterative corrections fails, then the system will be corrected by the second.

The implementation of double direction is additionally enhanced by Petrović [18], where the double step length scheme for the unconstrained optimization problem is presented as

$$x_{k+1} = x_k + \alpha_k d_k + \beta_k b_k, \tag{12}$$

where α_k and β_k are two different step lengths. The numerical results indicated the approach is quite effective compared to the double direction method in [19]. The authors in [8] incorporated the concept in (12) and transformed the double step length method for solving (1) to improve the numerical results and global convergence properties of the double direction scheme. The numerical results exhibited that the method in [8] is more reasonable than the method in [10] because it converges faster. Furthermore, the method [8] is globally converged using the line search proposed in [14]. Motivated by the work in [8], Halilu and Waziri [11] presented an inexact double step length method for solving (1). The attractive feature of this method is that it has a double step length and a single direction that satisfies the decent properties independent of line search. Despite the good convergence properties of the method in [10], its numerical performance is defined as weaker. Therefore, motivated by this reason, we aim to develop a globally converged derivative-free method with a line search to solve a system of nonlinear equations without calculating the Jacobian matrix.

Table 1: Authors' contribution table

Author's Name	Derivative-free	Matrix-free	Double Direction	Global Convergence	Application
Duranović-Miličić [6]	✓	✓	✓	✓	
Halilu and Waziri [10]	✓	✓	✓	✓	
Duranovic-Milicic [7]	✓	✓	✓	✓	
Musa, Waziri, and Halilu [17]	✓			✓	
Abdullahi, Waziri, and Halilu [1]	✓	✓		✓	✓
Petrović and Stanimirović [19]	✓	✓	✓	✓	
Kanzow et al. [12]	✓			✓	
Halilu and Waziri [11] ✓	✓	✓	✓	✓	
Yuan and Lu [26]	✓			✓	
Waziri et al. [23]	✓	✓			✓
Halilu and Waziri [8]	✓	✓	✓	✓	
Li and Fukushima [14]	✓			✓	
Halilu and Waziri [9]	✓	✓	✓	✓	
petrović [18]	✓	✓	✓	✓	
This article	✓	✓	✓	✓	✓

The research gap between the existing method and this article is described in Table 1 above. The table clearly shows that only the proposed method is derivative-free, matrix-free, double direction method, globally convergent, and can be applied to solve discretized Chandrasekhar's integral equation among the listed articles.

We now describe how the paper is structured. The proposed method's algorithm will be presented in section 2. Section 3 illustrates the convergence

results. Section 4 contains a list of numerical experiments and applications of the proposed method to Chandrasekhar's integral equation, which arises in radiative heat transfer. Section 5 concludes the paper.

2 Main result

In this section, we present the algorithm of our method. We suggest that the d_k and b_k in (5) are defined as follows:

$$d_k = -\gamma_k^{-1} F_k \tag{13}$$

and

$$b_k = -F_k, \tag{14}$$

where $\gamma_k > 0$ is an acceleration parameter. By substituting (13) and (14) into (5), we obtain

$$x_{k+1} = x_k - (\alpha_k + \alpha_k^2 \gamma_k)\gamma_k^{-1} F_k. \tag{15}$$

The acceleration parameter can be obtained using Taylor's series expansion below:

$$F_{k+1} \approx F_k + F'(\psi)(x_{k+1} - x_k). \tag{16}$$

By multiplying (16) through by θ_k, we have

$$\theta_k F_{k+1} \approx \theta_k F_k + \theta_k F'(\psi)(\alpha_k + \alpha_k^2 \gamma_k) d_k, \tag{17}$$

where $\theta_k > 0$ and ψ satisfies the conditions $\psi \in [x_k, x_{k+1}]$ and

$$\psi = x_k + \zeta(x_{k+1} - x_k), \quad 0 \le \zeta \le 1. \tag{18}$$

Taking $\zeta = 1$ in (18), obtain $\psi = x_{k+1}$.
We like to make Jacobian approximations via

$$\theta_k F'(\psi) \approx \gamma_{k+1} I. \tag{19}$$

Using (17) and (19), it is easy to confirm that

$$\gamma_{k+1} s_k = \theta_k y_k, \tag{20}$$

where $s_k = (\alpha_k + \alpha_k^2 \gamma_k) d_k$, $y_k = F_{k+1} - F_k$, and $\theta_k = \dfrac{s_k^T s_k}{y_k^T s_k}$ (see [24]).

The proposed acceleration parameter is defined by multiplying y_k^T on both sides of (20)

$$\gamma_{k+1} = \frac{\|s_k\|^2 \|y_k\|^2}{(\alpha_k + \alpha_k^2 \gamma_k)^2 (y_k^T d_k)^2}. \tag{21}$$

Our proposed scheme is given by equation (22) below based on (13) and (15):

$$x_{k+1} = x_k + (\alpha_k + \alpha_k^2\gamma_k)d_k. \tag{22}$$

Algorithm 1 Modification of the double direction approach (MDFDD)

Input: Given x_0, $\gamma_0 = 1$, $\epsilon = 10^{-5}$, $\phi_1 > 0$, $\phi_2 > 0$, and $r \in (0,1)$, set $k = 0$.
Step 1: Compute F_k.
Step 2: If $\|F_k\| \leq \epsilon$, then stop; otherwise, proceed to Step 3.
Step 3: Calculate search direction $d_k = -\gamma_k^{-1}F_k$.
Step 4: Set $x_{k+1} = x_k + (\alpha_k + \alpha_k^2\gamma_k)d_k$, where $\alpha_k = r^{a_k}$ with a_k being the smallest nonnegative integer a such that

$$f(x_k + (\alpha_k + \alpha_k^2\gamma_k)d_k) - f(x_k) \leq -\phi_1\|\alpha_k F_k\|^2 - \phi_2\|\alpha_k d_k\|^2 + \tau_k f(x_k). \tag{23}$$

Let $\{\tau_k\}$ be a given positive sequence such that

$$\sum_{k=0}^{\infty} \tau_k < \tau < \infty. \tag{24}$$

Step 5: Compute F_{k+1}.
Step 6: Determine $\gamma_{k+1} = \dfrac{\|s_k\|^2\|y_k\|^2}{(\alpha_k + \alpha_k^2\gamma_k)^2(y_k^T d_k)^2}$.
Step 7: Consider $k = k + 1$ and go to Step 2.

3 Convergence Analysis

We present how the proposed Algorithm 2 (MDFDD) converges globally in this section. Let us start by defining the level set

$$\Omega = \{x \| \|F(x)\| \leq \|F(x_0)\|\}. \tag{25}$$

However, we require the following assumptions:

Assumption 1. However, we state the following assumptions:

1. There exists $x^* \in \mathbb{R}^n$ such that $F(x^*) = 0$.
2. F is continuously differentiable in some neighborhood say Q of x^* containing Ω.
3. The Jacobian of F is bounded and positive definite on Q. That is, there exist positive constants $H > h > 0$ such that

$$\|F'(x)\| \leq H \quad \text{for all } x \in Q, \tag{26}$$

and

$$h\|d\|^2 \leq d^T F'(x)d \quad \text{for all } x \in Q, d \in \mathbb{R}^n. \tag{27}$$

Remark 1. We make the following remark:
Assumption 1 implies that there exist constants $H > h > 0$ such that

$$h\|d\| \leq \|F'(x)d\| \leq H\|d\| \quad \text{for all } x \in Q, d \in \mathbb{R}^n, \tag{28}$$

$$h\|x-y\| \leq \|F(x)-F(y)\| \leq H\|x-y\| \quad \text{for all } x, y \in Q. \tag{29}$$

Since $\gamma_k I$ approximates F'_k along s_k, the following assumption can be made.

Assumption 2. $\gamma_k I$ is a good approximation to F'_k, that is,

$$\|(F'_k - \gamma_k I)d_k\| \leq \varepsilon\|F_k\|, \tag{30}$$

where $\varepsilon \in (0,1)$ is a small quantity [26].

Lemma 1. Suppose that Assumption 2 holds, and let $\{x_k\}$ be generated by the MDFDD algorithm. Then d_k is a sufficient descent direction for $f(x_k)$ at x_k, that is,

$$\nabla f(x_k)^T d_k < c\|F_k\|^2, \quad c > 0. \tag{31}$$

Proof. From (13), we have

$$\begin{aligned} \nabla f(x_k)^T d_k &= F_k^T F'_k d_k \\ &= F_k^T[(F'_k - \gamma_k I)d_k - F_k] \\ &= F_k^T(F'_k - \gamma_k I)d_k - \|F_k\|^2, \end{aligned} \tag{32}$$

by the Cauchy–Schwarz inequality, we have

$$\begin{aligned} \nabla f(x_k)^T d_k &\leq \|F_k\|\|(F'_k - \gamma_k I)d_k\| - \|F(x_k)\|^2 \\ &\leq -(1-\epsilon)\|F(x_k)\|^2. \end{aligned} \tag{33}$$

This lemma is true for $\varepsilon \in (0,1)$.

We can conclude from Lemma 1 that the norm function $f(x_k)$ is a descent along d_k, which means that $\|F_{k+1}\| \leq \|F_k\|$ is true. □

Lemma 2. Suppose that Assumption 1 holds, and let $\{x_k\}$ be generated by the MDFDD algorithm. Then $\{x_k\} \subset \Omega$.

Proof. From Lemma 1, we have $\|F_{k+1}\| \leq \|F_k\|$. Furthermore, for all k,

$$\|F_{k+1}\| \leq \|F_k\| \leq \|F_{k-1}\| \leq \cdots \leq \|F_0\|.$$

This means that $\{x_k\} \subset \Omega$. □

Lemma 3 (see [26])**.** Suppose that Assumption 1 holds, and let $\{x_k\}$ be generated by the MDFDD algorithm. Then there exists a constant $m > 0$ such that for all k,

$$y_k^T s_k \geq h\|s_k\|^2. \tag{34}$$

Lemma 4. Suppose that Assumption 1 holds and that $\{x_k\}$ is generated by the MDFDD algorithm. Then

$$\lim_{k\to\infty} \|\alpha_k d_k\| = \lim_{k\to\infty} \|s_k\| = 0 \tag{35}$$

and

$$\lim_{k\to\infty} \|\alpha_k F_k\| = 0. \tag{36}$$

Proof. By (23) for all $k > 0$

$$\begin{aligned} \phi_2\|\alpha_k d_k\|^2 &\leq \phi_1\|\alpha_k F_k\|^2 + \phi_2\|\alpha_k d_k\|^2 \\ &\leq \|F_k\|^2 - \|F_{k+1}\|^2 + \tau_k\|F_k\|^2. \end{aligned} \tag{37}$$

By summing the above inequality, we have

$$\begin{aligned} \phi_2 \sum_{i=0}^{k} \|\alpha_i d_i\|^2 &\leq \sum_{i=0}^{k} \left(\|F_i\|^2 - \|F_{i+1}\|^2\right) + \sum_{i=0}^{k} \eta_i\|F_i\|^2, \\ &= \|F_0\|^2 - \|F_{k+1}\|^2 + \sum_{i=0}^{k} \tau_i\|F_i\|^2, \\ &\leq \|F_0\|^2 + \|F_0\|^2 \sum_{i=0}^{k} \tau_i, \\ &\leq \|F_0\|^2 + \|F_0\|^2 \sum_{i=0}^{\infty} \tau_i. \end{aligned} \tag{38}$$

From the level set and the fact that $\{\tau_k\}$ satisfies (24), then the series $\sum_{i=0}^{\infty} \|\alpha_i d_i\|^2$ converges. This implies (35). Using the same logic as above, but this time with $\phi_1\|\alpha_k F_k\|^2$ on the left, we obtain (36). □

Lemma 5. Suppose that Assumption 1 holds, and let $\{x_k\}$ be generated by the MDFDD algorithm. Then there exists a constant $m_1 > 0$ such that for all $k > 0$,

$$\|d_k\| \leq B. \tag{39}$$

Proof. From (13) and (21), we have

$$\begin{aligned}\|d_k\| &= \left\| -\frac{(y_{k-1}^T s_{k-1})^2 F_k}{\|y_{k-1}\|^2 \|s_{k-1}\|^2} \right\| \\ &\leq \frac{\|F_k\| \|s_{k-1}\|^2 \|y_{k-1}\|^2}{\|s_{k-1}\|^2 \|y_{k-1}\|^2} \\ &\leq \|F_0\|.\end{aligned} \tag{40}$$

Choosing $B = \|F_0\|$, we have (39). □

Theorem 1. Suppose that Assumption 1 holds, and let $\{x_k\}$ be generated by the MDFDD algorithm. Assume further, for all $k > 0$,

$$\alpha_k \geq \lambda \frac{|F_k^T d_k|}{\|d_k\|^2}, \tag{41}$$

where λ is some positive constant. Then

$$\lim_{k\to\infty} \|F_k\| = 0. \tag{42}$$

Proof. From Lemma 5, we have (39). Also, from (35) and the boundedness of $\{\|d_k\|\}$, we have

$$\lim_{k\to\infty} \alpha_k \|d_k\|^2 = 0. \tag{43}$$

From (41) and (43), we have

$$\lim_{k\to\infty} |F_k^T d_k| = 0. \tag{44}$$

Also, from (13), we have

$$F_k^T d_k = -\gamma_k^{-1} \|F_k\|^2, \tag{45}$$

$$\begin{aligned}\|F_k\|^2 &= \| - F_k^T d_k \gamma_k \| \\ &\leq |F_k^T d_k| |\gamma_k|.\end{aligned} \tag{46}$$

Since

$$\gamma_k^{-1} = \frac{(y_{k-1}^T s_{k-1})^2}{\|y_{k-1}\|^2 \|s_{k-1}\|^2} \geq \frac{h^2 \|s_{k-1}\|^4}{\|y_{k-1}\|^2 \|s_{k-1}\|^2} \geq \frac{h^2 \|s_{k-1}\|^2}{H^2 \|s_{k-1}\|^2} = \frac{h^2}{H^2},$$

then

$$|\gamma_k^{-1}| \geq \frac{h^2}{H^2}.$$

Therefore from (46), we have

$$\|F_k\|^2 \leq |F_k^T d_k| \left(\frac{H^2}{h^2} \right). \tag{47}$$

As a result,

$$0 \leq \|F_k\|^2 \leq |F_k^T d_k| \left(\frac{H^2}{h^2}\right) \longrightarrow 0. \tag{48}$$

Hence,

$$\lim_{k\to\infty} \|F_k\| = 0. \tag{49}$$

□

4 Numerical experiments

The first part of this section provides numerical results to demonstrate the efficacy of the proposed method by comparing it with existing methods.

- IDFDD: Algorithm 1 proposed in [10].
- IDSL: Algorithm 1 proposed in [11].

The MDFDD method is used in the second part to solve the problem of Chandrasekhar's integral equation in radiative heat transfer. The computer codes used were written in MATLAB 9.4.0 (R2018a) and ran on a computer with a 1.80 GHz CPU processor and 8 GB RAM.

4.1 Experiment of some nonlinear systems of equations

In the experiments, we implemented the three algorithms using the same line search (23), with $\phi_1 = \phi_2 = 10^{-4}$, $r = 0.2$, and $\tau_k = \dfrac{1}{(k+1)^2}$. The iteration is set to stop for the three methods if $\|F_k\| \leq 10^{-5}$ or when the number of iterations overreaches 1000, but there is no x_k meeting the stopping criterion. The numerical effects of the three methods are shown in Tables 3–9, where "ITRN," "CTM(S)," and "IP" represent the total number of iterations, CPU time (in seconds), and initial points, respectively. In addition, $\|Fk\|$ represents the residual value at the stopping point. The symbol "-" indicates failure due to a memory requirement or when some iterations exceed 1000. We tested the three methods on the current seven test problems, each with a different set of initial points and dimensions (n values). The experiment was carried out with the dimensions 100, 1,000, 2,000, 10,000, 50,000, and 100,000 to demonstrate the comprehensive numerical experiments of the MDFDD, IDFDD, and IDSL methods. Table 2 contains the starting points for the test problems.

The experiments made use of the following test problems:

Problem 1 [8]

$F_1 = x_1 - e^{cos\left(\frac{x_1+x_2}{n+1}\right)}$,

Table 2: Initial points used in test problems

INITIAL POINTS (IP)	VALUES
IP1	$\left(\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}\right)^T$
IP2	$\left(\frac{1}{5}, \frac{1}{5}, \ldots, \frac{1}{5}\right)^T$
IP3	$\left(\frac{3}{2}, \frac{3}{2}, \ldots, \frac{3}{2}\right)^T$
IP4	$\left(\frac{2}{5}, \frac{2}{5}, \ldots, \frac{2}{5}\right)^T$
IP5	$\left(0, \frac{1}{2}, \frac{2}{3}, \ldots, 1 - \frac{1}{n}\right)^T$
IP6	$\left(\frac{1}{4}, \frac{-1}{4}, \ldots, \frac{(-1)^n}{4}\right)^T$
IP7	$\left(1, \frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{n}\right)^T$.

$F_i = x_i - e^{cos\left(\frac{x_{i-1}+x_i+x_{i+1}}{n+1}\right)}$,
$F_n = x_n - e^{cos\left(\frac{x_{n-1}+x_n}{n+1}\right)}, \quad i = 2, 3, \ldots, n-1.$

Problem 2 [11]
$F_i(x) = x_i(1 + x_i x_{n-2} x_{n-1} x_n) - 2 + (1 - x_i^2), \quad i = 1, 2, \ldots, n.$

Problem 3 [24]
$F_i(x) = x_i - x_i\left(\sin x_i - \frac{11}{50}\right) + 2, \quad i = 1, 2, \ldots, n.$

Problem 4 [10]
$F_1(x) = (x_1^2 + x_2^2)x_1 - 1,$
$F_i(x) = (x_{i-1}^2 + 2x_i^2 + x_{i+1}^2)x_i - 1,$
$F_n(x) = (x_{n-1}^2 + x_n^2)x_n, \quad i = 2, 3, \ldots, n-1.$

Problem 5 [10]
$F_i(x) = 2x_i - \sin|x_i|, \quad i = 1, 2, \ldots, n.$

Problem 6 [11]
$F(x) = Ax + b_1,$
where $A = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}$, and $b_1 = (e_1^x - 1, \ldots, e_n^x - 1)^T$.

Problem 7 [8]

$$F(x) = Bx + b_2,$$

where $B = \begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}$, and $b_2 = (\sin x_1 - 1, \ldots, \sin x_n - 1)^T$.

Table 3: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 1

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	6	0.009	7.44E-06	60	0.031	8.35E-06	41	0.015	9.86E-06
	IP2	11	0.020	2.67E-06	60	0.023	9.48E-06	42	0.013	7.83E-06
	IP3	6	0.013	9.57E-06	58	0.022	7.93E-06	40	0.022	7.72E-06
	IP4	7	0.009	7.55E-06	60	0.020	8.73E-06	42	0.015	7.21E-06
	IP5	7	0.023	7.17E-06	59	0.023	8.78E-06	41	0.014	7.88E-06
	IP6	8	0.024	7.36E-06	61	0.018	8.49E-06	42	0.026	9.23E-06
	IP7	10	0.027	6.68E-06	61	0.019	7.64E-06	42	0.023	8.3E-06
1,000	IP1	3	0.005	4.35E-07	96	0.057	8.59E-06	45	0.031	7.51E-06
	IP2	3	0.005	5.13E-07	96	0.077	9.76E-06	45	0.051	8.52E-06
	IP3	3	0.007	2.09E-07	94	0.056	8.17E-06	43	0.029	8.41E-06
	IP4	3	0.005	4.6E-07	96	0.058	8.98E-06	45	0.057	7.84E-06
	IP5	3	0.009	3.17E-07	95	0.057	8.8E-06	44	0.035	8.34E-06
	IP6	3	0.007	6.38E-07	97	0.058	8.74E-06	46	0.055	7.03E-06
	IP7	3	0.011	5.66E-07	97	0.092	7.98E-06	45	0.051	9.17E-06
10,000	IP1	3	0.017	1.38E-10	111	0.486	9.05E-06	48	0.239	8.14E-06
	IP2	3	0.035	1.63E-10	112	0.492	7.81E-06	48	0.206	9.24E-06
	IP3	3	0.022	6.64E-11	109	0.477	8.6E-06	46	0.266	9.13E-06
	IP4	3	0.034	1.46E-10	111	0.485	9.45E-06	48	0.182	8.51E-06
	IP5	3	0.037	1E-10	110	0.483	9.23E-06	47	0.232	9.01E-06
	IP6	3	0.021	2.03E-10	112	0.500	9.2E-06	49	0.182	7.63E-06
	IP7	3	0.034	1.8E-10	112	0.490	8.42E-06	48	0.206	9.97E-06
50,000	IP1	3	0.103	4.96E-13	114	1.948	8.88E-06	50	0.752	8.92E-06
	IP2	3	0.087	5.96E-13	115	1.985	7.66E-06	51	0.778	7.09E-06
	IP3	2	0.072	8.87E-06	112	1.910	8.44E-06	48	0.718	1E-05
	IP4	3	0.113	4.96E-13	114	1.949	9.28E-06	50	0.748	9.32E-06
	IP5	3	0.101	2.99E-13	113	1.934	9.05E-06	49	0.903	9.87E-06
	IP6	3	0.093	6.95E-13	115	1.978	9.03E-06	51	0.752	8.36E-06
	IP7	3	0.079	5.96E-13	115	1.933	8.27E-06	51	0.760	7.65E-06
100,000	IP1	2	0.082	6.57E-06	115	4.097	9.54E-06	51	1.465	8.83E-06
	IP2	2	0.116	7.75E-06	116	4.347	8.23E-06	52	1.534	7.02E-06
	IP3	2	0.104	3.14E-06	113	3.826	9.07E-06	49	1.392	9.9E-06
	IP4	2	0.108	6.95E-06	115	3.872	9.97E-06	51	1.497	9.23E-06
	IP5	2	0.104	4.76E-06	114	3.852	9.73E-06	50	1.473	9.77E-06
	IP6	2	0.095	9.65E-06	116	3.923	9.71E-06	52	1.536	8.27E-06
	IP7	2	0.099	8.57E-06	116	3.827	8.89E-06	52	1.537	7.57E-06

From Tables 3–9, we can observe that the three methods are trying to solve (1). However, the improvement and effectiveness of the proposed method are pretty straightforward. The tables indicated that modifying the IDFDD method in the proposed scheme is a good improvement. The MDFDD method remarkably outperforms the IDFDD and IDSL methods for nearly all the problems assessed since it has the least number of iterations, which are far below the number of iterations for the IDFDD and IDSL methods. Moreover, the proposed method has less CPU time than the IDFDD method. However, the MDFDD method has a higher CPU time than

Table 4: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 2

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	13	0.026	6.15E-06	26	0.022	7.35E-06	37	0.010	8.96E-06
	IP2	20	0.036	9.95E-06	19	0.014	9.06E-06	28	0.012	9.49E-06
	IP3	15	0.043	5.55E-06	30	0.022	7.09E-06	43	0.021	7.75E-06
	IP4	18	0.018	4.46E-06	27	0.018	9.54E-06	33	0.017	8.71E-06
	IP5	20	0.024	6.27E-06	64	0.041	9.19E-06	34	0.009	8.09E-06
	IP6	11	0.017	5.10E-06	30	0.023	7.61E-06	38	0.012	9.29E-06
	IP7	23	0.043	6.62E-06	39	0.024	6.34E-06	43	0.021	9.82E-06
1,000	IP1	19	0.062	2.86E-07	28	0.020	9.52E-06	40	0.013	9.72E-06
	IP2	17	0.040	7.93E-06	22	0.025	7.51E-06	32	0.025	7.21E-06
	IP3	20	0.072	5.27E-06	32	0.032	9.18E-06	46	0.028	8.4E-06
	IP4	16	0.050	6.52E-06	30	0.024	7.91E-06	36	0.027	9.45E-06
	IP5	18	0.062	8.70E-06	71	0.054	7.79E-06	34	0.027	7.27E-06
	IP6	12	0.045	5.73E-06	32	0.020	9.85E-06	42	0.039	7.06E-06
	IP7	24	0.065	6.80E-06	43	0.018	5.72E-06	43	0.032	9.51E-06
10,000	IP1	33	0.300	5.24E-06	31	0.174	7.89E-06	44	0.153	7.38E-06
	IP2	41	0.345	3.48E-06	24	0.143	9.72E-06	35	0.131	7.82E-06
	IP3	14	0.208	7.43E-06	35	0.129	7.61E-06	49	0.185	9.11E-06
	IP4	35	0.269	3.35E-06	33	0.174	6.55E-06	40	0.101	7.17E-06
	IP5	17	0.220	6.73E-06	70	0.209	8.45E-06	33	0.119	9.98E-06
	IP6	14	0.207	4.16E-07	35	0.119	8.17E-06	45	0.141	7.65E-06
	IP7	29	0.350	5.4E-06	44	0.157	8E-06	46	0.137	7.38E-06
50,000	IP1	37	0.719	2.83E-06	33	0.443	7.23E-06	46	0.389	8.08E-06
	IP2	45	0.792	2.38E-07	26	0.339	8.91E-06	37	0.334	8.57E-06
	IP3	16	0.594	2.96E-06	37	0.461	6.97E-06	51	0.472	9.99E-06
	IP4	43	1.100	9.23E-07	34	0.433	9.38E-06	42	0.408	7.86E-06
	IP5	14	0.629	4.38E-06	73	0.825	8.01E-06	33	0.281	9.93E-06
	IP6	14	0.515	8.34E-06	37	0.556	7.48E-06	47	0.435	8.39E-06
	IP7	32	1.103	8.78E-06	45	0.527	8.09E-06	48	0.447	7.49E-06
100,000	IP1	36	1.114	8.69E-06	34	0.759	6.54E-06	47	0.786	8E-06
	IP2	50	1.775	8.2E-07	27	0.645	8.06E-06	38	0.650	8.48E-06
	IP3	11	0.595	7.05E-07	37	0.865	9.86E-06	52	0.873	9.89E-06
	IP4	42	1.368	5.53E-06	35	0.760	8.49E-06	43	0.697	7.78E-06
	IP5	17	1.597	6.23E-06	72	1.480	8.15E-06	33	0.578	9.92E-06
	IP6	21	1.756	3.55E-06	38	0.833	6.77E-06	48	0.775	8.3E-06
	IP7	34	2.416	6.9E-06	46	1.029	9.52E-06	49	0.823	7.3E-06

Table 5: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 3

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	16	0.034	8.52E-06	32	0.012	9.96E-06	37	0.010	8.32E-06
	IP2	12	0.033	9.99E-06	31	0.023	8.84E-06	26	0.015	7.7E-06
	IP3	20	0.044	2.25E-06	35	0.021	6.42E-06	41	0.008	8.47E-06
	IP4	17	0.021	1.76E-06	32	0.025	8.45E-06	36	0.017	7.56E-06
	IP5	20	0.045	6.07E-06	34	0.012	6.95E-06	40	0.020	7.81E-06
	IP6	10	0.023	1.06E-06	28	0.016	7.37E-06	35	0.017	8.38E-06
	IP7	17	0.030	7.24E-06	31	0.023	7E-06	35	0.007	8.61E-06
1,000	IP1	22	0.071	6.32E-06	35	0.047	8.26E-06	40	0.015	9.02E-06
	IP2	15	0.039	4.47E-06	34	0.048	7.33E-06	29	0.012	8.35E-06
	IP3	26	0.074	2.66E-07	37	0.044	8.32E-06	44	0.047	9.19E-06
	IP4	24	0.038	3.26E-06	35	0.032	7.01E-06	39	0.030	8.2E-06
	IP5	28	0.051	7.39E-07	36	0.049	9.28E-06	43	0.040	8.83E-06
	IP6	12	0.037	3.85E-06	30	0.026	9.54E-06	38	0.033	9.09E-06
	IP7	13	0.082	4.29E-06	33	0.036	7.05E-06	38	0.028	7.25E-06
10,000	IP1	26	0.295	6.95E-06	38	0.200	6.85E-06	43	0.114	9.78E-06
	IP2	24	0.345	1.95E-06	36	0.194	9.5E-06	32	0.168	9.05E-06
	IP3	37	0.506	3.4E-06	40	0.181	6.9E-06	47	0.211	9.97E-06
	IP4	27	0.319	9.21E-06	37	0.178	9.09E-06	42	0.118	8.89E-06
	IP5	30	0.263	9.76E-06	39	0.164	7.73E-06	46	0.166	9.64E-06
	IP6	17	0.331	9.04E-06	33	0.136	7.92E-06	41	0.139	9.86E-06
	IP7	21	0.256	7.66E-06	35	0.166	8.76E-06	41	0.154	7.64E-06
50,000	IP1	29	0.760	6.52E-06	39	0.506	9.81E-06	46	0.477	7.5E-06
	IP2	29	0.941	7.04E-06	38	0.520	8.7E-06	34	0.334	9.92E-06
	IP3	40	1.244	5.85E-06	41	0.529	9.87E-06	50	0.618	7.65E-06
	IP4	28	0.823	3.9E-06	39	0.540	8.32E-06	44	0.491	9.74E-06
	IP5	35	0.871	3.34E-06	41	0.540	7.08E-06	49	0.575	7.4E-06
	IP6	22	0.825	6.22E-06	35	0.488	7.25E-06	44	0.433	7.56E-06
	IP7	24	1.087	8.32E-06	37	0.486	7.99E-06	43	0.446	8.36E-06
100,000	IP1	36	2.601	8.24E-06	40	0.985	8.88E-06	47	0.888	7.43E-06
	IP2	33	2.573	6.18E-06	39	1.132	7.88E-06	35	0.727	9.82E-06
	IP3	43	2.423	9.86E-07	42	1.332	8.94E-06	51	1.080	7.57E-06
	IP4	34	2.529	5.42E-06	40	0.998	7.54E-06	45	0.887	9.65E-06
	IP5	42	2.712	9.63E-06	42	1.056	6.41E-06	50	0.924	7.32E-06
	IP6	20	1.531	2.37E-06	36	0.904	6.56E-06	45	0.862	7.49E-06
	IP7	24	1.634	5.4E-06	38	0.980	7.23E-06	44	0.825	8.27E-06

Table 6: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 4

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	37	0.061	9.93E-06	45	0.018	8.01E-06	45	0.011	8.46E-06
	IP2	32	0.066	9.00E-06	47	0.029	8.47E-06	46	0.023	8.83E-06
	IP3	36	0.053	8.51E-06	53	0.032	8.04E-06	54	0.025	9.45E-06
	IP4	37	0.067	7.28E-06	47	0.030	8.34E-06	41	0.019	8.61E-06
	IP5	33	0.056	8.14E-06	49	0.030	9.87E-06	51	0.026	8.19E-06
	IP6	41	0.044	6.93E-06	49	0.022	9.01E-06	52	0.010	7.39E-06
	IP7	26	0.029	7.11E-06	41	0.022	8.36E-06	44	0.027	9.03E-06
1,000	IP1	33	0.084	9.83E-06	46	0.036	1E-05	50	0.040	9.31E-06
	IP2	32	0.090	9.25E-06	48	0.030	9.59E-06	49	0.039	7.86E-06
	IP3	46	0.102	5.74E-06	53	0.043	7.68E-06	59	0.043	7.05E-06
	IP4	28	0.063	7.2E-06	47	0.052	8.51E-06	47	0.017	9.56E-06
	IP5	40	0.080	9.51E-06	52	0.059	9.66E-06	50	0.018	8.99E-06
	IP6	37	0.101	8.49E-06	49	0.032	9.95E-06	55	0.052	7.38E-06
	IP7	24	0.052	7.93E-06	43	0.032	9.41E-06	48	0.038	7.61E-06
10,000	IP1	36	0.657	9.61E-06	48	0.212	8.89E-06	53	0.189	9.81E-06
	IP2	40	0.567	3.06E-06	50	0.215	9.42E-06	54	0.227	9.17E-06
	IP3	58	0.594	9.09E-06	53	0.234	8.38E-06	64	0.216	9.44E-06
	IP4	39	0.761	9.19E-06	50	0.235	9.73E-06	50	0.190	9.54E-06
	IP5	51	0.718	7.54E-06	54	0.283	1E-05	51	0.244	8.49E-06
	IP6	49	0.751	5.9E-06	54	0.254	7.38E-06	58	0.190	7.3E-06
	IP7	34	0.341	9.06E-06	43	0.221	9.2E-06	48	0.165	8.37E-06
50,000	IP1	36	2.321	6.59E-06	48	0.698	9.53E-06	56	0.595	7.59E-06
	IP2	39	1.581	7.05E-06	53	0.770	9.97E-06	54	0.666	8.81E-06
	IP3	68	2.331	9.27E-06	55	0.810	8.88E-06	63	0.670	8.29E-06
	IP4	34	1.625	5.01E-06	51	0.752	9.55E-06	51	0.537	8.47E-06
	IP5	53	2.435	9.69E-06	55	0.795	8.83E-06	53	0.638	7.7E-06
	IP6	48	2.340	8.73E-06	54	0.834	9.04E-06	61	0.676	9.28E-06
	IP7	38	1.301	2.77E-06	44	0.655	8.59E-06	50	0.560	8.04E-06
10,0000	IP1	41	4.922	9.92E-06	49	1.467	9.82E-06	54	1.132	9.69E-06
	IP2	46	4.246	9.16E-06	53	1.650	9.8E-06	59	1.311	9.21E-06
	IP3	73	4.694	5.2E-06	55	1.908	9.38E-06	63	1.342	9.92E-06
	IP4	46	4.739	6.53E-06	52	1.455	8.6E-06	55	1.113	9.46E-06
	IP5	59	5.184	3.46E-06	56	1.598	8.14E-06	58	1.240	9.03E-06
	IP6	53	3.656	9.1E-06	55	1.586	8.23E-06	61	1.309	8.11E-06
	IP7	39	2.125	5.45E-06	44	1.418	9.99E-06	51	1.079	7.92E-06

Table 7: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 5

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	19	0.026	2.41E-06	48	0.031	9.76E-06	37	0.023	9.63E-06
	IP2	16	0.015	5.8E-06	45	0.015	8.7E-06	35	0.007	7.63E-06
	IP3	16	0.037	6.63E-06	53	0.033	8.1E-06	41	0.008	7.85E-06
	IP4	13	0.026	1.27E-06	48	0.036	7.74E-06	37	0.007	7.61E-06
	IP5	14	0.032	2.97E-06	51	0.032	8.6E-06	39	0.012	9.66E-06
	IP6	6	0.007	7.47E-07	48	0.027	7.65E-06	33	0.014	8.53E-06
	IP7	12	0.029	8.6E-06	44	0.019	7.78E-06	34	0.012	7.58E-06
1,000	IP1	14	0.042	3.31E-06	53	0.035	7.83E-06	41	0.025	7.31E-06
	IP2	10	0.036	5.19E-06	49	0.054	9.18E-06	38	0.019	8.27E-06
	IP3	17	0.041	6.63E-06	57	0.048	8.55E-06	44	0.039	8.51E-06
	IP4	19	0.040	3.13E-06	52	0.060	8.17E-06	40	0.013	8.26E-06
	IP5	9	0.028	4.13E-06	55	0.057	9.45E-06	43	0.036	7.66E-06
	IP6	9	0.039	2.42E-07	52	0.030	8.07E-06	36	0.018	9.25E-06
	x7	11	0.052	9.45E-06	44	0.040	7.8E-06	34	0.019	7.6E-06
10,000	IP1	13	0.237	5.88E-07	57	0.215	8.26E-06	44	0.108	7.93E-06
	IP2	15	0.333	6.48E-06	53	0.171	9.69E-06	41	0.112	8.97E-06
	IP3	16	0.202	1.66E-06	61	0.192	9.02E-06	47	0.159	9.23E-06
	IP4	13	0.150	7.8E-06	56	0.254	8.62E-06	43	0.132	8.95E-06
	IP5	19	0.402	3.93E-06	60	0.225	7.63E-06	46	0.213	8.36E-06
	IP6	13	0.213	3.62E-06	56	0.252	8.52E-06	40	0.093	7.02E-06
	IP7	13	0.211	2.74E-07	44	0.153	7.8E-06	34	0.079	7.6E-06
50,000	IP1	10	0.618	1.34E-07	60	0.664	8.1E-06	46	0.398	8.69E-06
	IP2	12	0.552	4.71E-06	56	0.604	9.51E-06	43	0.349	9.83E-06
	IP3	12	0.465	5.39E-06	64	0.722	8.85E-06	50	0.422	7.08E-06
	IP4	17	1.063	2.64E-06	59	0.689	8.46E-06	45	0.441	9.81E-06
	IP5	17	0.961	5.8E-06	62	0.693	9.86E-06	48	0.379	9.17E-06
	IP6	11	0.534	5.8E-06	59	0.655	8.36E-06	42	0.332	7.69E-06
	IP7	12	0.682	1.68E-06	44	0.500	7.8E-06	34	0.287	7.6E-06
100,000	IP1	11	1.156	7.49E-06	61	1.299	8.71E-06	47	1.125	8.61E-06
	IP2	9	0.684	4.11E-06	58	1.145	7.77E-06	44	0.871	9.73E-06
	IP3	14	1.359	7.52E-06	65	1.336	9.51E-06	51	0.805	7.01E-06
	IP4	10	1.360	7E-06	60	1.480	9.09E-06	46	0.873	9.71E-06
	IP5	15	1.204	4.48E-06	64	1.405	8.05E-06	49	0.877	9.08E-06
	IP6	17	1.563	1.77E-06	60	1.206	8.99E-06	43	0.699	7.62E-06
	IP7	11	1.106	1.63E-06	44	1.113	7.8E-06	34	0.520	7.6E-06

Table 8: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 6

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	27	0.205	8.46E-06	52	0.128	9.42E-06	38	0.073	8.21E-06
	IP2	20	0.195	6.73E-06	47	0.104	7.62E-06	35	0.072	8.98E-06
	IP3	44	0.264	8.86E-06	60	0.132	9.66E-06	40	0.068	8.38E-06
	IP4	31	0.319	7.88E-06	51	0.115	9.37E-06	37	0.068	9.22E-06
	IP5	34	0.264	4.6E-06	55	0.123	9.69E-06	40	0.071	7.5E-06
	IP6	24	0.239	5.94E-06	47	0.116	8.84E-06	35	0.061	9.78E-06
	IP7	22	0.213	7.94E-06	45	0.106	7.77E-06	37	0.067	9.35E-06
1,000	IP1	31	2.441	9.82E-06	54	1.154	9.19E-06	41	0.664	8.53E-06
	IP2	25	2.532	8.98E-06	50	1.092	9.64E-06	38	0.608	9.09E-06
	IP3	59	2.687	8.13E-06	57	1.178	8.11E-06	43	0.697	8.2E-06
	IP4	31	2.273	8.28E-06	53	1.135	8.6E-06	40	0.638	9.51E-06
	IP5	45	2.836	8.66E-06	55	1.133	9.24E-06	43	0.680	8.2E-06
	IP6	25	2.066	8.18E-06	49	1.047	7.81E-06	38	0.614	9.21E-06
	IP7	22	2.083	6.55E-06	45	0.988	7.83E-06	37	0.587	9.37E-06
2,000	IP1	26	5.618	4.72E-06	56	3.854	7.96E-06	42	2.135	8.42E-06
	IP2	25	6.920	9.43E-06	52	3.626	8.66E-06	39	2.011	8.95E-06
	IP3	59	8.466	7.28E-06	59	4.029	9.91E-06	44	2.260	8.07E-06
	IP4	27	6.605	8.86E-06	55	3.972	8.4E-06	41	2.090	9.39E-06
	IP5	43	7.773	7.29E-06	60	4.223	9.13E-06	44	2.283	8.13E-06
	IP6	27	7.955	6.38E-06	51	3.586	7.86E-06	39	1.978	8.99E-06
	IP7	25	8.424	6.73E-06	45	3.235	7.83E-06	37	1.891	9.37E-06

Table 9: Numerical outcomes of MDFDD, IDFDD, and IDSL methods for problem 7

		MDFDD			**IDFDD**			**IDSL**		
Dimension	IP	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$	ITRN	CTM(S)	$\|F_k\|$
100	IP1	17	0.226	9.74E-06	34	0.080	7.68E-06	31	0.053	9.24E-06
	IP2	23	0.253	5.6E-06	41	0.099	8.19E-06	36	0.063	9.62E-06
	IP3	26	0.221	8.98E-06	45	0.102	7.32E-06	40	0.067	7.41E-06
	IP4	16	0.175	8.8E-06	38	0.082	7.76E-06	34	0.063	7.14E-06
	IP5	22	0.174	6.76E-06	42	0.110	9.28E-06	38	0.076	7.42E-06
	IP6	24	0.168	8.55E-06	44	0.100	7.63E-06	39	0.088	8.13E-06
	IP7	21	0.164	5.67E-06	42	0.094	9.17E-06	38	0.073	7.26E-06
1,000	IP1	17	1.582	8.61E-06	35	0.743	8.31E-06	31	0.497	9.65E-06
	IP2	24	1.553	9.77E-06	44	0.940	9.98E-06	40	0.646	7.28E-06
	IP3	28	1.807	4.01E-06	48	1.012	8.61E-06	43	0.687	7.64E-06
	IP4	22	1.580	7.03E-06	41	0.870	9.4E-06	37	0.592	7.62E-06
	IP5	25	1.827	7.8E-06	46	1.024	8.31E-06	41	0.657	8.08E-06
	IP6	25	1.422	7.42E-06	47	1.046	9.26E-06	42	0.670	8.73E-06
	IP7	28	1.849	8.2E-06	46	1.069	8.51E-06	41	0.655	8.27E-06
2,000	IP1	18	5.262	7.49E-06	36	2.459	7.66E-06	32	1.656	8.08E-06
	IP2	28	5.929	8.16E-06	46	3.167	7.43E-06	41	2.121	7.21E-06
	IP3	31	6.115	6.11E-06	49	3.397	8.82E-06	44	2.235	7.54E-06
	IP4	27	6.253	9.63E-06	42	2.907	9.64E-06	38	1.995	7.54E-06
	IP5	30	7.231	5.2E-06	47	3.298	8.55E-06	42	2.194	8.01E-06
	IP6	31	6.675	4.99E-06	48	3.387	9.5E-06	43	2.283	8.64E-06
	IP7	25	6.070	5.32E-06	47	3.219	8.77E-06	42	2.188	8.22E-06

the IDSL method due to the computation of double direction in the MDFDD methods.

Figures 1–2 display the interpretation of the numerical results of each of the three methods using Dolan and Moré [5] performance profiles. We achieve this by plotting fraction $p(\tau)$ of problems for each method within τ of the smallest number of iterations and CPU time. As shown in Figures 1 and 2, the curves representing the MDFDD method remain above the IDFDD and IDSL methods in number iterations. Furthermore, it is above the curve representing the IDFDD method for the CPU time. Therefore, the proposed method outperforms the IDFDD and IDSL methods in fewer iterations and is thus the most efficient method. Finally, from the results in Tables 3–9, it is evident that the MDFDD method successfully solves problem (1).

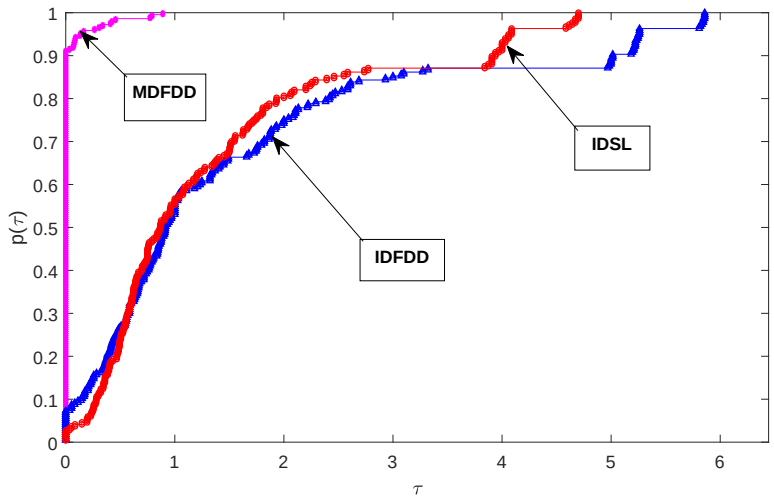


Figure 1: Performance profile with respect to the number of iterations

4.2 Application in integral equations

Chandrasekhar and Breen [3] computed H-equation as the solution of the nonlinear integral equation that gives the complete nonlinear equations technique. The nonlinear integral equation arising in radiative heat transfer problem is given by

$$H(x) = 1 + c\frac{x}{2}H(x)\int_0^1 \frac{H(y)}{x+y}dy, \tag{50}$$

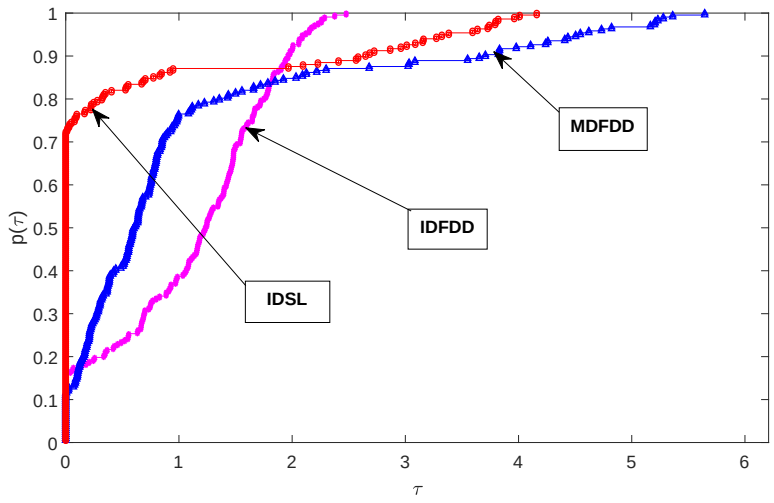


Figure 2: Performance profile with respect to the CPU time (in second)

with parameter $c \in [0,1]$ and $H : [0,1] \to \mathbb{R}$ is an unknown function.

Equation (50) can be written as

$$H(x) = \left[1 - \frac{c}{2}\int_0^1 \frac{xH(y)}{x+y}dy\right] = 1. \tag{51}$$

By multiplying both sides of (51) with $\left(1 - \frac{c}{2}\int_0^1 \frac{xH(y)}{x+y}dy\right)^{-1}$, we have

$$F(H)(x) = H(x) - \left(1 - \frac{c}{2}\int_0^1 \frac{xH(y)}{x+y}dy\right)^{-1} = 0, \tag{52}$$

which is called the Chandrasekhar H-equation [23]. However, (52) can be discretized by using the midpoint quadrature formula

$$\int_0^1 f(\mu)d\mu = h\sum_{j=1}^{n} f(\mu_j), \tag{53}$$

for $\mu_j = (j-0.5)h$, $0 \le j \le 1$, and $h = \frac{1}{n}$.

As a result, we have the following system of nonlinear equations:

$$F_i(x) = x_i - \left(1 - \frac{c}{2n}\sum_{j=1}^{n}\frac{\mu_i x_j}{\mu_i + \mu_j}\right)^{-1} \quad i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, n, \tag{54}$$

which is known as the discretized Chandrasekhar H-equation that can be solved by using some iterative methods. If the initial point $x_0 = (1, 1, 1, \ldots, 1)^T$, then the system in (54) has a solution for all $c \in (0, 1)$. However, the hardest part of the problem (54) is that the Jacobian is singular at $c = 1$. Therefore as c approaches 1, the Jacobian approaches the singularity point. Since our method is derivative-free, then it has the advantage to solve problem (54) even when c approaches 1.

To highlight the performance of the MDFDD approach furthermore, we conduct some numerical experiments by comparing it with the classical NM, IDFDD method [10], and IDSL method [11]. The iteration is also set to terminate when $\|x_{k+1} - x_k\| + \|F_k\| \leq 10^{-5}$ or when the iterations exceed 1000, but no point of x_k satisfying the stopping criterion. We have tried the three methods with the starting point of $x_0 = (1, 1, 1, \ldots, 1)^T$. Furthermore, we use the dimensions (n values) 100 to 20,000 to show the performance of each of the three methods.

Table 10: Numerical results of discretized Chandrasekhar H-equation

		NM		**IDFDD**		**IDSL**		**MDFDD**	
	Dimension	ITER	TIME	ITER	TIME	ITER	TIME	ITER	TIME
c=0.1	100	12	0.078	-	-	38	0.029	13	0.015
	500	15	0.785	-	-	41	0.017	14	0.015
	1000	15	3.056	-	-	42	0.018	11	0.018
	10000	20	1747	-	-	45	0.150	12	0.141
	20000	-	-	-	-	46	0.219	20	0.349
c=0.9	100	13	0.111	-	-	38	0.009	9	0.009
	500	15	0.789	-	-	41	0.022	17	0.029
	1000	18	3.729	-	-	42	0.026	15	0.024
	10000	-	-	-	-	45	0.231	15	0.215
	20000	-	-	-	-	46	0.216	14	0.377
c=0.99	100	13	0.111	-	-	38	0.012	12	0.011
	500	17	0.874	-	-	41	0.032	17	0.020
	1000	18	3.733	-	-	42	0.019	12	0.018
	10000	-	-	-	-	45	0.123	11	0.125
	20000	-	-	-	-	46	0.218	13	0.285
c=0.999	100	13	0.112	-	-	38	0.019	13	0.014
	500	17	0.883	-	-	41	0.013	16	0.018
	1000	18	3.766	-	-	42	0.025	16	0.029
	10000	-	-	-	-	45	0.130	13	0.176
	20000	-	-	-	-	46	0.318	12	0.282

The numerical results of the methods used to solve Chandrasekhar H-equation with different values of parameter c, are shown in Table 10. The table clearly indicates that the proposed method outperformed the NM be-

cause the NM failed when the number of dimension increased. This is due to the fact that as c approaches 1, the Jacobian approaches the singularity point. Moreover, the CPU time (in second) of the NM is higher than other methods because it solved the Jacobian matrix at each iteration. From Table 10, we can also observe that the IDFDD method has totally failed because it has poor numerical performance, as we have made mentioned earlier in the introduction section of this article. Although, the IDSL method solved problem (54) completely, but it has more number of iterations than the MDFDD method. This shows that our method has effectively solved the discretized Chandrasekhar H-equation with the least number of iterations and CPU time.

5 Conclusion

In this article, numerical comparisons were made using a set of large-scale test problems. Furthermore, Tables 3–9 and Figures 1–2 showed that the presented method is practically quite efficient because it has fewer iterations than the IDFDD and IDSL methods. Furthermore, we have successfully used the proposed method to deal with experiments on the Chandrasekhar H-equation in radiative heat transfer. The experiments were carried out and reported in Table 10 with different c values, demonstrating a better efficiency for the MDFDD method. The numerical results showed that the employed method solved the discretized integral equation with fewer iterations and CPU time than the NM, IDFDD, and IDSL methods. Future research includes applying the MDFDD scheme to solve the discretized three-dimensional nonlinear Poisson problem.

References

1. Abdullahi, H., Halilu, A.S. and Waziri M.Y. *A modified conjugate gradient method via a double direction approach for solving large-scale symmetric nonlinear systems*, J. numer. math. stoch. 10(1) (2018), 32–44.

2. Bouaricha, A. and Schnabel, R.B. *Tensor methods for large sparse systems of nonlinear equations,* Math. Program. 82 (1998) 377–400.

3. Chandrasekhar, S. and Breen, F. H. *On the radiative equilibrium of a stellar atmosphere*, XIX, Astrophys. J. 106 (1947) 143–144.

4. Dennis, J.E., and Schnabel, R.B. Numerical methods for unconstrained optimization and non-linear equations. Prentice Hall, Englewood Cliffs, NJ, 1983.

5. Dolan, E. and Moré, J. *Benchmarking optimization software with performance profiles*, Math. Program. 91(2) (2002), Ser. A, 201–213.

6. Duranović-Miličić, N.I. *A multi-step curve search algorithm in nonlinear optimization,* Yugosl. J. Oper. Res. 18 (2008), 47–52.

7. Duranović-Miličić, N. I. and Gardasevic-Filipovic, M. *A multi-step curve search algorithm in nonlinear convex case: nondifferentiable convex case,* Facta Univ. Ser. Math. Inform. 25 (2010), 11–24.

8. Halilu, A.S. and Waziri, M.Y. *A transformed double step length method for solving large-scale systems of nonlinear equations,* J Num. Math. Stoch. 9 (2017) 20–32.

9. Halilu, A.S. and Waziri M.Y. *Enhanced matrix-free method via double step length approach for solving systems of nonlinear equations,* Int. J. appl. Math Res. 6 (2017) 147–156.

10. Halilu, A.S. and Waziri, M.Y. *An improved derivative-free method via double direction approach for solving systems of nonlinear equations,* J. Ramanujan Math. Soc. 33 (2018), no. 1, 75–89.

11. Halilu, A.S., and Waziri, M.Y. *Inexact double step length method for solving systems of nonlinear equations,* Stat., Optim. Inf. Comput., 8 (2020) 165–174.

12. Kanzow, C., Yamashita, N. and Fukushima, M. *Levenberg-Marquardt methods for constrained nonlinear equations with strong local convergence properties,* J. Comput. Appl. Math. 172 (2004) 375–397.

13. Levenberg, K. *A method for the solution of certain non-linear problems in least squares,* Quart. Appl. Math. 2 (1944), 164–168.

14. Li, D. and Fukushima M. *A globally and superlinearly convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations,* SIAM J. Numer. Anal. 37 (1999), 152–172.

15. Marquardt, D.W. *An algorithm for least-squares estimation of nonlinear parameters*, SIAM J. Appl. Math. 11 (1963) 431–441.

16. Meintjes, K. and Morgan, A.P. *A methodology for solving chemical equilibrium systems*, Appl. Math. Comput. 22 (1987), 333–361.

17. Musa, Y. B., Waziri, M.Y., and Halilu, A.S. *On computing the regularization Parameter for the Levenberg-Marquardt method via the spectral radius approach to solving systems of nonlinear equations,*. J. Numer. Math. Stoch. 9 (2017), 80–94.

18. Petrović, M.J. *An accelerated double step size model in unconstrained optimization,* Appl. Math. Comput. 250 (2015), 309–319.

19. Petrović, M.J. and Stanimirović, P.S. *Accelerated double direction method for solving unconstrained optimization problems*, Math. Probl. Eng. 2014, Art. ID 965104, 8 pp.

20. Sun, M., Tian, M.Y. and Wang, Y.J. *Multi-step discrete-time Zhang neural networks with application to time-varying nonlinear optimization,* Discrete Dyn. Nat. Soc. 2019, Art. ID 4745759, 1–14.

21. Waziri, M.Y., Leong, W.J. and Hassan M.A. *Jacobian-free diagonal Newton's method for solving nonlinear systems with singular Jacobian,* Malays. J. Math. Sci. 5(2) (2011), 241–255.

22. Waziri, M.Y., Leong W.J., Hassan M.A. and Monsi M. *A new Newton's method with diagonal Jacobian approximation for system of nonlinear equations,* J. Math. Stat. 6 (2010) 246–252.

23. Waziri, M.Y., Leong W.J., Hassan, M.A, and Monsi, M. *A low memory solver for integral equations of Chandrasekhar type in the radiative transfer problems*, Math. Probl. Eng. 2011, Art. ID 467017, 12 pp.

24. Waziri, M.Y., and Sabiu, J. *A derivative-free conjugate gradient method and its global convergence for symmetric nonlinear equations,* Int. J. Math. Math. Sci. 2015, Art. ID 961487, 8 pp.

25. Xiao, Y.H. and Zhu, H. A *conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing*, J. Math. Anal. Appl. 405 (2013) 310–319.

26. Yuan, G. and Lu, X. *A new backtracking inexact BFGS method for symmetric nonlinear equations,* Comp. Math. App. 55 (2008) 116–129.

How to cite this article
A.I. Kiri, M.Y. Waziri and A.S. Halilu Modification of the double direction approach for solving systems of nonlinear equations with application to Chandrasekhar's Integral equation. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 426-448. doi: 10.22067/IJNAO.2022.74201.1083.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 449–466
DOI: 10.22067/ijnao.2022.72818.1064
https://ijnao.um.ac.ir/

On a class of Bézier-like model for shape-preserving approximation

B. Nouri and J. Saeidian*

Abstract

A class of Bernstein-like basis functions, equipped with a shape parameter, is presented. Employing the introduced basis functions, the corresponding curve and surface in rectangular patches are defined based on some control points. It is verified that the new curve and surface have most properties of the classical Bézier curves and surfaces. The shape parameter helps to adjust the shape of the curve and surface while the control points are fixed. We prove that the proposed Bézier-like curves can preserve monotonicity and that Bézier-like surfaces can preserve axial monotonicity. Moreover, the presented curves and surfaces preserve bound constraints implied by the original data.

AMS subject classifications (2020): 65D17, 65D10, 68U05.

Keywords: Blending functions; Bézier curve; Shape adjustment; Monotonicity preservation; Shape-preserving; Boundedness.

1 Introduction

In computer-aided geometric design (CAGD) and computer graphics, curves are often constructed by the following relation:

* Corresponding author

Received 6 October 2022; revised 21 February 2022; accepted 12 March 2022

Bahareh Nouri
Faculty of Mathematical Sciences and Computer, Kharazmi University, Tehran, Iran.
e-mail: std_nouri411@khu.ac.ir

Jamshid Saeidian
Faculty of Mathematical Sciences and Computer, Kharazmi University, Tehran, Iran.
e-mail: j.saeidian@khu.ac.ir

$$C(t) = \sum_{i=0}^{n} F_i(t) P_i, \qquad t \in [a, b] \subseteq \mathbb{R},\ n \in \mathbb{N}, \tag{1}$$

where the real functions $\{F_i\}$ are known as blending functions, $\{P_i\} \subseteq \mathbb{R}^s$ are control points ($s \geq 2$), and the polyline constructed by control points is called the control polygon. One of the most important blending functions are the Bernstein polynomials, which in turn lead to well-known classical Bézier curves. They have a simple structure and are widely used in many engineering and technology fields.

In a classical Bézier curve, once the control points are fixed, the shape of the curve cannot be changed. This is a drawback that reduces the shape adjustment and therefore limits their applicability. In order to overcome this deficiency, many researchers have tried to construct basis functions equipped with shape parameter(s) to create new curves whose structures and properties are similar to the Bézier curves, besides they also have the shape adjustment property [6, 13, 16, 17, 19, 21, 22, 23, 30]

In curves with shape parameter(s), by changing the parameter(s), the curve either approaches to the control polygon [17, 18, 19], or moves away from it [20, 31]. Another difference between parameter based curves originates from their construction, some basis functions are defined by a recursive formula [23, 22, 5, 20, 32], while others are based on a general formula [17, 19, 16, 18, 24, 8].

Here, we propose a new family of parameter based blending functions by using square roots of polynomials. Thus we call them *sq-basis functions*. The new blending functions are generated from three initial basis functions employing a recursive relation. This is a procedure that could be traced back in the literature [5, 22, 23, 32]. The corresponding Bézier-like curves, which in turn are referred to *sq-curves*, are in common with the classical Bézier curves in many features such as geometric invariance, endpoint interpolation, formal symmetry, and convex hull property. The sq-basis functions are equipped with one shape parameter, which can adjust the shape of the curve, and they would either move closer to or away from the control polygon. In the sq-curves with three control points, by altering the shape parameter, one would see a family of curves traveling from the control polygon to the straight line joining the first and last control point. In comparison to previously defined blending functions with the same structure, sq-curves show a higher rate of change by employing just one shape parameter, and this could be considered as an advantage. The construction could be extended to surfaces [11, 14], which share the most common features of Bézier surfaces.

Once we have a suitable model for data approximation, one can apply it in a shape-preserving approximation. Shape-preserving approximation is defined to be a method of constructing a curve (surface), which also preserves the shape implied by the data. It is known as an essential curve/surface design technique in CAD/CAM and geometric design. Convexity, monotonicity, positivity, and boundedness are the most important shape features,

which have extensively been studied in literature [1, 2, 3, 25, 27, 28, 29]. The proposed Bézier-like curves and surfaces prove to be a promising approximation tool that preserves the monotonicity and the boundedness implied by the data set.

The present paper is outlined as follows. In Section 2, the new blending functions are defined and their properties are studied. The corresponding curves and surfaces are defined in Section 3. Section 4 studies the shape-preserving properties of the new curves and surfaces.

2 New blending functions

The classical Bernstein functions have a number of important features that makes them a suitable basis for constructing control point based curves. Non-negativity, partition of unity, symmetry in some sense, and their end point values are among the most cited properties of classical Bernstein functions.

Here we address a general representation for a class of blending functions that have similar properties to Bernstein polynomials. We present a special case that uses the square root of quadratic (sq) polynomials.

Definition 1. Based on a real valued shape parameter ν, the sq-starting basis functions are defined as follows:

$$\begin{aligned} b_{2,0}(t) &= \frac{1}{2} - t + \varphi(t), \\ b_{2,1}(t) &= 1 - 2\varphi(t), \\ b_{2,2}(t) &= t - \frac{1}{2} + \varphi(t), \end{aligned} \tag{2}$$

where $0 \le t \le 1,\ 0 < \nu \le 1$, and

$$\varphi(t) = \sqrt{(1-\nu)\,(t^2 - t) + \frac{1}{4}}.$$

We generate the sq-basis functions of order n ($n \ge 3$) with the recursion relation of the classical Bernstein basis functions [9] as follows:

$$b_{n,i}(t) = (1-t)b_{n-1,i}(t) + tb_{n-1,i-1}(t), \quad t \in [0,1], \tag{3}$$

where $i = 0, 1, 2, \ldots, n$. For $k = -1$ or $k > n$, we set $b_{n,k}(t) = 0$.

The sq-basis functions of order $n = 3$ can be expressed as

$$\begin{aligned} b_{3,0}(t) &= t^2 - \frac{3}{2}t + \frac{1}{2} + (1-t)\,\varphi(t), \\ b_{3,1}(t) &= -t^2 - \frac{1}{2}t + 1 + (-2+3t)\,\varphi(t), \end{aligned} \tag{4}$$

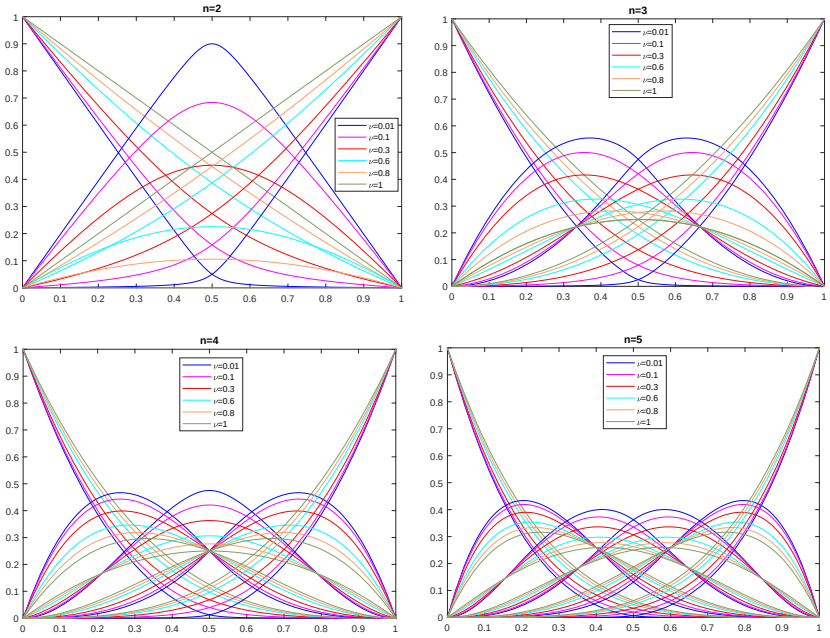


Figure 1: The sq-basis functions with different values of ν.

$$b_{3,2}(t) = -t^2 + \frac{5}{2}t - \frac{1}{2} + (1-3t)\,\varphi(t),$$
$$b_{3,3}(t) = t^2 - \frac{1}{2}t + t\varphi(t).$$

Figure 1 shows the sq-basis functions generated with $n = 2, 3, 4, 5$ and different values of ν.

Theorem 1. The sq-basis functions generated by (2) and (3) have the following properties:

(a) Nonnegativity: $b_{n,i}(t) \geq 0$ for $i = 0, 1, 2, \ldots, n$.

(b) Partition of unity (Normalization): $\sum_{i=0}^{n} b_{n,i}(t) = 1$.

(c) Symmetry: $b_{n,i}(t) = b_{n,n-i}(1-t)$ for $i = 0, 1, 2, \ldots, n$.

(d) End-point values:

$$b_{n,i}(0) = \begin{cases} 1, & i = 0, \\ 0, & i \neq 0, \end{cases} \qquad b_{n,i}(1) = \begin{cases} 1, & i = n, \\ 0, & i \neq n, \end{cases} \tag{5}$$

$$b'_{n,i}(0)=\begin{cases}-n+\nu, & i=0,\\ n-2\nu, & i=1,\\ \nu, & i=2,\\ 0, & o.w.,\end{cases}\qquad b'_{n,i}(1)=\begin{cases}n-\nu, & i=n,\\ 2\nu-n, & i=n-1,\\ -\nu, & i=n-2,\\ 0, & o.w.\end{cases}\tag{6}$$

Proof. Mathematical induction is used to prove this theorem.

(a) Nonnegativity is obvious from (2) and (3).

(b) From (2), we obtain $\sum_{i=0}^{2} b_{2,i}(t)=1$. Now suppose that the equality is true for m. Then according to recurrence relation(3) and the inductive hypothesis, we obtain

$$\sum_{i=0}^{m+1} b_{m+1,i}(t)=(1-t)\sum_{i=0}^{m+1} b_{m,i}(t)+t\sum_{i=0}^{m+1} b_{m,i-1}(t)=1-t+t=1.$$

(c) The sq-starting basis functions are symmetrical. Assume that the sq-basis functions of order m are symmetrical. Then from this inductive hypothesis and recurrence relation (3), one has

$$\begin{aligned}b_{m+1,i}(1-t)&=(1-(1-t))b_{m,i}(1-t)+(1-t)b_{m,i-1}(1-t)\\&=tb_{m,m-i}(t)+(1-t)b_{m,m-i+1}(t)=b_{m+1,m-i+1}(t).\end{aligned}$$

(d) From (2), we have

$$\begin{aligned}b'_{2,0}(t)&=-1+\varphi'(t),\\ b'_{2,1}(t)&=-2\varphi'(t),\\ b'_{2,2}(t)&=1+\varphi'(t),\end{aligned}\tag{7}$$

where

$$\varphi'(t)=\frac{(1-\nu)\,(2t-1)}{2\sqrt{(1-\nu)\,(t^2-t)+\frac{1}{4}}}.$$

By a simple deduction from (2) and (7), we conclude that the results in (5) and (6) hold for $n=2$.

Suppose that the properties at the endpoints hold for the sq-basis functions of order m. Then

$$\begin{aligned}b_{m+1,i}(0)=b_{m,i}(0)&=\begin{cases}1 & i=0,\\ 0, & i\neq 0,\end{cases}\\ b_{m+1,i}(1)=b_{m,i}(1)&=\begin{cases}1 & i=m,\\ 0, & i\neq m,\end{cases}\end{aligned}\tag{8}$$

which can be obtained by the inductive hypothesis and (3).

From (3), we have

$$b'_{m+1,i}(0) = -b_{m,i}(0) + b'_{m,i}(0) + b_{m,i-1}(0), \tag{9}$$
$$b'_{m+1,i}(1) = -b_{m,i}(1) + b_{m,i-1}(1) + b'_{m,i-1}(1). \tag{10}$$

From (8) and the inductive hypothesis and by using relation (9), the following results are obtained:

(i) For $i = 0$, $b'_{m+1,0}(0) = -b_{m,0}(0) + b'_{m,0}(0) = -1 - (-m + \nu) = -(m+1) + \nu$,
(ii) For $i = 1$, $b'_{m+1,1}(0) = -b_{m,1}(0) + b'_{m,1}(0) + b_{m,0}(0) = (m+1) - 2\nu$,
(iii) For $i = 2$, $b'_{m+1,2}(0) = -b_{m,2}(0) + b'_{m,2}(0) + b_{m,1}(0) = 0 + \nu + 0 = \nu$,
(iv) For $i \neq 0, 1, 2$, $b'_{m+1,i}(0) = 0$.

Similarly, from (10), the following results are deduced:

(i) For $i = m+1$, $b'_{m+1,m+1}(1) = (m+1) - \nu$,
(ii) For $i = m$, $b'_{m+1,m}(1) = -(m+1) + 2\nu$,
(iii) For $i = m-1$, $b'_{m+1,m-1}(1) = -\nu$,
(iv) For $i \neq m-1, m, m+1$, $b'_{m+1,i}(1) = 0$.

□

3 sq-curves and sq-surfaces

The sq-basis functions can be considered as blending functions to construct Bézier-like curves. Here we present the sq-curves and also extend the idea to surfaces.

Definition 2. Given control points $\{P_i\}_{i=0}^{n} \subseteq \mathbb{R}^2$, for $0 \leq t \leq 1$, $0 < \nu \leq 1$,

$$r_n(t) = \sum_{i=0}^{n} b_{n,i}(t) P_i, \tag{11}$$

is called a sq-curve, where $\{b_{n,i}(t)\}_{i=0}^{n}$ $(n \geq 3)$ are the sq-basis functions defined in (2) and (3).

From Theorem 1, the sq-curve introduced in (11) has the following properties:

(a) Geometric property at the endpoints: From the properties at the endpoints of the sq-basis functions, we obtain

$$\begin{aligned} &r_n(0) = P_0, \quad r_n(1) = P_n, \\ &r'_n(0) = (\nu)(P_2 - 2P_1 + P_0) + n(P_1 - P_0), \\ &r'_n(1) = (-\nu)(P_n - 2P_{n-1} + P_{n-2}) + n(P_n - P_{n-1}). \end{aligned}$$

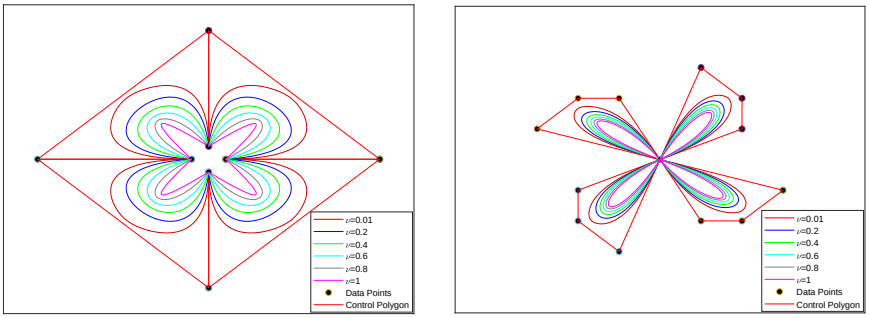


Figure 2: sq-curve with different values of ν.

(b) Symmetry: The symmetry property of the sq-basis functions leads us to

$$r_n(t; P_0, P_1, \ldots, P_n) = \sum_{i=0}^{n} b_{n,i}(t) P_i = \sum_{i=0}^{n} b_{n,n-i}(1-t) P_i$$
$$= \sum_{j=0}^{n} b_{n,j}(1-t) P_{n-j} = r_n(1-t; P_n, P_{n-1}, \ldots, P_0).$$

(c) Geometric invariance: As $r_n(t)$ is an affine combination of the control points, so the shape of the sq-curve is independent of the choice of coordinate system.

(d) Convex hull property: Because of the nonnegativity and normalization properties of the sq-basis functions, one concludes that the sq-curve must be inside the convex hull of its control polygon.

Figure 2 shows two specific curves generated by sq-curves, in which the control points are fixed and values of ν are set to $\nu = 0.01, 0.2, 0.4, 0.6, 0.8, 1$, respectively, from outside to inside. One can see that the sq-curves are to approach the control polygon as the shape parameter decreases.

3.1 Comparison with existing basis functions

The construction of a suitable set of blending function has attracted a great deal of interest among scientists. The main goal of most papers in this filed is to construct parameter based blending functions, which by altering this parameter(s), one can make most changes in the corresponding parametric curve (1). Some works focus on a specific application, and others studied

the structure and construction strategies. Here, we aim to emphasize on advantages of the sq-basis functions. As the sq-basis functions are generated recursively, we compare them with those who have a recursion formula.

Figure 3 represents a visual comparison. Figure 3(a) shows the sq-curves; Figure 3(b) represents the curves constructed in [32]; and the curves proposed in [23] are depicted in Figures 3(c) and 3(d), for different values of shape parameters. Compared to [32], the rate of change in sq-curves is greater than changing the corresponding shape parameter. The curves generated in [23] show a good rate of change for different values of shape parameters, but this is achieved by employing two separate parameters. In the sq-curves, this much of flexibility is gained by altering just one shape parameter.

In [20], no curve can be constructed for three control points because the starting basis functions have four members. Authors in [5, 6, 22, 23] employed three starting basis functions, but several shape parameters were used to make further changes to the curves. However, in the case of sq-curves, the same changes are gained with just one parameter. In some works [31, 15, 4], only four [15, 4] or five [31] bases were presented, and curves with fewer control points cannot be represented by these basis functions. For more control points, the continuity conditions are checked.

Generally, among the Bézier-like curves, which aim to present curves that move from the convex combination of the first and last control points to the control polygon by altering shape parameter, the sq-curves serve as a good choice and compete well with the other bases.

3.2 sq-surface

The corresponding sq-surfaces can be constructed by a tensor product approach. Given control points $\{P_i\}_{i=0}^{n} \subseteq \mathbb{R}^3$, the surface represented by

$$S(s,t) = \sum_{i=0}^{m} \sum_{j=0}^{n} P_{i,j} b_{m,i}(s)\, b_{n,j}(t), \qquad 0 \le s,t \le 1, \tag{12}$$

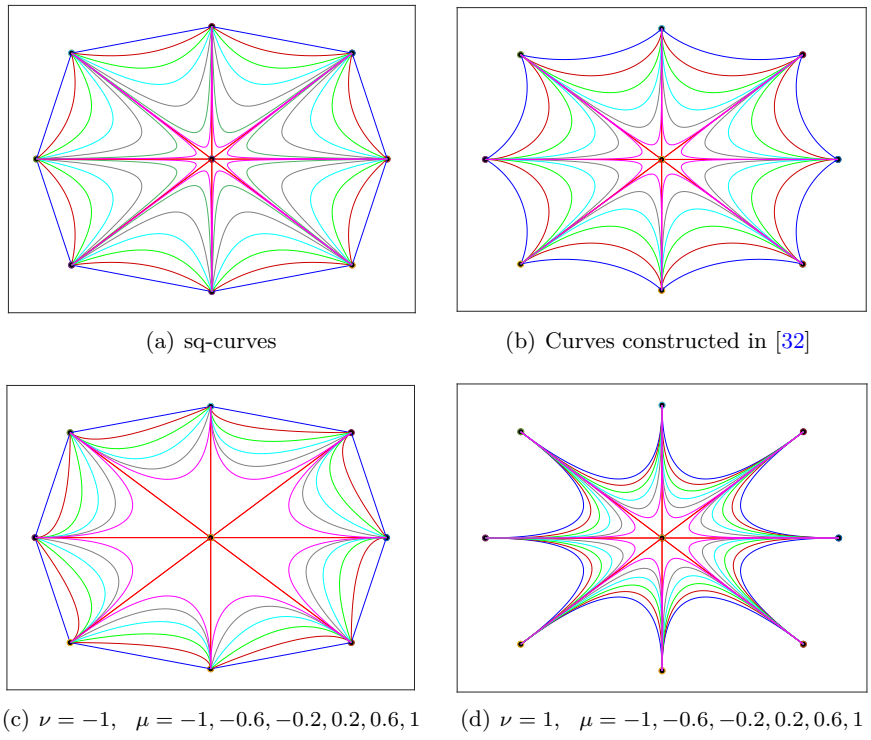
is called a sq-surface, in which $b_{m,i}(s)$ and $b_{n,j}(t)$ are the sq-basis functions.

Figure 4 shows that different surfaces can be obtained by altering the shape parameter when the control net is fixed.

From the properties of the sq-basis functions, it is easy to verify the following properties of the corresponding sq-surfaces:

(a) Interpolation at the corner points: $S(s,t)$ passes through corner points $P_{0,0}, P_{m,0}, P_{0,n}$, and $P_{m,n}$. In fact, we have $S(0,0) = P_{0,0}, S(1,0) = P_{m,0}, S(0,1) = P_{0,n}$, and $S(1,1) = P_{m,n}$.

(b) Nonnegativity: It is obvious that $b_{m,i}(s)\, b_{n,j}(t)$ is nonnegative for all m, n, i, j and $s, t \in [0,1]$.



(a) sq-curves (b) Curves constructed in [32]

(c) $\nu = -1, \quad \mu = -1, -0.6, -0.2, 0.2, 0.6, 1$ (d) $\nu = 1, \quad \mu = -1, -0.6, -0.2, 0.2, 0.6, 1$

Figure 3: Comparison of sq-curves with curves proposed in [32] and [23]

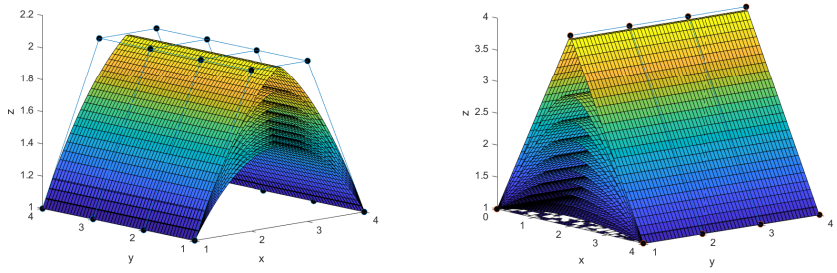


Figure 4: Shape of sq-surfaces with different shape parameters.

(c) Partition of unity:

$$\sum_{i=0}^{m}\sum_{j=0}^{n} b_{m,i}(s)\, b_{n,j}(t) = \sum_{i=0}^{m} b_{m,i}(s) \sum_{j=0}^{n} b_{n,j}(t) = 1, \qquad 0 \le s,t \le 1.$$

Proof of this is due to the partition of unity of the sq-basis functions.

(d) Convex hull property: The surface lies in the convex hull of its control points, since $S(s,t)$ is the linear combination of its control points with positive coefficients that sums to 1 (partition of unity).

(e) Affine invariance: This means that the surface defined by control points that are an affine transformation of all control points is the same surface that is obtained by applying the same transformation to the surface's equation. As $S(s,t)$ is an affine combination of the control points, so the shape of the sq-surfaces is independent of the choice of coordinate system.

(f) Boundaries of surface: The four surface boundaries are exactly the four sq-curves defined by the boundary edges of the surface as follows:

$$S(0,t) = \sum_{j=0}^{n} P_{0,j} b_{n,j}(t), \quad S(1,t) = \sum_{j=0}^{n} P_{m,j} b_{n,j}(t),$$
$$S(s,1) = \sum_{i=0}^{m} P_{i,n} b_{m,i}(s), \quad S(s,0) = \sum_{i=0}^{m} P_{i,0} b_{m,i}(s).$$

4 Shape-preserving approximation by sq-curves and sq-surfaces

The sq-curves preserve monotonicity and the sq-surfaces are axially monotonicity preserving. The newly defined models preserve boundedness in both two-dimensional and three-dimensional data. Here we state these facts in a detailed discussion.

4.1 Monotonicity preservation of sq-curves

Here, we present a proof of the monotonicity preservation of the sq-basis functions and sq-curves.

Definition 3. A system of functions $(g_0, \ldots, g_n)$ is monotonicity preserving if for any $\lambda_0 \le \lambda_1 \le \cdots \le \lambda_n$ in $\mathbb{R}$, the function $\sum_{i=0}^{n} \lambda_i g_i$ is increasing.

The following result, which characterizes monotonicity preserving systems, appears in [7, Proposition 2.3].

Proposition 1. Let $(g_0, \ldots, g_n)$ be a system of functions defined on an interval $[a, b]$. Let $v_i := \sum_{j=i}^{n} g_j$ for $i \in \{0, 1, \ldots, n\}$. Then $(g_0, \ldots, g_n)$ is monotonicity preserving if and only if v_0 is a constant function and the functions v_i are increasing for $i = 1, \ldots, n$.

We obtain the following result using the previous proposition.

Proposition 2. The sq-basis functions $(b_{n,0}, b_{n,1}, \ldots, b_{n,n})$ defined by (2) and (3) are monotonicity preserving.

Proof. Let us consider sq-basis functions (2) with $n = 2$. Using Proposition 1, $(b_{2,0}, b_{2,1}, b_{2,2})$ is monotonicity preserving if v_0 is a constant function and the functions v_1 and v_2 are increasing. Moreover,

$$v_0 = \sum_{j=0}^{2} b_{2,j}(t) = 1$$

and

$$v_1 = \sum_{j=1}^{2} b_{2,j}(t) = b_{2,1}(t) + b_{2,2}(t) = \frac{1}{2} + t - \varphi(t) \Rightarrow v_1'(t) = 1 - \varphi'(t).$$

The function $\varphi(t)$ is a convex function in the interval $[0, 1]$, so the decreasing function $-\varphi'(t)$ takes its minimum at $t = 1$. Thus we have $v_1'(1) = \nu$, so $v_1'(t) \geq 0$ for all $t \in [0, 1]$. Therefore the function $v_1(t)$ is increasing. We have

$$v_2 = \sum_{j=2}^{2} b_{2,j}(t) = b_{2,2}(t) = t - \frac{1}{2} + \varphi(t) \Rightarrow v_2'(t) = 1 + \varphi'(t).$$

Again, employing the convexity of $\varphi(t)$, one observes that the increasing function $\varphi'(t)$ takes its minimum at $t = 0$. Since $v_2'(0) = \nu$ results in $v_2'(t) \geq 0$, for all $t \in [0, 1]$, so we can conclude that $v_2(t)$ is an increasing function.

Suppose that the sq-basis functions $(b_{n-1,0}, b_{n-1,1}, \ldots, b_{n-1,n-1})$ is monotonicity preserving. Then using Proposition 1, $(b_{n,0}, b_{n,1}, \ldots, b_{n,n})$ is monotonicity preserving if v_0 is a constant function and the functions v_i are increasing for $i = 1, \ldots, n$.

We have

$$v_0 = \sum_{j=0}^{n} b_{n,j}(t) = 1$$

and

$$v_i = \sum_{j=i}^{n} b_{n,j}(t) \Rightarrow v_i'(t) = \sum_{j=i}^{n} b_{n,j}'(t). \tag{13}$$

By combining formulas (3) and (13), we have

$$\begin{aligned} v_i'(t) &= -\sum_{j=i}^{n} b_{n-1,j}(t) + (1-t)\sum_{j=i}^{n} b'_{n-1,j}(t) + \sum_{j=i}^{n} b_{n-1,j-1}(t) + t\sum_{j=i}^{n} b'_{n-1,j-1}(t) \\ &= b_{n-1,n} + \left(-\sum_{j=i}^{n-1} b_{n-1,j}(t) + \sum_{j=i}^{n-1} b_{n-1,j}(t) \right) + b_{n-1,i-1}(t) \\ &\quad +(1-t)\sum_{j=i}^{n} b'_{n-1,j}(t) + t\sum_{j=i}^{n} b'_{n-1,j-1}(t) \\ &= b_{n-1,i-1}(t) + (1-t)\sum_{j=i}^{n-1} b'_{n-1,j}(t) + t\sum_{j=i-1}^{n-1} b'_{n-1,j}(t). \end{aligned} \tag{14}$$

Since $(b_{n-1,0}, b_{n-1,1}, \ldots, b_{n-1,n-1})$ is monotonicity preserving, we deduce

$$\sum_{j=i}^{n-1} b'_{n-1,j}(t) \geq 0, \qquad \sum_{j=i-1}^{n-1} b'_{n-1,j}(t) \geq 0. \tag{15}$$

Now, from (14) and (15), for $t \in [0,1]$, it is obvious that $v_i'(t) \geq 0$. □

The sq-curve is a curve in $\mathbb{R}^2$ plane, so we need the following statement to assure the monotonicity preservation of sq-curves. The proof is straightforward.

Proposition 3. Let $(g_0, \ldots, g_n)$ be a system of functions that is monotonicity preserving. If $\{P_i\}_{i=0}^{n} \subseteq \mathbb{R}^2$ is the monotone data, then the function $\sum_{i=0}^{n} P_i g_i$ is increasing.

4.2 Monotonicity preservation of sq-surfaces

The sq-surfaces (12), defined on rectangular patches, are axially monotonicity-preserving. To verify this assertion, we state our reasoning based on some known results from the literature.

Given two strictly increasing sequences of abscissas $\alpha = (\alpha_0, \alpha_1, \ldots, \alpha_m)$ and $\beta = (\beta_0, \beta_1, \ldots, \beta_n)$ and control points $\left\{ \binom{\alpha_i}{\beta_j}_{c_{ij}} \right\}_{i=0,\ldots,m}^{j=0,\ldots,n}$, we define the control net $p : [\alpha_0, \alpha_m] \times [\beta_0, \beta_n] \to \mathbb{R}$ to be the unique function that satisfies the interpolation conditions $p(\alpha_j, \beta_j) = c_{ij}$, for all $i = 0, 1, \ldots, m$, and $j = 0, 1, \ldots, n$, and is bilinear on each rectangle $R_{ij} = [\alpha_i, \alpha_{i+1}] \times [\beta_j, \beta_{j+1}]$.

A bivariate function g is increasing in a direction $d = (d_1, d_2) \in \mathbb{R}^2$, if $g(x + \lambda d_1, y + \lambda d_2) \geqslant g(x, y), \lambda > 0$. In particular, the control net p can be

increasing in a direction d. This case has been characterized by Floater and Peña [10] in the following way.

Lemma 1. The control net p is increasing in the direction $d = (d_1, d_2) \in \mathbb{R}^2$ if and only if for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$,

$$d_1 \Delta_1 c_{i,j+l} + d_2 \Delta_2 c_{i+k,j} \geq 0, \quad k, l \in \{0, 1\},$$

where $\Delta_1 c_{ij} := (c_{i+1,j} - c_{i,j}) / (\alpha_{i+1} - \alpha_i)$ and $\Delta_2 c_{ij} := (c_{i,j+1} - c_{i,j}) / (\beta_{j+1} - \beta_j)$.

Given a sequence $(c_{ij})_{0 \leq i \leq m}^{0 \leq j \leq n}$, we have $\Lambda_1 c_{ij} : c_{i+1,j} - c_{i,j}$ for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$ and $\Lambda_2 c_{ij} : c_{i,j+1} - c_{i,j}$ for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$.

Remark 1. As a consequence of Lemma 1, we have that the control net p is increasing in the X-axis direction $d = (1, 0)$ if and only if for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$, $\Lambda_1 c_{ij} \geq 0$. Analogously, the control net p is increasing in the Y-axis direction $d = (0, 1)$ if and only if for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$, $\Lambda_2 c_{ij} \geq 0$.

Theorem 2. If the control net p is increasing in the X-axis direction $d = (1, 0)$ or in the Y-axis direction $d = (0, 1)$, then so is sq-surface $S(s, t)$.

Proof. Suppose that the control net p is increasing in the X-axis direction $d = (1, 0)$. Then for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, n-1$, we have $\Lambda_1 c_{ij} \geq 0$. We show that, in this case, $S(s, t)$ is increasing in the direction $d = (1, 0)$. Therefore,

$$S(s,t) = \sum_{i=0}^{m} \sum_{j=0}^{n} \begin{pmatrix} \alpha_i \\ \beta_j \\ c_{ij} \end{pmatrix} b_{m,i}(s)\, b_{n,j}(t), \qquad 0 \leq s, t \leq 1.$$

The surface $S(s, t)$ in the direction $d = (1, 0)$ for specified $y = \gamma$ is a curve that is obtained from the collision of the plane $y = \gamma$ and the surface $S(s, t)$, so

$$\sum_{i=0}^{m} \sum_{j=0}^{n} \beta_j b_{m,i}(s)\, b_{n,j}(t) = \gamma \Rightarrow \sum_{i=0}^{m} b_{m,i}(s) \sum_{j=0}^{n} \beta_j b_{n,j}(t) = \gamma$$
$$\Rightarrow \sum_{j=0}^{n} \beta_j b_{n,j}(t) = \gamma.$$

Therefore t is fixed and the curve on plane $y = \gamma$ is as follows:

$$\sum_{j=0}^{n} b_{n,j}(t) \left[\sum_{i=0}^{m} \begin{pmatrix} \alpha_i \\ c_{ij} \end{pmatrix} b_{m,i}(s) \right], \qquad 0 \leq s \leq 1. \tag{16}$$

Now from Proposition 3, one observes that $\sum_{i=0}^{m} \binom{\alpha_i}{c_{ij}} b_{m,i}(s)$ is increasing for any j, which lead us to the conclusion that the curve (16) would be increasing. Because the value of γ was arbitrary, the surface $S(s,t)$ is increasing in the direction $d = (1,0)$. One can prove the desired result in the direction $d = (0,1)$, in a similar way. □

Example 1. The set of data presented in Table 1 is generated from the function

$$F(x,y) = \ln(x^2 + y^2). \tag{17}$$

We have $\left\{ \binom{x_i}{y_j}_{F(x_i,y_j)} \right\}_{i=0,\dots,6}^{j=0,\dots,6}$ as a data set in a three-dimensional space. These provide us 49 control points. By these control points, we have a control net p, which is increasing in the X-axis direction $d = (1,0)$ and in the Y-axis direction $d = (0,1)$. Figure 5 is the visual model of monotone surfaces in the X-axis direction and in the Y-axis direction that are obtained by control points and (12) with different values of free parameter ν. The error rate of monotone surfaces has been reported.

Table 1: Data generated from the function $F(x,y) = \ln(x^2 + y^2)$

y	x						
	20	40	60	80	100	200	300
20	6.6846	7.6009	8.2940	8.8247	9.2496	10.6066	11.4120
40	7.6009	8.0709	8.5564	8.9872	9.3588	10.6359	11.4252
60	8.2940	8.5564	8.8818	9.2103	9.5178	10.6828	11.4468
80	8.8247	8.9872	9.2103	9.4572	9.7050	10.7451	11.4763
100	9.2496	9.3588	9.5178	9.7050	9.9035	10.8198	11.5129
200	10.6066	10.6359	10.6828	10.7451	10.8198	11.2898	11.7753
300	11.4120	11.4252	11.4468	11.4763	11.5129	11.7753	12.1007

4.3 Bound preservation of the sq-model

Suppose that we have a discrete data that is generated from a bounded phenomenon, either in a two-dimensional space or in dimension three. When we attempt to visualize the data using approximation methods, it is natural to expect the approximation to adhere to the boundedness [26, 25]. The sq-curves and sq-surfaces fulfill the convex hull property, which is the powerful feature that guarantees the bound-preserving approximation. Actually this is a feature that is in common for all Bézier-like model with convex hull property. When we approximate a data with a Bézier-like model, the curve or surface lies in the convex hull of the control points. If we set the data to be the control points, then the curve (surface) would be entirely inside the control polygon (control net), which guarantees the boundedness.

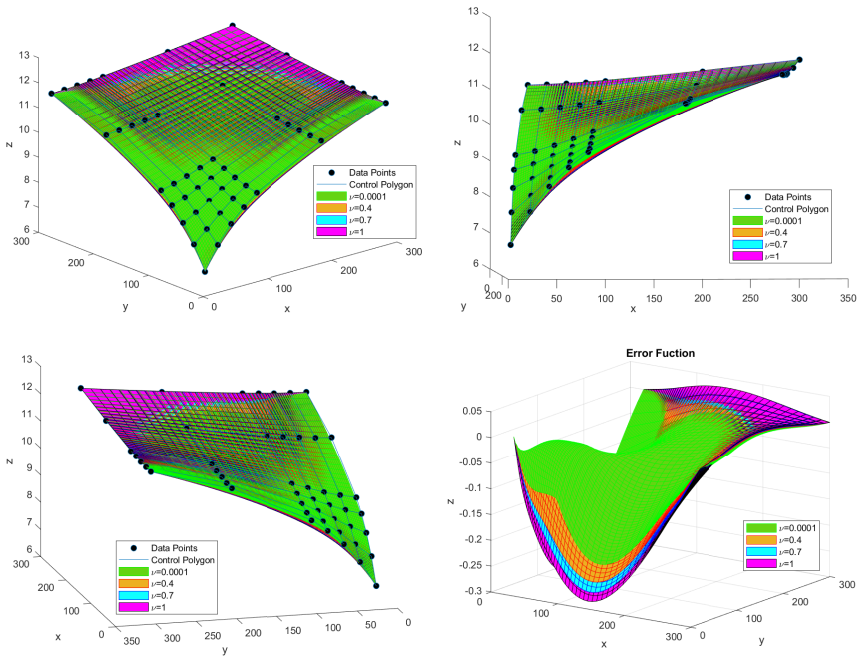


Figure 5: Axial monotonicity-preserving approximation with different shape parameters and the error functions

5 Conclusion

We presented Bézier-like parametric curves and surfaces through introducing a new class of blending functions with one shape parameter. These curves and surfaces, in addition to covering many properties of classical Bézier models, can be adjusted in a shape by altering the shape parameter when the control points are fixed. It is been verified that the newly defined sq-models can be successfully used for monotonicity-preserving approximation. The bound preservation property of the sq-models has also been reviewed.

Note that the introduction of shape parameter not only brings an advantage to the shape modification of the Bézier-like curves but also provides an optimized parameter for the shape optimization design of the Bézier-like ones. Hence, the research on how to utilize the SDABWO [12] to solve the model of curve shape optimization, which takes the shape parameter as the optimization variables, could be addressed in future works.

References

1. Abbas, M., Majid, A.A. and Ali, J.M. *Monotonicity-preserving C2 rational cubic spline for monotone data*, Appl. Math. Comput., 219(6) (2012), 2885–2895.

2. Abbas, M., Majid, A.A., Awang, M.H. and Ali, J.M. *Shape-preserving rational bi-cubic spline for monotone surface data*, 'WSEAS Trans. Math. 11(7) (2012), 660–673.

3. Abbas, M., Majid, A.A., Awang, M.N.H. and Ali, J.M. *Monotonicity-preserving rational bi-cubic spline surface interpolation*, Sci. Asia S, 40 (2014), 22–30.

4. Bashir, U., and Ali, J. M. *Rational cubic trigonometric Bézier curve with two shape parameters*, Comput. Appl. Math. 35(1) (2016), 285–300.

5. BiBi, S., Abbas, M., Misro, M.Y. and Hu, G. *A novel approach of hybrid trigonometric Bézier curve to the modeling of symmetric revolutionary curves and symmetric rotation surfaces*, IEEE Access, 7 (2019), 165779–165792.

6. BiBi, S., Abbas, M., Miura, K.T. and Misro, M.Y. *Geometric modeling of novel generalized hybrid trigonometric Bézier-like curve with shape parameters and its applications*, Mathematics, 8(6) (2020), 967.

7. Carnicer, J.M., Garcia-Esnaola, M. and Peña, J.M. *Convexity of rational curves and total positivity*, J. Comput. Appl. Math. 71(2) (1996), 365–382.

8. Chen, J. and Wang, G.J. *A new type of the generalized Bézier curves*, Appl. Math. J. Chinese Univ. Ser. B, 26(1) (2011), 47–56.

9. Farin, G. E., and Farin, G. *Curves and surfaces for CAGD: a practical guide*, Morgan Kaufmann, 2002.

10. Floater, M.S. and Peña, J.M. *Tensor-product monotonicity preservation*, Adv. Comput. Math. 9(3) (1998), 353–362.

11. Hu, G., Bo, C., Wei, G. and Qin, X. *Shape-adjustable generalized Bézier Surfaces: Construction and its geometric continuity conditions*, Appl. Math. Comput. 378 (2020), 125215.

12. Hu, G., Du, B., Wang, X. and Wei, G. *An enhanced black widow optimization algorithm for feature selection*, Knowl Based Syst. 235 (2022), 107638.

13. Hu, X., Hu, G., Abbas, M. and Misro, M.Y. *Approximate multi-degree reduction of Q-Bézier curves via generalized Bernstein polynomial functions*, Adv. Differ. Equ., 2020(1) (2020), 1–16.

14. Hu, G. and Wu, J.L. *Generalized quartic H-Bézier curves: Construction and application to developable surfaces*, Adv. Eng. Softw. 138 (2019), 102723.

15. Hu, G., Wu, J. and Qin, X. *A new approach in designing of local controlled developable H-Bézier surfaces*, Adv. Eng. Softw. 121 (2018), 26–38.

16. Hu, G., Wu, J. and Qin, X. *A novel extension of the Bézier model and its applications to surface modeling*, Adv. Eng. Softw. 125 (2018), 27–54.

17. Juhász, I. *A NURBS transition between a Bézier curve and its control polygon*, J. Comput. Appl. Math. 396 (2021), 113626.

18. Kovács, I. and Várady, T. *P-curves and surfaces: Parametric design with global fullness control*, Computer-Aided Design, 90 (2017), 113–122.

19. Kovács, I. and Várady, T. *P-Bézier and P-Bspline curves–new representations with proximity control*, Computer Aided Geom. Des. 62 (2018), 117–132.

20. Li, J. *A novel Bézier curve with a shape parameter of the same degree*, Results Math. 73(4) (2018), 1–11.

21. Majeed, A., Abbas, M., Qayyum, F., Miura, K.T., Misro, M.Y. and Nazir, T. *Geometric modeling using new Cubic trigonometric B-Spline functions with shape parameter*, Mathematics, 8(12) (2020), 2102.

22. Maqsood, S., Abbas, M., Hu, G., Ramli, A.L.A. and Miura, K.T. *A novel generalization of trigonometric Bézier curve and surface with shape parameters and its applications*, Math. Probl. Eng. 2020, Art. ID 4036434, 25 pp.

23. Maqsood, S., Abbas, M., Miura, K.T., Majeed, A. and Iqbal, A. *Geometric modeling and applications of generalized blended trigonometric Bézier curves with shape parameters*, Adv. Differ. Equ. 2020(1) (2020), 1–18.

24. Qin, X., Hu, G., Zhang, N., Shen, X. and Yang, Y. *A novel extension to the polynomial basis functions describing Bézier curves and surfaces of degree n with multiple shape parameters*, Appl. Math. Comput. 223 (2013), 1–16.

25. Saeidian, J., Sarfraz, M., Azizi, A. and Jalilian, S. *A new approach of constrained interpolation based on cubic Hermite splines*, J. Math. 2021, Art. ID 5925163, 10 pp.

26. Saeidian, J., Sarfraz, M. and Jalilian, S. *Bound-preserving interpolation using quadratic splines*, Journal of Mathematical Modeling, (2020), 1–13.

27. Sarfraz, M., Hussain, M.Z. and Hussain, F. *Shape preserving convex data interpolation*, Appl. Comput, Math. 16(3) (2017), 205–227.

28. Sarfraz, M., Samreen, S. and Hussain, M.Z. *A Quadratic Trigonometric Nu Spline with Shape Control*, Int. J. Image Graph. 17(03) (2017), 1750015.

29. Tariq, Z., Ibraheem, F., Hussain, M.Z. and Sarfraz, M. *Monotone data modeling using rational functions*, Turk. J. Electr. Eng. Comput. Sci. 27(3) (2019), 2331–2343.

30. Usman, M., Abbas, M., and Miura, K.T. *Some engineering applications of new trigonometric cubic Bézier-like curves to free-form complex curve modeling*, J. Adv. Mech. Des. Syst. Manuf. 14(4) (2020), JAMDSM0048-JAMDSM0048.

31. Yan, L. *Adjustable Bézier curves with simple geometric continuity conditions*, Math. Comput. Appl. 21(4) (2016), 44.

32. Yan, L. and Liang, J. *An extension of the Bézier model*, Appl. Math. Comput. 218(6) (2011), 2863–2879.

How to cite this article
B. Nouri and J. Saeidian On a class of Bézier-like model for shape-preserving approximation. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 449-466. doi: 10.22067/ijnao.2022.72818.1064.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 467–482
DOI: 10.22067/IJNAO.2022.74347.1086
https://ijnao.um.ac.ir/

A numerical treatment based on Bernoulli Tau method for computing the open-loop Nash equilibrium in nonlinear differential games

M. Dehghan Banadaki*, H. Navidi

Abstract

The Tau method based on the Bernoulli polynomials is implemented efficiently to approximate the Nash equilibrium of open-loop kind in nonlinear differential games over a finite time horizon. By this treatment, the system of two-point boundary value problems of differential game extracted from Pontryagin's maximum principle is transferred to a system of algebraic equations that Newton's iteration method can be used for solving it. Also, for the mentioned approximation by the Bernoulli polynomials, the convergence analysis and the error upper bound are discussed. To demonstrate the applicably and accuracy of the proposed approach, some illustrated examples are presented at the final.

AMS subject classifications (2020): 35A01, 65L10, 65L12, 65L20, 65L70.

Keywords: Nonlinear differential games; Open-loop Nash equilibrium; Pontryagin's maximum principle; Bernoulli Tau method.

* Corresponding author

Received 24 December 2021; revised 20 February 2022; accepted 22 March 2022

Mojtaba Dehghan Banadaki
Department of Applied Mathematics, Shahed University, Tehran, Iran. e-mail: m_dehghan@shahed.ac.ir

Hamidreza Navidi
Department of Applied Mathematics, Shahed University, Tehran, Iran. e-mail: navidi@shahed.ac.ir

1 Introduction

A differential game is an extension of optimal control theory that describes a conflict situation between some players, who seek their maximum or minimum own payoffs under a dynamical system. It has arisen in practical problems from economics to engineering applications in recent years [29, 30, 44, 11, 15, 31].

One of the most important and crucial solution concepts in game theory is the Nash equilibrium, in which players have no incentive to deviate from their original plans [23] and is classified into the following two cases in differential games based on the information of the state of the game that players know at different times of game:

- The players have no information during the game and only know the game state at the initial time. This kind of equilibrium is known as open-loop Nash equilibrium.
- The current game state is known to players. Such equilibrium is often called feedback Nash equilibrium.

The main approaches to computing the open-loop Nash equilibrium in differential games are indirect methods and direct methods. In indirect methods, the nonzero-sum differential game is reduced to a system of two-point boundary value problems (TPBVPs) by using the necessary optimality conditions of the Pontryagin's maximum principle that can be solved analytically or numerically [5]. In direct approaches that are optimization based methods, differential game problem is transferred to mathematical programming [12, 20]. However, the drawback of direct approaches is that there is no guarantee that the solution obtained is feasible for the original problem [27, 42].

Regarding the indirect methods, most researchers have focused on a special kind of differential game, namely linear-quadratic dynamic games that the state equation of the game is linear with respect to control and state variables, and both are quadratic in the performance indices. For this kind of differential game, the systems of TPBVPs are linear in general, and hence the open-loop Nash equilibrium can be obtained analytically based on solving Riccati equations [14, 17]. Indeed in practice, we face with differential games that their systems of TPBVPs are nonlinear generally. Therefore, using suitable numerical methods is necessary [35].

To the best of our knowledge, there are a few research works carried out to compute open-loop Nash differential games in nonlinear case. In [27], a pseudospectral method based on Chebyshev polynomials was applied for finding the players' open-loop strategies in nonlinear differential games. In [24], by Riccati equations, the open-loop Nash equilibrium of differential games in polynomial case was obtained. In [9], the coordinate transformation approach was extended for computing open-loop Nash equilibrium, and complementarity theory was applied for a class of zero-sum differential games to be solved

in [43]. In [32], a combined quasilinearization method with exponential Bernstein functions was introduced for a numerical solution to TPBVPs.

There are several methods for solving differential equations numerically, such as spectral methods [8], shooting and multiple shooting methods [25], variational iteration method [16], and homotopy analysis method [1] that each of which has its own implementation. Spectral methods are based on the weighted residual method that has high accurate results in solving differential equations and are classified into three methods, namely collocation, Galerkin, and Tau methods [19, 22, 34, 41].

The Tau method is one of the most accurate spectral methods for a numerical solution to differential equations of different kinds [39, 3, 38, 18]. The goal of this paper is to propose an implementation of the Bernoulli Tau method (BTM), in which the solution functions are defined by means of a truncated Bernoulli series expansion, to compute the open-loop Nash equilibrium in nonlinear differential games with finite horizons.

The remainder of the paper is organized into the following sections. In Section 2, the nonzero-sum nonlinear differential games are defined, and the extraction of the systems of TPBVPs from Pontryagin's maximum principle is described. In Section 3, the Tau approach based on the Bernoulli polynomials is introduced and applied for computing the open-loop strategies of these differential games. In Section 4, some numerical examples are presented to validate the accuracy and applicability of the present method. Finally, conclusions are presented in section 5.

2 Problem statement

A family of nonzero-sum nonlinear differential games with finite horizon is described in the following definition.

Definition 1. A nonzero-sum nonlinear differential game is defined as follows [6]:

$$\begin{aligned}
&\max_{u_i(\cdot)} J_i(u_i(\cdot), u_{-i}(\cdot)) = \int_0^T K_i(t, x(t), u_1(t), u_2(t), \ldots, u_m(t))\, dt + \Phi_i(x(T)),\\
&\dot{x}(t) = f(t, x(t), u_1(t), u_2(t), \ldots, u_m(t)),\\
&x(0) = x_0 \in \mathbb{R},
\end{aligned} \tag{1}$$

where $u_i(t) \in U_i \subset \mathbb{R}$ is the player i's control (strategy), $x(t) \in \mathbb{R}$ is the state vector of the differential game, and $M = \{1, 2, \ldots, m\}$ is the set of players. The functions $K_i(t, x(t), u_1(t), u_2(t), \ldots, u_m(t))$ and $\Phi_i(x(T)), i = 1, 2, \ldots, m$, are continuously differentiable functions that describe the player i's running payoff and terminal payoff, respectively. The goal of this differential game for each player i, $i = 1, 2, \ldots, m$, is to maximize his payoff by choosing a suitable strategy $u_i(t) \in U_i \subset \mathbb{R}$.

For differential game (1), the open-loop Nash equilibrium is described as follows.

Definition 2. The control actions $u_i^\star(\cdot), i = 1, 2, \ldots, m$, are considered as a Nash equilibrium for differential game (1), if the following inequalities hold:

$$J_i(u_i^\star(\cdot), u_{-i}^\star(\cdot)) \geqslant J_i(u_i(\cdot), u_{-i}^\star(\cdot)), \quad \text{for all } u_i \in U_i,$$

where u_i is the ith player's strategy and u_{-i} state the other players' strategies, that is, $u_{-i} = u_j, j \neq i$.

For deriving the first-order optimality necessary conditions of nonlinear differential game (1) and characterizing an open-loop strategy, the Hamiltonian functions are defined as follows:

$$H_i(t, x, u_i, u_{-i}, \lambda_i) = K_i(t, x, u_i, u_{-i}) + \lambda_i\, f(t, x, u_i, u_{-i}), \qquad i = 1, 2, \ldots, m,$$

where the variables λ_i, $i = 1, 2, \ldots, m$, are the adjoint functions.

Pontryagin's maximum principle provides a set of optimality conditions for control actions to construct an open-loop strategy in nonlinear differential game (1) as follows:

$$\begin{cases} \dot{x}(t) = f(t, x(t), u_1(t), \ldots, u_m(t)), \quad x(0) = x_0, & (2)\\ \dot{\lambda}_i(t) = -\dfrac{\partial H_i}{\partial x}(t, x(t), u_i(t), u_{-i}(t), \lambda_i(t)), \quad \lambda_i(T) = \dfrac{\partial \Phi_i(x(T))}{\partial x}, & (3)\\ \dfrac{\partial H_i}{\partial u_i}(t, x(t), u_i(t), u_{-i}(t), \lambda_i(t)) = 0, \quad i = 1, 2, \ldots, m. & (4) \end{cases}$$

An expression for $u_i(t)$, $i = 1, 2, \ldots, m$, with respect to $x(t)$ and $\lambda_i(t)$ can be obtained by solving the algebraic equations (4) as follows:

$$u_i = \Psi_i(t, x(t), \lambda_i(t)).$$

This expression is replaced in (2) and (3) to obtain the system of TPBVPs based on $x(t)$ and $\lambda_i(t)$, $i = 1, 2, \ldots, m$, as follows:

$$\begin{cases} \dot{x}(t) = f(t, x(t), \Psi_1(t), \Psi_2(t), \ldots, \Psi_m(t)),\\ \dot{\lambda}_i(t) = -\dfrac{\partial H_i}{\partial x}(t, x(t), \Psi_i(t), \Psi_{-i}(t), \lambda_i(t)),\\ x(0) = x_0,\\ \lambda_i(T) = \dfrac{\partial \Phi_i(x(T))}{\partial x}, \end{cases} \tag{5}$$

where $\Psi_i = \Psi_i(t, x(t), \lambda_i(t))$, $i = 1, 2, \ldots, m$.

This system of differential equations with split boundary conditions is nonlinear generally, which makes it difficult or impossible to be solved analytically. Therefore, using an appropriate numerical approach is required.

3 The Bernoulli Tau method for nonlinear differential games

In this part, an efficient formulation of the Tau method for a numerical solution to the system of TPBVPs is established by obtaining the open-loop strategy of nonlinear differential game (1).

The Tau method is a highly accurate spectral method for differential equations to be solved numerically. Implementing this method is based on expanding the solution functions $f(t)$ of differential equations in terms of suitable basis polynomials such as Bernoulli [40, 33], Jacobi [4, 28], and Bernstein polynomials [21] as follows:

$$f(t) = \sum_{i=0}^{\infty} f_i \, P_i(t),$$

where f_i and $P_i(t)$, $i = 0, 1, 2, \ldots$, are unknown coefficients and basis polynomials, respectively [7].

In practice, we use only a finite number of these basis polynomials, meaning that $f^n(t) = \sum_{i=0}^{n} f_i \, P_i(t)$ is a numerical approximation of the exact solution $f(t)$.

In this paper, the Bernoulli polynomials are considered as basis polynomials, in which the definition and properties of these polynomials in a function approximation are stated below.

Definition 3. Bernoulli polynomials of order n are defined on $[0, 1]$ by (see [26])

$$\beta_n(t) = \sum_{i=0}^{n} \binom{n}{i} \alpha_{n-i} t^i,$$

where α_i, $i = 0, 1, \ldots, n$, are Bernoulli numbers and defined as

$$\frac{t}{e^t - 1} = \sum_{i=0}^{\infty} \alpha_i \frac{t^i}{i!}.$$

The first few Bernoulli numbers are

$$\alpha_0 = 1, \quad \alpha_1 = \frac{-1}{2}, \quad \alpha_2 = \frac{1}{6}, \quad \alpha_4 = \frac{-1}{30}, \ldots,$$

with $\alpha_{2i+1} = 0$, for $i = 1, 2, 3, \ldots$.

The first seven Bernoulli polynomials are

$$\beta_0(t) = 1, \quad \beta_1(t) = t - \frac{1}{2}, \quad \beta_2(t) = t^2 - t + \frac{1}{6}, \quad \beta_3(t) = t^3 - \frac{3}{2}t^2 + \frac{1}{2}t,$$
$$\beta_4(t) = t^4 - 2t^3 + t^2 - \frac{1}{30}, \quad \beta_5(t) = t^5 - \frac{5}{2}t^4 + \frac{5}{3}t^3 - \frac{1}{6}t,$$

$$\beta_6(t) = t^6 - 3t^5 + \frac{5}{2}t^4 - \frac{1}{2}t^2 + \frac{1}{42}.$$

A complete basis is formed by these polynomials over the interval $[0, 1]$.

Any function $f(t)$ belonging to $L^2[0, 1]$ can be approximated by Bernoulli functions as follows:

$$f(t) \approx f^n(t) = \sum_{i=0}^{n} f_i \beta_i(t).$$

To apply the Tau method based on the Bernoulli polynomials for solving the system of TPBVPs (5), for simplification matters and without loss of generality, we consider $T = 1$, and then the unknown functions $x(t)$ and $\lambda_i(t)$, $i = 1, \ldots, m$, can be approximated as finite expansions of Bernoulli polynomials as follows:

$$\begin{aligned} x(t) &\approx x^n(t) = \sum_{j=0}^{n} a_j \beta_j(t) = A^T \beta(t) \\ \lambda_i(t) &\approx \lambda_i^n(t) = \sum_{j=0}^{n} b_{ij} \beta_j(t) = B_i^T \beta(t), \quad i = 1, 2, \ldots, m\,, \end{aligned}$$

where $A^T = [a_0, a_1, \ldots, a_n]$ and $B_i^T = [b_{i0}, b_{i1}, \ldots, b_{in}]$, $i = 1, \ldots, m$, are the vectors of unknown coefficients and $\beta(t) = [\beta_0(t), \beta_1(t), \ldots, \beta_n(t)]^T$ is the vector of Bernoulli polynomials.

The residual functions are defined by substituting these expansions in the differential equations of the system of TPBVPs (5) as follows:

$$\begin{aligned} R_0(t) &= \dot{x}^n(t) - f(t, x^n(t), \Psi_1^n(t), \Psi_2^n(t), \ldots, \Psi_m^n(t)), \\ R_i(t) &= \dot{\lambda}_i^n(t) + \frac{\partial H}{\partial x^n}(t, x^n(t), \Psi_i^n(t), \Psi_{-i}^n(t)), \quad i = 1, \ldots, m. \end{aligned}$$

Then, multiplying these residuals by $\beta_j(t)$, $j = 0, 1, \ldots, n-1$, integrating over the interval $[0, 1]$, and setting equal to zero, together with the boundary values, the following system of $(m+1)(n+1)$ algebraic equations is created, which Newton's iteration method can be applied to solve it and to determine the unknown vectors A^T *and* B_i^T, $i = 1, \ldots, m$:

$$\begin{cases} \displaystyle\int_0^1 R_0(t) B_j(t) dt = 0, \\ \displaystyle\int_0^1 R_i(t) B_j(t) dt = 0, \\ x^n(0) = x_0, \\ \lambda_i^n(1) = \dfrac{\partial \Phi_i(x^n(1))}{\partial x^n}. \end{cases}$$

By the following theorem, the convergence analysis and the error upper bound for the mentioned approximation obtained by the Bernoulli polynomials is discussed.

Theorem 1. Suppose that $x(t)$ and $\lambda_i(t), i = 1, \ldots, m$, belong to $C^{n+1}[0,1]$ and that $S_n = span\{\beta_0(t), \beta_1(t), \ldots, \beta_n(t)\}$. If $A^T\beta(t) \in S_n$ and $B_i^T\beta(t) \in S_n, i = 1, 2, \ldots, m$, are the best approximations of $x(t)$ and $\lambda_i(t)$, $i = 1, \ldots, m$, respectively, then

$$\|x(t) - A^T\beta(t)\|_{L^2[0,1]} \leq \frac{C}{(n+1)!\sqrt{2n+3}}$$

and

$$\|\lambda_i(t) - B_i^T\beta(t)\|_{L^2[0,1]} \leq \frac{C_i}{(n+1)!\sqrt{2n+3}}, \quad i = 1, 2, \ldots, m,$$

where $C = \max\limits_{t\in[0,1]} |x^{(n+1)}(t)|$ and $C_i = \max\limits_{t\in[0,1]} |\lambda_i^{(n+1)}(t)|$, $i = 1, 2, \ldots, m$.

Proof. The proof will be done for the first inequality and the other inequalities can be proved in a similar manner.

Since $x(t) \in C^{n+1}[0,1]$, there exists $C \in \mathbb{N}$ such that for every $t \in [0,1]$, we have $|x^k(t)| \leq C, k = 0, 1, \ldots, n+1$, and $x(t)$ can be expanded into the Taylor formula as

$$x(t) = \sum_{k=0}^{n} \frac{x^{(k)}(0)}{k!}t^k + \frac{x^{(n+1)}(\xi)}{(n+1)!}t^{n+1} = \tilde{x}(t) + \frac{x^{(n+1)}(\xi)}{(n+1)!}t^{n+1},$$

where $\tilde{x}(t) = \sum\limits_{i=0}^{n} \frac{x^{(k)}(0)}{k!}t^k$ and $\xi \in [0,t]$. Hence, we have

$$|x(t) - \tilde{x}(t)| = \frac{x^{(n+1)}(\xi)}{(n+1)!}t^{n+1}.$$

Because $A^T\beta(t)$ is the best approximation of $x(t)$ out of S_n, $\tilde{x}(t) \in S_n$, and considering the above equality, it is concluded that

$$\begin{aligned}\|x(t) - A^T\beta(t)\|_{L^2[0,1]} \leq \|x(t) - \tilde{x}(t)\|_{L^2[0,1]} &= \left\| \frac{x^{(n+1)}(\xi)}{(n+1)!}t^{n+1} \right\|_{L^2[0,1]} \\ &= \left(\int_0^1 \left| \frac{x^{(n+1)}(\xi)}{(n+1)!}t^{n+1} \right|^2 dt \right)^{\frac{1}{2}} \\ &\leq \frac{C}{(n+1)!} \left(\int_0^1 t^{2n+2} dt \right)^{\frac{1}{2}}\end{aligned}$$

$$= \frac{C}{(n+1)!\sqrt{2n+3}},$$

and this completes the proof. □

Remark 1. It should be noted that for practical use of Bernoulli polynomials on the interval $[a, b]$, it is necessary to shift the defining domain by the following variable substitution and construct the shifted Bernoulli polynomials:

$$t = \frac{x}{b-a} - \frac{a}{b-a}.$$

4 Numerical illustrations

In this part, three differential game problems are presented to illustrate the accuracy and efficiency of the proposed approach. Example 1 is a linear-quadratic differential game that its exact solution can be obtained. By this example and comparing it with the exact solution, we can verify and validate the proposed approach. Example 2 is also a linear quadratic differential game with attainable exact solution. In this example, we compare the results of the proposed method with the Chebyshev pseudospectral method (CPM) presented in [27]. Example 3 is a differential game arising from an economic model with a nonlinear system of TPBVPs that the exact solution is not available. To check the performance of the proposed method for this problem, a residual function is defined.

All the computations associated with the proposed method have been performed by Maple 17 software with 32 digits precision on a Core (TM) i7 PC with 2.70GHz of CPU and 16GB of RAM.

Example 1. For this differential game, the state equation is [13]

$$\dot{x}(t) = u_1(t) + u_2(t), \quad x(0) = 1,$$

and two players' performance indices are

$$\min_{u_1} J_1 = \int_0^1 (-x^2(t) + u_1^2(t))dt,$$
$$\min_{u_2} J_2 = \int_0^1 (2x^2(t) + u_2^2(t))dt + x^2(1).$$

The exact open-loop Nash equilibrium of this differential game is [13]

$$u_1^* = -\frac{1}{e} + e^{-t},$$
$$u_2^* = \frac{1}{e} - 2e^{-t}.$$

Hence, the exact values of players' performance indices are

$$J_1^* = -0.32975303263305,$$
$$J_2^* = 1.9344880850240.$$

The system of TPBVPs for the mentioned game is stated as

$$\begin{cases} \dot{x}(t) = -\dfrac{\lambda_1(t)}{2} - \dfrac{\lambda_2(t)}{2}, \\ \dot{\lambda}_1(t) = 2x(t), \\ \dot{\lambda}_2(t) = -4x(t), \\ x(0) = 1, \\ \lambda_1(1) = 0, \quad \lambda_2(1) = 2x(1). \end{cases}$$

The values of performance indices obtained by the proposed approach and the comparison of the analytical solutions are shown in Table 1. Also, the approximate solutions and the exact solutions with $n = 10$ together with absolute errors are plotted in Figure 1.

Table 1: Comparison of optimal payoff functionals J_1 and J_2 obtained by BTM with the exact solutions and also the CPU time(s) for Example 1.

n	J_{1BTM}	J_{2BTM}	$\lvert J_{1BTM} - J_1^* \rvert$	$\lvert J_{2BTM} - J_2^* \rvert$	CPU time(s)
4	−0.32975302954236861650	1.93448814833633875533	3.09×10^{-9}	5.23×10^{-8}	0.124
6	−0.32975303263303305145	1.93448808502434431993	1.35×10^{-14}	2.27×10^{-13}	0.156
8	−0.32975303263304656749	1.93448808502406878964	1.69×10^{-20}	2.84×10^{-19}	0.218
10	−0.32975303263304656750	1.93448808502406878929	8.17×10^{-27}	1.37×10^{-25}	0.328

Example 2. For this differential game, the state equation is [13]

$$\dot{x}(t) = 2x(t) + u_1(t) + u_2(t), \quad x(0) = 1,$$

and two players' performance indices are

$$\min_{u_1} J_1 = \int_0^3 (x^2(t) + u_1^2(t))dt,$$
$$\min_{u_2} J_2 = \int_0^3 (4x^2(t) + u_2^2(t))dt + 5x^2(3).$$

The exact open-loop Nash equilibrium of this differential game is [13]

$$u_1^* = -e^{-st} + \frac{1}{e^3}e^{-2t},$$
$$u_2^* = -4e^{-3t} - \frac{1}{e^3}e^{-2t}.$$

Hence, the exact values of players' performance indices are

$$J_1^* = 0.3140381912,$$

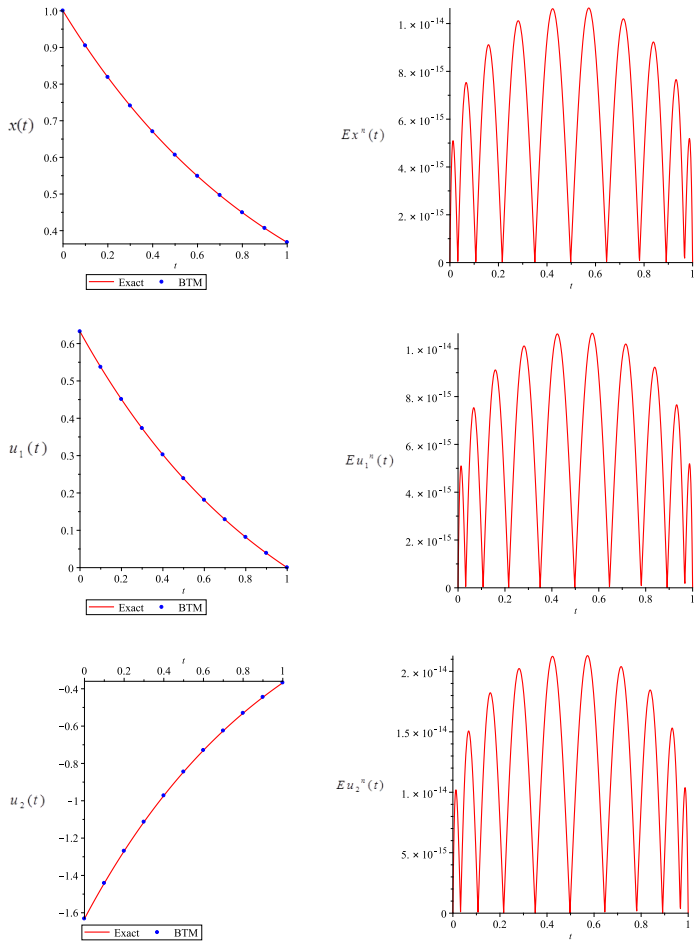


Figure 1: Plots of the approximate solutions and the analytical solutions together with absolute errors with $n = 10$ for Example 1.

$$J_2^* = 3.4136123279.$$

The system of TPBVPs for the mentioned game is stated as

$$\begin{cases} \dot{x}(t) = 2x(t) - \dfrac{\lambda_1(t)}{2} - \dfrac{\lambda_2(t)}{2}, \\ \dot{\lambda}_1(t) = -2x(t) - 2\lambda_1(t), \\ \dot{\lambda}_2(t) = -8x(t) - 2\lambda_2(t), \\ x(0) = 1, \\ \lambda_1(3) = 0, \quad \lambda_2(3) = 10x(3). \end{cases}$$

The values of performance indices obtained by the proposed approach and the comparison of the analytical solutions are shown in Table 2. Besides, to compare the BTM results with an existing approach, the results obtained by the CPM [27] are shown in Table 3.

Table 2: Comparison of optimal payoff functionals J_1 and J_2 obtained by BTM with the exact solutions and also the CPU time(s) for Example 2.

n	J_{1BTM}	J_{2BTM}	$\lvert J_{1BTM} - J_1^* \rvert$	$\lvert J_{2BTM} - J_2^* \rvert$	CPU time(s)
10	0.31403763402282	3.41361478021289	5.57×10^{-7}	2.45×10^{-6}	0.437
15	0.31403819123820	3.41361232797387	1.07×10^{-14}	4.58×10^{-14}	0.640
20	0.31403819123819	3.41361232797391	8.77×10^{-24}	3.85×10^{-23}	1.154

Table 3: Comparison of optimal payoff functionals J_1 and J_2 obtained by CPM with the exact solutions for Example 2.

n	J_{1CPM}	J_{2CPM}	$\lvert J_{1CPM} - J_1^* \rvert$	$\lvert J_{2CPM} - J_2^* \rvert$
10	0.3140689582	3.4134809955	0.0000307670	0.0001313324
15	0.3140381906	3.4136123306	6.1×10^{-10}	2.7×10^{-9}
20	0.3140381912	3.4136123279	5.2×10^{-15}	2.2×10^{-14}

Table 2 indicates that in the same situation in terms of the number of basis functions, the results obtained by the proposed method are more accurate than the results obtained by the CPM in this example.

Example 3. The following differential game describes the competition between two players in an effort for harvesting a natural renewable resource. The state equation of this game is expressed as

$$\dot{x}(t) = 0.1x(t) - 0.001x^2(t) - x(t)u_1(t) - x(t)u_2(t), \quad x(0) = 1.$$

The players' payoffs are given by

$$J_1(u_1, u_2) = \int_0^1 (3x(t)u_1(t) - \frac{1}{2}u_1^2(t))dt,$$
$$J_2(u_1, u_2) = \int_0^1 (2x(t)u_2(t) - \frac{1}{2}u_2^2(t))dt,$$

where the value $x(t) > 0$ is the resource level and the amounts $u_1(t) \geq 0$ and $u_2(t) \geq 0$ are the players' efforts for harvesting this resource, all at time t. Moreover, $\frac{1}{2}u_1^2$ and $\frac{1}{2}u_2^2$ indicate the costs for harvesting at effort levels u_1 and u_2, respectively [9].

Remark 2 (see [35])**.** By the linearity of the state equation of this differential game with respect to the control variables u_i, $i = 1, 2$, and the concavity of integrand of performance index J_i, $i = 1, 2$, with respect to u_i, $i = 1, 2$,

(because $\dfrac{\partial^2 J_i}{\partial u_i^2} = -1 < 0, i = 1, 2$), it yields that the open-loop strategy exists and is unique for this dynamic game regrading the Filippov–Cesari existence theorem [10].

The nonlinear system of TPBVPs extracted from Pontryagin's maximum principle for this differential game is stated as follows:

$$\begin{cases} \dot{x} = 0.1x - 5.001x^2 + x^2\lambda_1 + x^2\lambda_2, \\ \dot{\lambda}_1 = -9x - 0.1\lambda_1 + 8.002x\lambda_1 - x\lambda_1^2 - x\lambda_1\lambda_2, \\ \dot{\lambda}_2 = -4x - 0.1\lambda_2 + 7.002x\lambda_2 - x\lambda_2^2 - x\lambda_1\lambda_2, \\ x(0) = 1, \\ \lambda_1(1) = 0, \quad \lambda_2(1) = 0. \end{cases}$$

The numerical results for various amounts of n are presented in Table 4. It is worth mentioning that since the exact solution to this differential game is not available, to check the accuracy and validity of the proposed method for the differential game under consideration, the error of residuals is defined as follows:

$$\|Res\|^2 = \int_0^1 (R_1^2(t) + R_2^2(t) + R_3^2(t))dt,$$

where $R_i(t), i = 1, 2, 3$, are the residuals defined in the previous section.

Table 4: Optimal payoff functionals J_1 and J_2 for Example 3 with error norms and also the CPU time(s).

n	J_{1BTM}	J_{2BTM}	$\|Res\|^2$	CPU time(s)
6	0.946161437829	0.452174552034	3.74×10^{-5}	3.931
8	0.946161294373	0.452174505059	5.44×10^{-7}	12.683
10	0.946161293220	0.452174504702	7.27×10^{-9}	35.334
12	0.946161293210	0.452174504699	9.16×10^{-11}	92.486

It is notable that due to the nonlinearity of the system of TPBVPs for this example and also the process of implementing the Tau method, we expect that it consumes more time than the previous examples to be solved. Table 4 verifies this matter.

5 Conclusions

In this paper, a formulation of the Bernoulli Tau method (BTM) was established efficiently for approximating the open-loop Nash equilibrium in nonlinear differential games over a finite time horizon. Using this approach, the system of TPBVPs extracted from Pontryagin's maximum principle was

reduced to a system of algebraic equations by expanding the solution functions in terms of Bernoulli polynomials, which can be solved numerically to determine the unknown coefficients. At last, three examples were presented and solved by this approach to validate the applicably and accuracy of the present method. The approximate solutions were obtained with an excellent agreement with the exact solutions.

References

1. Abbasbandy, S., Shivanian, E., and Vajravelu, K. *Mathematical properties of ħ-curve in the frame work of the homotopy analysis method*, Commun. Nonlinear Sci. Numer. Simul., 16(11) (2011), 4268–4275.

2. Bhrawy, A., Tharwat, M., and Yildirim, A. *A new formula for fractional integrals of Chebyshev polynomials: Application for solving multi-term fractional differential equations*, Appl. Math. Model., 37(6) (2013), 4245–4252.

3. Bhrawy, A. H. and Zaky, M. A. *A method based on the Jacobi tau approximation for solving multi-term time-space fractional partial differential equations*, J. Comput. Phys., 281 (2015), 876–895.

4. Bhrawy, A. H., Zaky, M. A., and Baleanu, D. *New numerical approximations for space-time fractional burgers' equations via a Legendre spectral-collocation method*, Rom. Rep. Phys., 67(2) (2015), 340–349.

5. Bressan, A. *Bifurcation analysis of a non-cooperative differential game with one weak player*, J. Differ. Equ., 248(6), (2010), 1297–1314.

6. Bressan, A. *Noncooperative differential games*, Milan J. Math., 79(2) (2011), 357–427.

7. Canuto, C., Hussaini, M. Y., Quarteroni, A., Thomas Jr, A., et al. *Spectral methods in fluid dynamics*, Springer Science & Business Media, Berlin/Heidelberg, Germany, (2012).

8. Canuto, C., Hussaini, M. Y., Quarteroni, A., and Zang, T. A. *Spectral methods: fundamentals in single domains*, Springer Science & Business Media, Berlin/Heidelberg, Germany, (2007).

9. Carlson, D. A. and Leitmann, G. *An extension of the coordinate transformation method for open-loop Nash equilibria*, J. Optim. Theory Appl., 123(1) (2004), 27–47.

10. Cesari, L. *Optimization-theory and applications: problems with ordinary differential equations*, Springer Science & Business Media, Berlin/Heidelberg, Germany, (2012).

11. Dockner, E. J., Jorgensen, S., Van Long, N., and Sorger, G. *Differential games in economics and management science*, Cambridge University Press, London, (2000).

12. Ehtamo, H. and Raivio, T. *On applied nonlinear and bilevel programming or pursuit-evasion games*, J. Optim. Theory Appl., 108(1) (2001), 65–96.

13. Engwerda, J. *LQ dynamic optimization and differential games*, John Wiley & Sons, NJ, USA, (2005).

14. Engwerda, J. C. *On the open-loop Nash equilibrium in lq-games*, J. Econ. Dyn. Control, 22(5) (1998), 729–762.

15. Erickson, G. *Dynamic models of advertising competition*, Springer Science & Business Media, Berlin/Heidelberg, Germany, (2002).

16. Ghaneai, H. and Hosseini, M. *Variational iteration method with an auxiliary parameter for solving wave-like and heat-like equations in large domains*, Comput. Math. Appl., 69(5) (2015), 363–373.

17. Grosset, L. *A note on open loop Nash equilibrium in linear-state differential games*, Appl. Math. Sci., 8 (2014), 7239–7248.

18. Heydari, M., Loghmani, G. B., Rashidi, M. M. and Hosseini, S. M. *A numerical study for off-centered stagnation flow towards a rotating disc*, Propuls. Power Res., 4(3) (2015), 169–178.

19. Heydari M., Loghmani G. B. and Hosseini S. M. *Exponential Bernstein functions: an effective tool for the solution of heat transfer of a micropolar fluid through a porous medium with radiation*, Comput. Appl. Math., 36(1) (2017), 647–675.

20. Horie, K. and Conway, B. A. *Genetic algorithm preprocessing for numerical solution of differential games problems*, J. Guid. Control Dyn., 27(6), (2004), 1075–1078.

21. Hosseini, E., Barid Loghmani, G., Heydari, M., and Wazwaz, A.-M. *A numerical study of electrohydrodynamic flow analysis in a circular cylindrical conduit using orthonormal Bernstein polynomials*, Comput. Methods Differ. Equ., 5(4) (2017), 280–300.

22. Hosseini, E., Loghmani, G. B., Heydari, M. and Rashidi, M. M. *Investigation of magneto-hemodynamic flow in a semi-porous channel using orthonormal Bernstein polynomials*, Eur. Phys. J. Plus, 132(7) (2017), 1–16.

23. Jafari, S. and Navidi, H. *A game-theoretic approach for modeling competitive diffusion over social networks*, Games, 9(1) (2018), 8.

24. Jiménez-Lizárraga, M., Basin, M., Rodríguez, V., and Rodríguez, P. *Open-loop Nash equilibrium in polynomial differential games via state-dependent Riccati equation*, Automatica J. IFAC, 53 (2015), 155–163.

25. Johnson, P. A. *Numerical solution methods for differential game problems*, PhD thesis, Massachusetts Institute of Technology, (2009).

26. Keshavarz, E., Ordokhani, Y., and Razzaghi, M. *A numerical solution for fractional optimal control problems via Bernoulli polynomials*, J. Vib. Control, 22(18) (2016), 3889–3903.

27. Nikooeinejad, Z., Delavarkhalafi, A., and Heydari, M. *A numerical solution of open-loop Nash equilibrium in nonlinear differential games based on Chebyshev pseudospectral method*, J. Comput. Appl. Math., 300 (2016), 369–384.

28. Nikooeinejad, Z., Delavarkhalafi, A., and Heydari, M. *Application of shifted Jacobi pseudospectral method for solving (in) finite-horizon min-max optimal control problems with uncertainty*, Int. J. Control, 91(3) (2018), 725–739.

29. Nikooeinejad, Z., Heydari, M., Saffarzadeh, M., Loghmani, G. B. and Engwerda, J. *Numerical Simulation of Non-cooperative and Cooperative Equilibrium Solutions for a Stochastic Government Debt Stabilization Game*, Comput. Econ., (2021), 1–27.

30. Nikooeinejad, Z., Delavarkhalafi, A., Heydari, M. and Wazwaz, A. M. *A computational method for solving the system of Hamilton-Jacobi-Bellman PDEs in nonzero-sum fixed-final-time differential games*, Trans. A. Razmadze Math. Inst., 175(1) (2021), 83–100.

31. Nikooeinejad, Z. and Heydari, M. *Nash equilibrium approximation of some class of stochastic differential games: A combined Chebyshev spectral collocation method with policy iteration*, J. Comput. Appl. Math., 362 (2019), 41–54.

32. Nikooeinejad, Z., Heydari M. and Loghmani G. B. *Numerical solution of two-point BVPs in infinite-horizon optimal control theory: A combined quasilinearization method with exponential Bernstein functions*, Int. J. Comput. Math., 98(11) (2021), 2156–2174.

33. Rabiei, K., Ordokhani, Y., and Babolian, E. *Numerical solution of 1d and 2d fractional optimal control of system via Bernoulli polynomials*, Int. J. Appl. Comput., 4(1) (2018), 1–17.

34. Shen, J., Tang, T., and Wang, L.-L. *Spectral methods: algorithms, analysis and applications*, Springer Science & Business Media, Berlin/Heidelberg, Germany, (2011).

35. Sorger, G. *Competitive dynamic advertising: A modification of the case game*, J. Econ. Dyn. Control, 13(1) (1989), 55–80.

36. Starr, A. W. and Ho, Y.-C. *Further properties of nonzero-sum differential games*, J. Optim. Theory Appl., 3(4) (1969a), 207–219.

37. Starr, A. W. and Ho, Y.-C. *Nonzero-sum differential games*, J. Optim. Theory Appl., 3(3) (1969b), 184–206.

38. Tameh, M. S. and Shivanian, E. *Fractional shifted Legendre tau method to solve linear and nonlinear variable-order fractional partial differential equations*, Math. Sci., 15(1) (2021), 11–19.

39. Tari, A., Rahimi, M., Shahmorad, S., and Talati, F. *Development of the tau method for the numerical solution of two-dimensional linear Volterra integro-differential equations*, Comput. Methods Appl. Math., 9(4) (2009), 421–435.

40. Tohidi, E., Bhrawy, A., and Erfani, K. *A collocation method based on Bernoulli operational matrix for numerical solution of generalized pantograph equation*, Appl. Math. Model., 37(6) (2013), 4283–4294.

41. Torabi, S. M., Tari, A., and Shahmorad, S. *Two-step collocation methods for two-dimensional Volterra integral equations of the second kind*, J. Appl. Anal., 25(1) (2019), 1–11.

42. Wu, C., Teo, K. L., and Wu, S. *Min-max optimal control of linear systems with uncertainty and terminal state constraints*, Automatica J. IFAC, 49(6) (2013), 1809–1815.

43. Yazdaniyan, Z., Shamsi, M., Foroozandeh, Z., and de Pinho, M. d. R. *A numerical method based on the complementarity and optimal control formulations for solving a family of zero-sum pursuit-evasion differential games*, J. Comput. Appl. Math., 368, (2020), 112535

44. Yeung, D. W. and Petrosjan, L. A. *Cooperative stochastic differential games*, Springer Science & Business Media, Berlin/Heidelberg, Germany, (2006).

How to cite this article
M. Dehghan Banadaki and H. Navidi A numerical treatment based on Bernoulli Tau method for computing the open-loop Nash equilibrium in nonlinear differential games. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 467-482. doi: 10.22067/IJNAO.2022.74347.1086.

Iranian Journal of Numerical Analysis and Optimization
Vol. 12, No. 2, pp 483–497
DOI: 10.22067/ijnao.2022.74627.1091
https://ijnao.um.ac.ir/

A fourth-order method for solving singularly perturbed boundary value problems using nonpolynomial splines

S. Khan* and A. Khan

Abstract

In this paper, a class of second-order singularly perturbed interior layer problems is examined. A nonpolynomial mixed spline is used to develop the tridiagonal scheme. The developed method is second as well as fourth-order accurate based on the parameters. Error analysis is also carried out. The method is shown to converge point-wise to the true solution with higher accuracy. Linear and nonlinear second-order singularly perturbed boundary value problems have been solved by the presented method. Five numerical illustrations are given to demonstrate the applicability of the proposed method. Absolute errors are given in tables, which show that our method is more efficient than previously existing methods.

Keywords: Singularly perturbed, Second-order problems, Nonpolynomial splines, Convergence

AMS subject classifications (2020): Primary 65L10; Secondary 65D07

1 Introduction

Second-order singularly perturbed boundary-value problems (SPBVPs) have gained more importance for two reasons. Firstly, they occur in many areas of

* Corresponding author

Received 8 January 2022; revised 3 April 2022; accepted 4 April 2022

S. Khan
Department of Mathematics, Sri Venkateswara College, University of Delhi, New Delhi, India. e-mail: khan.shahana5@gmail.com

A. Khan
Department of Mathematics, Jamia Millia Islamia, New Delhi, India. e-mail: akhan1234in@rediffmail.com

science and engineering, like combustion, control theory, nuclear engineering, fluid mechanics, chemical reactor theory, and so on. Secondly, the occurrence of sharp boundary-layers as ϵ, the coefficient of the highest order derivative, approaches zero, makes it difficult for standard numerical methods. The approximate solution to boundary-value problems (BVPs), in which the highest derivative is multiplied with a small positive parameter, is described here. It is a well-known fact that the solution to the SPBVP exhibits a multiscale character. There are a number of methods available in the literature for solving SPBVPs [3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 18]. Some other recent methods for solving nonlinear problems are given in [15, 16, 17].
We consider second-order SPBVPs of the form

$$\epsilon z^{(2)}(x) = G(x, z, z'), \qquad r \leq x \leq s, \tag{1}$$

with

$$z(r) = \eta_0, \qquad z(s) = \eta_n, \tag{2}$$

where $0 < \epsilon \ll 1$ is a parameter multiplied by the highest derivative, $G(x, z, z')$ are continuously differentiable functions on $[r, s]$, and η_0 and η_n are constants.
Consider the following assumptions, for $r < x < s$ and $-\infty < z < z' < \infty$:
(i) G is continuous,
(ii) $\frac{\partial G}{\partial z}$ and $\frac{\partial G}{\partial z'}$ exist and are continuous,
(iii) $\frac{\partial G}{\partial z} > 0$ and $|\ \frac{\partial G}{\partial z}\ | \leq D$, for an arbitrary constant D.

The above assumptions from [11] confirm the existence and uniqueness of the solution to the BVP (1).
Here, we develop a method using a nonpolynomial spline for solving the generalized form of second-order SPBVPs. We use three-point finite difference approximations, which gives in place of give the accuracy of order four. When we implement the method, a tridiagonal system is obtained and solved using MATLAB.
The paper is organized as follows: In Section 2, the derivation of the numerical method is presented. The obtained scheme is applied to the second-order SPBVPs in Section 3. Error analysis of the method is discussed in Section 4. Numerical results are given in Section 5 to prove the efficiency of the proposed scheme. Finally, concluding remarks are presented.

2 Nonpolynomial mixed spline

Let the interval $[r,s]$ is divided into n equal parts as $r = x_0 < x_1 < x_2 < \cdots < x_n = s$, by introducing

$$x_i = r + ih, \quad where\ h = (s - r)/n, \quad i = 0, 1, \ldots, n.$$

Let

$$R_i(x) = a_i e^{\delta(x-x_i)} + b_i[\sin(\delta(x-x_i)) + \cos(\delta(x-x_i))] + c_i, \tag{3}$$

be a function defined on [r,s], which becomes an ordinary quadratic spline as parameter $\delta \longrightarrow 0$ and $\delta > 0$.

To evaluate $a_i, b_i,$ *and* c_i, we let

$$R_i(x_i) = z_i, \quad R_i(x_{i+1}) = z_{i+1}, \quad R_i^{(2)}(x_i) = \frac{1}{2}(S_i + S_{i+1}), \quad 0 \le i \le n. \tag{4}$$

Using the above conditions, we get

$$\begin{aligned}
a_i &= \frac{z_{i+1} - z_i}{\zeta} - \frac{1}{2\theta^2}[\frac{e^\theta - 1}{\zeta} - 1]h^2(S_i + S_{i+1}),\\
b_i &= \frac{z_{i+1} - z_i}{\zeta} - \frac{1}{2\theta^2}[\frac{e^\theta - 1}{\zeta}]h^2(S_i + S_{i+1}),\\
c_i &= \frac{z_{i+1} + (2+\zeta)z_i}{\zeta} + \frac{1}{2\theta^2}[\frac{2e^\theta - 1}{\zeta} - 1]h^2(S_i + S_{i+1}),
\end{aligned}$$

where $\zeta = -2 + \sin\theta + \cos\theta + e^\theta$ and $\theta = \lambda h$.
Using the first derivative continuity condition, $R_{i-1}^k(x_i) = R_i^k(x_i),\ k = 0, 1,$ the following scheme is obtained:

$$\phi z_{i-1} + \xi z_i + \psi z_{i+1} = h^2(\phi_1 S_{i-1} + \xi_1 S_i + \psi_1 S_{i+1}), \qquad 1 \le i \le n-1, \tag{5}$$

where

$$\begin{aligned}
\phi &= \frac{\sin\theta + e^\theta + \cos\theta}{2},\\
\xi &= \frac{2 + \sin\theta - e^\theta - \cos\theta}{2},\\
\psi &= 1,\\
\phi_1 &= \frac{(2\sin\theta - 1)e^\theta + \cos\theta - \sin\theta}{4\theta^2},\\
\xi_1 &= \frac{\sin\theta e^\theta - \sin\theta}{2\theta^2},\\
\psi_1 &= \frac{e^\theta - \sin\theta - \cos\theta}{4\theta^2}.
\end{aligned}$$

Remark: Our scheme reduces to [2] when

$$(\phi, \xi, \psi, \phi_1, \xi_1, \psi_1) = (1, -2, 1, \frac{1}{4}, \frac{1}{2}, \frac{1}{4}).$$

Error

After expanding (5), using the Taylor series, we obtain the truncation error (TE) as follows:

$$\begin{aligned} t_i = (\phi+\xi+\psi)z_i + (-\phi+\psi)hz_i' + h^2[\frac{\phi+\psi}{2!} - (\phi_1+\xi_1+\psi_1)]z_i^{(2)} \\ +h^3[\frac{-\phi+\psi}{3!} - (-\phi_1+\psi_1)]z_i^{(3)} + h^4[\frac{\phi+\psi}{4!} - (\frac{\phi_1+\psi_1}{2!})]z_i^{(4)} \\ +h^5[\frac{-\phi+\psi}{5!} - (\frac{-\phi_1+\psi_1}{3!})]z_i^{(5)} + h^6[\frac{\phi+\psi}{6!} - (\frac{\phi_1+\psi_1}{4!})]z_i^{(6)} \\ +h^7[\frac{-\phi+\psi}{7!} - (\frac{-\phi_1+\psi_1}{5!})]z_i^{(7)} + O(h^8), \qquad 1 \le i \le n-1. \end{aligned} \tag{6}$$

For various values of coefficients obtained in (5), we obtain the second-order method as well as the fourth-order method. For $\phi_1+\xi_1+\psi_1 = 1$ and $\phi_1 = \psi_1$, we obtain the second-order method. The TE for $(\phi, \xi, \psi, \phi_1, \xi_1, \psi_1) = (1, -2, 1, 1/6, 4/6, 1/6)$ is given as follows:

$$t_i = -\frac{1}{12}h^4 z_i^{(4)} + O(h^5), \quad 1 \le i \le n-1.$$

For $(\phi, \xi, \psi, \phi_1, \xi_1, \psi_1) = (1, -2, 1, 1/12, 10/12, 1/12)$, we obtain a higher order method, that is, fourth-order. The TE is as follows:

$$t_i = -\frac{1}{120}h^6 z_i^{(6)} + O(h^7), \quad 1 \le i \le n-1.$$

3 Implementation of the scheme

We consider the second-order SPBVP of the generalized form (1) as

$$\epsilon z^{(2)}(x) = G(x, z, z'), \qquad r \le x \le s.$$

In particular, we take the linear second-order SPBVP as follows:

$$\epsilon z^{(2)}(x) = -p(x)z'(x) - q(x)z(x) + g(x), \qquad r \le x \le s. \tag{7}$$

Now, implementing the scheme (5) on problem (7), we get the method, which is of second-order accuracy as follows:

$$\epsilon\phi z_{i-1} + \epsilon\xi z_i + \epsilon\psi z_{i+1} = h^2(\phi_1 G_{i-1} + \xi_1 G_i + \psi_1 G_{i+1}), \quad 1 \le i \le n-1, \tag{8}$$

where $\phi_1 = \psi_1$ and $\phi_1 + \xi_1 + \psi_1 = 1$. The method of fourth-order accuracy is given as follows:

$$\epsilon\phi z_{i-1} + \epsilon\xi z_i + \epsilon\psi z_{i+1} = h^2(\phi_1 G_{i-1} + \xi_1 \tilde{G}_i + \psi_1 G_{i+1}), \quad 1 \le i \le n-1, \tag{9}$$

where $(\phi_1, \xi_1, \psi_1) = (1/12, 10/12, 1/12)$ and

$$\begin{aligned} G_i &= G(x, z_i, z_i'), \\ G_{i-1} &= G(x, z_{i-1}, z_{i-1}'), \\ G_{i+1} &= G(x, z_{i+1}, z_{i+1}'), \\ \tilde{G}_i &= G(x, z_i, \tilde{z}_i'), \end{aligned}$$

in which

$$\begin{aligned} z_i' &= \frac{z_{i+1} - z_{i-1}}{2h}, \\ z_{i-1}' &= \frac{-3z_{i-1} + 4z_i - z_{i+1}}{2h}, \\ z_{i+1}' &= \frac{z_{i-1} - 4z_i + 3z_{i+1}}{2h}, \\ \tilde{z}_i' &= \frac{z_{i+1} - z_{i-1}}{2h} - \frac{h}{20}(G_{i+1} - G_{i-1}). \end{aligned}$$

Using the above approximations, we obtain an expression for the second-order method as follows:

$$\begin{aligned} &(\phi\epsilon - \frac{3}{2}h\phi_1 p_{i-1} - \frac{1}{2}h\xi_1 p_i + \frac{1}{2}h\psi_1 p_{i+1} + h^2\phi_1 q_{i-1})z_{i-1} \\ &+ (\epsilon\xi + 2h\phi_1 p_{i-1} - 2h\psi_1 p_{i+1} + h^2\xi_1 q_i)z_i \\ &+ (\epsilon\psi - \frac{1}{2}h\phi_1 p_{i-1} + \frac{1}{2}h\xi_1 p_i + \frac{3}{2}h\psi_1 p_{i+1} + h^2\psi_1 q_{i+1})z_{i+1} \\ &\quad = h^2(\phi_1 g_{i-1} + \xi_1 g_i + \psi_1 g_{i+1}), \qquad 1 \le i \le n-1. \end{aligned} \tag{10}$$

Similarly for the fourth-order method, it reads as follows:

$$\begin{aligned} &\big[\epsilon\phi - \frac{3}{2}h(\phi_1 - \frac{\xi_1 h}{20}p_i)p_{i-1} - \frac{1}{2}h\xi_1 p_i + \frac{1}{2}h(\psi_1 + \frac{\xi_1 h}{20}p_i)p_{i+1} \\ &\quad + h^2(\phi_1 - \frac{\xi_1 h}{20}p_i)q_{i-1}\big]z_{i-1} \\ &+ \big[\epsilon\xi + 2h\phi_1 p_{i-1} - 2h(\psi_1 + \frac{\xi_1 h}{20}p_i)p_{i+1} + h^2\xi_1 q_i\big]z_i \\ &+ \big[\epsilon\psi - \frac{1}{2}h(\phi_1 - \frac{\xi_1 h}{20}p_i)p_{i-1} + \frac{1}{2}h\xi_1 p_i + \frac{3}{2}h(\psi_1 + \frac{\xi_1 h}{20}p_i)p_{i+1} \\ &\quad + h^2(\psi_1 + \frac{\xi_1 h}{20}p_i)q_{i+1}\big]z_{i+1} \\ &\quad = h^2\big[(\phi_1 - \frac{\xi_1 h}{20}p_i)g_{i-1} + \xi_1 g_i + (\psi_1 + \frac{\xi_1 h}{20}p_i)g_{i+1}\big], \qquad 1 \le i \le n-1. \end{aligned} \tag{11}$$

4 Convergence

Here, we discuss the error analysis of higher order method, that is, fourth-order. In the matrix form, (11) can be written as

$$MZ = U, \tag{12}$$

where $M = (m_{i,j})$ is the tridiagonal matrix whose entries are given by (11) and $U = [u_1, u_2, \ldots, u_{n-1}]^T$ is given as

$$\begin{aligned}
u_1 =& h^2[(\phi_1 - \frac{h\xi_1}{20}p_1)g_0 + \xi_1 g_1 + (\psi_1 + \frac{h\xi_1}{20}p_1)g_2] \\
&- (\epsilon\phi - \frac{3}{2}h(\phi_1 - \frac{h\xi_1}{20}p_1)p_0 - \frac{1}{2}h\xi_1 p_1 + \frac{1}{2}h(\psi_1 + \frac{h\xi_1}{20}p_1)p_2 \\
&+ h^2(\phi_1 - \frac{h\xi_1}{20}p_1)q_0)y_0, \\
u_i =& h^2[(\phi_1 - \frac{h\xi_1}{20}p_i)g_{i-1} + \xi_1 g_i + (\psi_1 + \frac{h\xi_1}{20}p_i)g_{i+1}], \qquad 2 \le i \le n-2, \\
u_{n-1} =& h^2[(\phi_1 - \frac{h\xi_1}{20}p_{n-1})g_{n-2} + \xi_1 g_{n-1} + (\psi_1 + \frac{h\xi_1}{20}p_{n-1})g_n] \\
&- (\epsilon\psi - \frac{1}{2}h(\phi_1 - \frac{h\xi_1}{20}p_{n-1})p_{n-2} + \frac{1}{2}h\xi_1 p_{n-1} + \frac{3}{2}h(\psi_1 + \frac{\xi_h}{20}p_{n-1})p_n \\
&+ h^2(\psi_1 + \frac{h\xi_1}{20}p_{n-1})q_n)y_n,
\end{aligned}$$

and $Z = [z_1, z_2, \ldots, z_{n-1}]^T$. We also have

$$M\tilde{Z} = U + T, \tag{13}$$

where $\tilde{Z} = [\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_{n-1}]^T$ and $T = [t_1, t_2, \ldots, t_{n-1}]^T$.
Also $t_i = [\tilde{z}_i - z_i]^T, i = 1, 2, \ldots, n-1$, is the local TE. From (12) and (13), we have

$$\begin{gathered}
M(\tilde{Z} - Z) = T, \\
ME = T, \\
E = \tilde{Z} - Z.
\end{gathered}$$

Now we calculate the sum of elements of each row of the matrix M as

$$Sum_i = \begin{cases} \epsilon\xi + \epsilon\psi + \frac{3}{2}h(\phi_1 - \frac{h\xi_1}{20}p_1)p_0 + \frac{1}{2}h\xi_1 p_1 \\ -\frac{1}{2}h(\psi_1 + \frac{h\xi_1}{20}p_1)p_2 + h^2\xi_1 q_1 + h^2(\psi_1 + \frac{h\xi_1}{20}p_1)q_2, \qquad i{=}1, \end{cases}$$

$$Sum_i = \begin{cases} \epsilon\phi + \epsilon\xi + \epsilon\psi \\ +h^2((\phi_1 - \frac{h\xi_1}{20}p_i)q_{i-1} + \xi_1 q_i + (\psi_1 + \frac{h\xi_1}{20}p_i)q_{i+1}), \qquad 2 \le i \le n-2, \end{cases}$$

$$Sum_i = \begin{cases} \phi\epsilon + \epsilon\xi + \frac{1}{2}h(\phi_1 - \frac{h\xi_1}{20}p_{n-1})p_{n-2} - \frac{1}{2}h\xi_1 p_{n-1} \\ -\frac{3}{2}h(\psi_1 + \frac{h\xi_1}{20}p_{n-1})p_n + h^2(\phi_1 - \frac{h\xi_1}{20}p_{n-1})q_{n-2} + h^2\xi_1 q_{n-1}, \quad i{=}n\text{-}1. \end{cases}$$

Furthermore,

$$\begin{aligned} &Sum_i \geq h^2(\xi_1 q_1 + \psi_1 q_2 - \frac{3}{40}\xi_1 p_0 p_1 - \frac{1}{40}\xi_1 p_1 p_2) + O(h^3), \quad i{=}1, \\ &Sum_i \geq h^2(\phi_1 q_{i-1} + \xi_1 q_i + \psi_1 q_{i+1}), \quad 2 \leq i \leq n-2, \\ &Sum_i \geq h^2(\phi_1 q_{n-2} + \xi_1 q_{n-1} - \frac{3}{40}\xi_1 p_n p_{n-1} - \frac{1}{40}\xi_1 p_{n-1} p_{n-2}) + O(h^3), \quad i{=}n\text{-}1. \end{aligned}$$

Let $0 < K \in Z^+$ be the minimum of $\mid p_i \mid$ and $\mid q_i \mid$. For sufficiently small h, we can say that

$$Sum_1 \geq h^2[(\xi_1 + \psi_1 - \frac{1}{10}\xi_1)K],$$

$$Sum_i \geq h^2[(\phi_1 + \xi_1 + \psi_1)K], \qquad 2 \leq i \leq n-2,$$

$$Sum_{n-1} \geq h^2[(\phi_1 + \xi_1 - \frac{1}{10}\xi_1)K].$$

Therefore, we get

$$\begin{aligned} &\frac{1}{Sum_1} \leq \frac{1}{h^2[(\frac{9}{10}\xi_1 + \psi_1)K]}, \\ &\frac{1}{Sum_i} \leq \frac{1}{h^2[(\phi_1 + \xi_1 + \psi_1)K]}, \qquad 2 \leq i \leq n-2, \\ &\frac{1}{Sum_{n-1}} \leq \frac{1}{h^2[(\phi_1 + \frac{9}{10}\xi_1)K]}. \end{aligned}$$

Furthermore,

$$\frac{1}{Sum_i} \leq \begin{cases} \frac{1}{h^2[(\frac{9}{10}\xi_1 + \psi_1)K]}, & i{=}1 \\ \frac{1}{h^2[(\phi_1 + \xi_1 + \psi_1)K]}, & 2 \leq i \leq n-2 \\ \frac{1}{h^2[(\phi_1 + \frac{9}{10}\xi_1)K]}, & i{=}n\text{-}1. \end{cases}$$

The matrix M is monotone, irreducible and diagonally dominant for suiciently small h. Therefore, M^{-1} exist and is positive. Hence,

$$\|E\| = \|M^{-1}\|\|T\|.$$

Let $M^{-1} = (m^*_{i,j})$. Then by [19], we get

$$\sum_{i=1}^{n-1} m_{i,j}^* Sum_i = 1, \qquad 1 \le j \le n-1.$$

Therefore,

$$\begin{aligned} m_{i,j}^* &\le \frac{1}{Sum_i}, \\ \|M^{-1}\| &= \max_{1\le i\le n-1} \sum_{j=1}^{n-1} |m_{i,j}^*| \le \sum_{i=1}^{n-1} \frac{1}{Sum_i} \\ &= [\frac{1}{h^2(\frac{9}{10}\xi_1+\psi_1)K} + \frac{1}{h^2(\phi_1+\xi_1+\psi_1)K} + \frac{1}{h^2(\phi_1+\frac{9}{10}\xi_1)K}], \\ \|T_i\| &= \max_{1\le i\le n-1} \sum_{i=1}^{n-1} |T_i|, \qquad 1 \le i \le n-1. \end{aligned}$$

Finally,

$$\|E\| = \|M^{-1}\|\|T\| \le \frac{1}{h^2K}[\frac{1}{\frac{9}{10}\xi_1+\psi_1} + \frac{1}{\phi_1+\xi_1+\psi_1} + \frac{1}{\phi_1+\frac{9}{10}\xi_1}]\|T\|.$$

As the fourth-order method, $\|T\| = O(h^6)$ using (6). Then

$$\|E\| \le \frac{1}{h^2K}[\frac{1}{\frac{9}{10}\xi_1+\psi_1} + \frac{1}{\phi_1+\xi_1+\psi_1} + \frac{1}{\phi_1+\frac{9}{10}\xi_1}]O(h^6) = O(h^4).$$

Hence, the error is of order four. For the second-order method, $\|T\| = O(h^4)$ using (6). Then $\|E\| = O(h^2)$. We can prove the convergence of the second-order method by following the above procedure.

5 Numerical illustrations and discussion

To check the applicability of the developed method to existing problems, we solve three linear and one nonlinear problem of the type (1). The maximum absolute errors (MAEs) are tabulated in Tables 1–8. We have also solved the heat flow problem in Example 5.

Example 1. Consider the nonlinear SPBVP from [18] as

$$\epsilon z^{(2)}(x) + 2z'(x) + e^{z(x)} = 0, \qquad 0 \le x \le 1, \tag{14}$$

with

$$z(0) = z(1) = 0.$$

The analytical solution is

$$z(x) = \ln\Big(2/(1+x)\Big) - e^{-2x/\epsilon}\ln 2.$$

The results of our method along with comparison are given in Table 1. Also we have tabulated the MAE in Table 2.

Table 1: Comparison of approximate solution of Example 1 with the method of [18] for $\epsilon = 10^{-3}$ and $n = 1000$

x	Presented method	[18]	Analytical solution
0.001	0.5933418	0.6913641	0.5983404
0.010	0.6835355	0.6825219	0.6831968
0.020	0.6736751	0.6726859	0.6733446
0.030	0.6639109	0.6629456	0.6635884
0.040	0.6542413	0.6532992	0.6539264
0.050	0.6446647	0.6437448	0.6443570
0.100	0.5981091	0.5972949	0.5978370
0.300	0.4309488	0.4304523	0.4307829
0.500	0.2877780	0.2874905	0.2876821
0.700	0.1625668	0.1624234	0.1625189
0.900	0.0513068	0.0512663	0.0512933
1.000	0.0000000	0.0000000	0.0000000

Table 2: MAE of Example 1 for $(\phi_1, \xi_1, \psi_1) = (1/12, 10/12, 1/12)$

$n \setminus \epsilon$	2^{-2}	2^{-4}	2^{-6}	2^{-8}
16	3.010×10^{-2}	5.450×10^{-2}	3.950×10^{-2}	3.208×10^{-1}
32	2.900×10^{-2}	2.010×10^{-2}	8.300×10^{-2}	1.275×10^{-1}
64	2.870×10^{-2}	1.600×10^{-2}	4.430×10^{-2}	3.280×10^{-2}
128	2.860×10^{-2}	1.550×10^{-2}	1.310×10^{-2}	7.760×10^{-2}
256	2.860×10^{-2}	1.540×10^{-2}	5.300×10^{-2}	4.150×10^{-2}

Example 2. Consider the linear SPBVP from [14, 18] as

$$\epsilon z^{(2)}(x) + z'(x) = 1 + 2x, \qquad 0 \le x \le 1, \tag{15}$$

with

$$z(0) = 0, \qquad z(1) = 1.$$

The analytical solution is

$$z(x) = x(x + 1 - 2\epsilon) + \frac{(2\epsilon - 1)(1 - e^{-x/\epsilon})}{(1 - e^{-1/\epsilon})}.$$

The results of our method have been compared with the methods of [14, 18] in Table 3. We have also tabulated the MAE in Table 4.

Table 3: Comparison of the approximate solution of Example 2 with the methods of [14, 18] for $\epsilon = 10^{-3}$ and $n = 1000$

x	[18]	[14]	Presented method	Analytical solution
0.001	-1.0009970	-0.6311195	-0.6293169	-0.6298573
0.010	-0.9918800	-0.9898546	-0.9878740	-0.9878747
0.020	-0.9815600	-0.9796000	-0.9776400	-0.9776400
0.030	-0.9710400	-0.9691000	-0.9671600	-0.9671600
0.040	-0.9603199	-0.9584000	-0.9564800	-0.9564800
0.050	-0.9493999	-0.9475000	-0.9456000	-0.9456000
0.100	-0.8918000	-0.8900000	-0.8882000	-0.8882000
0.300	-0.6114000	-0.6100000	-0.6086000	-0.6086000
0.500	-0.2510000	-0.2500000	-0.2490000	-0.2490000
0.700	0.1894000	0.1900000	0.1906000	0.1906000
0.900	0.7098000	0.7099999	0.7102000	0.7102000
1.000	1.0000000	1.0000000	1.0000000	1.0000000

Example 3. Consider the two-parameter SPBVP from [20] as

$$-\epsilon_d z^{(2)}(x) + \epsilon_c z'(x) + z(x) = \cos(\pi x), \qquad 0 \le x \le 1, \tag{16}$$

with

$$z(0) = 0, \qquad z(1) = 0.$$

The analytical solution is

$$z(x) = A\cos(\pi x) + B\sin(\pi x) + Ce^{\lambda_1 x} + De^{-\lambda_2(1-x)},$$

where

$$A = \frac{\epsilon_d \pi^2 + 1}{\epsilon_c^2 \pi^2 + (\epsilon_d \pi^2 + 1)^2}, \qquad B = \frac{\epsilon_c \pi}{\epsilon_c^2 \pi^2 + (\epsilon_d \pi^2 + 1)^2},$$

Table 4: MAE of Example 2 for $(\phi_1, \xi_1, \psi_1) = (1/12, 10/12, 1/12)$

$n \setminus \epsilon$	2^{-2}	2^{-4}	2^{-6}	2^{-8}
16	8.8950×10^{-7}	4.7391×10^{-4}	5.6800×10^{-2}	4.6880×10^{-1}
32	5.5531×10^{-8}	2.8359×10^{-5}	7.3000×10^{-3}	2.2370×10^{-1}
64	3.4736×10^{-9}	1.7529×10^{-6}	5.2469×10^{-4}	5.8100×10^{-2}
128	2.1708×10^{-10}	1.0925×10^{-7}	3.1398×10^{-5}	7.5000×10^{-3}
256	1.3640×10^{-11}	6.8234×10^{-6}	1.9407×10^{-6}	5.3738×10^{-4}

Table 5: Comparison of the approximate solution of Example 3 with the method of [20] for $n = 128$

ϵ_c	[20]	Presented method	[20]	Presented method
	$\epsilon_d = 10^{-2}$	$\epsilon_d = 10^{-2}$	$\epsilon_d = 10^{-4}$	$\epsilon_d = 10^{-4}$
10^{-3}	9.2993×10^{-7}	2.6109×10^{-8}	1.3294×10^{-3}	2.7811×10^{-4}
10^{-4}	1.1557×10^{-7}	2.5996×10^{-8}	3.6708×10^{-4}	2.7239×10^{-4}
10^{-5}	3.4933×10^{-8}	2.5984×10^{-8}	2.8085×10^{-4}	2.7148×10^{-4}
10^{-6}	2.6878×10^{-8}	2.5983×10^{-8}	2.7232×10^{-4}	2.7138×10^{-4}
10^{-7}	2.6072×10^{-8}	2.5982×10^{-8}	2.7147×10^{-4}	2.7139×10^{-4}

$$C = -A\frac{1+e^{-\lambda_2}}{1-e^{\lambda_1-\lambda_2}}, \qquad D = A\frac{1+e^{\lambda_1}}{1-e^{\lambda_1-\lambda_2}},$$
$$\lambda_1 = \frac{\epsilon_c - \sqrt{\epsilon_c^2+4\epsilon_d}}{2\epsilon_d}, \qquad \lambda_2 = \frac{\epsilon_c + \sqrt{\epsilon_c^2+4\epsilon_d}}{2\epsilon_d}.$$

The results of our method along with comparison are given in Table 5.

Example 4. Consider the SPBVP from [14, 18] as

$$\epsilon z^{(2)}(x) + z'(x) - z(x) = 0, \qquad 0 \le x \le 1, \tag{17}$$

with

$$z(0) = 1, \qquad z(1) = 1.$$

The analytical solution is

$$z(x) = \frac{(e^{m_4}-1)e^{m_3 x} + (1-e^{m_3})e^{m_4 x}}{e^{m_4}-e^{m_3}},$$

where

$$m_3 = \frac{-1+\sqrt{1+4\epsilon}}{2\epsilon}, \quad m_4 = \frac{-1-\sqrt{1+4\epsilon}}{2\epsilon}.$$

The results of our method have been compared with methods [14, 18] in Table 6. We have also tabulated the MAE in Table 7.

Example 5. Consider the heat flow problem of viscous incompressible fluid over a stretching plate from [1] as

$$z^{(2)}(x) + P_r[1-e^{-x}]z'(x) + \epsilon[z(x)z^{(2)}(x) + [z'(x)]^2] = 0, \quad 0 \le x \le 1, \tag{18}$$

with

$$z(0) = 1, \qquad z(1) = 0,$$

Table 6: Comparison of approximate solution of Example 4 with the methods of [14, 18] for $\epsilon = 10^{-3}$ and $n = 1000$

x	[18]	[14]	Presented method	Analytical solution
0.001	0.6003270	0.6007916	0.6011354	0.6007918
0.010	0.3712379	0.3716054	0.3719728	0.3719724
0.020	0.3749439	0.3753111	0.3756784	0.3756784
0.030	0.3787160	0.3790830	0.3794502	0.3794502
0.040	0.3825260	0.3828929	0.3832599	0.3832599
0.050	0.3863742	0.3867410	0.3871079	0.3871079
0.100	0.4062043	0.4065697	0.4069350	0.4069350
0.300	0.4962382	0.4965853	0.4969323	0.4969323
0.500	0.6062278	0.6065307	0.6068334	0.6068334
0.700	0.7405963	0.7408182	0.7410400	0.7410400
0.900	0.9047471	0.9048374	0.9049277	0.9049277
1.000	1.0000000	1.0000000	1.0000000	1.0000000

Table 7: MAE of Example 4 for $(\phi_1, \xi_1, \psi_1) = (1/12, 10/12, 1/12)$

$n \setminus \epsilon$	2^{-2}	2^{-4}	2^{-6}	2^{-8}
16	1.9745×10^{-6}	4.1185×10^{-4}	3.8100×10^{-2}	2.9880×10^{-1}
32	1.2290×10^{-7}	2.4523×10^{-5}	5.0000×10^{-3}	1.4300×10^{-1}
64	7.6864×10^{-9}	1.5138×10^{-6}	3.6087×10^{-4}	3.7300×10^{-2}
128	4.8037×10^{-10}	9.4320×10^{-8}	2.1567×10^{-5}	4.8000×10^{-3}
256	3.0033×10^{-11}	5.9003×10^{-9}	1.3326×10^{-6}	3.4706×10^{-4}

where P_r is the Prandtl number, which can take different values. Problem (18) in [1] is perturbed by the small parameter ϵ with the first part consisting of the terms independent of ϵ and the second part involving ϵ. Here, we take the first part of (18) as

$$z^{(2)}(x) + P_r[1 - e^{-x}]z'(x) = 0, \qquad 0 \le x \le 1, \tag{19}$$

with

$$z(0) = 1, \qquad z(1) = 0.$$

The MAE of (19) is calculated using the double mesh principle due to the unavailability of an exact solution. The MAEs are given in Table 8 for P_r=1.5.

Table 8: MAE of Example 5

n	16	32	64	128	256	512	1024
Second-order method for $(\phi_1, \xi_1, \psi_1) = (1/6, 4/6, 1/6)$	2.1451×10^{-5}	2.7598×10^{-6}	3.4967×10^{-7}	4.3996×10^{-8}	5.5172×10^{-9}	6.9075×10^{-10}	8.6421×10^{-11}
Fourth-order method for $(\phi_1, \xi_1, \psi_1) = (1/12, 10/12, 1/12)$	7.1156×10^{-6}	9.5771×10^{-7}	1.2409×10^{-7}	1.5788×10^{-8}	1.9909×10^{-9}	2.4996×10^{-10}	3.1305×10^{-11}

6 Conclusion

In this paper, we solved second-order SPBVPs using a tridiagonal scheme obtained by a nonpolynomial spline. The presented method is based on a lower degree spline that could reduce computational time and is more feasible working. The error obtained in the developed scheme (5) is of order two as well as four depending upon the choice of parameters. Point-wise convergence and MAE are given in Tables 1–8. Here, linear, nonlinear, and two-parameter perturbed problems are being solved. Comparison with the result of [14, 18, 20] was given in Tables 1,3,5, and 6, which proves the efficiency of the method and also shows that our method is better than these existing methods.

References

1. Ahmad, N., Siddiqui, Z.U. and Mishra, M.K., *Boundary layer flow and heat transfer past a stretching plate with variable thermal conductivity,* Int. J. Non-Linear Mech. 45 (2010) 306–309.

2. Al-Said, E.A., *Smooth spline solution for a system of second order boundary value problems,* J. Nat. Geometry 16 (1999) 19–28.

3. Ascher, U.M., Matheij, R.M.M. and Russell, R.D., *Numerical solution of boundary value problems for ordinary differential Equations,* (Englewood Cliffs, NJ: Prentice-Hall) 1988.

4. Ayalew, M., Kiltu, G.G. and Duressa, G.F., *Fitted numerical scheme for second-order singularly perturbed differential-difference equations with mixed shifts,* Abstr. Appl. Anal. Volume 2021, Article ID 4573847, 11 pages.

5. Aziz, T. and Khan, A., *A spline method for second-order singularly perturbed boundary-value problems,* J. Comput. Appl. Math. 147 (2002) 445–452.

6. Chekole, A.T., Duressa, G.F. and Kiltu, G.G., *Non-polynomial septic spline method for singularly perturbed two point boundary value problems of order three,* J. Taibah Univ. Sci. 13 (2019) 651–660.

7. El-Zahar, E.R., *Approximate analytical solution of singularly perturbed boundary value problems in MAPLE,* AIMS Mathematics, 5 (2020) 2272–2284.

8. Gupta, R., Bhagyamma, D. and SharathBabu, K., *Numerical solution of singularly perturbed two-point boundary value problem using transformation technique using quadrature method,*Turk. J. Comput. Math. Educ. 12 (2021) 2084–2091.

9. Kadalbajoo, M.K. and Bawa, R.K., *Variable mesh difference scheme for singularly-perturbed boundary value problems using splines,* J. Optim. Theory Appl. 90 (1996) 405–416.

10. Kadalbajoo, M.K. and Reddy, Y.N., *Asymptotic and numerical analysis of singular perturbation problems: A survey,* Appl. Math. Comput. 30 (1989) 223–259.

11. Keller, H.B., *Numerical methods for two point boundary value problems*, Blaisdell Publications Co., New York, 1968.

12. Khan, A., Khan I., Aziz, T. and Stojanovic, M., *A Variable-Mesh Approximation Method for Singularly Perturbed Boundary-Value Problems using Cubic Spline in Tension,* Int. J. Comput. Math. 81 (2004) 1513–1518.

13. Kiltu, G.G., Duressa, G.F. and Bullo, T.A., *Computational method for singularly perturbed delay differential equations of the reaction-diffusion type with negative shift,* 'J. Ocean Eng. Sci. 6 (2021) 285–291.

14. Kumar, M., Mishra, H.K. and Singh, P., *A boundary value approach for a class of linear singularly perturbed boundary value problems,* Adv. Eng. Softw. 10(2009) 298–304.

15. Nikan, O., Avazzadeh, Z. and Rasoulizadeh, M.N., *Soliton solutions of the nonlinear sine-Gordon model with Neumann boundary conditions arising in crystal dislocation theory,* Nonlinear Dyn. 106 (2021) 783–813.

16. Rasoulizadeh, M.N., Ebadi, M.J., Avazzadeh, Z. and Nikan, O., *An efficient local meshless method for the equal width equation in fluid mechanics,* Eng. Anal. Bound. Elem. 131 (2021) 258–268.

17. Rasoulizadeh, M.N., Nikan, O. and Avazzadeh, Z., *The impact of LRBF-FD on the solutions of the nonlinear regularized long wave equation,* Math. Sci. 15 (2021) 365–376.

18. Reddy, Y.N. and Chakravarthy, P.P., *An initial-value approach for solving singularly perturbed two-point boundary value problems,* Appl. Math. Comput. 155 (2004) 95–110.

19. Varga, R.S., *Matrix iterative analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.

20. Zahra, W.K. and Mhlawy, A.M.El. *Numerical solution of two-parameter singularly perturbed boundary value problems via exponential spline,* J. King Saud Univ. Sci. 25 (2013) 201–208.

How to cite this article
S. Khan and A. Khan A fourth-order method for solving singularly perturbed boundary value problems using nonpolynomial splines. *Iranian Journal of Numerical Analysis and Optimization*, 2022; 12(2): 483-497. doi: 10.22067/ijnao.2022.74627.1091.

Aims and scope

Iranian Journal of Numerical Analysis and Optimization (IJNAO) is published twice a year by the Department of Applied Mathematics, Faculty of Mathematical Sciences, Ferdowsi University of Mashhad. Papers dealing with different aspects of numerical analysis and optimization, theories and their applications in engineering and industry are considered for publication.

Journal Policy

All submissions to IJNAO are first evaluated by the journal's Editor-in-Chief or one of the journal's Associate Editors for their appropriateness to the scope and objectives of IJNAO. If deemed appropriate, the paper is sent out for review using a single blind process. Manuscripts are reviewed simultaneously by reviewers who are experts in their respective fields. The first review of every manuscript is performed by at least two anonymous referees. Upon the receipt of the referee's reports, the paper is accepted, rejected, or sent back to the author(s) for revision. Revised papers are assigned to an Associate Editor who makes an evaluation of the acceptability of the revision. Based upon the Associate Editor's evaluation, the paper is accepted, rejected, or returned to the author(s) for another revision. The second revision is then evaluated by the Editor-in-Chief, possibly in consultation with the Associate Editor who handled the original paper and the first revision, for a usually final resolution.

The authors can track their submissions and the process of peer review via: http://ijnao.um.ac.ir

All manuscripts submitted to IJNAO are tracked by using "iThenticate" for possible plagiarism before acceptance.

Instruction for Authors

The Journal publishes all papers in the fields of numerical analysis and optimization. Articles must be written in English.
All submitted papers will be refereed and the authors may be asked to revise their manuscripts according to the referee's reports. The Editorial Board of the Journal keeps the right to accept or reject the papers for publication.
The papers with more than one authors, should determine the corresponding author. The e-mail address of the corresponding author must appear at the end of the manuscript or as a footnote of the first page.
It is strongly recommended to set up the manuscript by Latex or Tex, using the template provided in the web site of the Journal. Manuscripts should be typed double-spaced with wide margins to provide enough room for editorial remarks.

References should be arranged in alphabetical order by the surname of the first author as examples below:

[1] Stoer, J. and Bulirsch, R. *Introduction to Numerical Analysis*, Springer-verlag, New York, 2002.

[2] Brunner, H. *A survey of recent advances in the numerical treatment of Volterra integral and integro-differential equations*, J. Comput. Appl. Math. 8 (1982), 213-229.

Iranian Journal of Numerical Analysis and Optimization

Former : MRJMS

CONTENTS Vol. 12, No.2, pp 250-499, 2022

The rest of contents is on back of the page

web site : http://ijnao.um.ac.ir

Email : ijnao@um.ac.ir

ISSN-Print : 2423-6977

ISSN-Online : 2423-6969