mathematics

Special Issue Reprint

Advances in Computational Mathematics and Applied Mathematics

Edited by
Tao Liu, Fazlollah Soleymani and Qiang Ma

mdpi.com/journal/mathematics

MDPI

Advances in Computational Mathematics and Applied Mathematics

Advances in Computational Mathematics and Applied Mathematics

Guest Editors

Tao Liu
Fazlollah Soleymani
Qiang Ma

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Guest Editors

Tao Liu
College of Sciences
Northeastern University
Shenyang
China

Fazlollah Soleymani
Department of Mathematics
Institute for Advanced
Studies in Basic Sciences
(IASBS)
Zanjan
Iran

Qiang Ma
Department of Mathematics
Harbin Institute of
Technology
Harbin
China

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Mathematics* (ISSN 2227-7390), freely accessible at: https://www.mdpi.com/journal/mathematics/special_issues/66D2HHG7PB.

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-4585-9 (Hbk)
ISBN 978-3-7258-4586-6 (PDF)
https://doi.org/10.3390/books978-3-7258-4586-6

© 2025 by the authors. Articles in this book are Open Access and distributed under the Creative Commons Attribution (CC BY) license. The book as a whole is distributed by MDPI under the terms and conditions of the Creative Commons Attribution-NonCommercial-NoDerivs (CC BY-NC-ND) license (https://creativecommons.org/licenses/by-nc-nd/4.0/).

Contents

About the Editors

Tao Liu

Tao Liu is an Associate Professor at the College of Sciences, Northeastern University. He received his Ph.D. degree in Mathematics and M.S. degree in Applied Mathematics from Harbin Institute of Technology and Harbin Engineering University, respectively. He was a Post-Doctoral Research Fellow at the School of Electrical and Electronic Engineering, Nanyang Technological University from 2022 to 2023. His research interests include deep learning, reinforcement learning, multiscale methods (multigrid and wavelet), the homotopy method, inverse and ill-posed problems, computational mathematics, and applied mathematics. In related subjects, he has published more than 50 SCI journal papers and more than 30 EI conference papers. He was appointed as the leader of the Outstanding Project of Nature Science Foundation of Hebei Province of China in 2019. His research is funded by the Natural Science Foundation of Hebei Province of China and the Marine Ecological Restoration and Smart Ocean Engineering Research Center of Hebei Province of China, among others. Furthermore, he serves as a reviewer for *Mathematical Reviews*, published by the American Mathematical Society, an Editorial Board Member and Topical Advisory Panel member for several academic journals indexed in SCI, and has served as an invited speaker and member of international technical committees for many conferences indexed in EI.

Fazlollah Soleymani

Fazlollah Soleymani pursued a postdoctoral fellowship at the Polytechnic University of Valencia in Spain under the Marie Curie Action - FP7 scholarship. In 2017, he began his work at the Institute for Advanced Studies in Basic Sciences (IASBS) in Iran. Subsequently, in 2023, he joined East China Normal University in Shanghai, China, as a research associate. Soleymani's primary research interests lie in computational mathematics, evidenced by his Scopus h-index of 28 as of January 2025. His recent work encompasses various problem domains, including RBF(-HFD) meshfree schemes for financial partial (integro-) differential equations, high-order iterative methods, numerical solutions of stochastic ODEs, risk management using various risk measures, and the application of machine learning in finance.

Qiang Ma

Qiang Ma is an Associate Professor at the Department of Mathematics, Harbin Institute of Technology. He received his Ph.D. degree in Basic Mathematics and M.S. degree in Computational Mathematics from Harbin Institute of Technology. His interests, in both research and teaching, include structure-preserving algorithms for differential equations and numerical methods for stochastic differential equations. He has published over 50 papers in prestigious international journals, such as the *International Journal of Computer Mathematics*, *Numerical Methods for Partial Differential Equations*, *Mathematical Methods in the Applied Sciences*, *Calcolo*, *Numerical Algorithms*, among others. He has presided over one National Natural Science Foundation of China Youth Science Foundation Project, one National Key Research and Development Special Sub-Project, and one Shandong Provincial Natural Science Foundation General Project.

Preface

Dear Colleagues,

Computational mathematics and applied mathematics are closely related fields within mathematics. Computational mathematics research focuses on numerical analysis and scientific calculation methods, such as interpolation and approximation, numerical methods of differential equations, numerical integration, matrix computation, and linear equation systems. With the development of large-scale computing and parallel computing technology, computational mathematics has been able to handle large-scale data and complex problems. Applied mathematics research tends to focus on its practical applications in various fields, such as physics, engineering, economics, finance, geophysics, computer science, social sciences, biology, and medicine. Research in applied mathematics has made significant breakthroughs in optimization algorithms, data mining, and machine learning, providing strong support for the development and application of science and technology.

This reprint features a selection of 16 papers that present groundbreaking findings in theoretical studies, along with the latest advancements in addressing practical scientific and technological challenges.

This reprint brought together mathematicians, physicists, and engineers, as well as other scientists. Topics covered in this reprint include the following:

Nonlinear Schrödinger equations;
Klein–Gordon equations;
Impulsive neutral differential equations;
Kepler's equation;
Parabolic heat equations;
Nonlinear monotone equations;
Fractional differential equations;
Elliptic curves;
Inverse and ill-posed problems;
The impulsive Runge–Kutta method;
The Steffensen-type method without memory;
Reinforcement learning;
The constrained Markov decision process;
Time series;
Financial mathematics;
Biomathematics.

Tao Liu, Fazlollah Soleymani, and Qiang Ma
Guest Editors

mathematics

MDPI

Article

An Efficient Iterative Approach for Hermitian Matrices Having a Fourth-Order Convergence Rate to Find the Geometric Mean

Tao Liu [1], Ting Li [1], Malik Zaka Ullah [2], Abdullah Khamis Alzahrani [2] and Stanford Shateyi [3,*]

[1] School of Mathematics and Statistics, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China; liutao@neuq.edu.cn (T.L.); 202115110@stu.neu.edu.cn (T.L.)
[2] Mathematical Modelling and Applied Computation (MMAC) Research Group, Department of Mathematics, Faculty of Science, King Abdulaziz University, Jeddah 21589, Saudi Arabia; zmalek@kau.edu.sa (M.Z.U.); akalzahrani@kau.edu.sa (A.K.A.)
[3] Department of Mathematics and Applied Mathematics, School of Mathematical and Natural Sciences, University of Venda, P. Bag X5050, Thohoyandou 0950, South Africa
* Correspondence: stanford.shateyi@univen.ac.za

Abstract: The target of this work is to present a multiplication-based iterative method for two Hermitian positive definite matrices to find the geometric mean. The method is constructed via the application of the matrix sign function. It is theoretically investigated that it has fourth order of convergence. The type of convergence is also discussed, which is global under an appropriate choice of the initial matrix. Numerical experiments are reported based on input matrices of different sizes as well as various stopping termination levels with comparisons to methods of the same nature and same number of matrix–matrix multiplications. The simulation results confirm the efficiency of the proposed scheme in contrast to its competitors of the same nature.

Keywords: iterative approach; geometric mean; matrix sign; fractal; basins of attraction; fourth order of convergence

MSC: 41A25; 65F60

1. Introduction

1.1. The Sign for a Matrix

The matrix sign function (MSF) [1], alternatively referred to as the matrix signum function or sign of a matrix, is an operation applied to matrices, producing a matrix of identical dimensions. The origin of the function sign is taken from its scalar counterpart, which works on real numbers, assigning +1 to positive scalars, -1 to negative scalars, and 0 to zero ([2], chapter 11). Extending the sign function to matrices was a progression aimed at aiding the exploration of matrix theory and the development of novel algorithms for addressing matrix equations and systems [3]. The MSF for an invertible matrix $W \in \mathbb{C}^{n\times n}$ can be written as

$$\operatorname{sign}(W) = U, \tag{1}$$

and then, computed by ([4]; p. 107)

$$U = \frac{2}{\pi} W \int_0^\infty (t^2 I + W^2)^{-1} dt, \tag{2}$$

where I is the identity matrix.

Since its inception, this function has been applied across diverse fields of mathematics and scientific computation, playing crucial roles in numerical analysis [4,5]. Recent investigations [6,7] have concentrated on enhancing methods for effectively finding the MSF,

developing high-order techniques, and investigating its correlations with other matrix functions and properties. This function has emerged as a valuable instrument for manipulating and characterizing matrices in various domains.

For small to moderately sized matrices, it is feasible to compute the spectral factorization and subsequently assess $f(W)$. Higham [4] details numerous methods for computing functions of matrices within this size range. Ref. [8] proposed a foundational framework for computing several matrix functions, including (1). However, for large matrices W, the computational cost associated with computing the spectral factorization may become prohibitively high. Similarly, other techniques that rely on factorizing W to compute $f(W)$ may also become impractical for large matrices lacking an exploitable structure. In such scenarios, iterative methods emerge as viable options.

1.2. Matrix Geometric Mean (MGM)

The geometric mean serves as a measure of central tendency for a finite set of real numbers, computed by taking the product of their values, and then, finding the nth root. When focusing on matrices, the MGM is a tool that averages a set of matrices with positive definiteness. In some recent papers [9,10], the geometric mean for two positive definite (PD) matrices has been identified as the midpoint of the geodesic joining the two matrices.

When faced with two matrices, the determination of the MGM necessitates the consideration of the following function:

$$\phi : \mathbb{A}^n \times \mathbb{A}^n \to \mathbb{A}^n, \tag{3}$$

wherein $\mathbb{A}^n$ stands for the set for all $n \times n$ HPD (Hermitian PD) matrices. Here, $GMean(W, Z)$ could be provided by [11]

$$W\#Z := GMean(W, Z) = W(W^{-1}Z)^{\frac{1}{2}}, \tag{4}$$

which is a sub-case of the formulation below for $t \in \mathbb{R}$ [12]:

$$W\#_t B := W(W^{-1}Z)^t. \tag{5}$$

In [13] (p. 105), the following formulation was provided for computing the MGM:

$$W\#Z = W^{\frac{1}{2}}(W^{-\frac{1}{2}}ZW^{-\frac{1}{2}})^{\frac{1}{2}}W^{\frac{1}{2}}, \tag{6}$$

for the HPD W and Z matrices of suitable dimensions. For $GMean(\cdot, \cdot)$, we have

$$GMean(V, I) := \mathrm{diag}(\sqrt{v_1}, \sqrt{v_2}, \ldots, \sqrt{v_n}), \tag{7}$$

wherein $V = \mathrm{diag}(v_1, v_2, \ldots, v_n)$ stands for a diagonal matrix with $v_i > 0$, and I stands for the unit matrix. It can be asserted that $W\#Z$ possesses all the attributes essential for a geometric mean [14], such as

$$Z\#W = W\#Z. \tag{8}$$

If Z and W commute with each other, then we have

$$W\#Z = (WZ)^{\frac{1}{2}}.$$

Here, X stands for the matrix square root (principal) of W and it is given by $X = W^{\frac{1}{2}}$ as the solution of the following matrix equation:

$$X^2 = W, \tag{9}$$

where here W does not have real non-positive eigenvalues. In fact, the matrix $W\#Z$ solves the following Riccati equation ([13]; p. 106):

$$Z = XW^{-1}X. \tag{10}$$

Additionally, by using the characteristics of the square root (principal), one has

$$W(W^{-1}Z)^{\frac{1}{2}} = (ZW^{-1})^{\frac{1}{2}}W = Z(ZW^{-1})^{\frac{1}{2}} = (WZ^{-1})^{\frac{1}{2}}Z = W\#Z. \tag{11}$$

The MGM has several important features as follows:

$$\begin{array}{lcl} W\#W & = & W, \\ (W\#Z)^{-1} & = & W^{-1}\#Z^{-1}, \\ W\#Z & \leq & \frac{1}{2}(W+Z). \end{array} \tag{12}$$

1.3. Goals

The objective of this article is to introduce a novel approach for computing (11) for two appropriate matrices by initially determining the MSF.

- It is demonstrated that this iterative technique achieves global convergence for this purpose, provided a suitable initial approximation, with a fourth-order convergence rate.
- Detailed convergence proofs and numerical simulations are provided.
- It can be inferred that the proposed scheme serves as an effective tool for computing (4) of two HPD matrices.
- An advantage of the proposed method is its ability to obtain larger attraction basins, resulting in a larger convergence radius compared to similar methods for computing the matrix sign function. This leads to faster convergence, thereby reducing the total number of matrix multiplications.

From both practical and theoretical standpoints, the quest for the computation of the MGM holds significance. This endeavor often relies on iterative techniques, prominently leveraging various matrix–matrix products.

1.4. Structure

The rest of this work is structured as follows. Section 2 furnishes some techniques for determining the MSF. Subsequently, Section 3 elucidates the utility of high-order schemes and introduces a solver tailored for addressing nonlinear equations. The iterative approach is extended to handle matrices and its efficacy is substantiated via analysis, showing a convergence order of four in Section 4. In Section 5, we investigate the attraction basins to guarantee the global convergence behavior compared to analogous methods. The stability of the proposed matrix iteration is discussed in Section 6. Section 7 discusses the extension of the scheme for computing the MGM for two HPD matrices. Section 8 presents the outcomes of our numerical investigation, validating our theoretical insights and highlighting the practicality of our approach. Finally, Section 9 offers our concluding remarks.

2. Several Existing Iterations

Let f represent a real-valued function and have the nonlinear problem [15]

$$f(t) = 0. \tag{13}$$

In the case where $f(\xi) = 0$, ξ is identified as a root of f. Given that (13) typically lacks an exact solution in a general context, it becomes imperative to seek an approximate solution through iterative approaches [16,17]. Newton's method stands out as a foundational iterative technique for this purpose, boasting convergence order and efficiency index values of 2 and 2.41, respectively. Alternatively, the root can be sought employing the fixed-point scheme in the format below:

$$k_{q+1} = g(k_q), \qquad q = 0, 1, 2, \cdots. \tag{14}$$

The field of iterative approaches finds fruitful application in solving matrix-related challenges, including the computation of matrix functions, as highlighted in works such as [18,19].

Let us recall an efficient general family of methods for finding the MSF here. The authors in [20] provided a general framework as a family of methods for calculating (1). Considering $\iota_1 + \iota_2 \geq 1$, then Ref. [20] discussed that the iteration structure below,

$$k_{q+1} = \frac{k_q P_{\iota_1}(1-k_q^2)}{Q_{\iota_2}(1-k_q^2)} := \psi_{2\iota_1+1,2\iota_2}, \tag{15}$$

converges with convergence speed $\iota_1 + \iota_2 + 1$ to ± 1. Therefore, the quadratically convergent Newton's solver can be obtained by

$$K_{q+1} = \frac{1}{2}\Big(K_q^{-1} + K_q\Big), \tag{16}$$

where

$$K_0 = W, \tag{17}$$

is the initial guess and W represents the input matrix based on (1). Observing that the reciprocal Padé approximations can be formulated using the inverses of (15), we recognize that Newton's method offers an iterative strategy for approximating (1); for more, see [21,22].

Employing (15), the following famous methods, specifically, the locally convergent Newton–Schulz solver that does not require matrix inversion,

$$K_{q+1} = \frac{1}{2}K_q(3I - K_q^2), \tag{18}$$

and the globally convergent Halley's solver,

$$K_{q+1} = [K_q^2 + I][K_q(3I + K_q^2)]^{-1}, \tag{19}$$

can be extracted. Further state-of-the-art developments can be observed in [23,24].

3. A Multi-Step Method for Nonlinear Equations

Initially, we introduce the following secant-type iterative technique devoid of memory to address (30); see also the discussions in [25–27]. Let us consider the following structure:

$$\begin{cases} d_q = k_q - f'(k_q)^{-1} f(k_q), & q = 0, 1, \ldots, \\ y_q = k_q - \dfrac{i_1 f(k_q) - i_2 f(d_q)}{i_3 f(k_q) - i_4 f(d_q)} \dfrac{f(k_q)}{f'(k_q)}, \\ k_{q+1} = y_q - f[y_q, k_q]^{-1} f(y_q), \end{cases} \tag{20}$$

which gives the following error equation:

$$\epsilon_{q+1} = \frac{a_2(i_3 - i_1)}{i_3}\epsilon_q^2 + \mathcal{O}(\epsilon_q^3), \tag{21}$$

where

$$a_j = (f'(\xi)j!)^{-1}(f^{(j)}(\xi)), \text{ and } \epsilon_q = k_q - \xi.$$

The connection described in (21) leads to the choice of $i_1 = i_3$, thus converting the error equation to

$$\epsilon_{q+1} = \frac{a_2^2(i_1 + i_2 - i_4)}{i_1}\epsilon_q^3 + \mathcal{O}(\epsilon_q^4). \tag{22}$$

Hence, it is imperative to ascertain the remaining undetermined coefficients in such a manner that guarantees

$$i_1 + i_2 - i_4 = 0,$$

thereby diminishing the newly appeared asymptotic (22). Furthermore, their choice should strive to minimize the ensuing error equation, specifically,

$$\frac{\left(3i_1a_2a_3(i_1+i_2-i_4) - a_2^3\left(-i_4(5i_1+i_2)+i_1(3i_1+5i_2)+i_4^2\right)\right)}{i_1^2}\epsilon_q^4.$$

We choose now $i_1 = i_3 = 29$, $i_2 = 30$, and $i_4 = 59$. The second substep outlined in (23) marks an advancement from the procedure outlined in [28]. Moreover, this methodology is devised to widen the attraction basins, offering a comparative advantage over other methods with similar attributes. Hence, we derive the following iterative method:

$$\begin{cases} d_q = k_q - f'(k_q)^{-1}f(k_q), \\ y_q = k_q - \dfrac{29f(k_q) - 30f(d_q)}{29f(k_q) - 59f(d_q)}\dfrac{f(k_q)}{f'(k_q)}, \\ k_{q+1} = y_q - f[y_q, k_q]^{-1}f(y_q), \end{cases} \tag{23}$$

where the divided difference operator (see, e.g., [29]) is obtained via $f[l, j] := (f(j) - f(l))(j - l)^{-1}$.

Theorem 1. *Assume ξ in D as a single zero of $f : D \subseteq \mathbb{C} \to \mathbb{C}$ that is a differentiable function (sufficiently). Additionally, let us consider that k_0 is sufficiently close to the solution. Consequently, the iterates produced by (23) exhibit a convergence of at least fourth order.*

Proof. By expanding $f(k_q)$ and $f'(k_q)$ around ξ, we obtain

$$f(k_q) = f'(\xi)[\epsilon_q + a_2\epsilon_q^2 + a_3\epsilon_q^3 + a_4\epsilon_q^4 + a_5\epsilon_q^5 + \mathcal{O}(\epsilon_q^6)], \tag{24}$$

and

$$f'(k_q) = f'(\xi)[1 + 2a_2\epsilon_q + 3a_3\epsilon_q^2 + 4a_4\epsilon_q^3 + 5a_5\epsilon_q^4 + \mathcal{O}(\epsilon_q^5)]. \tag{25}$$

Now, from (24) and (25), one obtains

$$d_q = \xi + a_2\epsilon_l^2 + \left(-2a_2^2 + 2a_3\right)\epsilon_l^3 - \left(-4a_2^3 + 7a_2a_3 - 3a_4\right)\epsilon_l^4 + \mathcal{O}(\epsilon_l^5). \tag{26}$$

By expanding $f(d_q)$ around ξ and using (26), it is possible to write

$$y_q = \xi - \frac{1}{29}a_2^2\epsilon_l^4 + \left(\frac{927a_2^3}{841} - \frac{33a_2a_3}{29}\right)\epsilon_l^4 + \mathcal{O}(\epsilon_l^5). \tag{27}$$

From (27) and (24), one obtains that

$$f[y_q, k_q] = f'(k_q) + a_2f'(k_q)\epsilon_l^1 + a_3f'(k_q)\epsilon_l^2 \left(a_4f'(k_q) - \frac{a_2^3f'(k_q)}{29}\right)\epsilon_l^3 + \mathcal{O}(\epsilon_l^4). \tag{28}$$

Now, by the use of (27) and (28) we attain

$$\epsilon_{q+1} = y_q - f[y_q, k_q]^{-1}f(y_q) - \xi = -\frac{1}{29}a_2^3\epsilon_q^4 + \mathcal{O}(\epsilon_q^5). \tag{29}$$

The proof is complete. □

4. Expanding to the Matrix Context

Using (23) to address solving

$$F(U) := U^2 - I = 0, \tag{30}$$

leads to the following scheme:

$$K_{q+1} = 2K_q\left(37I + 72K_q^2 + 7K_q^4\right)\left[15I + 146K_q^2 + 71K_q^4\right]^{-1}. \tag{31}$$

Since the convergence of the iterative methods must be performed for $\pm$, so each constructed iteration in this category can be written as a fraction in the scalar form, which means that its reciprocal can be convergent to $\mp$. Due to this similarity, one can derive the reciprocal version of (31) and express it as follows:

$$K_{q+1} = \left(15I + 146K_q^2 + 71K_q^4\right)\left[2K_q\left(37I + 72K_q^2 + 7K_q^4\right)\right]^{-1}. \tag{32}$$

The process begins with an initial value and progressively refines the estimation with each iterate until reaching convergence. This iterative characteristic proves beneficial when handling intricate or sizable matrices, as direct methods might entail high computational costs. Currently, we are examining the convergence properties of (32) to establish a convergence outcome.

Theorem 2. *When determining the sign of matrix W, under the condition of no eigenvalues residing on the imaginary axis, we begin with an initial approximation K_0 sufficiently near to U, selected using (17). This choice ensures commutativity with W. Consequently, the scheme (32) (or equivalently (31)) converges towards the sign matrix U with a convergence rate of four.*

Proof. The method we introduce requires matrix multiplications, similar to its competitors. However, much of the convergence theory for our method relies on computing eigenvalues (see, e.g., [30,31]) from one iteration to the next. Let us employ the Jordan block matrix J to decompose W in the following manner using the invertible matrix L:

$$W = LJL^{-1}. \tag{33}$$

Utilizing this, in conjunction with the iterative approach, results in an iteration structure akin to the original iteration structure, albeit focusing on the eigenvalues transitioning from step q to step $q+1$, as demonstrated below:

$$\begin{aligned}\lambda_{q+1}^i =& \left(15 + 146{\lambda_q^i}^2 + 71{\lambda_q^i}^4\right)\\ &\times\left[2\lambda_q^i\left(37 + 72{\lambda_q^i}^2 + 7{\lambda_q^i}^4\right)\right]^{-1}, \quad 1 \le i \le n,\end{aligned} \tag{34}$$

where $b_i = \text{sign}\lambda_q^i(\lambda) = \pm 1$. Generally, (34) reveals that the eigenvalues tend to $b_i = \pm 1$, i.e.,

$$\lim_{q\to\infty}\left|\frac{\lambda_{q+1}^i - b_i}{\lambda_{q+1}^i + b_i}\right| = 0. \tag{35}$$

This indicates convergence and suggests that the eigenvalues approach ± 1 with each iteration, leading to eigenvalue clustering during the iterative process. Following the examination of convergence, determining the convergence rate becomes essential. For this purpose, it is taken into account that

$$\Theta_q = 2K_q(37I + 72K_q^2 + 7K_q^4). \tag{36}$$

Thus, we can write the following:

$$\begin{aligned} K_{q+1} - U &= (15I + 146K_q^2 + 71K_q^4)\Theta_q^{-1} - U \\ &= [15I + 146K_q^2 + 71K_q^4 - U\Theta_q]\Theta_q^{-1} \\ &= [15I + 146K_q^2 + 71K_q^4 - 74K_qU - 144K_q^3U - 14K_q^5U]\Theta_q^{-1} \\ &= [-15(K_q - U)^4 + 14K_qU\Big(K^4 - 4K_q^3U + 6K_q^2U^2 - 4K_qU^3 + I\Big)]\Theta_q^{-1} \\ &= [-15(K_q - U)^4 + 14K_qU(K_q - U)^4]\Theta_q^{-1} \\ &= (K_q - U)^4[-15I + 14K_qU]\Theta_q^{-1}. \end{aligned} \tag{37}$$

By employing (37), one can derive the following:

$$\|K_{q+1} - U\| \leq \Big(\|\Theta_q^{-1}\|\|15I - 14K_qU\|\Big)\|K_q - U\|^4, \tag{38}$$

indicating a convergence order of four for (32). This concludes the proof. The analysis of error for (31) can be inferred in a similar manner. □

5. Attraction Basins

It is essential to highlight how the suggested approach can be contrasted with its counterparts from the Padé iterations for computing the MSF. The fourth-order methods within the Padé family can be outlined as [20]

$$K_{q+1} = [I + 6K_q^2 + K_q^4][4K_q(I + K_q^2)]^{-1}, \qquad \text{Padé [1,2]}, \tag{39}$$

$$K_{q+1} = [4K_q(I + K_q^2)][I + 6K_q^2 + K_q^4]^{-1}, \qquad \text{Reciprocal of Padé [1,2]}. \tag{40}$$

Constructing high-order schemes is only practical if they can rival existing methods in performance and computational cost. Hence, comparing (31) and (32) to (39) and (40) is crucial to demonstrate that the presented solver maintains the same convergence order as the established Padé methods while requiring similar computational resources. Moreover, its larger convergence radii, as evidenced by the corresponding basins of attractions, underscore its superiority.

To assess the global convergence and broader attraction basins of the presented solver compared to its counterparts, attraction basins are plotted. The region $[-2, 2] \times [-2, 2] \in \mathbb{C}$ is partitioned into a grid of initial points, each tested for convergence based on the criterion $|k_q^2 - 1| \leq 10^{-2}$. Diverging points are marked in black. The numerical results are depicted in Figures 1 and 2, with shading indicating the number of iterations required for convergence.

While Newton's solver and iterative methods (31) and (39) exhibit global convergence, the attraction basins for (31) and (32) show lighter areas, suggesting faster convergence compared to their Padé counterparts.

Figure 1. Basins of attraction shaded based upon the number iterations required to fulfill the convergence criterion for (16) (**left**) and (39) (**right**).

Figure 2. Basins of attraction shaded based upon the number iterations required to fulfill the convergence criterion for (31) (**left**) and (32) (**right**).

6. Stability

The stability analysis regarding (32) is presented in the following theorem. Theorem 3 extends a fundamental outcome discussed in [32] concerning the stability of pure matrix iterations. Additionally, it mentions that:

- $U^2 = I$;
- $U^{-1} = U$.

Theorem 3. *Using (32) and considering that W does not possess any purely imaginary eigenvalues, we can conclude that the sequence* $\{K_q\}_{q=0}^{\infty}$*, with* $K_0 = W$*, remains asymptotically stable.*

Proof. Suppose that β_q represents a perturbation in the computational process of the iterative method at the q-th iteration, and express this as follows:

$$\widetilde{K}_q = K_q + \beta_q. \tag{41}$$

Now, let us conduct a first-order error analysis, indicating that for all $i \geq 2$,

$$(\beta_q)^i \approx 0. \tag{42}$$

If β_q is sufficiently small, then (42) holds well, allowing one to express

$$\widetilde{K}_{q+1} = [15I + 146\widetilde{K}_q^2 + 71\widetilde{K}_q^4][2\widetilde{K}_q(37I + 72\widetilde{K}_q^2 + 7\widetilde{K}_q^4)]^{-1}. \tag{43}$$

As we reach a large-enough value for q, indicating the convergence phase, we assume that K_q is approximately equal to $\text{sign}(W)$, denoted as U. Through significant simplifications, we derive that

$$\widetilde{K}_{q+1} \approx \left(U + \frac{1}{2}\beta_q - \frac{1}{2}U\beta_q U\right). \tag{44}$$

Using

$$\beta_{q+1} = \widetilde{K}_{q+1} - K_{q+1},$$

we can write

$$\beta_{q+1} \approx \frac{1}{2}\beta_q - \frac{1}{2}U\beta_q U. \tag{45}$$

This results in the fact that the next iteration, denoted as $q+1$, remains within certain limits, meaning

$$\|\beta_{q+1}\| \leq \frac{1}{2}\|\beta_0 - U\beta_0 U\|. \tag{46}$$

Hence, the sequence $\{K_q\}_{q=0}^{\infty}$ generated by (32) achieves asymptotic stability. With that, the proof comes to a close. □

7. Extension to MGM

An efficient way to calculate the geometric mean of two HPD matrices W and Z, without having to find the matrix square roots (principal), relies on (refer to [4] (page 131))

$$\operatorname{sign}\left(\begin{bmatrix} 0 & W \\ Z^{-1} & 0 \end{bmatrix}\right) = \begin{bmatrix} 0 & T \\ T^{-1} & 0 \end{bmatrix}, \tag{47}$$

and therefore, the mean can be obtained as follows:

$$T = W(Z^{-1}W)^{-\frac{1}{2}} = W(W^{-1}Z)^{\frac{1}{2}} = W\#Z. \tag{48}$$

If the starting matrix is chosen correctly and the matrices do not have eigenvalues on the imaginary axis, there will not be any breakdown when computing the inverse matrix for (31) or (16). It is worth mentioning that for any appropriate matrix E such that $W + E$ is PD, we have

$$\operatorname{sign}\begin{pmatrix} 0 & W+E \\ I & 0 \end{pmatrix}, \tag{49}$$

as a fixed value of (32).

8. Computational Aspects

Various methods examined previously are contrasted under equivalent conditions within Mathematica [33]. To demonstrate the effectiveness of the innovative approach, we conduct computational simulations of various sizes. The subsequent termination criterion in l_∞ is employed

$$\|K_{q+1} - K_q\|_\infty \leq \varepsilon. \tag{50}$$

The Cauchy stopping criterion (50) can be employed instead of the convergence criterion $K_{q+1}^2 - I = 0$, as it is significantly easier to implement in higher dimensions. This approach circumvents the need for additional matrix powers in the algorithmic step, leading to faster convergence by eliminating one further matrix–matrix multiplication.

The globally convergent methods (31), (32), (39), (40), and (16) are denoted in this section as PM1, PM2, PD1, PD2, and NM2, respectively. All the fourth-order methods require four matrix products and one matrix inverse per cycle. Here, we tackle the Riccati problem (10) for two HPD matrices specified by

$$W = \begin{pmatrix} 2 & 0 & 1 & & \\ 0 & 2 & 0 & 1 & \\ 1 & 0 & 2 & 0 & \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ & & 1 & 0 & 2 \end{pmatrix}_{n\times n},$$

$$Z = \begin{pmatrix} \frac{3}{2} & \frac{2}{3} & & & \\ \frac{2}{3} & \ddots & \ddots & & \\ & \ddots & \ddots & \frac{2}{3} & \\ & & \frac{2}{3} & \frac{3}{2} & \end{pmatrix}_{n\times n}.$$

Several details are in order:

- We consider different sizes and employ the same termination criterion.
- The inverse of matrix W in (10) was calculated directly, after which both matrices were used in the iterative methods for comparative analysis.
- The comparison outcomes for different iterative techniques are provided in Figures 3 and 4.
- All the iterative methods having fourth order examined here incur an equivalent computational expense concerning matrix–matrix products and inverse calculations.

The findings concerning the calculation of the geometric mean of the two HPD matrices demonstrate the superiority of PM1 and PM2 over their counterparts of similar order, showcasing their efficiency. Clearly the computational CPU time for PM1 and PM2 decreases in contrast to PD1 and PD2 since they use the same number of matrix–matrix products and inverses per computing cycle but PM1 and PM2 have larger attraction basins based on the discussions in Section 5. The MGM is applicable only to HPD matrices with positive real eigenvalues. In contrast, the proposed method for the matrix sign function can be applied to all matrices with complex eigenvalues, provided none of them lie on the imaginary axis.

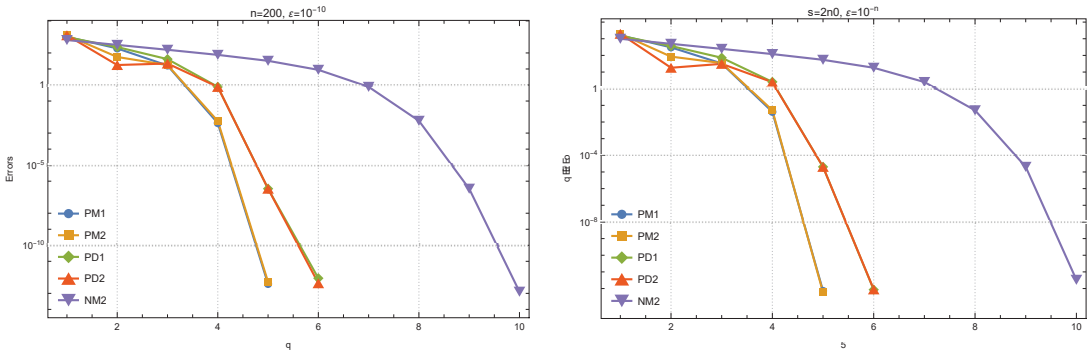


Figure 3. Simulation results for different tolerances in the stopping criterion. It shows PM1 and PM2 arrive at the convergence phase quicker than their competitors of the same order in a smaller number of iterations.

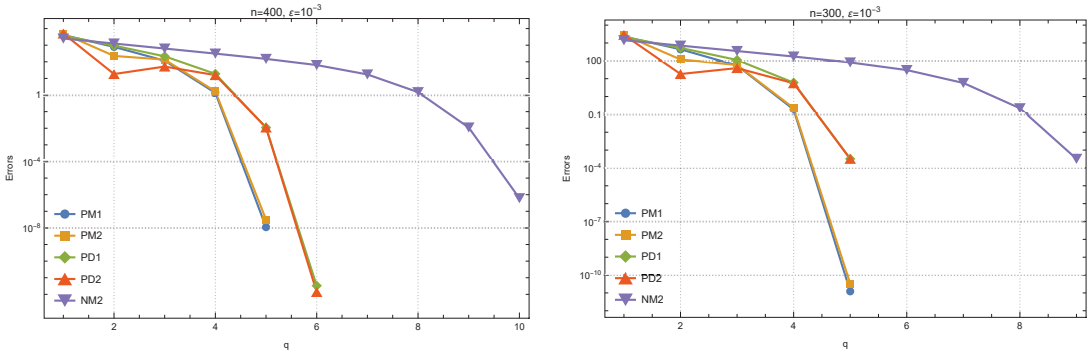


Figure 4. Simulation results for different dimensions. It shows PM1 and PM2 arrive at the convergence phase quicker than their competitors of the same order in a smaller number of iterations.

It is worth noting that such iterative approaches can be expedited (in a similar way as in Newton's method [4]) by computing an additional parameter at each iteration and substituting K_q with $\mu_q K_q$, as outlined below:

$$\mu_q = \begin{cases} |\det(K_q)|^{\frac{-1}{n}}, & \text{(determinantal scaling)}, \\ \sqrt{\dfrac{\rho(K_q^{-1})}{\rho(K_q)}}, & \text{(spectral scaling)}, \\ \sqrt{\dfrac{\|K_q^{-1}\|}{\|K_q\|}}, & \text{(norm scaling)}. \end{cases} \tag{51}$$

We conclude this section by emphasizing the significance of learning procedures within the realm of artificial intelligence and machine learning models [34–36]. Designing a strategy based on machine learning tools could efficiently accelerate the convergence of such iterative structures by developing a model that quickly transitions the initial matrix into the convergence phase. This could be focused on in future works on this field.

9. Conclusions

The concept of the geometric mean, initially defined for positive scalars, can be extended to HPD matrices in multiple ways. These extensions aim to capture essential properties akin to those expected of a mean, to varying degrees. A practical use of the MSF arises in determining the MGM of two HPD matrices. This is particularly necessary in addressing a specific category of nonlinear matrix equations like (10).

In this study:

- We introduced a computationally intensive approach for determining the sign of a matrix, which was subsequently demonstrated to exhibit a fourth-order convergence order.
- The new method demonstrates global convergence and competes favorably against prominent alternatives from the Padé solvers.
- The stability of the scheme was brought forward.
- Computational experiments were conducted to show the efficacy of our iterative technique (and its reciprocal) across various test scenarios.

Forthcoming research lines can be concentrated on two aspects. First, it would be more efficient if a sharper initial matrix could be designed so as to put the iterative approach much closer to the convergence phase, leading to a faster convergence. And second, it is favorable to improve the results by extending them to higher orders while possessing larger attraction basins when compared to the exiting multiplication-rich methods from the Padé family of methods.

Author Contributions: Conceptualization, T.L. (Tao Liu), T.L. (Ting Li), M.Z.U., A.K.A. and S.S.; formal analysis, T.L. (Tao Liu), T.L. (Ting Li), M.Z.U., A.K.A. and S.S.; funding acquisition, T.L. (Tao Liu) and S.S.; investigation, T.L. (Tao Liu), T.L. (Ting Li), M.Z.U., A.K.A. and S.S.; methodology, T.L. (Tao Liu), M.Z.U., A.K.A. and S.S.; supervision, T.L. (Tao Liu), M.Z.U., A.K.A. and S.S.; validation, T.L. (Tao Liu), M.Z.U., A.K.A. and S.S.; writing—original draft, T.L. (Tao Liu), T.L. (Ting Li), M.Z.U., A.K.A. and S.S.; writing—review and editing, T.L. (Tao Liu), T.L. (Ting Li), M.Z.U., A.K.A. and S.S. All authors have read and agreed to the published version of the manuscript.

Funding: The research of the first author was funded by the Research Project on Graduate Education and Teaching Reform of Hebei Province of China (YJG2024133).

Data Availability Statement: In terms of the data availability statement, it is affirmed that there is no data sharing applicable to this article, as there were no new data created throughout the course of this work.

Acknowledgments: The fourth author states that: This research work was funded by Institutional Fund Projects under grant no. (IFPIP: 1331-130-1443). The authors gratefully acknowledge technical and financial support provided by the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

Conflicts of Interest: The authors affirm that there are no personal relationships or identifiable conflicting financial interests that could be perceived to influence the research presented in this manuscript.

References

1. Denman, E.D.; Beavers, A.N. The matrix sign function and computations in systems. *Appl. Math. Comput.* **1976**, *2*, 63–94. [CrossRef]
2. Hogben, L. *Handbook of Linear Algebra*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2007.
3. Roberts, J.D. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *Int. J. Cont.* **1980**, *32*, 677–687. [CrossRef]
4. Higham, N.J. *Functions of Matrices: Theory and Computation*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2008.
5. Li, X.-P.; Nunes, R.W.; Vanderbilt, D. Density-matrix electronic-structure method with linear system-size scaling. *Phys. Rev. B* **1993**, *47*, 10891–10894. [CrossRef] [PubMed]
6. Shi, L.; Zaka Ullah, M.; Kumar Nashine, H.; Alansari, M.; Shateyi, S. An Enhanced Numerical Iterative Method for Expanding the Attraction Basins When Computing Matrix Signs of Invertible Matrices. *Fractal Fract.* **2023**, *7*, 684. [CrossRef]
7. Soleymani, F.; Kumar, A. A fourth-order method for computing the sign function of a matrix with application in the Yang—Baxter-like matrix equation. *Comput. Appl. Math.* **2019**, *38*, 64. [CrossRef]

8. Al-Mohy, A.; Higham, N. A scaling and squaring algorithm for the matrix exponential. *SIAM J. Matrix Anal. Appl.* **2009**, *31*, 970–989. [CrossRef]
9. Soleymani, F.; Sharifi, M.; Shateyi, S.; Khaksar Haghani, F. An algorithm for computing geometric mean of two Hermitian positive definite matrices via matrix sign. *Abstr. Appl. Anal.* **2014**, *2014*, 978629. [CrossRef]
10. Jebreen, H.B.; Akgül, A. A fast iterative method to find the matrix geometric mean of two HPD matrices. *Math. Meth. Appl. Sci.* **2019**, *42*, 5615–5625. [CrossRef]
11. Pusz, G.; Woronowicz, S.L. Functional calculus for sesquilinear forms and the purification map. *Rep. Math. Phys.* **1975**, *8*, 159–170. [CrossRef]
12. Lawson, J.D.; Lim, Y. The geometric mean, matrices, metrics and more. *Amer. Math. Month.* **2001**, *108*, 797–812. [CrossRef]
13. Bhatia, R. *Positive Definite Matrices, Princeton Series in Applied Mathematics*; Princeton University Press: Princeton, NJ, USA, 2007.
14. Iannazzo, B. The geometric mean of two matrices from a computational viewpoint. *Numer. Lin. Alg. Appl.* **2016**, *23*, 208–229. [CrossRef]
15. McNamee, J.M.; Pan, V.Y. *Numerical Methods for Roots of Polynomials—Part I*; Academic Press: Cambridge, MA, USA; Elsevier: Amsterdam, The Netherlands, 2007.
16. McNamee, J.M.; Pan, V.Y. *Numerical Methods for Roots of Polynomials—Part II*; Academic Press: Cambridge, MA, USA; Elsevier: Amsterdam, The Netherlands, 2013.
17. Shil, S.; Nashine, H.K.; Soleymani, F. On an inversion-free algorithm for the nonlinear matrix problem $X^{\alpha}A^{*}X^{-\beta}A + B^{*}X^{-\gamma}B = I$. *Int. J. Comput. Math.* **2022**, *99*, 2555–2567. [CrossRef]
18. Byers, R.; Xu, H. A new scaling for Newton's iteration for the polar decomposition and its backward stability. *SIAM J. Matrix Anal. Appl.* **2008**, *30*, 822–843. [CrossRef]
19. Soheili, A.R.; Soleymani, F. Iterative methods for nonlinear systems associated with finite difference approach in stochastic differential equations. *Numer. Algor.* **2016**, *71*, 89–102. [CrossRef]
20. Kenney, C.S.; Laub, A.J. Rational iterative methods for the matrix sign function. *SIAM J. Matrix Anal. Appl.* **1991**, *12*, 273–291. [CrossRef]
21. Greco, F.; Iannazzo, B.; Poloni, F. The Padé iterations for the matrix sign function and their reciprocals are optimal. *Lin. Algebra Appl.* **2012**, *436*, 472–477. [CrossRef]
22. Soleymani, F.; Stanimirović, P.S.; Shateyi, S.; Haghani, F.K. Approximating the matrix sign function using a novel iterative method. *Abstr. Appl. Anal.* **2014**, *2014*, 105301. [CrossRef]
23. Jung, D.; Chun, C.; Wang, X. Construction of stable and globally convergent schemes for the matrix sign function. *Lin. Alg. Appl.* **2019**, *580*, 14–36. [CrossRef]
24. Sharma, P.; Kansal, M. Extraction of deflating subspaces using disk function of a matrix pencil via matrix sign function with application in generalized eigenvalue problem. *J. Comput. Appl. Math.* **2024**, *442*, 115730. [CrossRef]
25. Haghani, F.K.; Soleymani, F. An improved Schulz-type iterative method for matrix inversion with application. *Trans. Inst. Meas. Control.* **2014**, *36*, 983–991. [CrossRef]
26. Ogbereyivwe, O.; Atajeromavwo, E.J.; Umar, S.S. Jarratt and Jarratt-variant families of iterative schemes for scalar and system of nonlinear equations. *Iran. J. Numer. Anal. Optim.* **2024**, *14*, 391–416.
27. Dehghani-Madiseh, M. Moore-Penrose inverse of an interval matrix and its application. *J. Math. Model.* **2024**, *12*, 145–155.
28. Zaka Ullah, M.; Muaysh Alaslani, S.; Othman Mallawi, F.; Ahmad, F.; Shateyi, S.; Asma, M. A fast and efficient Newton-type iterative scheme to find the sign of a matrix. *Aims Math.* **2023**, *8*, 19264–19274. [CrossRef]
29. Khdhr, F.W.; Soleymani, F.; Saeed, R.K.; Akgül, A. An optimized Steffensen-type iterative method with memory associated with annuity calculation. *The Euro. Phy. J. Plus* **2019**, *134*, 146. [CrossRef]
30. Cordero, A.; Soleymani, F.; Torregrosa, J.R.; Zaka Ullah, M. Numerically stable improved Chebyshev–Halley type schemes for matrix sign function. *J. Comput. Appl. Math.* **2017**, *318*, 189–198. [CrossRef]
31. Liu, T.; Zaka Ullah, M.; Alshahrani, K.M.A.; Shateyi, S. From fractal behavior of iteration methods to an efficient solver for the sign of a matrix. *Fractal Fract.* **2023**, *7*, 32. [CrossRef]
32. Iannazzo, B. Numerical Solution of Certain Nonlinear Matrix Equations. Ph.D. Thesis, Universita degli studi di Pisa, Pisa, Italy, 2007.
33. Hoste, J. *Mathematica Demystified*; McGraw-Hill: New York, NY, USA, 2009.
34. Larijani, A.; Dehghani, F. An efficient optimization approach for designing machine models based on combined algorithm. *FinTech* **2024**, *3*, 40–54. [CrossRef]
35. Mohammad, M.; Trounev, A.; Cattani, C. Stress state and waves in the lithospheric plate simulation: A 3rd generation AI architecture. *Results Phys.* **2023**, *53*, 106938. [CrossRef]
36. Mohammadabadi, S.M.S.; Yang, L.; Yan, F.; Zhang, J. Communication-efficient training workload balancing for decentralized multi-agent learning. *arXiv* **2024**, arXiv:2405.00839.

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

mathematics

MDPI

Article

Q-Sorting: An Algorithm for Reinforcement Learning Problems with Multiple Cumulative Constraints

Jianfeng Huang, Guoqiang Lu, Yi Li and Jiajun Wu *

College of Engineering, Shantou University, Shantou 515063, China; jfhuang@stu.edu.cn (J.H.); luguoqiang85@163.com (G.L.); 13487633473@139.com (Y.L.)
* Correspondence: wujiajun@ustc.edu

Abstract: This paper proposes a method and an algorithm called Q-sorting for reinforcement learning (RL) problems with multiple cumulative constraints. The primary contribution is a mechanism for dynamically determining the focus of optimization among multiple cumulative constraints and the objective. Executed actions are picked through a procedure with two steps: first filter out actions potentially breaking the constraints, and second sort the remaining ones according to the Q values of the focus in descending order. The algorithm was originally developed upon the classic tabular value representation and episodic setting of RL, but the idea can be extended and applied to other methods with function approximation and discounted setting. Numerical experiments are carried out on the adapted Gridworld and the motor speed synchronization problem, both with one and two cumulative constraints. Simulation results validate the effectiveness of the proposed Q-sorting in that cumulative constraints are honored both during and after the learning process. The advantages of Q-sorting are further emphasized through comparison with the method of lumped performances (LP), which takes constraints into account through weighting parameters. Q-sorting outperforms LP in both ease of use (unnecessity of trial and error to determine values of the weighting parameters) and performance consistency (6.1920 vs. 54.2635 rad/s for the standard deviation of the cumulative performance index over 10 repeated simulation runs). It has great potential for practical engineering use.

Keywords: reinforcement learning; cumulative constraint; constrained Markov decision process (CMDP)

MSC: 60J20

1. Introduction

Reinforcement learning [1] has been successful in areas like Atari games [2] and the game of Go [3]. The learning processes of these applications happen in simulator environments rather than real worlds. The sole objective is to find policies that maximize the return without having to consider any constraints. However, there are also problems with constraints. For example, imagine a recycling robot whose objective is to figure out a route from the origin to the destination to collect as much garbage as possible. Apart from the objective, the robot must keep the battery from running out before it reaches the destination [4]. Another example is the cellular network, where the objective is the maximum throughput and the constraints are transmission delay, service level, package loss rate, etc. [5]. Also, in the problem of energy management for hybrid electric vehicles, apart from the objective of minimum fuel consumption, the physical characteristics of motors and engines should be enforced as constraints [6]. Zhang et al. [7] considered an energy efficiency maximization problem with the power budget at the transmitter and the quality of service as constraints and tackled it using the proximal policy optimization framework. For heterogeneous networks, the achievable sum information rate is to be maximized with the achievable information rate requirements and the energy harvesting requirements as constraints [8]. In a word, there are lots of circumstances in practical engineering projects where objectives and constraints are to be considered simultaneously.

Decision-making problems with constraints are typically modeled and solved under the framework of the constrained Markov decision process (CMDP). There are two kinds of constraints: instantaneous and cumulative. The former requires that the action A_t taken must be a member of an admissible set A_C, which may be dependent on the current state S_t. The latter can be divided into two groups: probabilistic and expected. In cases of probabilistic constraints, the probability that the cumulative costs violate a constraint is required to be within a certain threshold. Expected constraints, on the other hand, pose requirements on the cumulated/averaged values of the costs. It can be further divided into two categories: discounted sum and mean value. Liu et al. [9] provided a summary and classification of RL problems with constraints. In this paper, the problem studied is restricted to discounted sum constraints in an episodic setting. Details are to be provided in Section 2.

MDPs with cumulative constraints (both discounted sum and mean value) were first studied in [10]. It is found that if the model is completely known, the CMDP problem can be transformed into a linear programming problem and solved. However, in practical problems, transition dynamics are seldom known in advance, making the theoretical solution inapplicable. Among other methods, Lagrangian relaxation is a popular one that turns the original constrained learning problem into an unconstrained one by adding the constraint functions weighted by corresponding Lagrange multipliers to the original objective function [11–14]. Drawbacks of the Lagrangian relaxation include sensitivity to the initialization of the multipliers as well as the learning rate, large performance variation during learning, no guarantee of constraint satisfaction during learning, too slow learning pace, etc. [9]. Furthermore, to derive the adaptive Lagrange multiplier, one has to solve the saddle point problem in an iterative way, which may be numerically unstable [15].

Lyapunov-based methods are also popular. Originally, Lyapunov functions were a kind of scalar function to describe the stability property of a system [16]. They can also represent the steady-state performance of a Markov process [17] and serve as a tool to transform the global properties of a system into local ones and vice versa [18]. The first attempts to utilize the Lyapunov functions to tackle CMDP problems can be found in [18], where an algorithm based on linear programming is proposed to construct the Lyapunov functions for the constraints. It is a value-function-based algorithm and not suitable for continuous action space. Another Lyapunov-based algorithm specifically for large and continuous action spaces using policy gradients (PG) to update the policies is proposed in [19]. The idea is to use the state-dependent linearized Lyapunov constraints to derive the set of feasible solutions and then project the policy parameters or the actions onto it. Compared with the Lagrangian relaxation methods, Lyapunov-based methods ensure constraint satisfaction both during and after learning. The drawbacks of the Lyapunov methods are in two aspects. First, to derive the Lyapunov functions on each policy evaluation step, a linear programming problem has to be solved, which may be numerically intractable if the state space is large [19]. Although it is possible to use heuristic constant Lyapunov functions depending only on the initial state and the horizon, theoretical guarantees are lost [20]. Second, Lyapunov methods require the initial policy π_0 to be feasible, whereas in some problems, feasible initial policies are unavailable, and it is usually more desirable to start with random policies [19].

Constrained Policy Optimization (CPO) [21] is an extension of the popular trust region policy optimization (TRPO) [22] to make it applicable to problems with discounted sum constraints. It respects the constraints both during and after learning and ensures monotonic performance improvement. It uses a conjugate gradient to approximate the Fisher Information Matrix and backtracking line search to determine feasible actions, which makes it computationally expensive and susceptible to approximation error [9,20]. CPO does not support mean-valued constraints and is difficult to extend to cases of multiple constraints [23]. Finally, the methodology of CPO can hardly be applied to other RL algorithms, which are not in the category of proximal policy gradient [18].

Interior-point policy optimization (IPO) proposed by [23] is a promising algorithm for RL problems with cumulative constraints. It is a first-order policy optimization algorithm inspired by the interior-point method [24]. The core idea of IPO is to augment the objective function with logarithmic barrier functions whose values go to negative infinity if the corresponding constraint is violated and zero if it is satisfied. IPO has a lot of merits, like its applicability to general types of cumulative constraints, including both discounted sum and mean-valued ones, its easy extension to handle multiple constraints; easy tuning of hyperparameters; and its robustness in stochastic environments. It is also noteworthy that IPO is one of the few that provides simulation results for multiple constraints. The main drawback of IPO is that the initial policy must be feasible [9]. This issue is addressed in later works by dividing the learning process into two phases [25,26]. In the first phase, the objective is totally ignored, and the cumulative costs are successively optimized to obtain a feasible policy. In the second phase, the original IPO algorithm is initiated with the feasible policy found at the end of the first phase. However, it is still not clear what should be performed if the agent gets stuck on an infeasible policy during the learning process of the second phase.

Although IPO demonstrates promising performances in empirical results, it does not provide adequate theoretical guarantees other than the performance bound. Comparatively, Triple-Q [27] is the first model-free and simulator-free RL algorithm for CMDP with proof on sublinear regret and zero constraint violation. It has the same low computational complexity as SARSA [28]. Although it is claimed that Triple-Q can be extended to accommodate multiple constraints, the corresponding simulation results are not provided in the paper. Triple-Q is designed for episodic CMDPs with discounted sum constraints only. In later works, it is integrated with optimistic Q-learning [29] to obtain another model-free algorithm named Triple-QA for infinite-horizon CMDPs with mean-valued constraints. Triple-QA also provides sublinear regret and zero constraint violations. In general, thorough performance bounds are usually provided by model-based methods like [30,31]. Triple-Q and Triple-QA are among the few exceptions.

Projection-based Constrained Policy Optimization (PCPO) [32] is an algorithm for expected cumulative constraints. It learns optimal and feasible policies iteratively in two steps. In the first step, it uses TRPO to learn an intermediate policy, which is better in terms of the objective but may be infeasible. In the second step, it projects the intermediate policy back into the constraint set to get the nearest feasible policy. The scheme of projection ensures improvement of the policy as well as satisfaction of the constraints. The main drawbacks of PCPO are expensive computation and limited generality, which are similar to those of CPO since they both use TRPO to perform policy updates [9].

Backward value functions (BVF) are another useful tool for solving CMDP problems. In typical RL settings [1], value functions are "forward," representing expected discounted cumulative rewards from the current state to the terminal state or the infinite end. Comparatively, BVF describes the expected sum of returns or costs collected by the agent *so far*. It builds upon the concept of the backward Markov chain, which is first discussed in [33]. Pankayaraj and Varakantham [34] employed BVF to tackle safety in hierarchical RL problems. Satija et al. [20] proposed a method for translating trajectory-level constraints into instantaneous state-dependent ones. This approach respects constraints both during and after learning. It requires fewer approximations as compared to other methods, and the only approximation error is from the function approximation. As a result, it is computationally efficient. One problem that has not been addressed well by [20], but is critical to the practical application, as has been discussed before, is the recovery mechanism from infeasible policies in the case of multiple constraints. This paper aims to fill this gap.

State augmentation is also another promising solution for CMDP problems. Calvo-Fullana et al. [35] proposed a systematic procedure to augment the state with Lagrange multipliers to solve RL problems with constraints. They also demonstrated that CMDP and regularized RL problems are *not* equivalent, meaning that there exist some constrained RL problems that cannot be solved by using a weighted linear combination of rewards (the

method of which is called lumped performances in this paper). McMahan and Zhu [36] proposed augmenting the state space to take constraints into consideration. They emphasized anytime constraint satisfaction in their methods, which requires the agent to never violate the constraint both during and after the learning process.

Primal-dual approaches are also popular. Bai et al. [37] proposed a conservative stochastic primal-dual algorithm that is able to achieve ϵ-optimal cumulative reward with zero constraint violations. However, it has also been demonstrated that classic primal-dual methods cannot solve all constrained RL problems [35].

Model error may significantly influence the ability of the agent to satisfy the constraints. Ma et al. [38] proposed a model-based safe RL framework named Conservative and Adaptive Penalty (CAP), which considers model uncertainty by calculating it and adaptively using it to trade off optimality and feasibility.

For safe RL applications, learning from offline data is also attractive since it avoids the dangerous actions of trial and error online. Xu et al. [39] proposed constraints penalized Q-learning (CPQ) to solve the distributional shift problem in offline RL.

Gaps: In RL problems with multiple cumulative constraints, the final learned policy should have two properties, which are optimality and feasibility. In other words, the return should be maximized, whereas the constraints should be satisfied. The two requirements are usually in opposite directions, however, meaning that purely pursing one would cause the other to fail. The learning process thus consists of two kinds of components, namely, optimization and recovery. The former is to drive the policy towards a larger return. The latter is to make it more feasible. For the existing literature, one point that has not gained much attention but is vital to practical applications of the algorithms, however, is the mechanism of recovery from infeasible policies. In other words, most algorithms are expected to work with feasible policies. They operate under the assumption that updating the current feasible policy would result in another feasible one. This property is called *consistent feasibility* [18,20]. For example, it is theoretically proven that CPO, Lyapunov-based, and BVF-based algorithms all maintain the feasibility of the policy upon updates once the base policies being updated are feasible [18,20,23]. However, the problem remains: what should be performed if the initial policy is infeasible, or if it is feasible at the beginning but turns infeasible in the middle of learning due to effects like function approximation error. In these cases, a mechanism to recover the infeasible policy back to a feasible one is important. The design of the recovery mechanism is not the focus of the existing literature but rather an implementation issue. A recovery method was originally proposed along with CPO in [21], which performs policy updates to purely optimize the constraints, ignoring the objective temporarily. This strategy is also adopted by the Lyapunov-based algorithm [19] and the BVF-based one [20]. However, the recovery method originally proposed with CPO only covers the case of a single constraint. It is unclear how to extend it to accommodate multiple constraints. Chow et al. [19] suggest extending this recovery update to the multiple-constraint scenario by doing gradient descent over the constraint that has the worst violation but provides simulation results on the case of single constraint only. This paper aims to fill the gap by proposing a systematic mechanism for policy recovery that is applicable to the case of multiple cumulative constraints and accompanied by corresponding simulation results.

Contributions: A simple method and algorithm named Q-sorting are proposed for CMDP problems with discounted sum constraints in a tabular and episodic setting with deterministic environments and policies. It is similar to the BVF-based algorithm in terms of the way to predict whether a certain action potentially violates a constraint, but additionally provides a systematic mechanism for recovering from infeasible policies. Compared to existing recovery methods used in CPO, Lyapunov-based, and BVF-based algorithms, it covers cases of multiple constraints. It also provides the possibility to rank the constraints according to their importance and specify the order in which they are to be considered, enabling finer control and configuration of the learning process. It is model-free and can be applied online. It pursues constraint satisfaction both during and after learning. Although

Q-sorting was originally developed in a tabular and episodic setting, it can be extended to methods with function approximation and discounted settings, as long as they are value-based. By using the BVF to estimate cumulative costs incurred so far, it can also be extended to accommodate stochastic environments and policies.

The rest of this paper is organized as follows. Section 2 introduces the problem. Section 3 discusses the proposed Q-sorting algorithm. Section 4 presents simulation results of Q-sorting on problems of Gridworld and motor speed synchronization control with one and two constraints and compares it to the conventional method of lumped performances. Section 5 gives a conclusion.

2. RL Problems with Multiple Cumulative Constraints

Consider a typical MDP: an agent is in some state S_t and takes an action A_t, transits to the next state S_{t+1} and receives a reward R_{t+1}. The process continues until the terminal state, or some exit condition, is reached. The whole *objective* is to learn an optimal policy $A_t = \pi(S_t)$ (supposing a deterministic one) to maximize the discounted cumulative rewards $R_1 + \gamma R_2 + \ldots$, namely, the return. In problems with multiple cumulative constraints, however, the agent has to take care of not only the objective but also the *constraints*. After A_t is taken on S_t, it receives not only R_{t+1} but also a vector $\mathbf{R}_{t+1,constraints}$ corresponding to "rewards" of different constraints:

$$\mathbf{R}_{\mathrm{t+1,constraints}} = [\mathrm{R}_{\mathrm{t+1,constraint1}}, \mathrm{R}_{\mathrm{t+1,constraint2}}, \ldots] \tag{1}$$

After the end of the episode, the cumulated values of $R_{t+1,constraint1}, R_{t+1,constraint2}, \ldots$ should be above some prespecified thresholds:

$$\begin{cases} \mathrm{R}_{1,\mathrm{constraint1}} + \gamma_{\mathrm{c}_1} \mathrm{R}_{2,\mathrm{constraint1}} + \ldots \geq \mathrm{c}_1 \\ \mathrm{R}_{1,\mathrm{constraint2}} + \gamma_{\mathrm{c}_2} \mathrm{R}_{2,\mathrm{constraint2}} + \ldots \geq \mathrm{c}_2 \\ \vdots \end{cases} \tag{2}$$

where $\gamma_{c_1}, \gamma_{c_2}, \ldots$, are the discount rates for different constraints. The difference between the returns of the objective and the constraints is that the former are what we are seeking to maximize, whereas the latter only have to stay above some value. Assumptions are as follows:

1. MDP is finite-time and episodic, which means that all discount rates $\gamma, \gamma_{c_1}, \gamma_{c_2}, \ldots$ are 1.
2. MDP as well as the policy are deterministic.

It should be noted that for CMDP problems, there may be one or multiple cumulative constraints, but there should always be only one objective.

3. Q-Sorting

RL problems with one objective and multiple cumulative constraints are analogous to those with multiple objectives. The core of the learning algorithm is to allocate learning resources, for example, computing time and service, between different constraints/objective. Due to safety requirements, it is also desired that the times when constraints are violated be as few as possible, both during and after learning. These problems could be solved by imposing some predefined rules specifying at each time step which objective/constraint should be solely considered.

The idea is more obvious by supposing a value-based RL algorithm like Monte-Carlo or Q-learning. Naturally, one Q table could be learned for each objective/constraint. And if no constraints are imposed, the action is typically produced according to some ε-greedy mechanism:

$$A_t = \begin{cases} \underset{a}{\mathrm{argmax}}\, Q_{objective}(S_t, a), a \in \mathbb{A}, & r \geq \varepsilon \\ \text{randomly pick one from } \mathbb{A}, & r < \varepsilon \end{cases} \tag{3}$$

where r represents a uniform random number in $(0, 1)$, ε is the exploration rate, and $\mathbb{A}$ is the set of all possible actions. The subscript in $Q_{objective}$ emphasizes that the Q table being used corresponds to the objective, namely, the return of which we are seeking to maximize.

Now consider the problem with one objective and multiple cumulative constraints. To predict the effects of a certain action on satisfying or violating constraints, it is necessary to record the rewards "*up until now*" and have them summed/accumulated. For example, suppose that the cumulative constraint refers to the fact that the fuel consumption on a trip should be within a certain amount. At each time step, to predict whether a future route satisfies the constraint, one should first check out how much fuel has been consumed. By subtracting the fuel already consumed from the total available amount (the constraint), one gets the surplus quota. And by comparing the surplus quota to the predicted fuel consumption *from now on* till the end, one gets a (predicted) conclusion on whether a certain route (action) violates the constraint.

To make it clear, suppose that only one cumulative constraint exists. When making decisions (choosing actions), two circumstances are possible. First, there is *at least one action satisfying the constraint* (in terms of prediction rather than reality). To maximize the return of the objective, one simply filters all actions violating the constraint out of $\mathbb{A}$ to get $\mathbb{A}_c$, which represents the set of all *feasible* actions, and then replace $\mathbb{A}$ with $\mathbb{A}_c$ in the greedy component of Equation (3) to get A_t. Next, consider the second circumstance, where *no actions satisfy the constraint*. In this case, the greedy action regarding the *objective* violates the constraint and thus cannot be used. Rather, if the "constraint-first" principle is adopted, the greedy action regarding the *constraint* should be used, which means that $Q_{objective}$ in Equation (3) should be replaced with $Q_{constraint}$. In other words, the *focus of optimization* is switched from the objective to the constraint when no actions are feasible. This seems natural if one observes Equation (2): requirements state that the value of cumulated $R_{t+1,constraint}$ be greater than or equal to some threshold, and not satisfying the constraint implies that this cumulative value is too small. To move the policy in the direction of satisfying the constraint, it is reasonable to pick the action a maximizing $Q_{constraint}(S_t, a)$.

In the presence of multiple cumulative constraints, however, things get complicated. On each time step, one has to decide the "focus of optimization", not between one objective and *one* constraint but among one objective and *multiple* constraints. Figure 1 illustrates the idea, using an example with one objective and four constraints. In a specific state, S_t, suppose that there are five action candidates. The Q values of each candidate are queried for different constraints/objectives. The satisfaction of a certain action candidate regarding a certain constraint is evaluated using the following equation:

$$isSatisfied(A_{t,candidate}) = \begin{cases} \text{true}, & Q_{constraint}(S_t, A_{t,candidate}) + RTN_{constraint} \geq c_i \\ \text{false}, & Q_{constraint}(S_t, A_{t,candidate}) + RTN_{constraint} < c_i \end{cases} \quad (4)$$

where RTN is for "return till now", that is, the cumulated rewards of the constraint up until now. Equation (4) is called a "test" for a certain *action candidate* $A_{t,candidate}$ regarding a certain *constraint* c_i on the time step t.

The table in Figure 1 shows a possible case of the test results, where a check mark is for satisfying the constraint and a cross mark is for violating it. Each column (except the last one) corresponds to a specific constraint, and each row corresponds to an action candidate. The last column corresponds to the objective.

The procedure is to test and filter all action candidates with each of the constraints, one by one, starting from the first. For example, for the first constraint, $A_{t,candidate1}$, $A_{t,candidate2}$, $A_{t,candidate3}$, and $A_{t,candidate4}$ pass the test, whereas $A_{t,candidate5}$ fails and is filtered out right away. Then, calculate $Q_{constraint2}(S_t, a)$ for the four survivors and test them with Equation (4). $A_{t,candidate1}$, $A_{t,candidate2}$, and $A_{t,candidate3}$ pass the second test, whereas $A_{t,candidate4}$ fails. Abandon $A_{t,candidate4}$ and repeat the process until no candidates pass the test or the last column (the objective) is reached. The column where all survivors settle on becomes the *focus of optimization*, and all survivors become *candidates to pick*. In this example, the focus is constraint3 and the candidates to pick are $A_{t,candidate1}$, $A_{t,candidate2}$, and$A_{t,candidate3}$. Among the three, the action

that maximizes $Q_{constraint3}(S_t, a)$ is ultimately picked. Specifically, $Q_{constraint3}(S_t, A_{t,candidate1})$, $Q_{constraint3}(S_t, A_{t,candidate2})$, and $Q_{constraint3}(S_t, A_{t,candidate3})$ are sorted in descending order, and the action candidate corresponding to the first is picked. With the learning process going on, the focus of optimization shall move from constraint1 to constraint2, constraint3, ... consecutively, and finally settle on the objective. The agent focuses on one constraint/objective at a time and strikes to find a policy that maximizes the objective performance while satisfying all the cumulative constraints.

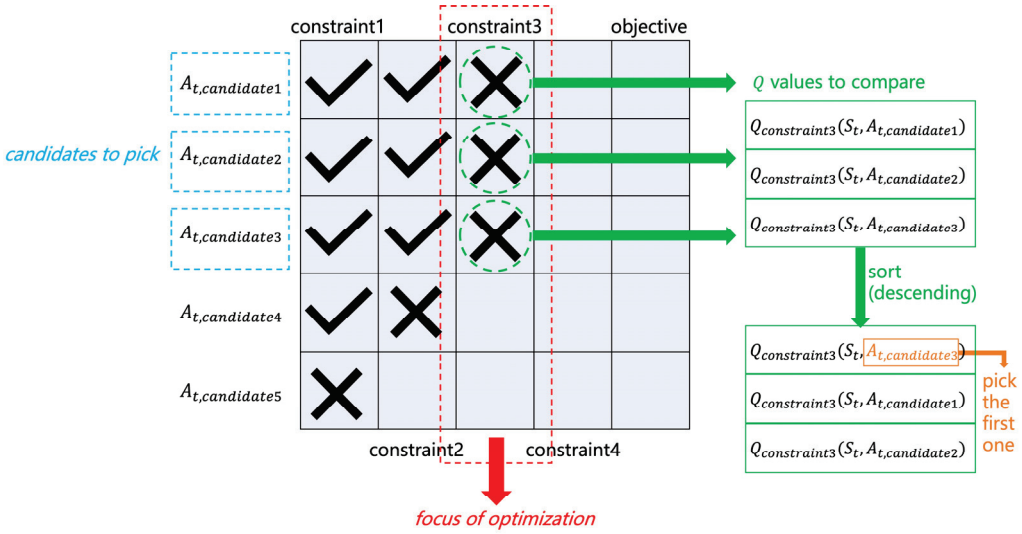


Figure 1. Q-sorting.

A typical optimization process for the policy is illustrated in Figure 2.

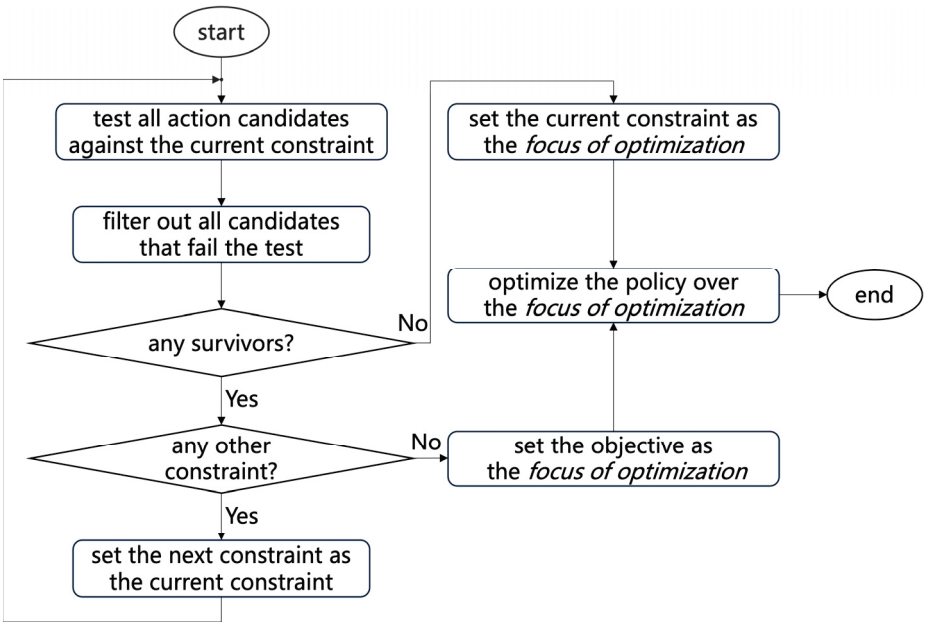


Figure 2. Optimization of the policy.

The pseudocode of the Q-sorting algorithm is summarized in Algorithm 1.

Algorithm 1. Q-sorting

Algorithm parameter: small $\varepsilon > 0$
Initialize $Q_{objective}(s,a), Q_{constraint1}(s,a), Q_{constraint2}(s,a), \ldots$ for the objective and each constraint arbitrarily except that $Q(terminal, \cdot) = 0$, for all $s \in \mathbb{S}$, $a \in \mathbb{A}$
Loop for each episode:
Initialize S_t
Initialize an empty array *trajectory*
Loop for each step of the episode:
Generate a uniform random number $r \in [0,1]$
IF $r < \varepsilon$
Randomly pick $A_t \in \mathbb{A}$
ELSE

1. Test and filter action candidates, starting from the first constraint, until no candidates pass a specific test or all tests are passed. If all candidates fail on a constraint, the *constraint* becomes the focus of optimization; on the other hand, if there is at least one candidate satisfying all constraints (passing all the tests), the *objective* becomes the focus of optimization. Record the index of focus as i, $i \in \{constraint1,\ constraint2,\ \ldots,\ objective\}$.
2. Record indices of candidates reaching i as $j_1, j_2, \ldots$, $j_k \in \{candidates1,\ candidates2, \ldots\}$.
3. Sort $Q_i(S_t, A_{j_1}),\ Q_i(S_t, A_{j_2}), \ldots$ in descending order and pick the action corresponding to the first as A_t. If multiple actions attain the maximum Q_i value at the same time, randomly pick one from them.

Take A_t, observe $R_{t+1}, R_{t+1,constraint1}, R_{t+1,constraint2}, \ldots$ and S_{t+1}
Append the vector
$$move = [S_t, A_t, S_{t+1}, R_{t+1}, R_{t+1,constraint1}, R_{t+1,constraint2}, \ldots]^{\mathrm{T}}$$
to *trajectory*: $trajectory \leftarrow [trajectory, move]$
Update the current state: $S_t \leftarrow S_{t+1}$
until the terminal state is reached
Update $Q_{objective}, Q_{constraint1}, Q_{constraint2}, \ldots$ with Monte Carlo, according to the *trajectory* recoded

4. Simulation Results

The effectiveness of the proposed Q-sorting is verified by two problems. The first one is the classic Gridworld, and the second one is the motor speed synchronization control. For both problems, cases of one constraint and two constraints are investigated. All learning processes start with random policies, which may be infeasible. The framework of Q-sorting can be applied to any value-based RL algorithm, like Monte Carlo or Q-learning. Here, for the simulation results, Monte Carlo is used. Source code and demos are provided in the supplementary materials.

4.1. Gridworld

A 5×5 Gridworld is considered, as shown in Figure 3. In the classic setting, the agent starts from the origin (the upper left) and tries to reach the destination (the lower right) with as few steps as possible. There are no constraints, but only the objective. In this paper, however, the Gridworld problem is adapted to include one or two constraints. In the case of one constraint, the agent collects three points upon each move (even those that leave the agent in the original position, like those against the wall). Throughout the whole episode, a minimum of 300 points is required. The objective is the same as that of the classic version: minimum steps. Upon simple inspection, it is quite easy to conclude that the optimal value of steps to reach the destination while satisfying the constraint is 100.

Simulation results are reported in Figure 4, which are obtained from 10 repeated simulations. The curve represents the mean, and the shaded region represents the standard deviation. The first two subplots show performances corresponding to the objective and the constraint, respectively. The last subplot illustrates the progress of the *focus of optimization*, as labeled in Figure 1. For a specific episode, the focus of optimization on each step is summed up and averaged over the whole episode to get the *focus of episode*, which is then plotted. All performance indices are averaged within a moving window containing the nearby 10 episodes.

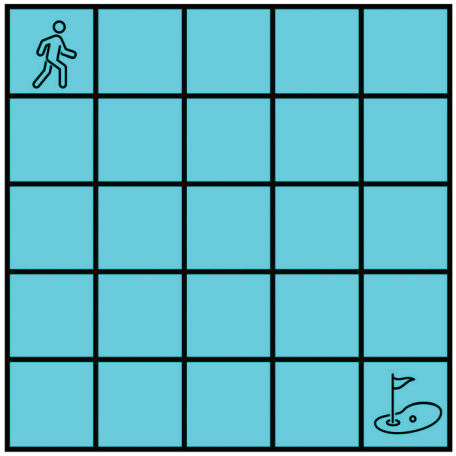


Figure 3. Gridworld.

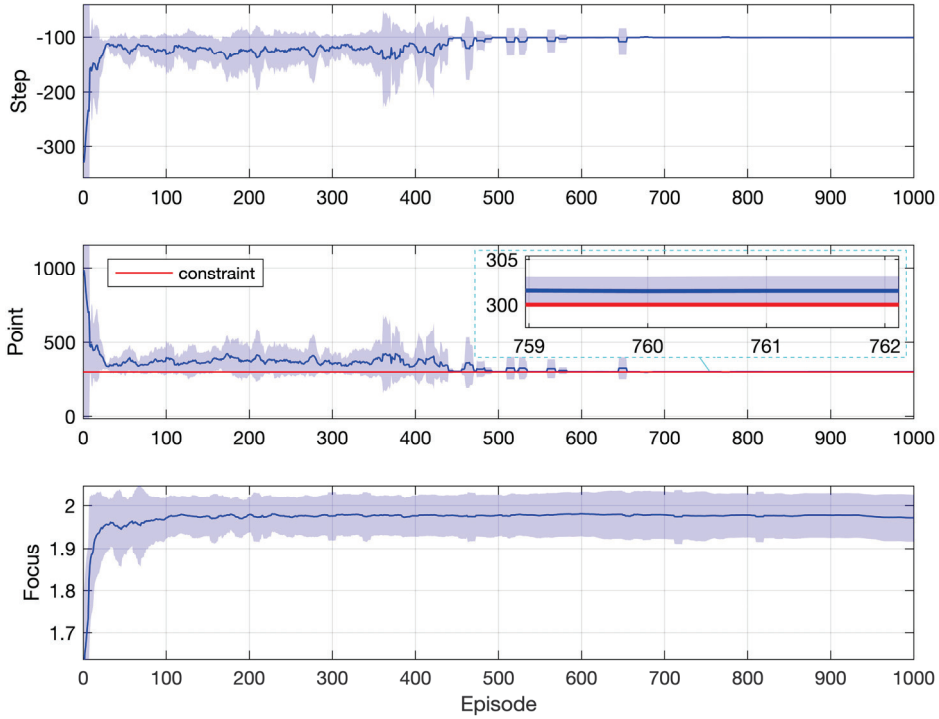


Figure 4. Simulation results: Gridworld with one constraint.

From Figure 4, it can be observed that the averaged points collected are far above the constraint threshold at the very beginning of learning, which is the result of random policies upon initialization. On average, it takes the agent about 300 steps to reach the destination in the first episode, and the corresponding collected points are about 1000, which is certainly nonoptimal since the constraint requires only 300 points. After that, steps taken quickly approach the optimal value of 100, with the averaged points collected decreasing while staying above the constraint. With the decay of the exploration rate, the policy gradually converges to the optimal and feasible one, achieving the theoretically minimum number of steps 100.

The last subplot of Figure 4 shows the progress of the focus. Within the first 30 episodes, the focus value quickly climbs from 1, which corresponds to the constraint, to 2, which corresponds to the objective. This is because the random policies upon initialization easily satisfy the constraint. However, it is not always the case. A more difficult situation is presented later, where the agent has to consider another extra constraint.

The mechanism of filtering and sorting (Section 3) ensures that the agent considers the constraint in the first place while learning to maximize the return. This is reflected in the second subplot, where the points collected stay above the required threshold (300) most of the time. In other words, satisfaction with the constraint is ensured during the learning process, not just after the end of it.

The optimal trajectory after learning is shown in Figure 5. It is produced by executing the learned policy greedily. The color bar shows the number of visits for each cell.

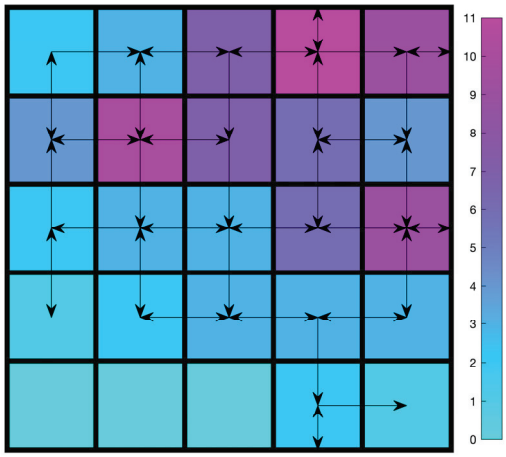


Figure 5. Trajectory after learning: Gridworld with one constraint.

The one-constraint Gridworld problem is then extended to include one more constraint: the number of turns. One turn is counted if both conditions are met: (a) the moving axis (either horizontal or vertical) of the current action is different from that of the previous one; (b) the current position is different from the previous one. At the beginning, the moving axis is undefined (null). The moving axis is updated only if the current position is different from the previous one.

In the simulation, two thresholds for the number of turns are tested. The first one is 20, the results of which are shown in Figure 6. The agent is required to collect at *least* 300 points with at *most* 20 turns, using as few steps as possible. It is to be noted that the constraint on the number of turns and the objective of the minimum steps are somehow contradictory. To constrain the number of turns, the agent is tempted to adopt a strategy that keeps going against the wall. However, doing so would probably increase the number of steps. It is difficult to balance the two. In the simulation, it was found that, apart from the horizontal and vertical positions, adding a third state variable, which represents the number of total steps taken so far, is helpful. And different from the first constraint, which is easily satisfied from the beginning by the random policies, the second constraint gets satisfied late in the middle of the learning. This is indicated by the third and last subplot of Figure 6, in which the focus index slowly increases from 1 to 3 and spends a lot of episodes around 2.

Trajectory after learning about the two-constraint Gridworld problem is shown in Figure 7. Comparing Figure 7 with Figure 5, the effects of the constraint on the number of turns are obvious. To satisfy the second constraint while collecting points, the agent adopts a strategy to keep going against the wall (the purple cell), which counts as no turns.

To make the comparison clearer, the threshold for the second constraint is further lowered from 20 to 5. Simulation results for the learning process and the trajectory after learning are shown in Figures 8 and 9, respectively. It is a more difficult mission for the agent since now it has to collect at least 300 points with at most 5 turns. As a result, the focus index climbs more slowly than in Figures 4 and 6. The agent spends a lot of time learning to constrain the number of turns to 5. After the end of learning, however, it successfully finds a policy that attains 300 points within 5 turns. In the final learned trajectory, the agent spends a lot of time on the lower left corner by going left against the wall.

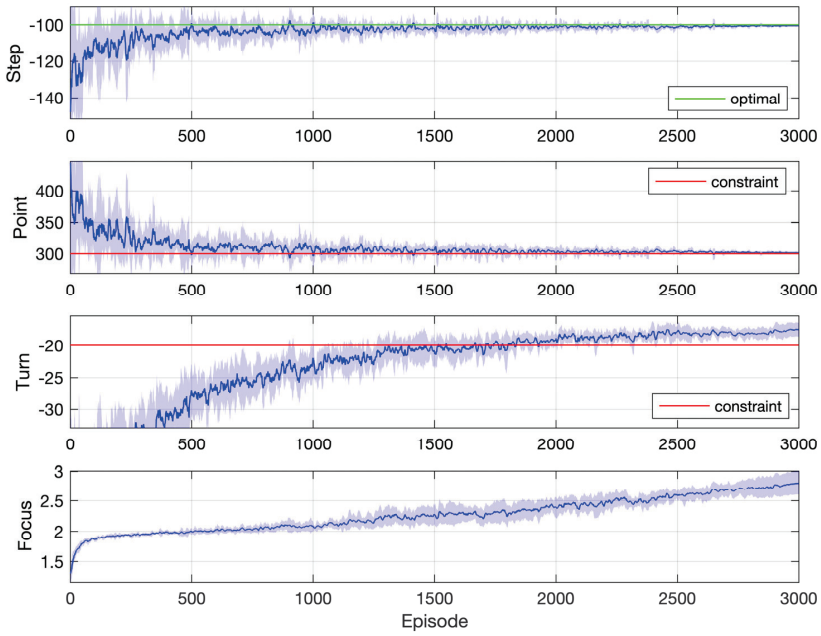


Figure 6. Simulation results: Gridworld with two constraints (20 turns at most).

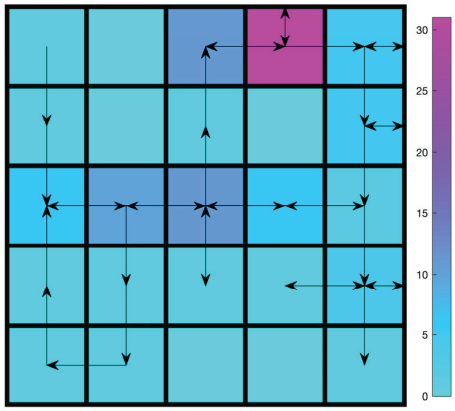


Figure 7. Trajectory after learning: Gridworld with two constraints (20 turns at most).

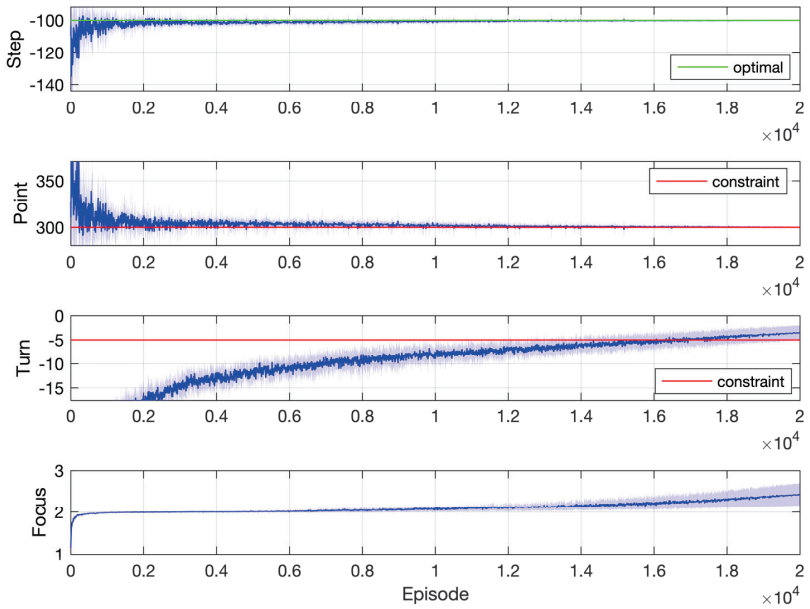


Figure 8. Simulation results: Gridworld with two constraints (5 turns at most).

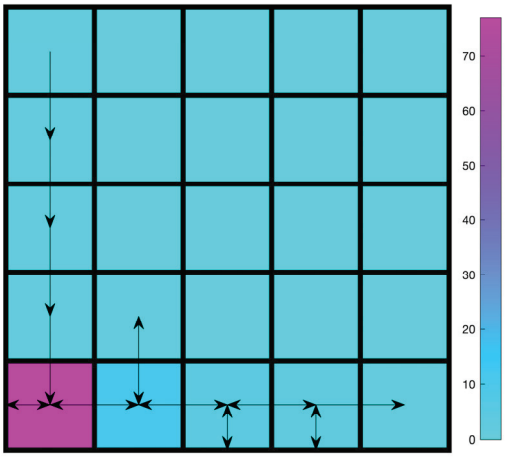


Figure 9. Trajectory after learning: Gridworld with two constraints (5 turns at most).

4.2. Motor Speed Synchronization Control

The motor speed synchronization problem can be found in [40,41]. To be short, the speed of the motor is required to change from an initial speed to a terminal one by following a reference trajectory. In this paper, it is modeled as a RL problem. The state variable is the time stamp, discretized with a step size of 0.1 s. The action variable is the motor torque, with the range $[-10, 0]$ Nm for cases of one constraint and $[-10, 5]$ Nm for cases of two constraints, both discretized with the quantization step 0.5 Nm. A detailed problem setup can be found in the supplementary materials.

Two kinds of objectives are considered here. In the first one, the motor speed is required to decrease to negative values as *fast* as possible ($R_{t+1} = -1$). Whereas in the second one, it should be as *slow* as possible ($R_{t+1} = 1$). As in the Gridworld problem, both cases of one constraint and two constraints are considered. In the case of one constraint, the sole requirement is that the deviation between the actual speed trajectory and the reference be *smaller* than a certain threshold. The deviation between two speed trajectories is calculated as $\sum_{t=0} R_{t+1,constraint1}$ where $R_{t+1,constraint1}$ is as follows:

$$R_{t+1,constraint1} = -\left|\omega_{t+1,ref} - \omega_{t+1,act}\right| \text{ (rad/s)} \tag{5}$$

The reference speed trajectory is a simple line given by the following equation:

$$\omega_{ref}(t) = \frac{\omega_0 - \omega_T}{0 - T} \cdot t + \omega_0 \tag{6}$$

where $\omega_0 = 3000$ rpm (revolutions per minute), $\omega_T = 0$, and $T = 1$ s. In the case of two constraints, apart from the trajectory deviation, the number of "turns" by the actual motor speed along the whole process is also constrained to be *larger* than a certain value. Here, one "turn" is defined as a change in the sign of the rotational acceleration. For example, if the rotational acceleration of the motor is negative (the speed is going down) on the previous time step and positive (the speed is going up) on the current time step, it is counted as one turn. It will be interesting to see how the agent manages to satisfy both constraints.

Figure 10 shows motor speed trajectories after the end of learning for the two cases of objective under different thresholds of c_1 ranging from -600 to -100 rad/s. Cumulated $R_{t+1,constraint1}$ in Equation (2) against the threshold c_1 is also shown in the form $\sum_{t=0} R_{t+1,constraint1}/c_1$. If the value of $\sum_{t=0} R_{t+1,constraint1}$ is larger than that of c_1, the constraint is successfully satisfied.

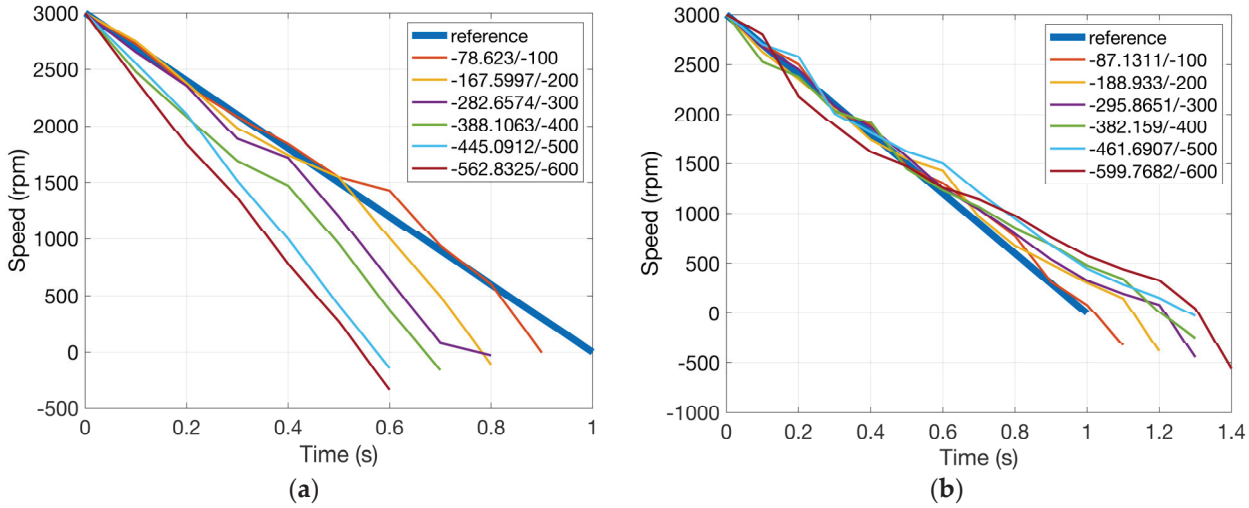


Figure 10. Motor speed trajectory after learning (one constraint), x/y is for $\sum_{t=0} R_{t+1,constraint1}/c_1$. (**a**) Objective: shortest duration and (**b**) objective: longest duration.

It can be seen that all learned motor speed trajectories successfully satisfy the corresponding constraints. The objective performance index is also maximized, which can be inferred from the simulation results. For example, in the case where the objective is the shortest duration for the motor to reach negative speed values, if we tighten the constraint on deviation between the actual speed trajectory and the reference, the duration would be longer since the actual speed trajectory would have to lean towards the reference. On the other hand, if the constraint is loose, the duration would be shorter. The effects of the objective can also be observed by comparing the two subplots in Figure 10. For the shortest duration objective, speed trajectories are all under the reference, whereas for the longest duration objective, they are all above, which is reasonable.

Figure 11 shows simulation results for one of the instances in Figure 10, where the objective is the longest duration and the constraint threshold c_1 is -100 rad/s. This requires that the cumulated speed deviation along the whole process does not exceed 100 rad/s, which is quite a difficult task. As a result, the focus index increases at a rather slow pace because the agent spends a lot of time learning to satisfy the constraint. It is near the end of learning that the agent starts to consider the objective completely. The return settles on the optimal value 11, which corresponds to a duration of 1.1 s (because the time step is 0.1 s and the agent gets one unit reward upon each transition along the timeline), after about 2000 episodes. The constraint is satisfied after about 1500 episodes.

To further emphasize the effectiveness of the algorithm in satisfying the constraint while pursuing optimality, performances regarding different levels of the constraint in the process of learning are shown in Figure 12. The objective is the longest duration for the motor to reach negative speed values. Three levels of the constraint are compared, namely, $c_1 = -1000$ rad/s, $c_1 = -600$ rad/s, and $c_1 = -\infty$ (which corresponds to the case of no constraints). It can be inferred that if no constraints are posed, the optimal policy is to output the maximum torque possible on each time step, which will also result in the maximum deviation from the reference speed trajectory. This is exactly the case in Figure 12, when $c_1 = -\infty$. However, if specific constraints are posed for the level of deviation, the objective performance will be compromised in order to satisfy the constraint. Notice how the agent pushes itself against the limit of the constraint to maintain satisfaction with it while pursuing optimal objective performance. The tighter the constraint, the smaller the return.

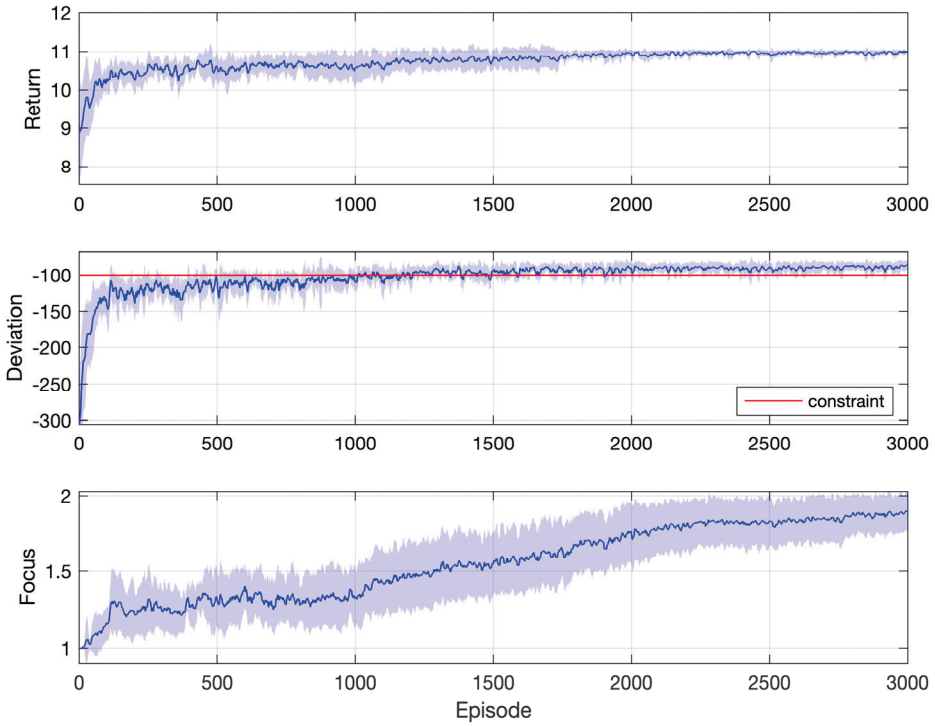


Figure 11. Simulation results: motor speed synchronization control with one constraint (objective: longest duration; constraint: $c_1 = -100$ rad/s).

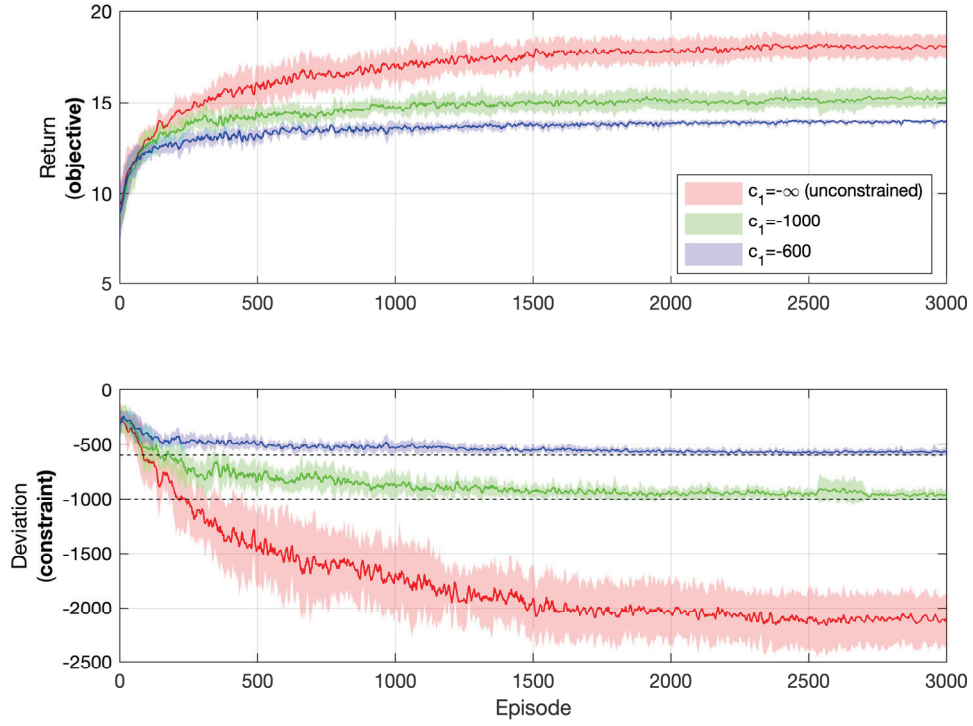


Figure 12. Simulation results: motor speed synchronization control with different levels of the constraint (objective: longest duration).

Simulation results for the case of two constraints are shown in Figure 13, where c_1 for the first constraint is fixed to -500 rad/s. As has been discussed before, the second constraint corresponds to the number of "turns", that is, the speed going up and down. A comparison between Figures 10 and 13 will reveal the effects of the second constraint. If the number of speed turns is unconstrained, under the threshold $c_1 = -500$ rad/s, the shortest

duration for the motor to reach negative speed values is 0.6 s, according to Figure 10. However, if at least three or four turns are required, the shortest duration will increase to 0.8 s, according to Figure 13. This seems natural since alternating between acceleration and deceleration takes the agent more time to reach the target speed than full deceleration. In the second subplot of Figure 13, the first constraint is slightly violated (-500.828 vs. -500), which may be due to the inaccuracy of the learned function values. Figure 14 shows simulation results for one of the instances in Figure 13a, where the objective is the shortest duration with cumulative constraints $c_1 = -500$ rad/s and $c_2 = 4$ turns. The agent learns to satisfy the first constraint after about 10 episodes and the second one after about 150 episodes.

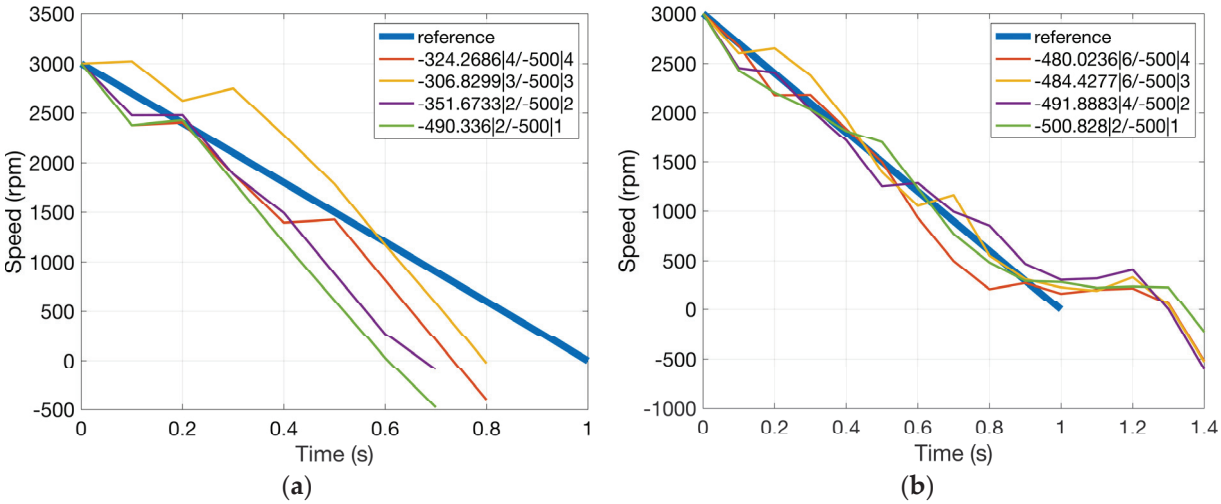


Figure 13. Motor speed trajectories after learning (two constraints), x1|x2/y1|y2 is for $\sum_{t=0} R_{t+1,constraint1} | \sum_{t=0} R_{t+1,constraint2} \,/\, c_1 | c_2$. (**a**) Objective: shortest duration and (**b**) objective: longest duration.

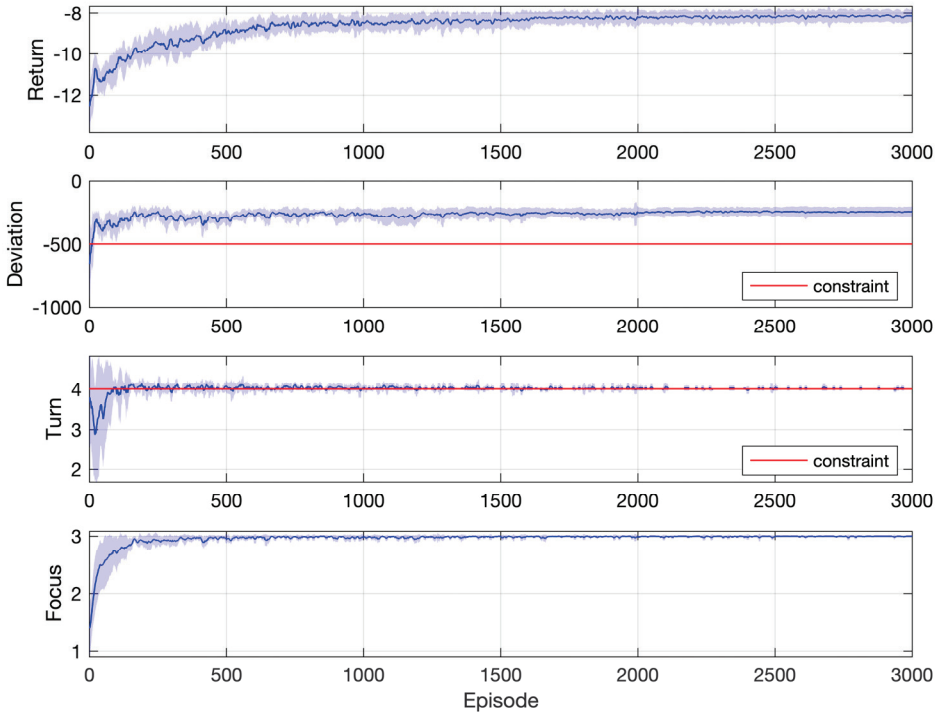
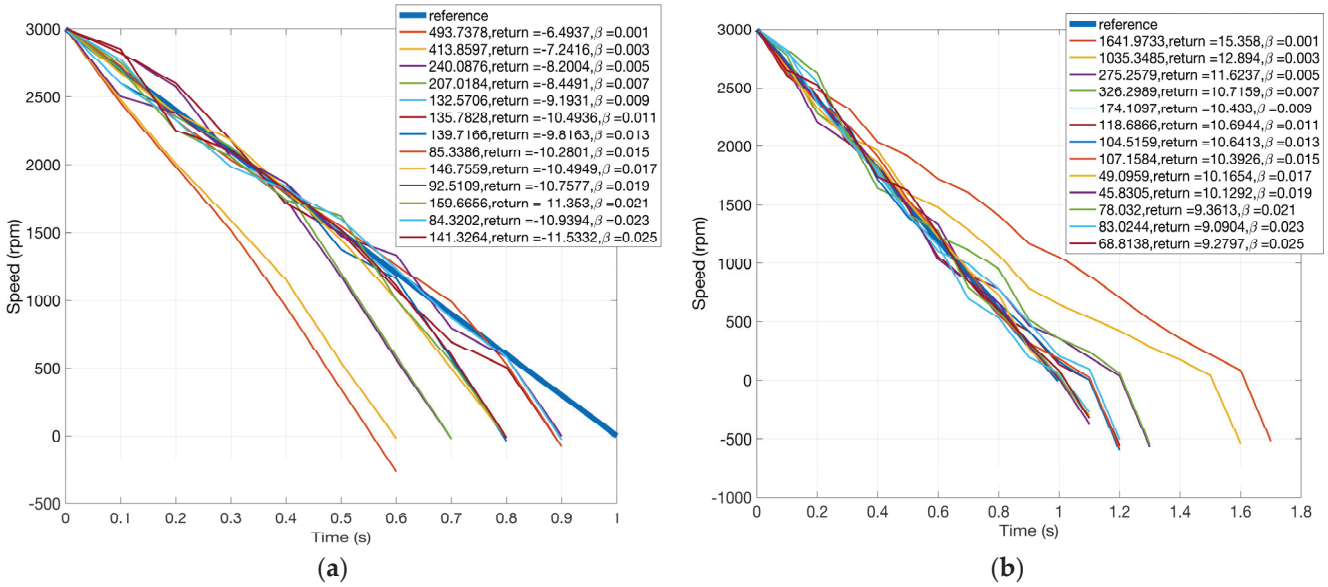


Figure 14. Simulation results: motor speed synchronization control with two constraints (objective: shortest duration; constraints: $c_1 = -500$ rad/s and $c_2 = 4$ turns).

Last but not least, the results of Q-sorting are compared with the method of lumped performances (LP), which integrates constraints into the objective function and turns the original problem into a constraint-free one. LP is, in effect, the Lagrangian relaxation with a fixed multiplier. Here, only one constraint is considered, namely, the deviation from the reference. Specifically, the reward signal is modified as follows:

$$R_{t+1,lumped} = \begin{cases} -1 - \beta\left|\omega_{t+1,ref} - \omega_{t+1,act}\right|, & \text{objective: shortest duration} \\ 1 - \beta\left|\omega_{t+1,ref} - \omega_{t+1,act}\right|, & \text{objective: longest duration} \end{cases} \tag{7}$$

The effects of the original constraint are controlled by the parameter β. Figure 15 shows learned speed trajectories for different values of β, ranging from 0.001, 0.003, . . ., to 0.025. For each trajectory, the deviation from the reference, the return, and the value of β are shown, respectively. Obviously, the smaller the β, the smaller the effects of the constraint, and the larger the deviation from the reference. Figure 15 coincides with intuition. It is also observed that speed trajectories resulting from different values of β exhibit similar shapes to those in Figure 10.



(**a**) (**b**)

Figure 15. Motor speed trajectories after learning for LP (one constraint). (**a**) Objective: shortest duration and (**b**) objective: longest duration.

The method of LP comes with two main drawbacks. First, the relationship between the effects of the constraints and the value of β is unclear. For example, one cannot easily determine the value of β to express the requirement that the cumulated speed deviation should be below 300 rad/s. To attain a proper value of β, lots of trials and experiments are needed. Comparatively, in Q-sorting, the constraint is fed directly into the algorithm; no other proxy parameters are needed.

The second drawback is related to the performance consistency of the algorithm. Figure 16 shows motor speed trajectories for Q-sorting and LP, simulations of which are run repeatedly for 10 times each. The objective for both is to decrease the motor speed to negative values with as long a duration as possible. For Q-sorting, the cumulated speed deviation from the reference is required to be within 300 rad/s. For LP, it is controlled through the proxy parameter β, whose value is fixed to 0.005. Here, the value of β is determined from Figure 15b, where the cumulated speed deviation (the constraint) is 275.2579 rad/s when $\beta = 0.005$. The idea is to choose a value that results in a cumulated speed deviation near 300 rad/s.

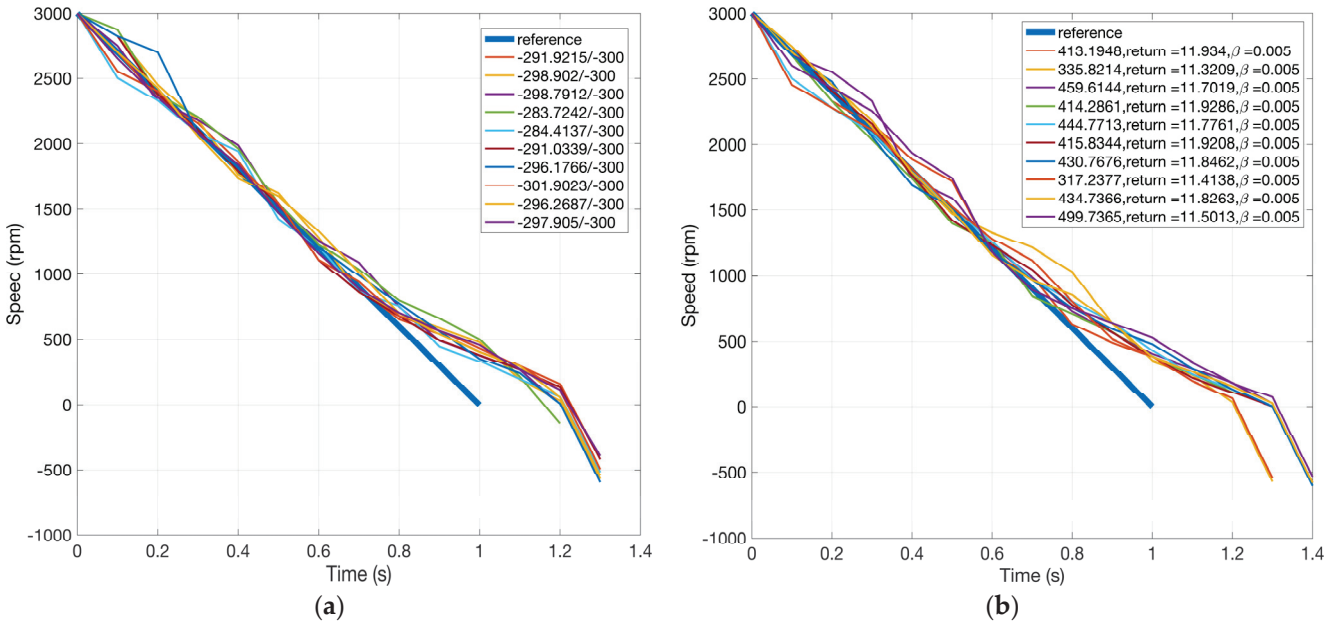


Figure 16. Motor speed trajectories for Q-sorting and LP over 10 runs (objective: longest duration). (**a**) Q-sorting and (**b**) LP.

Figure 17 shows the cumulated speed deviation for both methods in different simulation instances. Q-sorting provides great consistency, with the cumulative performance index concentrating around 300 rad/s, most of the time below it, just as the constraint requires. Comparatively, it ranges from 300 to 500 rad/s in the cases of LP, which implies that there is no deterministic relationship between the values of β and the cumulative performance index of the constraint. One cannot count on the fixed value of β for the satisfaction of a certain cumulative constraint. For reference, the standard deviations of the cumulated speed deviation for Q-sorting and LP over 10 repeated simulation runs are 6.1920 and 54.2635 rad/s, respectively.

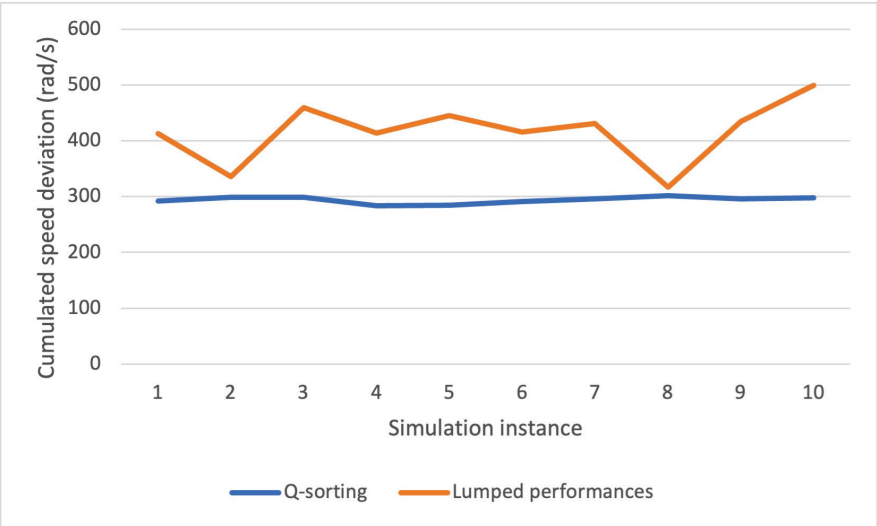


Figure 17. Cumulated speed deviation over 10 runs.

In a word, compared to the conventional LP, Q-sorting not only provides greater ease of use by requiring only the constraint thresholds rather than trials and errors on the values of proxy parameters but also ensures better performance consistency and is thus more suitable for practical use.

5. Conclusions

An algorithm named Q-sorting for RL problems with multiple cumulative constraints is proposed. The core is a mechanism that dynamically determines the focus of optimization among different constraints/objective, at each step of learning. The focus and the action are determined through filtering and sorting of the Q table, which gives it the name Q-sorting. It is a plugin that can be readily applied to any value-based RL algorithm to provide the capability of satisfying cumulative constraints while pursuing optimality. It is verified with two adapted problems, namely, Gridworld and the motor speed synchronization control, each with one or two cumulative constraints. Simulation results show that the proposed method is able to learn an optimal policy that honors all cumulative constraints both during and after the learning process. This makes it suitable for safety-critical applications.

It has to be emphasized that although the idea of Q-sorting is effective, its performance heavily depends on the accuracy of Q values. That is because the algorithm uses $Q_{constraint}(S_t, a)$ to predict whether a specific action violates the constraint. An implementation developed in MATLAB using Monte Carlo to learn the value function is provided in the supplementary materials. Other tabular methods, such as Q-learning and SARSA, are also possible, but performances may differ.

This paper restricts the scenario to finite-time, episodic problems with deterministic environments and policies. Under this assumption, a determined policy with the same initial state will always result in the same cumulative performance index, so there is no need to express the cumulative constraints as expected/averaged values over multiple episodes. In problems with stochastic environments and policies, however, cumulative constraints can only be represented in an expected/averaged manner. Also, for problems with a discounted rather than episodic setting, it is sometimes desired to limit the average resources consumed on each step rather than the cumulated quantities. How to extend the idea of Q-sorting to the two cases above can be a future topic.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/math12132001/s1. Source code and demos written in MATLAB are provided in the Supplementary Materials, which can be downloaded alongside the article.

Author Contributions: Conceptualization, J.H. and G.L.; methodology, J.H.; software, G.L. and Y.L.; validation, G.L. and Y.L.; formal analysis, J.H.; writing—original draft preparation, J.H.; writing—review and editing, J.W.; visualization, G.L. and Y.L.; supervision, J.W.; project administration, J.W.; funding acquisition, J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the STU Scientific Research Initiation Grant (grant number: NTF23037).

Data Availability Statement: Data are contained within the article.

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

References

1. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2018; ISBN 978-0-262-19398-6.
2. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *Nature* **2013**, *518*, 529–533. [CrossRef] [PubMed]
3. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature* **2016**, *529*, 484. [CrossRef] [PubMed]
4. Geibel, P. *Reinforcement Learning for MDPs with Constraints*; Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence Lecture Notes Bioinformatics); Springer: Berlin/Heidelberg, Germany, 2006; Volume 4212, pp. 646–653. [CrossRef]
5. Julian, D.; Chiang, M.; O'Neill, D.; Boyd, S. QoS and Fairness Constrained Convex Optimization of Resource Allocation for Wireless Cellular and Ad Hoc Networks. In Proceedings of the Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies, New York, NY, USA, 23–27 June 2002; IEEE: Piscataway, NJ, USA, 2002; Volume 2, pp. 477–486.
6. Yuan, J.; Yang, L. Predictive Energy Management Strategy for Connected 48V Hybrid Electric Vehicles. *Energy* **2019**, *187*, 115952. [CrossRef]

7. Zhang, R.; Xiong, K.; Lu, Y.; Fan, P.; Ng, D.W.K.; Letaief, K.B. Energy Efficiency Maximization in RIS-Assisted SWIPT Networks with RSMA: A PPO-Based Approach. *IEEE J. Sel. Areas Commun.* **2023**, *41*, 1413–1430. [CrossRef]
8. Zhang, R.; Xiong, K.; Lu, Y.; Gao, B.; Fan, P.; Letaief, K. Ben Joint Coordinated Beamforming and Power Splitting Ratio Optimization in MU-MISO SWIPT-Enabled HetNets: A Multi-Agent DDQN-Based Approach. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 677–693. [CrossRef]
9. Liu, Y.; Halev, A.; Liu, X. Policy Learning with Constraints in Model-Free Reinforcement Learning: A Survey. In Proceedings of the IJCAI International Joint Conference on Artificial Intelligence, Montreal, QC, Canada, 19–27 August 2021; pp. 4508–4515. [CrossRef]
10. Altman, E. *Constrained Markov Decision Processes*; Routledge: Oxfordshire, UK, 1999; ISBN 1315140225.
11. Chow, Y.; Ghavamzadeh, M.; Janson, L.; Pavone, M. Risk-Constrained Reinforcement Learning with Percentile Risk Criteria. *J. Mach. Learn. Res.* **2018**, *18*, 6070–6120.
12. Tessler, C.; Mankowitz, D.J.; Mannor, S. Reward Constrained Policy Optimization. In Proceedings of the 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, 6–9 May 2019; pp. 1–15.
13. Bohez, S.; Abdolmaleki, A.; Neunert, M.; Buchli, J.; Heess, N.; Hadsell, R. Value Constrained Model-Free Continuous Control. *arXiv* **2019**, arXiv:1902.04623.
14. Jayant, A.K.; Bhatnagar, S. Model-Based Safe Deep Reinforcement Learning via a Constrained Proximal Policy Optimization Algorithm. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 24432–24445.
15. Panageas, I.; Piliouras, G.; Wang, X. First-Order Methods Almost Always Avoid Saddle Points: The Case of Vanishing Step-Sizes. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 6474–6483.
16. Vidyasagar, M. *Nonlinear Systems Analysis*; SIAM: Philadelphia, PA, USA, 2002; ISBN 0898715261.
17. Glynn, P.W.; Zeevi, A. *Bounding Stationary Expectations of Markov Processes*; Institute of Mathematical Statistics: Waite Hill, OH, USA, 2008; Volume 4, pp. 195–214. [CrossRef]
18. Chow, Y.; Nachum, O.; Duenez-Guzman, E.; Ghavamzadeh, M. A Lyapunov-Based Approach to Safe Reinforcement Learning. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 8092–8101.
19. Chow, Y.; Nachum, O.; Faust, A.; Duenez-Guzman, E.; Ghavamzadeh, M. Lyapunov-Based Safe Policy Optimization for Continuous Control. *arXiv* **2019**, arXiv:1901.10031.
20. Satija, H.; Amortila, P.; Pineau, J. Constrained Markov Decision Processes via Backward Value Functions. In Proceedings of the 37th International Conference on Machine Learning, ICML 2020, Virtual, 13–18 July 2020; pp. 8460–8469.
21. Achiam, J.; Held, D.; Tamar, A.; Abbeel, P. Constrained Policy Optimization. In Proceedings of the International Conference on Machine Learning; PMLR, Sydney, Australia, 6–11 August 2017; pp. 22–31.
22. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust Region Policy Optimization. In Proceedings of the International Conference on Machine Learning, PMLR, Lille, France, 7–9 July 2015; Volume 67, pp. 1889–1897.
23. Liu, Y.; Ding, J.; Liu, X. IPO: Interior-Point Policy Optimization under Constraints. In Proceedings of the AAAI 2020-34th AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 4940–4947. [CrossRef]
24. Boyd, S.P.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004; ISBN 0521833787.
25. Liu, Y.; Ding, J.; Liu, X. A Constrained Reinforcement Learning Based Approach for Network Slicing. In Proceedings of the 2020 IEEE 28th International Conference on Network Protocols (ICNP), Madrid, Spain, 13–16 October 2020. [CrossRef]
26. Liu, Y.; Ding, J.; Liu, X. Resource Allocation Method for Network Slicing Using Constrained Reinforcement Learning. In Proceedings of the 2021 IFIP Networking Conference (IFIP Networking), Espoo and Helsinki, Finland, 21–24 June 2021; pp. 1–3. [CrossRef]
27. Wei, H.; Liu, X.; Ying, L. Triple-Q: A Model-Free Algorithm for Constrained Reinforcement Learning with Sublinear Regret and Zero Constraint Violation. *Proc. Mach. Learn. Res.* **2022**, *151*, 3274–3307.
28. Rummery, G.; Niranjan, M. *On-Line Q-Learning Using Connectionist Systems (Technical Report)*; University of Cambridge, Department of Engineering Cambridge: Cambridge, UK, 1994; Volume 37.
29. Wei, C.Y.; Jafarnia-Jahromi, M.; Luo, H.; Sharma, H.; Jain, R. Model-Free Reinforcement Learning in Infinite-Horizon Average-Reward Markov Decision Processes. In Proceedings of the 37th International Conference on Machine Learning ICML 2020, Virtual, 13–18 July 2020; pp. 10101–10111.
30. Singh, R.; Gupta, A.; Shroff, N.B. Learning in Constrained Markov Decision Processes. *IEEE Trans. Control Netw. Syst.* **2023**, *10*, 441–453. [CrossRef]
31. Bura, A.; HasanzadeZonuzy, A.; Kalathil, D.; Shakkottai, S.; Chamberland, J.F. DOPE: Doubly Optimistic and Pessimistic Exploration for Safe Reinforcement Learning. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 1047–1059.
32. Yang, T.Y.; Rosca, J.; Narasimhan, K.; Ramadge, P.J. Projection-Based Constrained Policy Optimization. In Proceedings of the 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, 26–30 April 2020; pp. 1–24.
33. Morimura, T.; Peters, J. Derivatives of Logarithmic Stationary Distributions for Policy Gradient Reinforcement Learning. *Neural Comput.* **2010**, *22*, 342–376. [CrossRef] [PubMed]
34. Pankayaraj, P.; Varakantham, P. Constrained Reinforcement Learning in Hard Exploration Problems. In Proceedings of the 37th AAAI Conference on Artificial Intelligence, AAAI 2023, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 15055–15063. [CrossRef]

35. Calvo-Fullana, M.; Paternain, S.; Chamon, L.F.O.; Ribeiro, A. State Augmented Constrained Reinforcement Learning: Overcoming the Limitations of Learning with Rewards. *IEEE Trans. Automat. Control*, 2023; *early access*. [CrossRef]
36. McMahan, J.; Zhu, X. Anytime-Constrained Reinforcement Learning. *Proc. Mach. Learn. Res.* **2024**, *238*, 4321–4329.
37. Bai, Q.; Bedi, A.S.; Agarwal, M.; Koppel, A.; Aggarwal, V. Achieving Zero Constraint Violation for Constrained Reinforcement Learning via Primal-Dual Approach. In Proceedings of the 36th AAAI Conference on Artificial Intelligence, AAAI 2022, Virtually, 22 February–1 March 2022; Volume 36, pp. 3682–3689. [CrossRef]
38. Ma, Y.J.; Shen, A.; Bastani, O.; Jayaraman, D. Conservative and Adaptive Penalty for Model-Based Safe Reinforcement Learning. In Proceedings of the 36th AAAI Conference on Artificial Intelligence, AAAI 2022, Virtually, 22 February–1 March 2022; Volume 36, pp. 5404–5412. [CrossRef]
39. Xu, H.; Zhan, X.; Zhu, X. Constraints Penalized Q-Learning for Safe Offline Reinforcement Learning. In Proceedings of the 36th AAAI Conference on Artificial Intelligence, AAAI 2022, Virtually, 22 February–1 March 2022; Volume 36, pp. 8753–8760. [CrossRef]
40. Huang, J.; Zhang, J.; Huang, W.; Yin, C. Optimal Speed Synchronization Control with Disturbance Compensation for an Integrated Motor-Transmission Powertrain System. *J. Dyn. Syst. Meas. Control* **2018**, *141*, 041001. [CrossRef]
41. Huang, J.; Zhang, J.; Yin, C. Comparative Study of Motor Speed Synchronization Control for an Integrated Motor–Transmission Powertrain System. *Proc. Inst. Mech. Eng. Part D J. Automob. Eng.* **2020**, *234*, 1137–1152. [CrossRef]

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

mathematics

MDPI

Article

Two Schemes of Impulsive Runge–Kutta Methods for Linear Differential Equations with Delayed Impulses

Gui-Lai Zhang * and Chao Liu

College of Sciences, Northeastern University, Shenyang 110819, China; liuchao@neuq.edu.cn
* Correspondence: zhangguilai@neuq.edu.cn

Abstract: In this paper, two different schemes of impulsive Runge–Kutta methods are constructed for a class of linear differential equations with delayed impulses. One scheme is convergent of order p if the corresponding Runge–Kutta method is p order. Another one in the general case is only convergent of order 1, but it is more concise and may suit for more complex differential equations with delayed impulses. Moreover, asymptotical stability conditions for the exact solution and numerical solutions are obtained, respectively. Finally, some numerical examples are provided to confirm the theoretical results.

Keywords: impulsive Runge–Kutta methods; impulsive θ methods; convergence; asymptotical stability

MSC: 37M05; 37M22

1. Introduction

Impulsive differential equations (IDEs) arise widely in numerous mathematics models of systems with instantaneous perturbations. Such models have been applied with huge success in lots of application fields, such as control theory, medicine, biotechnology, economics, population growth, etc. Some work on these systems is presented in [1–4]. Recently, more and more experts and scholars have begun to pay attention to different kinds of differential equations with delayed impulses (DEDIs) and have achieved many important results about the following equations: nonlinear ordinary differential equations with delayed impulses (see [4–8], etc.), time-delay differential equations with delayed impulses [4,9–14], and stochastic differential equations with delayed impulses [15–20]. In the present paper, asymptotical stability conditions for the zero solution of a class of linear DEDIs are obtained, and two interesting examples are provided to reveal the effect of delayed impulses on differential equations, which can potentially destabilize a stable system or stabilize an unstable one.

Recently, the theory of numerical methods for impulsive differential equations has also been developing rapidly. The convergence and stability of impulsive Runge–Kutta methods for scalar linear IDEs [21,22], multidimensional linear IDEs [23], semilinear IDEs [24], nonlinear IDEs [25–29], impulsive time-delay differential equations [30–35], and stochastic impulsive time-delay differential equations [36] have been studied, respectively. There is a lot of important and relevant literature (see [37–42], etc.). However, to our knowledge, most of the previous literature focused on numerical methods for IDEs or impulsive time-delay differential equations; the research on numerical methods for DEDIs is still lacking.

The rest of this paper is organized as follows. In Section 2, asymptotical stability conditions for the zero solution of a class of linear DEDIs and two examples are given to show how delayed impulsive actions influence the stability of the zero solution of the equations. In Section 3, the scheme 1 impulsive Runge–Kutta methods (S1IRKMs) are

constructed, and the convergence and asymptotical stability of the methods are studied. Moreover, the asymptotical stability of the scheme 1 impulsive θ method (S1IθM) is studied. In Section 4, the scheme 2 impulsive Runge–Kutta methods (S2IRKMs) are constructed based on classical Runge–Kutta methods, and the convergence and stability of the methods are studied. In general, a S2IRKM is only convergent of order 1. Therefore, it is very necessary to consider the scheme 2 impulsive θ method (S2IθM). Moreover, the asymptotical stability of S2IθMs is studied. In Section 5, we provide some numerical examples to confirm our theoretical results. Finally, in Section 6, conclusions and future work are provided.

2. Asymptotical Stability of DEDIs

In this section, not only are the asymptotical stability conditions for the zero solution of DEDIs obtained but also two examples are given to illustrate that delayed impulses can change a previously unstable problem into a stable one or a previously stable problem into an unstable one.

2.1. Asymptotical Stability of DEDIs

In this paper, we consider the impulsive differential equation:

$$\begin{cases} x'(t) = ax(t), & t \geq 0, t \neq k\tau, k \in \mathbb{Z}^+ = \{1, 2, \cdots\}, \\ x(k\tau^+) = bx(r_k), & k \in \mathbb{Z}^+, \\ x(0+0) = x_0, \end{cases} \tag{1}$$

where $a \neq 0$, b, x_0, τ, and r_k are real constants, $b \neq 0$, $k \in \mathbb{Z}^+$. There is a real constant $\sigma \in (0, 1]$, such that all r_k, $k \in \mathbb{Z}^+$ satisfy

$$r_k = \sigma k\tau + (1 - \sigma)(k - 1)\tau.$$

Definition 1. *$x(t)$ is said to be the solution of* (1) *if*

1. $\lim_{t \to 0^+} x(t) = x(0+0) = x_0$,
2. *For $t \in (0, +\infty)$, $t \neq k\tau$, $x(t)$ is differentiable and satisfies $x'(t) = ax(t)$,*
3. *$x(t)$ is left-continuous in $(0, +\infty)$ and if $t = k\tau$, then $x(k\tau^+) = bx(r_k)$, where $x(k\tau^+) = \lim_{t \to k\tau^+} x(t)$.*

Problem (1) has a unique solution as follows:

$$x(t) = (b\mathrm{e}^{a\sigma\tau})^k x_0 \mathrm{e}^{a(t-k\tau)}, \ \forall t \in (k\tau, (k+1)\tau]. \tag{2}$$

From (2), it is easy to obtain the following theorem.

Theorem 1. *The solution $x(t) \equiv 0$ of* (1) *is asymptotically stable ($x(t) \to 0$ as $t \to +\infty$) if and only if*

$$|b|\mathrm{e}^{a\sigma\tau} < 1. \tag{3}$$

When $\sigma = 1$, DEDI (1) is changed into an IDE (not delayed). Consequently, when $\sigma = 1$, the necessary and sufficient condition (3) for the asymptotical stability of DEDI (1) (Theorem 1) is changed into the special case of the necessary and sufficient condition for the asymptotical stability of an IDE (not delayed). Hence the result of ([21] Theorem 1.4) is the special case of Theorem 1 of the present paper. The difference between the asymptotical stability of the DEDI and the asymptotical stability of the IDE also can be seen from the following two examples.

2.2. Two Interesting Examples

In this subsection, we present two differential equations with delayed impulses which are interesting and offer simple examples to show how the delayed impulsive actions influence the stability of the zero solution of the equations. In Example 1, the zero solutions

of an ordinary differential equation and an impulsive differential equation are unstable, but the same equation with delayed impulses is asymptotically stable. Conversely, in Example 2, the zero solutions of an ordinary differential equation and an impulsive differential equation are asymptotically stable, but the same equation with delayed impulses is unstable.

Example 1. *First, consider the following simple scalar ordinary differential equation:*

$$x'(t) = x(t),\ t \geq 0,\ x(0) = x_0. \tag{4}$$

Solving this equation, we can obtain the exact solution of (4),

$$x(t) = x_0 \mathrm{e}^t,\ t \geq 0,$$

which implies that when $x_0 \neq 0$,

$$\lim_{t \to +\infty} x(t) = +\infty,$$

which also implies that the zero solution of (4) *is unstable.*

Second, consider the same equation with impulses (not delayed):

$$\begin{cases} x'(t) = x(t), & t \geq 0, t \neq k, k \in \mathbb{Z}^+, \\ x(k^+) = (\frac{1}{2})x(k), & k \in \mathbb{Z}^+, \\ x(0+0) = x_0. \end{cases} \tag{5}$$

Solving this equation, we can obtain the exact solution of (5),

$$x(t) = (0.5\mathrm{e})^k x_0 \mathrm{e}^{t-k},\ \forall t \in (k, k+1].$$

which implies that when $x_0 \neq 0$,

$$\lim_{t \to +\infty} x(t) = +\infty,$$

which also implies that the zero solution of (5) *is unstable.*

Finally, consider the same differential equation with delayed impulses:

$$\begin{cases} x'(t) = x(t), & t \geq 0, t \neq k, k \in \mathbb{Z}^+, \\ x(k^+) = (\frac{1}{2})x(k - \frac{9}{10}), & k \in \mathbb{Z}^+, \\ x(0+0) = x_0. \end{cases} \tag{6}$$

Solving this equation, we can obtain the exact solution of (6),

$$x(t) = \left(0.5\mathrm{e}^{\frac{1}{10}}\right)^k x_0 \mathrm{e}^{t-k},\ \forall t \in (k, k+1].$$

By Theorem 1 of the present paper, we can obtain that the zero solution of (6) *is asymptotically stable (see Figure 1).*

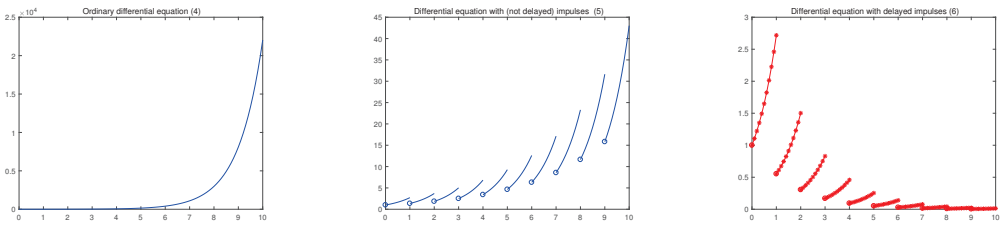


Figure 1. The exact solutions of an ordinary differential Equation (4), a differential equation with (not-delayed) impulses (5), and a differential equation with delayed impulses (6) when $x_0 = 1$.

Example 2. *First, consider the following simple scalar ordinary differential equation:*

$$x'(t) = -x(t),\ t \geq 0,\ x(0) = x_0. \tag{7}$$

Solving this equation, we can obtain the exact solution of (7)*,*

$$x(t) = x_0 \mathrm{e}^{-t},\ t \geq 0.$$

Obviously, the zero solution of (7) *is asymptotically stable.*
Second, consider the same differential equation with impulses (not delayed):

$$\begin{cases} x'(t) = -x(t), & t \geq 0, t \neq k, k \in \mathbb{Z}^+, \\ x(k^+) = 2x(k), & k \in \mathbb{Z}^+, \\ x(0+0) = x_0. \end{cases} \tag{8}$$

Solving this equation, we can obtain the exact solution of (5)*,*

$$x(t) = \left(\frac{2}{\mathrm{e}}\right)^k x_0 \mathrm{e}^{t-k},\ \forall t \in (k, k+1].$$

By ([21] Theorem 1.4), we can obtain that the zero solution of (5) *is asymptotically stable.*
Finally, consider the following differential equation with delayed impulses:

$$\begin{cases} x'(t) = -x(t), & t \geq 0, t \neq k, k \in \mathbb{Z}^+, \\ x(k^+) = 2x(k - \frac{9}{10}), & k \in \mathbb{Z}^+, \\ x(0+0) = x_0. \end{cases} \tag{9}$$

Solving this equation, we can obtain the exact solution of (6)*,*

$$x(t) = \left(2\mathrm{e}^{-\frac{1}{10}}\right)^k x_0 \mathrm{e}^{t-k},\ \forall t \in (k, k+1].$$

which implies that when $x_0 \neq 0$*,*

$$\lim_{t \to +\infty} x(t) = +\infty,$$

which implies that the zero solution of (9) *is unstable (see Figure 2).*

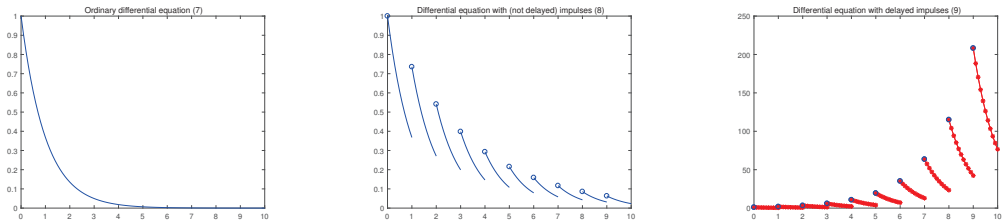


Figure 2. The exact solutions of an ordinary differential Equation (7), a differential equation with (not-delayed) impulses (8), and a differential equation with delayed impulses (9) when $x_0 = 1$.

3. S1IRKM for (1)

The special case of $\sigma = 1$ has already been studied in paper [21], and below we focus on the case of $0 < \sigma < 1$. All the points in the set $S = \{k\tau, r_k : k \in \mathbb{Z}^+\}$ are chosen as the numerical mesh. For convenience, we divide the intervals $[(k-1)\tau, r_k]$ and $[r_k, k\tau]$ $(k \in \mathbb{Z}^+)$ equally by m, where m is a positive integer, respectively. That means the step sizes are as follows for $\forall k \in \mathbb{N} = \{0, 1, 2, \cdots\}$:

$$h_{k,l} = \begin{cases} \bar{h}_1 := \frac{r_k-(k-1)\tau}{m} = \frac{\sigma\tau}{m}, & l = 1, 2, \cdots, m, \\ \bar{h}_2 := \frac{k\tau-r_k}{m} = \frac{(1-\sigma)\tau}{m}, & l = m+1, m+2, \cdots, 2m. \end{cases}$$

The mesh point $t_{k,0} = k\tau$, $t_{k,l} = k\tau + \sum_{j=0}^{l} h_{k,j}$, $\forall k \in \mathbb{N}$, $l = 1, 2, \cdots, 2m$.

$$\begin{cases} x_{k,l+1} = x_{k,l} + ah_{k,l+1} \sum_{i=1}^{v} b_i X_{k,l+1}^i, & l = 0, 1, \cdots, 2m-1, \\ X_{k,l+1}^i = x_{k,l} + ah_{k,l+1} \sum_{j=1}^{v} a_{ij} X_{k,l+1}^j, & i = 1, 2, \cdots, v, \\ x_{k+1,0} = bx_{k,m}, & k \in \mathbb{N}, \\ x_{0,0} = x_0, \end{cases} \tag{10}$$

where v refers to the number of stages. The weights b_i, the abscissae $c_i = \sum_{j=1}^{v} a_{ij}$, and the matrix $A = [a_{ij}]_{j=1}^{v}$ are denoted by (A, b, c). We denote the approximation to the solution $x(t_{k,l})$, $x(r_k)$, and $x(k\tau + 0)$ by $x_{k,l}$ ($l = 1, 2, \cdots, 2m$), $x_{k,m}$, and $x_{k,0}$, respectively.

Equation (10) can be written as

$$\begin{cases} x_{k,l+1} = \left(1 + z_1 b^T (I - z_1 A)^{-1} e\right) x_{k,l} = R(z_1) x_{k,l}, & l = 0, 1, \cdots, m-1, \\ x_{k,l+1} = \left(1 + z_2 b^T (I - z_2 A)^{-1} e\right) x_{k,l} = R(z_2) x_{k,l}, & l = m, \cdots, 2m-1, \\ x_{k+1,0} = bx_{k,m}, & k \in \mathbb{Z}^+, \\ x_{0,0} = x_0, \end{cases} \tag{11}$$

where $R(z) = 1 + zb^T(I - zA)^{-1}e$, $z_1 = a\bar{h}_1 = \frac{a\sigma\tau}{m}$ and $z_2 = a\bar{h}_2 = \frac{a(1-\sigma)\tau}{m}$.

3.1. Asymptotical Stability of S1IRKMs

Theorem 2. *Assume the condition* (3) *holds, and the stability function of the Runge–Kutta method is* $R(z) = \frac{Q_r(z)}{P_s(z)}$, *which is given by the* (r, s)*-Padé approximation to* e^z, $|z| < 1$ *for* $z = z_1$, *and* $z = z_2$:

(i) *if* $a > 0$ *and* s *is even, then S1IRKM* (10) *for* (1) *is asymptotically stable,*

(ii) *if* $a < 0$ *and* r *is odd, then S1IRKM* (10) *for* (1) *is asymptotically stable.*

Proof. From scheme (11), we can obtain that for $\forall k \in \mathbb{N}$, $0 \le l \le m$,

$$x_{k,l} = \left(b(R(z_1))^m\right)^k x_0 (R(z_1))^l,$$

and for $\forall k \in \mathbb{N}$, $m+1 \le l \le 2m$,

$$x_{k,l} = \left(b(R(z_1))^m\right)^k x_0 (R(z_1))^m (R(z_2))^{l-m}.$$

Hence, the numerical method (11) is asymptotically stable if and only if

$$|b(R(z_1))^m| < 1. \tag{12}$$

(i) If $a > 0$ and s is even, applying ([21] Lemmas 3.3 and 3.7) and the condition (3), we can obtain

$$|b|(R(z_1))^m \le |b|e^{z_1 m} = |b|e^{a\bar{h}_1 m} = |b|e^{a\sigma\tau} < 1,$$

which implies that (12) holds.

(ii) Similarly, if $a < 0$ and r is odd, applying ([21] Lemmas 3.3 and 3.7) and the condition (3), we can obtain

$$|b|(R(z_1))^m \le |b|e^{z_1 m} = |b|e^{a\bar{h}_1 m} = |b|e^{a\sigma\tau} < 1,$$

which implies that (12) holds. □

3.2. Convergence of S1RKM

In order to study the convergence of an S1RKM, the case where DEDI (1) is defined in the interval $[0, T]$ is considered in this subsection. For convenience, assume that there exists a positive integer N such that $T = N\tau$.

Lemma 1 ([21,43–45]). *There exists a unique (r, s)-Padé approximation $R_{rs}(z) = \frac{Q_r(z)}{P_s(z)}$ to e^z for $(r, s) \in \mathbb{N} \times \mathbb{N}$. Furthermore,*

$$e^z P_s(z) - Q_r(z) = \frac{(-1)^s z^{r+s+1}}{(r+s)!} \int_0^1 u^s (1-u)^r \mathrm{e}^{uz} du,$$

where

$$Q_r(z) = \frac{r!}{(r+s)!} \sum_{j=0}^{r} \frac{(r+s-j)!}{j!(r-j)!} z^j,$$

$$P_s(z) = \frac{s!}{(r+s)!} \sum_{j=0}^{s} \frac{(r+s-j)!}{j!(s-j)!} (-z)^j.$$

In order to analyze the local truncation errors of S1IRKM (10) for DEDI (1), consider the following problem:

$$\begin{cases} z_{k,l+1} = z_{k,l} + a h_{k,l+1} \sum\limits_{i=1}^{v} b_i Z_{k,l+1}^i, & l = 0, 1, \cdots, 2m-1, \\ Z_{k,l+1}^i = z_{k,l} + a h_{k,l+1} \sum\limits_{j=1}^{v} a_{ij} Z_{k,l+1}^j, & i = 1, 2, \cdots, v, \end{cases} \tag{13}$$

where $z_{k,0} = x(k\tau^+)$, $z_{k,l} = x(t_{k,l})$, $k = 0, 1, 2, \cdot, N$, $l = 1, 2, \cdot, 2m-1$.

Theorem 3. *If the corresponding Runge–Kutta method is convergent of order p, then the local truncation errors between* (13) *and DEDI* (1) *satisfy that there exists a constant C such that for arbitrary $k = 0, 1, 2, \cdot, N$, $l = 1, 2, \cdot, 2m-1$,*

$$R_{k,l+1} := |z_{k,l+1} - x(t_{k,l+1})| \le C h_{k,l+1}^{p+1}.$$

Proof. Because Runge–Kutta methods are convergent of order p, by Lemma 1, there exists a constant $C_1 > 0$ such that

$$R_{k,l+1} := |\mathrm{e}^{a h_{k,l+1}} - R(a h_{k,l+1})| \le C_1 h_{k,l+1}^{p+1}. \tag{14}$$

Obviously, (13) can be rewritten as

$$z_{k,l+1} = R(a h_{k,l+1}) z_{k,l},$$

where $R_{rs}(a h_{k,l+1}) = (1 + a h_{k,l+1} b^T (I - a h_{k,l+1} A e))$. From the expression (2) for the solution $x(t)$ of DEDI (1), we have

$$\sup_{t \in (0,T]} |x(t)| \le M.$$

Hence, the local errors satisfy

$$R_{k,l} = |x(t_{k,l+1}) - z_{k,l+1}| \le |\mathrm{e}^{a h_{k,l+1}} - R(a h_{k,l+1})||z_{k,l}| \le C h_{k,l+1}^{p+1}.$$

where $C = C_1 M$. □

Theorem 4. *If Runge–Kutta methods are convergent of order p, then S1IRKM* (10) *for* (1) *is also convergent of order p, and in the following sense, there exists a constant C_5 such that for all $k \in \mathbb{N}$, $l = 1, 2, \cdots, m$, the global errors satisfy*

$$e_{k,l} = |x(t_{k,l}) - x_{k,l}| \leq C_5 h^p,$$

where $h = \max\{\bar{h}_1, \bar{h}_2\} = \max\limits_{k,l}\{h_{k,l}\}$.

Proof. From (1) and (13), we have

$$\begin{aligned} |X^i_{k,l+1} - Z^i_{k,l+1}| &\leq |x_{k,l} - z_{k,l}| + |a| h_{k,l+1} \left(\sum_{j=1}^{v} |a_{ij}| |X^i_{k,l+1} - Z^i_{k,l}| \right) \\ &\leq |x_{k,l} - z_{k,l}| + |a| h \left(\max_{1\leq i\leq v} \sum_{j=1}^{v} |a_{ij}| \right) \max_{1\leq i\leq v} \{|X^i_{k,l+1} - Z^i_{k,l}|\} \end{aligned}$$

which implies

$$\max_{1\leq i\leq v} \{|X^i_{k,l+1} - Z^i_{k,l}|\} \leq \Lambda |x_{k,l} - z_{k,l}|$$

where $\Lambda = \left(1 - |a| h \left(\max_{1\leq i\leq v} \sum_{j=1}^{v} |a_{ij}| \right) \right)^{-1}$:

$$\begin{aligned} |x_{k,l+1} - z_{k,l+1}| &\leq |x_{k,l} - z_{k,l}| + |a| h_{k,l+1} \left(\sum_{i=1}^{v} |b_i| \right) \max_{1\leq i\leq v} \{|X^i_{k,l+1} - Z^i_{k,l}|\} \\ &\leq (1 + \beta\Lambda |a| h_{k,l+1}) |x_{k,l} - z_{k,l}| \end{aligned}$$

where $\beta = \left(\sum\limits_{i=1}^{v} |b_i| \right)$. From Theorem 3, we have

$$R_1 := \max_{0\leq k\leq N, 1\leq l\leq m} \{R_{k,l}\} \leq C\bar{h}_1 h^p$$

and

$$R_2 := \max_{0\leq k\leq N, m+1\leq l\leq 2m} \{R_{k,l}\} \leq C\bar{h}_2 h^p.$$

If $0 \leq l \leq m-1$,

$$\begin{aligned} e_{k,l+1} &:= |x(t_{k,l+1}) - x_{k,l+1}| \\ &\leq |x(t_{k,l+1}) - z_{k,l+1}| + |z_{k,l+1} - x_{k,l+1}| \\ &\leq (1 + \beta\Lambda |a| h_{k,l+1}) |x_{k,l} - z_{k,l}| + R_{k,l+1} \\ &\leq (1 + \beta\Lambda |a| \bar{h}_1) e_{k,l} + R_1 \\ &\leq \left(1 + \beta\Lambda |a| \bar{h}_1\right)^{l+1} e_{k,0} + \left[\left(1 + \beta\Lambda |a| \bar{h}_1\right)^{l+1} - 1 \right] \frac{R_1}{\beta\Lambda |a| \bar{h}_1} \\ &\leq \mathrm{e}^{(l+1)\beta\Lambda |a| \bar{h}_1} e_{k,0} + \left(\mathrm{e}^{(l+1)\beta\Lambda |a| \bar{h}_1} - 1 \right) \frac{R_1}{\beta\Lambda |a| \bar{h}_1} \\ &\leq \mathrm{e}^{\beta\Lambda |a| \sigma\tau} e_{k,0} + \left(\mathrm{e}^{\beta\Lambda |a| \sigma\tau} - 1 \right) \frac{R_1}{\beta\Lambda |a| \bar{h}_1} \end{aligned} \tag{15}$$

or else, if $m \le l \le 2m-1$,

$$\begin{aligned} e_{k,l+1} &= |x(t_{k,l+1}) - x_{k,l+1}| \\ &\le |x(t_{k,l+1}) - z_{k,l+1}| + |z_{k,l+1} - x_{k,l+1}| \\ &\le (1+\beta\Lambda|a|h_{k,l+1})|x_{k,l} - z_{k,l}| + R_{k,l+1} \\ &\le (1+\beta\Lambda|a|\bar{h}_2)e_{k,l} + R_2 \\ &\le \left(1+\beta\Lambda|a|\bar{h}_2\right)^{l-m+1} e_{k,m} + \left[\left(1+\beta\Lambda|a|\bar{h}_2\right)^{l-m+1} - 1\right]\frac{R_2}{\beta\Lambda|a|\bar{h}_2} \\ &\le \mathrm{e}^{(l-m+1)\beta\Lambda|a|\bar{h}_2} e_{k,m} + \left(\mathrm{e}^{(l-m+1)\beta\Lambda|a|\bar{h}_2} - 1\right)\frac{R_2}{\beta\Lambda|a|\bar{h}_2} \\ &\le \mathrm{e}^{\beta\Lambda|a|(1-\sigma)\tau} e_{k,0} + \left(\mathrm{e}^{\beta\Lambda|a|(1-\sigma)\tau} - 1\right)\frac{R_2}{\beta\Lambda|a|\bar{h}_2} \end{aligned} \tag{16}$$

otherwise,

$$\begin{aligned} e_{k+1,0} &= |x(t_{k+1,0}) - x_{k+1,0}| \\ &= |bx(r_k) - bx_{k,m}| \le |b| e_{k,m} \\ &\le |b|\mathrm{e}^{\beta\Lambda|a|m\bar{h}_1} e_{k,0} + \left(\mathrm{e}^{\beta\Lambda|a|m\bar{h}_1} - 1\right)\frac{R_1}{\beta\Lambda|a|\bar{h}_1} \\ &= |b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau} e_{k,0} + \left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{R_1}{\beta\Lambda|a|\bar{h}_1} \\ &\le \left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)^{k+1} e_{0,0} + \frac{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)^{k+1}-1}{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)-1}\left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{R_1}{\beta\Lambda|a|\bar{h}_1} \end{aligned} \tag{17}$$

Because $e_{0,0} = 0$, i.e., $x_{0,0} = x(0^+) = x_0$, it follows from (17) that we can obtain that for arbitrary $k = 0, 1, \cdot, N-1$,

$$\begin{aligned} e_{k+1,0} &\le \frac{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)^{k+1}-1}{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)-1}\left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{R_1}{\beta\Lambda|a|\bar{h}_1} \\ &\le \frac{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)^{k+1}-1}{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)-1}\left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{C\bar{h}_1^p}{\beta\Lambda|a|} \\ &\le C_2 h^p, \end{aligned} \tag{18}$$

where $C_2 = \frac{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)^{k+1}-1}{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)-1}\left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{C}{\beta\Lambda|a|}$. From (15) and (18), applying Theorem 3, we can obtain that for arbitrary $k = 0, 1, 2, \cdots, N-1$, $1 \le l \le m$,

$$e_{k,l} \le C_3 h^p, \tag{19}$$

where $C_3 = \mathrm{e}^{\beta\Lambda|a|\sigma\tau} C_2 + \left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{C}{\beta\Lambda|a|}$. Similarly, from (16) and (19), applying Theorem 3, we can obtain that for arbitrary $k = 0, 1, 2, \cdots, N-1$, $m \le l \le 2m$,

$$e_{k,l} \le C_4 h^p, \tag{20}$$

where $C_4 = \mathrm{e}^{\beta\Lambda|a|(1-\sigma)\tau} C_3 + \left(\mathrm{e}^{\beta\Lambda|a|(1-\sigma)\tau} - 1\right)\frac{C}{\beta\Lambda|a|}$.

Finally, summarizing Equations (18)–(20), we know that all the global errors satisfy

$$e_{k,l} \le C_5 h^p, \ \forall k = 0, 1, 2, \cdots, N-1, \ \forall l = 0, 1, 2, \cdots, 2m,$$

where $C_5 = \max\{C_2, C_3, C_4\}$. □

3.3. Asymptotical Stability of S1IθMs

Similarly, the scheme 1 impulsive θ method (S1IθM) for (1) can be constructed as follows:

$$\begin{cases} x_{k,l+1} = x_{k,l} + h_{k,l}(a(1-\theta)x_{k,l} + a\theta x_{k,l+1}), & l = 0, 1, \cdots, 2m-1, \\ x_{k+1,0} = bx_{k,m}, & k \in \mathbb{N}, \\ x_{0,0} = x_0. \end{cases} \tag{21}$$

Obviously, S1IθM (21) can be written as

$$\begin{cases} x_{k,l+1} = \left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)x_{k,l} = \left(\frac{1+(1-\theta)z_1}{1-\theta z_1}\right)x_{k,l}, & l = 0, 1, \cdots, m-1, \\ x_{k,l+1} = \left(\frac{1+(1-\theta)a\bar{h}_2}{1-\theta a\bar{h}_2}\right)x_{k,l} = \left(\frac{1+(1-\theta)z_2}{1-\theta z_2}\right)x_{k,l}, & l = m, \cdots, 2m-1, \\ x_{k+1,0} = bx_{k,m}, & k \in \mathbb{Z}^+, \\ x_{0,0} = x_0. \end{cases} \tag{22}$$

Theorem 5. *Assume the condition* (3) *holds,* $|z_1| < 1$, *and* $|z_2| < 1$*:*

(i) *if* $a > 0$ *and* $0 < \theta < \varphi(1)$, *then the impulsive* θ *method* (21) *for* (1) *is asymptotically stable,*

(ii) *if* $a < 0$ *and* $0 < \theta < \varphi(0)$, *then the impulsive* θ *method* (21) *for* (1) *is asymptotically stable,*

where $\varphi(x) = \frac{1}{x} - \frac{1}{e^x-1}$, $x \in \mathbb{R}$. *(The function of* φ *can be referred to in Lemma 2 in ref. [46]).*

Proof. From scheme (22), we can obtain that for $\forall k \in \mathbb{N}$, $0 \le l \le m$,

$$x_{k,l} = \left(b\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^m\right)^k x_0\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^l,$$

and for $\forall k \in \mathbb{N}$, $m+1 \le l \le 2m$,

$$x_{k,l} = \left(b\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^m\right)^k x_0\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^m\left(\frac{1+(1-\theta)a\bar{h}_2}{1-\theta a\bar{h}_2}\right)^{l-m}.$$

Hence, the numerical method (22) is asymptotically stable if and only if

$$|b\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^m| < 1. \tag{23}$$

(i) If $a > 0$ and $0 < \theta < \varphi(1)$, applying ([21] Lemma 2.3) and the condition (3), we can obtain

$$|b|\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^m \le |b|e^{a\bar{h}_1 m} = |b|e^{a\sigma\tau} < 1,$$

which implies that (33) holds.

(ii) Similarly, if $a < 0$ and $0 < \theta < \varphi(0)$, applying ([21] Lemma 2.3) and the condition (3), we can obtain

$$|b|\left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1}\right)^m \le |b|e^{a\bar{h}_1 m} = |b|e^{a\sigma\tau} < 1,$$

which implies that (33) holds. □

4. S2IRKMs for (1)

For the second scheme, we pay attention to a uniform grid with step size $h = \frac{\tau}{m}$, where m is an integer. So, the formula for the time points is

$$t_{k,l} = k\tau + lh,\ k \in N, l = 0, 1, 2, \cdots, m.$$

The impulsive Runge–Kutta method for (1) can be constructed as follows:

$$\begin{cases} y_{k,l+1} = y_{k,l} + ah \sum\limits_{i=1}^{s} b_i Y^i_{k,l+1}, & l = 0, 1, \cdots, m-1, \\ Y^i_{k,l+1} = y_{k,l} + ah \sum\limits_{j=1}^{s} a_{ij} Y^j_{k,l+1}, & i = 1, 2, \cdots, s, \\ y_{k+1,0} = b y_{k,\lfloor m\sigma \rfloor}, & k \in \mathbb{Z}^+, \\ y_{0,0} = x_0. \end{cases} \tag{24}$$

Here, $y_{k,l}$ is an approximation of the exact solution $x(t_{k,l})$, $\forall k \in \mathbb{N}$, $\forall l = 1, 2, \cdots, m$. $y_{k,0}$ is an approximation of $x(k\tau + 0)$, $\forall k \in \mathbb{Z}^+$. Obviously, if σm is an integer, $x_{k,\lfloor \sigma m \rfloor}$ is an approximation of the exact solution $x(r_k)$. Otherwise, we cannot find the numerical solutions at $t = r_k$. Now, $x_{k,\lfloor \sigma m \rfloor}$, which is an approximation of $x(t_{k,\lfloor \sigma m \rfloor})$ ($t_{k,\lfloor \sigma m \rfloor} \le r_k$ and $|r_k - t_{k,\lfloor \sigma m \rfloor}| \le h$), is viewed as an approximation of $x(r_k)$ to find the numerical solution of (1).

The impulsive Runge–Kutta method (24) can be written as

$$\begin{cases} y_{k,l+1} = \left(1 + zb^T (I - zA)^{-1} e\right) y_{k,l} = R(z) y_{k,l}, & l = 0, 1, \cdots, m-1, \\ y_{k+1,0} = b y_{k,\lfloor m\sigma \rfloor}, & k \in \mathbb{Z}^+, \\ y_{0,0} = x_0, \end{cases} \tag{25}$$

where $z = ha$.

4.1. Asymptotical Stability of Scheme 2 Impulsive Runge–Kutta Methods

Theorem 6. *Assume the condition* (3) *holds, and the stability function of the Runge–Kutta method is* $R(z) = \frac{Q_r(z)}{P_s(z)}$, *which is given by the* (r, s)*-Padé approximation to* e^z, $z = ah$, $|z| < 1$*:*

(i) *if* $a > 0$ *and* s *is even, then the impulsive Runge–Kutta method* (24) *for* (1) *is asymptotically stable,*

(ii) *if* $a < 0$ *and* r *is odd, then the impulsive Runge–Kutta method* (24) *for* (1) *is asymptotically stable when* $h < \frac{1}{a} \ln(|b| e^{a\sigma\tau})$.

Proof. From scheme (25), we can obtain that

$$y_{k,l} = \left(b(R(z))^{\lfloor \sigma m \rfloor}\right)^k y_{0,0} (R(z))^l, \quad \forall k \in \mathbb{N}, l = 0, 1, 2, \cdots, m,$$

which implies that the numerical method (25) is asymptotically stable if and only if

$$|b(R(z))^{\lfloor \sigma m \rfloor}| < 1. \tag{26}$$

(i) If $a > 0$ and s is even,

$$|b|(R(z))^{\lfloor \sigma m \rfloor} \le |b| e^{z \lfloor \sigma m \rfloor} \le |b| e^{ahm\sigma} = |b| e^{a\sigma\tau} < 1,$$

which implies that (26) holds.

(ii) If $a < 0$ and r is odd, $h < \frac{1}{a} \ln(|b| e^{a\sigma\tau})$ implies $|b| e^{a\sigma\tau - ah} < 1$. Hence, we can obtain

$$|b|(R(z))^{\lfloor \sigma m \rfloor} \le |b| e^{z \lfloor \sigma m \rfloor} \le |b| e^{ah(m\sigma - 1)} = |b| e^{a\sigma\tau - ah} < 1,$$

which implies that (26) holds. □

4.2. Convergence of S2IRKMs

In order to study the convergence of S2IRKM (24), the case where DEDI (1) is defined in the interval $[0, T]$ is considered in this subsection. For convenience, assume that there exists a positive integer N such that $T = N\tau$.

To analyze the local truncation errors of S2IRKM (24) for DEDI (1), consider the following problem:

$$\begin{cases} z_{k,l+1} = z_{k,l} + ah \sum\limits_{i=1}^{v} b_i Z^i_{k,l+1}, & l = 0, 1, \cdots, m-1, \\ Z^i_{k,l+1} = z_{k,l} + ah \sum\limits_{j=1}^{v} a_{ij} Z^j_{k,l+1}, & i = 1, 2, \cdots, v, \end{cases} \tag{27}$$

where $z_{k,0} = x(k\tau^+)$, $z_{k,l} = x(t_{k,l})$, $k = 0, 1, 2, \cdot, N$, $l = 1, 2, \cdot, m-1$.

Theorem 7. *If the corresponding Runge–Kutta method is convergent of order p, then the local truncation errors between* (27) *and DEDI* (1) *satisfy that there exists a constant C_6 such that for arbitrary $k = 0, 1, 2, \cdot, N$, $l = 1, 2, \cdot, m-1$,*

$$R_{k,l+1} := |z_{k,l+1} - x(t_{k,l+1})| \le C_6 h^{p+1}.$$

Theorem 8. *If Runge–Kutta methods are convergent of order p, then the impulsive Runge–Kutta methods* (24) *for* (1) *are convergent at least of order* 1*, and in the following sense, there exists a constant C_{10} such that for all $k = 0, 1, 2, \cdots, N-1$, $l = 0, 1, 2, \cdots, m$, the global errors satisfy*

$$e_{k,l} = |x(t_{k,l}) - x_{k,l}| \le C_{10} h.$$

Proof. From (24) and (27), we have

$$\begin{aligned} |X^i_{k,l+1} - Z^i_{k,l+1}| &\le |x_{k,l} - z_{k,l}| + |a|h \left(\sum_{j=1}^{v} |a_{ij}| |X^i_{k,l+1} - Z^i_{k,l}| \right) \\ &\le |x_{k,l} - z_{k,l}| + |a|h \left(\max_{1\le i\le v} \sum_{j=1}^{v} |a_{ij}| \right) \max_{1\le i\le v} \{|X^i_{k,l+1} - Z^i_{k,l}|\} \end{aligned}$$

which implies

$$\max_{1\le i\le v} \{|X^i_{k,l+1} - Z^i_{k,l}|\} \le \Lambda |x_{k,l} - z_{k,l}|$$

where $\Lambda = \left(1 - |a|h\left(\max_{1\le i\le v} \sum_{j=1}^{v} |a_{ij}|\right)\right)^{-1}$. So we can obtain that

$$\begin{aligned} |x_{k,l+1} - z_{k,l+1}| &\le |x_{k,l} - z_{k,l}| + |a|h \left(\sum_{i=1}^{v} |b_i| \right) \max_{1\le i\le v} \{|X^i_{k,l+1} - Z^i_{k,l}|\} \\ &\le (1 + \beta\Lambda|a|h)|x_{k,l} - z_{k,l}|, \end{aligned}$$

where $\beta = \left(\sum\limits_{i=1}^{v} |b_i| \right)$. By Theorem 7, we have

$$R := \max_{k=0,1,\cdots,N-1, l=0,1,\cdots,m} \{R_{k,l}\} \le C_6 h^{p+1}.$$

When $0 \le l \le m-1$,

$$\begin{aligned} e_{k,l+1} &:= |x(t_{k,l+1}) - x_{k,l+1}| \\ &\le |x(t_{k,l+1}) - z_{k,l+1}| + |z_{k,l+1} - x_{k,l+1}| \\ &\le (1 + \beta\Lambda|a|h)|x_{k,l} - z_{k,l}| + R_{k,l+1} \\ &\le (1 + \beta\Lambda|a|h)e_{k,l} + R \\ &\le (1 + \beta\Lambda|a|h)^{l+1} e_{k,0} + \left[\left(1 + \beta\Lambda|a|\bar{h}_1\right)^{l+1} - 1 \right] \frac{R}{\beta\Lambda|a|h} \\ &\le \mathrm{e}^{(l+1)\beta\Lambda|a|h} e_{k,0} + \left(\mathrm{e}^{(l+1)\beta\Lambda|a|h} - 1 \right) \frac{R}{\beta\Lambda|a|h} \\ &\le \mathrm{e}^{\beta\Lambda|a|\tau} e_{k,0} + \left(\mathrm{e}^{\beta\Lambda|a|\tau} - 1 \right) \frac{R}{\beta\Lambda|a|h}. \end{aligned} \tag{28}$$

For $\forall k = 1, 2, \cdots, N$, from Taylor's formula, it follows that

$$x(r_k) - x(t_{k,\lfloor \sigma m \rfloor}) = x'(t_{k,\lfloor \sigma m \rfloor})(r_k - t_{k,\lfloor \sigma m \rfloor}) + \frac{1}{2!}x''(\xi)(r_k - t_{k,\lfloor \sigma m \rfloor})^2$$

which implies that

$$|x(r_k) - x(t_{k,\lfloor \sigma m \rfloor})| \leq C_7 h,$$

which implies that

$$\begin{aligned}
e_{k+1,0} &= |x(k\tau^+) - x_{k+1,0}| \\
&= |bx(r_k) - bx_{k,\lfloor \sigma m \rfloor}| \\
&\leq |b|\Big(|x(r_k) - x(t_{k,\lfloor \sigma m \rfloor})| + |x(t_{k,\lfloor \sigma m \rfloor}) - x_{k,\lfloor \sigma m \rfloor}|\Big) \\
&\leq |b|\mathrm{e}^{\beta\Lambda|a|h\lfloor \sigma m \rfloor} e_{k,0} + |b|\Big(\mathrm{e}^{\beta\Lambda|a|h\lfloor \sigma m \rfloor} - 1\Big)\frac{R}{\beta\Lambda|a|h} + |b|C_7 h \\
&\leq |b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau} e_{k,0} + |b|\Big(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\Big)\frac{R}{\beta\Lambda|a|h} + |b|C_7 h \\
&\leq |b|\left(\frac{\Big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\Big)^{k+1} - 1}{\big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\big) - 1}\right)\left[\Big(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\Big)\frac{R}{\beta\Lambda|a|h} + |b|C_7 h\right] \\
&\quad + \Big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\Big)^{k+1} e_{0,0}.
\end{aligned}$$

Because $e_{0,0} = 0$, i.e., $x_{0,0} = x(0^+) = x_0$, we have

$$\begin{aligned}
e_{k+1,0} &\leq |b|\left(\frac{\Big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\Big)^{k+1} - 1}{\big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\big) - 1}\right)\left[\Big(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\Big)\frac{R}{\beta\Lambda|a|h} + |b|C_7 h\right] \\
&\leq |b|\left(\frac{\Big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\Big)^{k+1} - 1}{\big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\big) - 1}\right)\left[\Big(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\Big)\frac{C_6 h^p}{\beta\Lambda|a|} + |b|C_7 h\right] \\
&\leq |b|\left(\frac{\Big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\Big)^{k+1} - 1}{\big(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\big) - 1}\right)\left[\Big(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\Big)\frac{C_6 T^{p-1} h}{\beta\Lambda|a|} + |b|C_7 h\right] \\
&\leq C_8 h,
\end{aligned} \tag{29}$$

where $C_8 = |b|\left(\frac{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right)^{k+1} - 1}{\left(|b|\mathrm{e}^{\beta\Lambda|a|\sigma\tau}\right) - 1}\right)\left[\left(\mathrm{e}^{\beta\Lambda|a|\sigma\tau} - 1\right)\frac{C_6 T^{p-1}}{\beta\Lambda|a|} + |b|C_7\right]$. From (28) and (29), for $k = 0, 1, \cdots, N-1$, $l = 1, 2, \cdots, m$, we obtain

$$e_{k,l} \leq C_9 h, \tag{30}$$

where $C_9 = \mathrm{e}^{\beta\Lambda|a\tau} C_8 + \left(\mathrm{e}^{\beta\Lambda|a|\tau} - 1\right)\frac{C_6 T^{p-1}}{\beta\Lambda|a|}$.

Finally, we know that all the global errors satisfy

$$e_{k,l} \leq C_{10} h, \ \forall k = 0, 1, 2, \cdots, N-1, \ \forall l = 0, 1, 2, \cdots, 2m,$$

where $C_{10} = \max\{C_8, C_9\}$. □

4.3. Asymptotical Stability of S2IθMs

Similarly, the impulsive θ method for (1) can be constructed as follows:

$$\begin{cases} y_{k,l+1} = y_{k,l} + h(a(1-\theta)y_{k,l} + a\theta y_{k,l+1}), & l = 0, 1, \cdots, m-1, \\ y_{k+1,0} = by_{k,\lfloor \sigma m \rfloor}, & k \in \mathbb{N}, \\ y_{0,0} = x_0, \end{cases} \tag{31}$$

Obviously, the impulsive θ method (31) can be rewritten as

$$\begin{cases} y_{k,l+1} = \left(\frac{1+(1-\theta)ah}{1-\theta ah}\right) y_{k,l}, & l = 0, 1, \cdots, m-1, \\ y_{k+1,0} = b y_{k,\lfloor \sigma m \rfloor}, & k \in \mathbb{N}, \\ y_{0,0} = x_0, \end{cases} \tag{32}$$

Theorem 9. *Assume the condition* (3) *holds and* $|ah| < 1$*:*

(i) *if* $a > 0$ *and* $0 < \theta < \varphi(1)$*, then the impulsive* θ *method* (31) *for* (1) *is asymptotically stable.*

(ii) *if* $a < 0$ *and* $0 < \theta < \varphi(0)$*, then the impulsive* θ *method* (31) *for* (1) *is asymptotically stable when* $h < \frac{1}{a} \ln(|b|e^{a\sigma\tau})$.

In the above, $\varphi(x) = \frac{1}{x} - \frac{1}{e^x - 1}$, $x \in \mathbb{R}$.

Proof. From scheme (32), we can obtain that for $\forall k \in \mathbb{N}$, $0 \le l \le m$,

$$y_{k,l} = \left(b \left(\frac{1+(1-\theta)ah}{1-\theta ah} \right)^{\lfloor \sigma m \rfloor} \right)^k y_{0,0} \left(\frac{1+(1-\theta)ah}{1-\theta ah} \right)^l,$$

Hence, the numerical method (22) is asymptotically stable if and only if

$$|b| \left(\frac{1+(1-\theta)a\bar{h}_1}{1-\theta a\bar{h}_1} \right)^{\lfloor \sigma m \rfloor} < 1. \tag{33}$$

(i) If $a > 0$ and $0 < \theta < \varphi(1)$, applying ([21] Lemma 2.3) and the condition (3), we can obtain

$$|b| \left(\frac{1+(1-\theta)ah}{1-\theta ah} \right)^{\lfloor \sigma m \rfloor} \le |b| e^{ah\lfloor \sigma m \rfloor} \le |b| e^{ah\sigma m} = |b| e^{a\sigma\tau} < 1,$$

which implies that (33) holds.

(ii) Similarly, if $a < 0$ and $0 < \theta < \varphi(0)$, applying ([21] Lemma 2.3) and the condition (3), we can obtain

$$|b| \left(\frac{1+(1-\theta)ah}{1-\theta ah} \right)^{\lfloor \sigma m \rfloor} \le |b| e^{ah\lfloor \sigma m \rfloor} \le |b| e^{ah(\sigma m - 1)} = |b| e^{a\sigma\tau - ah} < 1,$$

which implies that (33) holds. □

5. Numerical Experiments

In this section, two simple numerical examples are given.

Example 3. *Consider the following DEDI:*

$$\begin{cases} x'(t) = 2x(t), & t \ge 0, t \ne k, k \in \mathbb{Z}^+, \\ x(k^+) = (\frac{1}{4}) x(k - \frac{2}{3}), & k \in \mathbb{Z}^+, \\ x(0+0) = x_0. \end{cases} \tag{34}$$

Solving (34)*, we can obtain*

$$x(t) = \left(0.25 e^{\frac{2}{3}}\right)^k x_0 e^{2(t-k)}, \ \forall t \in (k, k+1].$$

By Theorem 1, the zero solution of (34) *is asymptotically stable. By Theorems 2 and 6, both S1IRKM* (10) *and S2IRKM* (24) *for* (34) *are asymptotically stable if the stability function*

$R_{rs}(z) = \frac{P_r(z)}{Q_s(z)}$ *satisfies that s is even. By Theorems 5 and 9, both S1IθM* (21) *and S2IθM* (31) *for* (34) *are asymptotically stable if* $0 < \theta < \varphi(1)$.

In Tables 1–5, AE denotes the absolute errors between the numerical solutions and the exact solutions of DEDIs. Similarly, RE denotes the relative errors between the numerical solutions and the exact solutions of DEDIs.

As can be seen from Table 1, when the step size is halved, both the absolute and relative errors of the scheme 1 impulsive Euler method (S1IEM) and the scheme 2 impulsive Euler method (S2IEM) for DEDI (34) *become half of the original ones, which roughly indicates that both the S1IEM and S2IEM for DEDI* (34) *are convergent of order 1.*

As can be seen from Table 2, when the step size is halved, both the absolute and relative errors of the scheme 1 impulsive classical 4-stage 4-order Runge–Kutta method (S1ICRKM) for DEDI (34) *become one-sixteenth of the original ones, which roughly indicates that he S1ICRKM for DEDI* (34) *is convergent of order 4. On the other hand, when the step size is halved, both the absolute and relative errors of the scheme 2 impulsive classical 4-stage 4-order Runge–Kutta method (S2ICRKM) for DEDI* (34) *become half of the original ones, which roughly indicates that the S2ICRKM for DEDI* (34) *is convergent of order 1.*

Table 1. The errors between the exact solution of DEDI (34) and the numerical solutions obtained from the S1IEM and S2IEM for (34) at $t = 6$.

	S1IEM		**S2IEM**	
m	AE	RE	AE	RE
100	0.00441823021	0.02184293837	0.01660106964	0.08207271326
200	0.00222814668	0.01101555787	0.01171428183	0.05791331011
400	0.00111888657	0.00553157470	0.00432256363	0.02136997995
800	0.00056065360	0.00277177093	0.00300716613	0.01486689045
Ratio	1.98999866598	1.98999866598	1.85487233434	1.85487233434

Table 2. The errors between the exact solution of DEDI (34) and the numerical solutions obtained from the S1ICRKM and S2ICRKM for (34) at $t = 6$.

	S1ICRKM		**S2ICRKM**	
m	AE	RE	AE	RE
100	$8.34915193 \times 10^{-11}$	$4.12767109 \times 10^{-10}$	0.00663129	0.03278391
200	$5.24716381 \times 10^{-12}$	$2.59410376 \times 10^{-11}$	0.00663129	0.03278390
400	$3.28126415 \times 10^{-13}$	$1.62219819 \times 10^{-12}$	0.00167860	0.00829871
800	$2.04836148 \times 10^{-14}$	$1.01267321 \times 10^{-13}$	0.00167860	0.00829871
Ratio	15.97400003	15.97400003	1.98349430	1.98349430

Example 4. *Consider the following DEDI:*

$$\begin{cases} x'(t) = -2x(t), & t \geq 0, t \neq k, k \in \mathbb{Z}^+, \\ x(k^+) = 3x(k - 1 + \frac{\pi}{4}), & k \in \mathbb{Z}^+, \\ x(0 + 0) = x_0. \end{cases} \tag{35}$$

Solving (34), *we can obtain*

$$x(t) = \left(3\mathrm{e}^{-\frac{\pi}{2}}\right)^k x_0 \mathrm{e}^{-2(t-k)}, \ \forall t \in (k, k+1].$$

By Theorem 1, the zero solution of (35) *is asymptotically stable. By Theorems 2 and 6, both S1IRKM* (10) *and S2IRKM* (24) *for* (35) *are asymptotically stable if the stability function* $R_{rs}(z)$ *satisfies that r is odd. By Theorems 5 and 9, both S1IθM* (21) *and S2IθM* (31) *for* (35) *are asymptotically stable if* $0 < \theta < \varphi(0) = 0.5$.

As can be seen from Table 3, when the step size is halved, both the absolute and relative errors of the S1IEM and S2IEM for DEDI (35) *become half of the original ones, which roughly indicates that both the S1IEM and S2IEM for DEDI* (35) *are convergent of order 1.*

As can be seen from Table 4, when the step size is halved, both the absolute and relative errors of the S1ICRKM for DEDI (35) *become one-sixteenth of the original ones, which roughly indicates that the S1ICRKM for DEDI* (35) *is convergent of order 4. On the other hand, when the step size is halved, both the absolute and the relative errors of the S2ICRKM for DEDI* (34) *become one-fifth of the original ones, which roughly indicates that the S2ICRKM for DEDI* (35) *is convergent but not up to order 4.*

As can be seen from Table 5, which is different from Table 4 and Table 4, the S2ICRKM for DEDI (5) *is convergent of order 4 when the step sizes satisfy* $h = \frac{\tau}{m}$ *and* $m\sigma = \lfloor m\sigma \rfloor$*, i.e.,* $r_k = t_{k,\lfloor m\sigma \rfloor}$, $k = 0, 1, 2, \cdots$.

Table 3. The errors between the exact solution of DEDI (35) and the numerical solutions obtained from the S1IEM and S2IEM for (35) at $t = 10$.

	S1IEM		**S2IEM**	
m	AE	RE	AE	RE
100	$2.27936431 \times 10^{-4}$	0.11803342	$1.22261651 \times 10^{-4}$	0.06331134
200	$1.16948119 \times 10^{-4}$	0.06055981	$1.37787810 \times 10^{-4}$	0.07135133
400	$5.92344257 \times 10^{-5}$	0.03067365	$6.32341966 \times 10^{-5}$	0.03274487
800	$2.98092526 \times 10^{-5}$	0.01543627	$2.49977644 \times 10^{-5}$	0.01294471
Ratio	1.97016040	1.97016040	1.86530675	1.86530675

Table 4. The errors between the exact solution of DEDI (35) and the numerical solutions obtained from the S1ICRKM and S2ICRKM for (35) at $t = 10$.

	S1ICRKM		**S2ICRKM**	
m	AE	RE	AE	RE
100	$1.55947583 \times 10^{-11}$	$8.07550914 \times 10^{-9}$	$1.97059664 \times 10^{-4}$	0.10204436
200	$9.68306381 \times 10^{-13}$	$5.01422777 \times 10^{-10}$	$1.38899240 \times 10^{-5}$	0.00719269
400	$6.03074448 \times 10^{-14}$	$3.12292958 \times 10^{-11}$	$1.38899215 \times 10^{-5}$	0.00719269
800	$3.75264056 \times 10^{-15}$	$1.94324801 \times 10^{-12}$	$1.38899213 \times 10^{-5}$	0.00719269
Ratio	16.077341944	16.077341944	5.39574622	5.39574622

Table 5. The errors between the exact solution of DEDI (5) and the numerical solutions obtained from the S1ICRKM and S2ICRKM for (5) at $t = 10$.

	S1ICRKM		**S2ICRKM**	
m	AE	RE	AE	RE
10	$5.96296636 \times 10^{-9}$	$4.56638794 \times 10^{-7}$	$1.90245209 \times 10^{-8}$	$1.45688132 \times 10^{-6}$
20	$3.86900800 \times 10^{-10}$	$2.96285278 \times 10^{-8}$	$1.23953019 \times 10^{-9}$	$9.49221475 \times 10^{-8}$
40	$2.46385915 \times 10^{-11}$	$1.88680198 \times 10^{-9}$	$7.91000980 \times 10^{-11}$	$6.05741693 \times 10^{-9}$
80	$1.55444234 \times 10^{-12}$	$1.19037847 \times 10^{-10}$	$4.99548319 \times 10^{-12}$	$3.82549772 \times 10^{-10}$
Ratio	15.65520355	15.65520355	15.61763152	15.61763152

6. Conclusions and Future Works

In this paper, two different schemes of impulsive Runge–Kutta methods are constructed for DEDI (1) based on different ways to approximate the states $x(r_k)$, where $k \in \mathbb{Z}^+$ is required for the delayed impulses. When constructing S1IRKMs, the approximations of $x(r_k)$ are the numerical solutions obtained from Runge–Kutta methods at moments r_k, $k \in \mathbb{Z}^+$. The S1IRKMs have better convergence and are convergent of order p if the corresponding Runge–Kutta method is p order. On the other hand, when constructing S2IRKMs,

the approximations of $x(r_k)$ are the numerical solutions obtained from Runge–Kutta methods at moments at $t_{k,\lfloor \sigma m \rfloor}$, where $t_{k,\lfloor \sigma m \rfloor} \leq r_k$, $|r_k - t_{k,\lfloor \sigma m \rfloor}| \leq h$, $h = \frac{\tau}{m}$, $m, k \in \mathbb{Z}^+$. The S2IRKMs in the general case are only convergent of order 1, but they are more concise and may suit for more complex differential equations with delayed impulses. Therefore, it is very necessary to consider S2IθM. Moreover, the asymptotical stability of the exact solution and the numerical solutions of DEDI (1) was studied.

Here, we only studied the asymptotical stability of the exact solution of linear DEDI (1); the asymptotical stability of the exact solution of nonlinear DEDIs still needs further study. Moreover, applying S2IθMs to solve nonlinear DEDIs, time-delay differential equations with delayed impulses, and stochastic differential equations with delayed impulses will be future work. Applying impulsive continuous Runge–Kutta methods to solve these equations will also be future work.

Author Contributions: Conceptualization, G.-L.Z.; Methodology, G.-L.Z.; Writing—original draft, G.-L.Z. and C.L.; Writing—review & editing, G.-L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (No. 11701074) and Hebei Natural Science Foundation (No. A2020501005).

Data Availability Statement: The datasets generated during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no competing interests.

References

1. Bainov, D.D.; Simeonov, P.S. *Systems with Impulsive Effect: Stability, Theory and Applications*; Ellis Horwood: Chichester, UK, 1989.
2. Bainov, D.D.; Simeonov, P.S. *Impulsive Differential Equations: Asymptotic Properties of the Solutions*; World Scientific: Singapore, 1995.
3. Lakshmikantham, V.; Bainov, D.D.; Simeonov, P.S. *Theory of Impulsive Differential Equations*; World Scientific: Singapore, 1989.
4. Li, X.D.; Song, S.J. *Impulsive Systems with Delays: Stability and Control*; Science Press: Beijing, China, 2022.
5. He, Z.L.; Li, C.D.; Cao, Z.R.; Li, H.F. Stability of nonlinear variable-time impulsive differential systems with delayed impulses. *Nonlinear Anal. Hybrid Syst.* **2021**, *39*, 100970. [CrossRef]
6. Li, X.D.; Zhang, X.L.; Song, S.J. Effect of delayed impulses on input-to-state stability of nonlinear systems. *Automatica* **2017**, *76*, 378–382. [CrossRef]
7. Li, X.D.; Song, S.J.; Wu, J. Exponential stability of nonlinear systems with delayed impulses and applications. *IEEE Trans. Autom. Control* **2019**, *64*, 4024–4034. [CrossRef]
8. Liu, W.L.; Li, P.; Li, X.D. Impulsive systems with hybrid delayed impulses:Input-to-state stability. *Nonlinear Anal. Hybrid Syst.* **2022**, *46*, 101248. [CrossRef]
9. Chen, X.Y.; Liu, Y.; Ruan, Q.H.; Cao, J.D. Stabilization of nonlinear time-delay systems: Flexible delayed impulsive control. *Appl. Math. Model.* **2023**, *114*, 488–501. [CrossRef]
10. Chen, W.H.; Zheng, W.X. Exponential stability of nonlinear time-delay systems with delayed impulse effects. *Automatica* **2011**, *47*, 1075–1083. [CrossRef]
11. Jiang, B.; Lu, J.; Liu, Y. Exponential stability of delayed systems with average-delay impulses. *SIAM J. Control Optim.* **2020**, *58*, 3763–3784. [CrossRef]
12. Li, X.; Yang, X.; Cao, J. Event-triggered impulsive control for nonlinear delay systems. *Automatica* **2020**, *117*, 108981. [CrossRef]
13. Yang, H.L.; Wang, X.; Zhong, S.M.; Shu, L. Synchronization of nonlinear complex dynamical systems viadelayed impulsive distributed control. *Appl. Math. Comput.* **2018**, *320*, 75–85.
14. Yu, Z.Q.; Ling, S.; Liu, P.X. Exponential stability of time-delay systems with flexible delayed impulse. *Asian J. Control* **2024**, *26*, 265–279. [CrossRef]
15. Cui, Q.; Li, L.L.; Cao, J.D.Stability of inertial delayed neural networks with stochastic delayed impulses via matrix measure method. *Neurocomputing* **2022**, *471*, 70–78. [CrossRef]
16. Kuang, D.P.; Li, J.L.; Gao, D.D. Input-to-state stability of stochastic differential systems with hybrid delay-dependent impulses. *Commun. Nonlinear Sci. Numer. Simul.* **2024**, *128*, 107661. [CrossRef]
17. Lu, Y.; Zhu, Q.X. Exponential stability of impulsive random delayed nonlinearsystems with average-delay impulses. *J. Frankl. Inst.* **2024**, *361*, 106813. [CrossRef]
18. Niu, S.N.; Chen, W.H.; Lu, X.M.; Xu, W.X. Integral sliding mode control design for uncertain impulsive systems with delayed impulses. *J. Frankl. Inst.* **2023**, *360*, 13537–13573. [CrossRef]
19. Xu, H.F.; Zhu, Q.X. New criteria on pth moment exponential stability of stochastic delayed differential systems subject to average-delay impulses. *Syst. Control Lett.* **2022**, *164*, 105234. [CrossRef]

20. Zhang, M.M.; Zhu, Q.X. Stability of stochastic delayed differential systems with average-random-delay impulses. *J. Frankl. Inst.* **2024**, *361*, 106777. [CrossRef]
21. Ran, X.J.; Liu, M.Z.; Zhu, Q.Y. Numerical methods for impulsive differential equation. *Math. Comput. Model.* **2008**, *48*, 46–55. [CrossRef]
22. Zhang, Z.H.; Liang, H. Collocation methods for impulsive differential equations. *Appl. Math. Comput.* **2014**, *228*, 336–348. [CrossRef]
23. Liu, M.Z.; Liang, H.; Yang, Z.W. Stability of Runge-Kutta methods in the numerical solution of linear impulsive differential equations. *Appl. Math. Comput.* **2007**, *192*, 346–357. [CrossRef]
24. Zhang, G.L. Asymptotical stability of numerical methods for semi-linear impulsive differential equations. *Comput. Appl. Math.* **2020**, *39*, 17. [CrossRef]
25. Liang, H.; Song, M.H.; Liu, M.Z. Stability of the analytic and numerical solutions for impulsive differential equations. *Appl. Numer. Math.* **2011**, *61*, 1103–1113. [CrossRef]
26. Liang, H.; Liu, M.Z.; Song, M.H. Extinction and permanence of the numerical solution of a two-preyone-predator system with impulsive effect. *Int. J. Comput. Math.* **2011**, *88*, 1305–1325. [CrossRef]
27. Liang, H. hp-Legendre-Gauss collocation method for impulsive differential equations. *Int. J. Comput. Math.* **2015**, *94*, 151–172. [CrossRef]
28. Wen, L.P.; Yu, Y.X. The analytic and numerical stability of stiff impulsive differential equations in Banach space. *Appl. Math. Lett.* **2011**, *24*, 1751–1757. [CrossRef]
29. Zhang, G.L. Convergence, consistency and zero stability of impulsive one-step numerical methods. *Appl. Math. Comput.* **2022**, *423*, 127017. [CrossRef]
30. Ding, X.; Wu, K.N.; Liu, M.Z. The Euler scheme and its convergence for impulsive delay differential equations. *Appl. Math. Comput.* **2010**, *216*, 1566–1570. [CrossRef]
31. Zhang, G.L.; Song, M.H.; Liu, M.Z. Asymptotical stability of the exact solutions and the numerical solutions for a class of impulsive differential equations. *Appl. Math. Comput.* **2015**, *258*, 12–21. [CrossRef]
32. Zhang, G.L.; Song, M.H. Asymptotical stability of Runge–Kutta methods for advanced linear impulsive differential equations with piecewise constant arguments. *Appl. Math. Comput.* **2015**, *259*, 831–837. [CrossRef]
33. Zhang, G.L.; Song, M.H.; Liu, M.Z. Exponential stability of the exact solutions and the numerical solutions for a class of linear impulsive delay differential equations. *J. Comput. Appl. Math.* **2015**, *285*, 32–44. [CrossRef]
34. Zhang, G.L. High order Runge–Kutta methods for impulsive delay differential equations. *Appl. Math. Comput.* **2017**, *313*, 12–23. [CrossRef]
35. Zhang, G.L.; Song, M.H. Impulsive continuous Runge–Kutta methods for impulsive delay differential equations. *Appl. Math. Comput.* **2019**, *341*, 160–173. [CrossRef]
36. Wu, K.N.; Ding, X. Convergence and stability of Euler method for impulsive stochastic delay differential equations. *Appl. Math. Comput.* **2014**, *229*, 151–158. [CrossRef]
37. El Ahmadi, M.; Ayoujil, A.; Berrajaa, M. Existence and multiplicity of solutions for a class of double phase variable exponent problems with nonlinear boundary condition. *Adv. Math. Model. Appl.* **2023**, *8*, 401–414.
38. Gasimov, Y.S.; Jafari, H.; Mardanov, M.J.; Sardarova, R.A.; Sharifov, Y.A. Existence and uniqueness of the solutions of the nonlinear impulse differential equations with nonlocal boundary conditions. *Quaest. Math.* **2022**, *45*, 1399–1412. [CrossRef]
39. Syam, M.I.; Raja, M.A.; Syam, M.M.; Jarada, H.M. An accurate method for solving the undamped duffing equation with cubic nonlinearity. *Int. J. Appl. Comput. Math.* **2018**, *69*, 4. [CrossRef]
40. Syam, M.I.; Anwar, M.N.Y.; Yildirim, A.; Syam, M.M. The modified fractional power series method for solving fractional non-isothermal reaction-diffusion model equations in a spherical catalyst. *Int. J. Appl. Comput. Math.* **2019**, *5*, 38. [CrossRef]
41. Syam, S.M.; Siri, Z.; Altoum, S.H.; Kasmani, R.M. An efficient numerical approach for solving systems of fractional problems and their applications in science. *Mathematics* **2023**, *11*, 3132. [CrossRef]
42. Syam, S.M.; Siri, Z.; Altoum, S.H.; Md. Kasmani, R. Analytical and numerical methods for solving second-order two-dimensional symmetric sequential fractional integro-differential equations. *Symmetry* **2023**, *15*, 1263. [CrossRef]
43. Butcher, J.C. *Numerical Method for Ordinary Differential Equations*; Wiley: New York, NY, USA, 2016.
44. Dekker, K.; Verwer, J.G. *Stability of Runge–Kutta Methods for Stiff Nonlinear Differential Equations*; North-Holland: Amsterdam, The Netherlnads, 1984.
45. Hairer, E.; Nøsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations II, Stiff and Differential Algebraic Problems*; Springer: New York, NY, USA, 1993.
46. Song, M.H.; Yang, Z.W.; Liu, M.Z. Stability of θ-methods for advanced differential equations with piecewise continuous arguments. *Comput. Math. Appl.* **2005**, *49*, 1295–1301. [CrossRef]

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

mathematics

MDPI

Article

Quasi-Analytical Solution of Kepler's Equation as an Explicit Function of Time

A. N. Beloiarov, V. A. Beloiarov, R. C. Cruz-Gómez *, C. O. Monzón and J. L. Romero

Departamento de Física, Universidad de Guadalajara, Blvd. Marcelino García Barragán y Calzada Olimpica, Guadalajara 44840, JA, Mexico; viacheslav.beloiarov2962@alumnos.udg.mx (A.N.B.); sbeloiarov@lincolnschool.edu.mx (V.A.B.); cesar.monzon@academicos.udg.mx (C.O.M.); jose.ribarra@academicos.udg.mx (J.L.R.)

* Correspondence: raul.cruz@academicos.udg.mx

Abstract: Although Kepler's laws can be empirically proven by applying Newton's laws to the dynamics of two particles attracted by gravitational interaction, an explicit formula for the motion as a function of time remains undefined. This paper proposes a quasi-analytical solution to address this challenge. It approximates the real dynamics of celestial bodies with a satisfactory degree of accuracy and minimal computational cost. This problem is closely related to Kepler's equation, as solving the equations of motion as a function of time also provides a solution to Kepler's equation. The results are presented for each planet of the solar system, including Pluto, and the solution is compared against real orbits.

Keywords: Kepler's equation; quasi-analytical solution; celestial bodies; Kepler's laws

MSC: 65H10

1. Introduction

Kepler's laws offer an empirical mathematical model for describing the motion of orbiting bodies, later proven by Newton's analytical results [1,2]. Therefore, they are of fundamental importance in astrophysics, celestial mechanics, and Earth science [3]. Nevertheless, Kepler's laws do not describe the motion of celestial bodies as a function of time. It is the so-called Kepler's equation (KE) that provides the time dependence of the position of orbiting celestial bodies. Since Kepler first established this equation in 1609, several solutions of different types have been proposed, continuing up to the present day [2,4–9]. One of the significant early attempts was made by Carlini in 1819, later refined by Jacobi [9]. Because of its transcendental nature, an exact explicit solution remains elusive. There exist a vast variety of numerical methods renowned for their remarkable accuracy. These methods typically involve calculating infinite series or employing high-order iterative approaches. It is hardly possible to refer here to all these works, but we cite the most relevant ones [3,10–19]. Recently, some methods that use artificial intelligence have been proposed [20].

In this work, we present a quasi-analytical solution that gives explicit equations of motion as a function of time. This solution is quasi-analytical, because even though it gives explicit formulas, it depends on six numerical coefficients, which in turn depend on the orbital eccentricity. Our results provide an explicit solution to KE.

The Orbit Equation

The classical problem begins then, with the description of the motion of the planets according to the following equations [21]:

$$t(\theta) = \frac{\alpha^2 \mu}{l} \int_0^{\theta} \frac{1}{(1+\varepsilon \cos \varphi)^2} d\varphi, \tag{1}$$

and

$$r(\theta) = \frac{\alpha}{1+\varepsilon \cos \theta}. \tag{2}$$

Here, (r, θ) are the coordinates of the planet in the polar coordinate system, with the origin at the Sun; t is the time; ε is the eccentricity; and α, μ, and l are the focal parameter, the reduced mass, and the angular momentum, respectively [2]. The angle θ is called *the true anomaly*, defined as $\theta = 0$ at the pericenter [19].

The inverse of the solution of Equation (1), $\theta(t)$ is closely related to KE (the explicit relation will be shown in Section 4):

$$E - \varepsilon \sin E = M, \tag{3}$$

where E is the *eccentric anomaly* and M is the *mean anomaly*, expressed as

$$M = \frac{2\pi}{T} t \, , t \in [0, T] \, , \tag{4}$$

T is the orbital period. As was pointed out before, usually, E needs to be estimated by iterative methods [8] or series expansions [22]. Thus, the accuracy of the position and/or velocity of a celestial object moving in the Keplerian orbit, which may be obtained from the solution of KE, depends on the approximation method used. Figure 1 illustrates the variables involved in the problem.

Equations (1) and (3) are both transcendental. The solution of Equation (1) gives the solution of Equation (3), and vice-versa, which has led to a vast literature concerning alternative approaches [3]. A detailed description of the different works presented in the literature on this problem can be found in [3,19].

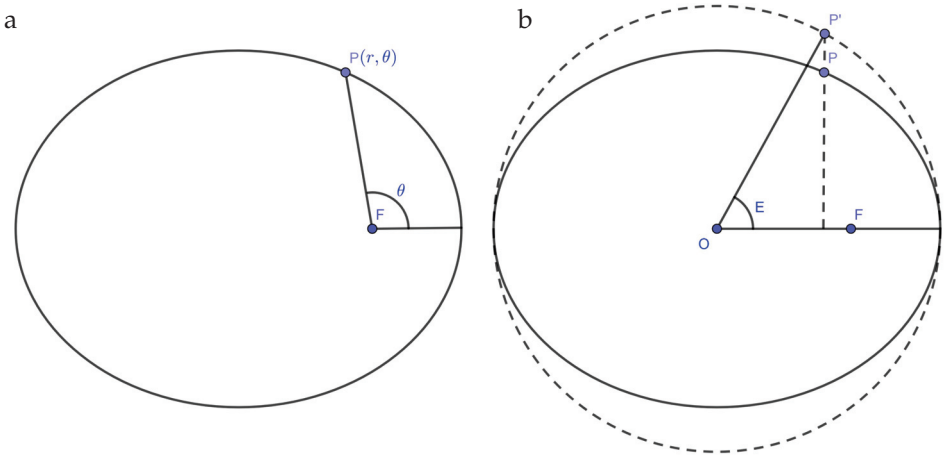


Figure 1. Illustration of the relative position of the Sun and planet (**a**) and Kepler's equation (**b**).

The remainder of this paper is structured as follows. In Section 2, we present in detail, step by step, a quasi-analytical solution of Equation (1). In Section 3, the results are presented and compared with the real orbit of each planet of the solar system. In Section 4 we present the solution to KE based on the solution of Equation (1). Finally, in Section 5, we give a brief discussion and conclusions about the results obtained in this work.

2. Methodology

In the present section, a quasi-analytical method is established for solving Equation (1). For clarity, we divide the method into sections that we call steps. All calculations and graphics were performed with MATLAB 2020b (MathWorks Inc., Natick, MA USA). Now, we briefly describe each step.

Step 0: In this step, we integrate Equation (1) and introduce a normalized time τ, which maps the real time t from the interval $[0, T]$ to the interval $[0, \pi]$, which leads us to Equation (7) for $\tau(\theta)$.

Step 1: In this step, we separate $\tau(\theta)$ into two parts, $\tau_a(\theta)$ and $\tau_b(\theta)$. Based on the observation that $\tau_a(\theta)$ resembles the behavior of $\tau(\theta)$ (especially for small values of ε), and, taking into account its analytical form, we find the inverse of $\tau_a(\theta)$, $\theta_0(\tau)$ as our zeroth approximation for $\theta(\tau)$.

Step 2: Assuming the exact solution has the form of Equation (14), we introduce the *true* argument of the arctangent function, $\varphi(\tau)$, which gives the exact solution. Then, we introduce the function $\psi_0(\tau)$ (Equation (17)) as the quotient of the *true* argument $\varphi(\tau)$ and the argument of the zeroth approximation, $\varphi_0(\tau)$. Finally, we define the first approximation $\theta_1(\tau)$ by adjusting the argument $\varphi_0(\tau)$ of the arctangent function, which appears in the zeroth approximation $\theta_0(\tau)$. The main idea is to make a series of adjustments in such a way that the quotient mentioned above approaches 1 as closely as possible.

Step 3: Following the logic of *Step 2*, we introduce the function $\psi_1(\tau)$ (Equation (22)) and the argument $\varphi_2(\tau)$ (Equation (25)) in order to obtain the second approximation $\theta_2(\tau)$ (Equation (26)). First, we propose two kinds of approximations for $\psi_1(\tau)$: linear and harmonic. This gives the approximations $\theta_{2.1}(\tau)$ and $\theta_{2.2}(\tau)$ (Equations (27) and (28)), respectively. These approximations give corresponding precisions of 3400 km and 470 km, respectively, for the position of the Earth and are completely analytical. They may be used in certain applications that do not require high precision.

Next, we proceed to improving the function $\psi_1(\tau)$ as an analytical expression (Equation (29)), involving an arctangent function, with an argument which, in turn, is an expression that depends on six numerical coefficients (Equation (30)). In order to calculate these coefficients, we propose two methods, namely *Method A* and *Method B*.

Method A consists in calculating the coefficients for each given value of the orbital eccentricity, while *Method B* allows us to express these coefficients as another analytical function with corresponding numeric coefficients that are calculated for ranges of the eccentricity.

2.1. Step 0

Our first step is integrating Equation (1) to obtain $t(\theta)$. The solution is (see Appendix A).

$$t(\theta) = \frac{\alpha^2\,\mu}{l}\left[\frac{2}{(1-\varepsilon^2)^{\frac{3}{2}}}\arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\tan\frac{\theta}{2}\right) - \frac{2\varepsilon}{1-\varepsilon^2}\,\frac{\tan\frac{\theta}{2}}{1+\varepsilon+(1-\varepsilon)\tan^2\frac{\theta}{2}}\right]. \quad (5)$$

This solution is periodic, with a period equal to 2π, and coincides with the actual time behavior in the interval $\theta \in [-\pi, \pi]$. Since the time evolution should be monotonic, we illustrate a way to extend Equation (5) to depict the time evolution by applying the following mapping:

$$t_1(\theta) = 2\,t(\pi)\left\lfloor 1 + \frac{\theta-\pi}{2\pi}\right\rfloor + t(\theta)\,, \quad (5a)$$

where $2t(\pi) = \max(t(\theta)) - \min(t(\theta))$. In order to avoid complications related to the periodicity of the functions of θ involved, and the monotony of time, in what follows, we focus our work only on the half-period of the orbit, namely $\theta \in [0, \pi]$. The second half of the orbit may be obtained from the first part using the symmetry of the orbit.

We now propose the following change of variable:

$$\tau = \frac{l}{\alpha^2\,\mu}\,\frac{(1-\varepsilon^2)^{\frac{3}{2}}}{2}\,t\,. \quad (6)$$

Thus, Equation (5), in terms of τ and θ has the form:

$$\tau(\theta) = \begin{cases} \arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\tan\frac{\theta}{2}\right) - \varepsilon\sqrt{1-\varepsilon^2}\,\frac{\tan\frac{\theta}{2}}{1+\varepsilon+(1-\varepsilon)\tan^2\frac{\theta}{2}} \text{ for } \theta \in [0,\pi)\,, \\ \\ \frac{\pi}{2} \text{ for } \theta = \pi\,. \end{cases} \tag{7}$$

Let us seek for the solution to Equation (7) in the form $\theta(\tau)$ on the interval $\theta \in [0, \pi]$, and, as deduced from Equation (7), in $\tau \in [0, \frac{\pi}{2}]$.

Note the relationship between τ and the mean anomaly M:

$$\tau = \frac{M}{2}\,. \tag{8}$$

2.2. Step 1

We rewrite Equation (7) as follows:

$$\tau(\theta) = \begin{cases} \tau_a(\theta) - \tau_b(\theta) \text{ for } \theta \in [0,\pi)\,, \\ \\ \frac{\pi}{2} \text{ for } \theta = \pi\,, \end{cases} \tag{9}$$

where

$$\tau_a(\theta) = \arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\tan\frac{\theta}{2}\right), \tag{10}$$

$$\tau_b(\theta) = \varepsilon\sqrt{1-\varepsilon^2}\,\frac{\tan\frac{\theta}{2}}{1+\varepsilon+(1-\varepsilon)\tan^2\frac{\theta}{2}}\,. \tag{11}$$

Figure 2 shows the behavior of $\tau(\theta)$, $\tau_a(\theta)$, and $\tau_b(\theta)$ with ε as parameter.

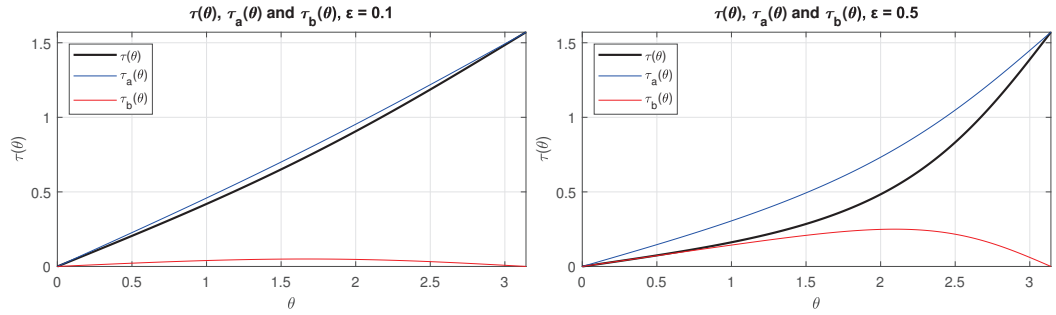


Figure 2. Plots of $\tau(\theta)$, $\tau_a(\theta)$, and $\tau_b(\theta)$ with $\varepsilon = 0.1$ (**left**) and $\varepsilon = 0.5$ (**right**).

Figure 2 shows how $\tau_a(\theta)$ is similar to $\tau(\theta)$; therefore, as our zeroth approximation $\theta_0(\tau)$, we will solve the following equation:

$$\tau(\theta) = \tau_a(\theta) = \arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\tan\frac{\theta}{2}\right). \tag{12}$$

To solve the Equation (12), we apply the tangent function to both parts. Solving for θ, we obtain

$$\theta_0(\tau) = \begin{cases} 2\arctan\left(\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\tan\tau\right) \text{ for } \tau \in [0,\frac{\pi}{2})\,, \\ \\ \pi \text{ for } \tau = \frac{\pi}{2}\,. \end{cases} \tag{13}$$

Figure 3 shows plots of $\theta(\tau)$ and $\theta_0(\tau)$ with ε as parameter. $\theta(\tau)$ was obtained using Equation (7) and interchanging axes.

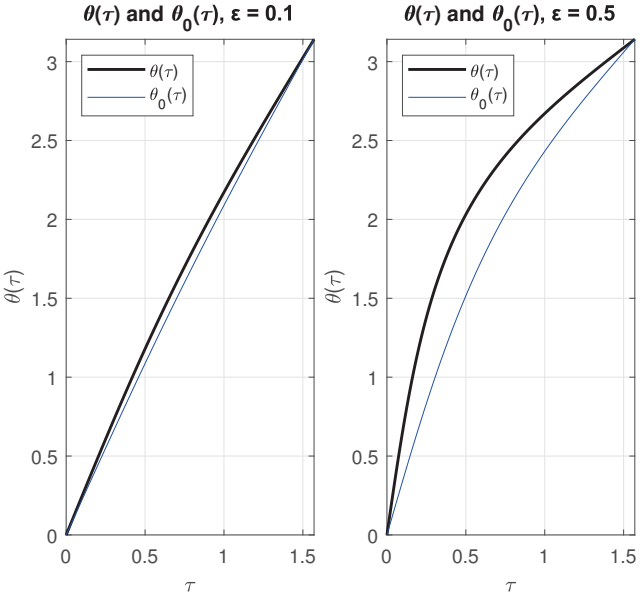


Figure 3. Plots of $\theta(\tau)$ and $\theta_0(\tau)$ with $\varepsilon = 0.1$ (**left**) y $\varepsilon = 0.5$ (**right**).

2.3. Step 2

We introduce the functions $\varphi(\tau)$ and $\varphi_0(\tau)$ as follows:

$$\theta(\tau) = 2\,\arctan[\varphi(\tau)], \tag{14}$$

$$\varphi_0(\tau) = \sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\,\tan\tau, \tag{15}$$

so that

$$\theta_0(\tau) = 2\,\arctan[\varphi_0(\tau)]. \tag{16}$$

Now, we introduce another function $\psi_0(\tau)$, defined on the open interval $\tau \in (0, \frac{\pi}{2})$:

$$\psi_0(\tau) = \frac{\varphi(\tau)}{\varphi_0(\tau)}\,. \tag{17}$$

Appendix B proves the following limits (the former is used immediately to introduce the function $\varphi_1(\tau)$, and the latter is used in step 3):

$$\lim_{\tau\to 0}\psi_0(\tau) = \frac{1}{1-\varepsilon}\,, \tag{18}$$

and

$$\lim_{\tau\to\frac{\pi}{2}}\psi_0(\tau) = 1+\varepsilon\,. \tag{19}$$

Now, we introduce, using Equations (15) and (18), the function $\varphi_1(\tau)$ as follows:

$$\varphi_1(\tau) = \lim_{\tau\to 0}\psi_0(\tau)\,\varphi_0(\tau) = \frac{1}{1-\varepsilon}\,\varphi_0(\tau) = \frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\,\tan\tau\,. \tag{20}$$

We define approximation one, $\theta_1(\tau)$, as follows:

$$\theta_1(\tau) = \begin{cases} 2\,\arctan[\varphi_1(\tau)] = 2\,\arctan\left(\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\,\tan\tau\right) \text{ for } \tau \in (0, \frac{\pi}{2})\,, \\ 0 \text{ for } \tau = 0\,, \\ \pi \text{ for } \tau = \frac{\pi}{2}\,. \end{cases} \tag{21}$$

Figure 4 shows plots of $\theta(\tau)$ and $\theta_1(\tau)$ with ε as parameter.

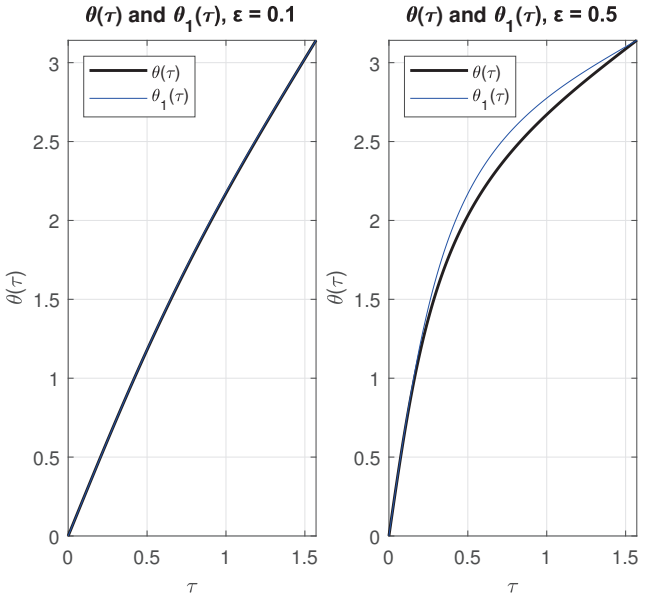


Figure 4. Plots of $\theta(\tau)$ and $\theta_1(\tau)$ with $\varepsilon = 0.1$ (**left**) and $\varepsilon = 0.5$ (**right**).

In Figure 4, we observe an evident improvement in the $\theta_1(\tau)$ approximation compared to $\theta_0(\tau)$ shown in Figure 3.

2.4. Step 3

We introduce a new function, $\psi_1(\tau)$, defined on the open interval $\tau \in (0, \frac{\pi}{2})$:

$$\psi_1(\tau) = \frac{\varphi(\tau)}{\varphi_1(\tau)}\,. \tag{22}$$

As follows from Equations (18)–(20),

$$\lim_{\tau \to 0} \psi_1(\tau) = 1\,, \tag{23}$$

$$\lim_{\tau \to \frac{\pi}{2}} \psi_1(\tau) = 1 - \varepsilon^2\,. \tag{24}$$

Figure 5 shows the plot of $\psi_1(\tau)$ for different values of the parameter ε.
We introduce the function $\varphi_2(\tau)$ as follows:

$$\varphi_2(\tau) = \psi_1(\tau)\,\varphi_1(\tau) = \psi_1(\tau)\,\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\,\tan\tau\,. \tag{25}$$

The definition of $\varphi_2(\tau)$ as the product of $\psi_1(\tau)$ and $\varphi_1(\tau)$ leads us to an apparent logical paradox, since, as follows from Equation (22), $\varphi_2(\tau) = \varphi(\tau)$. The solution to this

paradox is the fact that $\psi_1(\tau)$ is an unknown function, which we are trying to approximate with the best precision possible. The existence of such a function is obvious, for it can be theoretically constructed point-wise. However, in this work, we are looking for a quasi-analytical expression for $\psi_1(\tau)$, since the exact analytical expression for it, in principle, may not even exist.

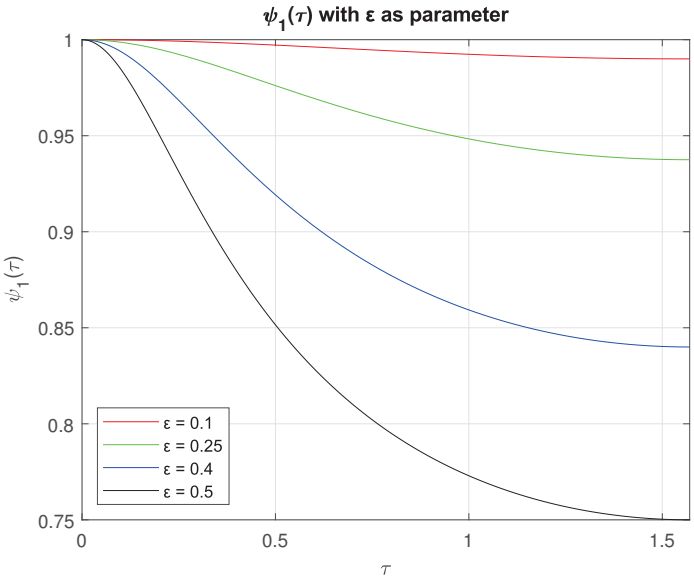


Figure 5. Plot of $\psi_1(\tau)$ with ε as parameter.

Now, we define approximation two, $\theta_2(\tau)$, as follows:

$$\theta_2(\tau) = \begin{cases} 2\arctan[\varphi_2(\tau)] = 2\arctan\left(\psi_1(\tau)\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\tan\tau\right) \text{ for } \tau \in (0, \frac{\pi}{2}), \\ 0 \text{ for } \tau = 0, \\ \pi \text{ for } \tau = \frac{\pi}{2}. \end{cases} \tag{26}$$

It should be noted that the approximation $\theta_1(\tau)$ (Equation (21)) can be considered as the approximation $\theta_2(\tau)$ with $\psi_1(\tau) = 1$. For small values of ε, such as Earth's ($\varepsilon = 0.0167$), $\psi_1(\tau)$ has a small range (for the Earth, it will be of [0.9997, 1]) so that the approximation $\psi_1(\tau) = 1$ already gives us an acceptable result for certain applications. Thus, for the Earth, the maximum error of $\theta_1(\tau)$ is of 1.8×10^{-4} rad, which, translated to the error in the position and taking into account the average distance from the Earth to the Sun as 1.5×10^8 km, is equivalent to 27,000 km, which is a little more than two diameters of the Earth. Now, approximating $\psi_1(\tau)$ in a linear form, and as a cosine with period π and amplitude adjusted to the range $[1-\varepsilon^2, 1]$, the following corresponding approximations are obtained:

$$\theta_{2.1}(\tau) = \begin{cases} 2\arctan\left[\left(1-\frac{2\varepsilon^2}{\pi}\tau\right)\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\tan\tau\right] \text{ for } \tau \in [0, \frac{\pi}{2}), \\ \pi \text{ for } \tau = \frac{\pi}{2}, \end{cases} \tag{27}$$

$$\theta_{2.2}(\tau) = \begin{cases} 2\arctan\left[\left(1+\frac{\varepsilon^2}{2}(\cos 2\tau - 1)\right)\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\tan\tau\right] \text{ for } \tau \in [0, \frac{\pi}{2}), \\ \pi \text{ for } \tau = \frac{\pi}{2}. \end{cases} \tag{28}$$

The approximation $\theta_{2.1}(\tau)$ gives a maximum error of 2.24×10^{-5} rad, which translates to 3400 km in this position (slightly more than half the Earth's radius), and the approximation $\theta_{2.2}(\tau)$ gives a maximum error of 3.11×10^{-6} rad, which corresponds to approximately 470 km in this position.

In Figure 5, we notice that the shape of the curves resembles an arctangent function, with a mapping of the argument from $(-\infty, \infty)$ to $(\frac{\pi}{2}, 0)$, with the range mapping from $(-\frac{\pi}{2}, \frac{\pi}{2})$ to $(1-\varepsilon^2, 1)$, and with the behavior of the argument being asymmetric and nonlinear. As before, we search for the approximation of the function $\psi_1(\tau)$ in the following form:

$$\psi_1(\tau) = 1 + \frac{\varepsilon^2}{2}\left(\frac{2}{\pi}\arctan[\xi(\varepsilon,\tau)] - 1\right), \tag{29}$$

where

$$\xi(\varepsilon,\tau) = a_1\,\tau^{-2} + a_2\,\tau^{-1} + a_3\,\tau + b_1\left(\tau - \frac{\pi}{2}\right)^{-2} + b_2\left(\tau - \frac{\pi}{2}\right)^{-1} + b_3\left(\tau - \frac{\pi}{2}\right), \tag{30}$$

with a_i y b_i $(i = 1, 2, 3)$ being functions of ε.

In order to evaluate the coefficients a_i and b_i (the six numerical coefficients mentioned in the introduction), two methods were developed, *Method A* and *Method B*. In *Method A*, the coefficients are calculated for each value of ε, while in *Method B*, they are expressed as analytical functions that, in turn, depend on other numeric coefficients, which are calculated for ranges of ε. As will be seen subsequently, *Method A* is more accurate, but less general, while *Method B* is slightly less accurate (especially after a certain value of ε), but more general and easier to use in applications.

2.4.1. Method A

The coefficients $a_i(\varepsilon)$ and $b_i(\varepsilon)$ were calculated numerically for each eccentricity value, corresponding to every planet of the solar system, by means of RMSE (root mean square error) minimization, using MATLAB 2020b.

Figure 6 shows the behavior of $a_i(\varepsilon)$ and $b_i(\varepsilon)$ in the range of $\varepsilon \in (0, 0.5)$ (this range was selected because some absolute values of $a_i(\varepsilon)$ and $b_i(\varepsilon)$ grow drastically after $\varepsilon = 0.5$, which does not allow us to appreciate the behavior before $\varepsilon = 0.5$).

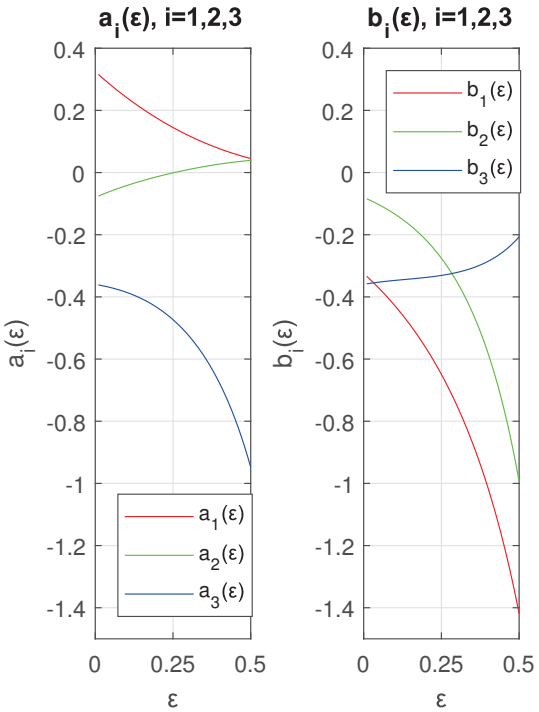


Figure 6. Plots of $a_i(\varepsilon)$ (**left**) and $b_i(\varepsilon)$ (**right**).

Table 1 shows the values of $a_i(\varepsilon)$ and $b_i(\varepsilon)$, as well as the error of $\theta(\tau)$ for the Earth and Pluto, in radians, where ME is the maximum absolute error, MAE is the mean absolute error, and RMSE is the root mean square error.

Table 1. Values of a_i and b_i and error in $\theta(\tau)$ for the Earth and Pluto.

Planet	ε	a_1	a_2	a_3	b_1	b_2	b_3	ME	MAE	RMSE
Earth	0.0167	0.310	−0.073	−0.363	−0.340	−0.087	−0.357	6.1×10^{-9}	2.9×10^{-9}	3.4×10^{-9}
Pluto	0.2488	0.146	−0.001	−0.472	−0.647	−0.274	−0.331	6.6×10^{-6}	2.8×10^{-6}	3.4×10^{-6}

Since Table 1 serves just for illustrative purposes, the values of $a_i(\varepsilon)$ and $b_i(\varepsilon)$ were rounded to three decimal places. A detailed table including more accurate values for all planets will be shown in Section 3. For now, let us mention that the absolute maximum error in the position of the Earth is 915 meters and approximately 39,000 km (close to the value of the circumference of the Earth, or 16 Pluto diameters) for the position of Pluto (the average distance of Pluto to the Sun is approximately 5.9 billion kilometers).

2.4.2. Method B

Analyzing Figure 6, it can be seen that the shapes of the curves have the behavior of a cubic polynomial, so we search for the coefficients $a_i(\varepsilon)$ and $b_i(\varepsilon)$ in the following form:

$$\begin{aligned} a_i(\varepsilon) &= a_{ij}\varepsilon^j\,, \\ b_i(\varepsilon) &= b_{ij}\varepsilon^j\,, \\ i &= 1,2,3\,, \\ j &= 0,1,2,3\,, \end{aligned} \tag{31}$$

where repeated indices imply summation.

We introduced the approximation $\theta_3(\tau)$ with the same form of $\theta_2(\tau)$ given by Equations (26), (29) and (30), where Equation (31) was now used to approximate the a_i and b_i in Equation (30).

The attempt to calculate the coefficients a_{ij} and b_{ij} in the complete range of $\varepsilon \in$ (0,1) was not successful. Thus, it was decided to divide the range into five intervals: $\varepsilon \in$ (0,0.1], $\varepsilon \in$ (0.1,0.25], $\varepsilon \in$ (0.25,0.5], $\varepsilon \in$ (0.5,0.7], and $\varepsilon \in$ (0.7,1). In the next section, we will present the values of a_{ij} and b_{ij} by ranges, and compare the precision of the approximation $\theta_2(\tau)$, which uses values of $\psi_1(\tau)$ calculated using *Method A*, with that of $\theta_3(\tau)$.

3. Results

We now compare the results of our quasi-analytical solution with the real orbits of the planets of the solar system. First, we review the results obtained above in the following equations. By a change of variable,

$$\tau = \frac{1}{\alpha^2\mu}\frac{(1-\epsilon^2)^{\frac{3}{2}}}{2}t\,, \tag{32}$$

the equations of motion in the polar coordinate system with the origin at the Sun, and in the interval of time $\tau \in [0, \frac{\pi}{2}]$ ($t \in [0, \frac{T}{2}]$), $\theta \in [0, \pi]$ are the following:

$$\theta(\tau) = \begin{cases} 2\arctan\left[\left(1+\frac{\varepsilon^2}{2}\left(\frac{2}{\pi}\arctan[\zeta(\varepsilon,\tau)]-1\right)\right)\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\tan\tau\right] & \text{for } \tau \in (0, \frac{\pi}{2})\,, \\ 0 \text{ for } \tau = 0\,, \\ \pi \text{ for } \tau = \frac{\pi}{2}\,, \end{cases} \tag{33}$$

$$r(\tau) = \frac{\alpha}{1 + \varepsilon \cos \theta(\tau)}, \tag{34}$$

where $\xi(\varepsilon, \tau)$ is given by the following expression:

$$\xi(\varepsilon, \tau) = a_1\tau^{-2} + a_2\tau^{-1} + a_3\tau + b_1\left(\tau - \frac{\pi}{2}\right)^{-2} + b_2\left(\tau - \frac{\pi}{2}\right)^{-1} + b_3\left(\tau - \frac{\pi}{2}\right), \tag{35}$$

a_i and b_i $(i = 1, 2, 3)$ are functions of ε. In what follows, we present the two methods developed for the calculation of the coefficients a_i and b_i.

3.1. Method A

As mentioned above, in this method, the values of a_i and b_i were calculated for each eccentricity by means of RMSE minimization in MATLAB 2020b. Table 2 shows the values of a_i and b_i, ME, MAE, and RMSE errors in $\theta(\tau)$, in radians, with D_s being the average distance of the planet to the Sun in kilometers, and E_p being the maximum error of the position in kilometers with respect to the real orbit.

Table 2. Values of a_i, b_i, corresponding errors in $\theta(\tau)$, and absolute position for celestial bodies.

	Mercury	**Venus**	**Earth**	**Mars**	**Jupiter**	**Saturn**	**Uranus**	**Neptune**	**Pluto**
ε	**0.2056**	**0.0067**	**0.0167**	**0.0935**	**0.0489**	**0.0565**	**0.0457**	**0.0113**	**0.2488**
a_1	0.17053517	0.31862395	0.30977503	0.24694331	0.28234256	0.27609996	0.28499728	0.31453377	0.14561949
a_2	−0.01166110	−0.07705003	−0.0728542	−0.04432541	−0.06014655	−0.05731287	−0.06135776	−0.07510434	−0.0012376
a_3	−0.43861374	−0.36086494	−0.36274460	−0.38289299	−0.36985666	−0.37179675	−0.36907139	−0.36171170	−0.47217706
b_1	−0.57282655	−0.33074668	−0.34005421	−0.42023391	−0.37170483	−0.37957121	−0.36843946	−0.33499871	−0.64694223
b_2	−0.22240872	−0.08288098	−0.08742593	−0.12920751	−0.10334339	−0.10741403	−0.10166725	−0.08494978	−0.27396065
b_3	−0.33681377	−0.35855961	−0.35697508	−0.34764211	−0.35253989	−0.35161930	−0.35294044	−0.35781706	−0.33108759
ME	3.6×10^{-6}	9.5×10^{-10}	6.1×10^{-9}	3.5×10^{-7}	6.6×10^{-8}	9.4×10^{-8}	5.6×10^{-8}	2.7×10^{-9}	6.6×10^{-6}
MAE	1.5×10^{-6}	4.5×10^{-10}	2.9×10^{-9}	1.5×10^{-7}	2.9×10^{-8}	4.1×10^{-8}	2.5×10^{-8}	1.3×10^{-9}	2.8×10^{-6}
RMSE	1.9×10^{-6}	5.3×10^{-10}	3.4×10^{-9}	1.8×10^{-7}	3.5×10^{-8}	5.0×10^{-8}	3.0×10^{-8}	1.5×10^{-9}	3.4×10^{-6}
D_s	5.8×10^{7}	1.1×10^{8}	1.5×10^{8}	2.3×10^{8}	7.9×10^{8}	1.4×10^{9}	2.9×10^{9}	4.5×10^{9}	5.9×10^{9}
E_p	208	0.11	0.92	80	52	131	163	12	39,000

The results in Table 2 show that for small values of ε, the accuracy was high. However, for values of ε∼0.2 the error increased significantly (for Mercury and Pluto the error increased by 2–3 orders in comparison to the errors for the other planets).

3.2. Method B

As was pointed out before, in this method, we calculate the coefficients of a third-degree polynomial (Equation (31)) by minimizing the RMSE for the ranges of ε. Therefore, in this case, it is not necessary to calculate the a_i and b_i for each value of the eccentricity. Table 3 shows the values of the coefficients a_{ij} and b_{ij} for the different ranges of ε. Obviously, using this method results in errors greater than those of *Method A*.

Table 3. Values of a_{ij} and b_{ij} calculated in ranges of ε.

Range of ε	**(0,0.1]**	**(0.1,0.25]**	**(0.25,0.5]**	**(0.5,0.7]**	**(0.7,1.0)**
a_{10}	0.32464090	0.32455984	0.32493519	0.33117795	0.34799892
a_{11}	−0.90342437	−0.90136299	−0.90443788	−0.94434644	−1.02378174
a_{12}	0.79798292	0.77956682	0.78688870	0.87179816	0.99672027
a_{13}	−0.24897861	−0.19074605	−0.19470806	−0.25486105	−0.32027157
a_{20}	−0.07992819	−0.07920359	−0.07197522	−0.06646582	−0.11098658
a_{21}	0.43404410	0.41648507	0.33665965	0.29615322	0.50088661
a_{22}	−0.64017363	−0.49188569	−0.19208276	−0.09396227	−0.40748236
a_{23}	0.75341116	0.31132787	−0.07256170	−0.15093252	0.00894242
a_{30}	−0.35968044	−0.35742442	−0.15675162	6.29837377	154.50791377
a_{31}	−0.17220655	−0.22278632	−2.20357719	−41.92862343	−715.24316797
a_{32}	−0.64666864	−0.26055305	6.28024901	87.40520959	1105.82612363
a_{33}	−1.78408496	−2.80408480	−10.06884121	−65.08749455	−577.96129585
b_{10}	−0.32463507	−0.32142203	−0.09294215	6.21933680	138.39317241
b_{11}	−0.90434048	−0.97722350	−3.24616805	−42.18802414	−643.15766068
b_{12}	−1.11071449	−0.54525397	7.00356134	86.74796714	996.53861794
b_{13}	−1.63153199	−3.15712303	−11.61829996	−65.86504267	−524.44755555

Table 3. *Cont.*

Range of ε	(0,0.1]	(0.1,0.25]	(0.25,0.5]	(0.5,0.7]	(0.7,1.0)
b_{20}	−0.07992299	−0.07527391	0.29080466	11.97863173	287.23614950
b_{21}	−0.43571269	−0.54094614	−4.16198706	−76.07319110	−1326.12648115
b_{22}	−0.78063318	0.03353877	12.02291626	158.84151291	2048.91985911
b_{23}	−2.10660888	−4.29567159	−17.65734989	−117.20310761	−1068.68369232
b_{30}	−0.35968350	−0.36025835	−0.44346730	−3.54853992	−81.17535211
b_{31}	0.17171182	0.18402819	0.99888552	20.06864682	372.42195877
b_{32}	−0.59524043	−0.68312683	−3.34688052	−42.20159066	−574.67488149
b_{33}	1.45594036	1.66666199	4.58818969	30.87266047	298.77577747

Figure 7 shows a comparison of the errors between methods A and B. As can be seen, the error in *Method B* was slightly higher (especially for ME) up to values of $\varepsilon = 0.8$. For values of $\varepsilon > 0.8$, the errors of *Method B* increased significantly with respect to those of *Method A*.

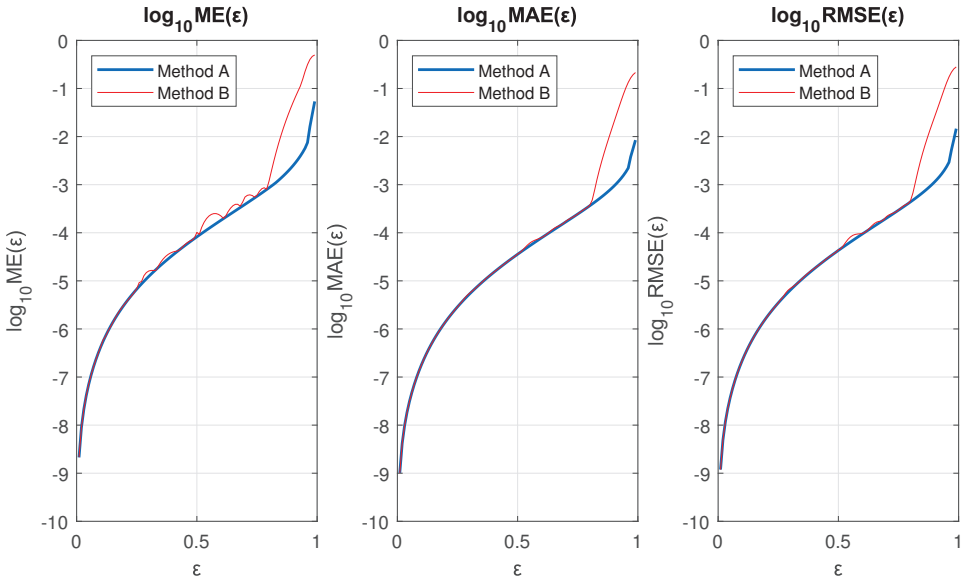


Figure 7. Logarithmic plots of the error in $\theta(\tau)$ as a function of ε for both methods.

3.3. Equations of Motion for the Planets

Finally, let us express the equations of motion of both position and velocity in the Cartesian coordinate system, with the origin at the Sun.

The equations for the position are as follows:

$$\begin{aligned} x(\tau) &= r(\tau)\,\cos\,\theta(\tau)\,, \\ y(\tau) &= r(\tau)\,\sin\,\theta(\tau)\,, \end{aligned} \tag{36}$$

where $\theta(\tau)$ and $r(\tau)$ are given by Equations (33) and (34), respectively.

The equations for the velocity are as follows:

$$\begin{aligned} v_x &= \dot{r}(\tau)\,\cos\,\theta(\tau) - r\dot{\theta}(\tau)\,\sin\,\theta(\tau)\,, \\ v_y &= \dot{r}(\tau)\,\sin\,\theta(\tau) + r\dot{\theta}(\tau)\,\cos\,\theta(\tau)\,, \end{aligned} \tag{37}$$

where

$$\dot{\theta}(\tau) = 2\,\frac{\sqrt{1+\varepsilon}}{(1-\varepsilon)^{\frac{3}{2}}}\,\frac{\dot{\psi}(\tau)\,\tan\tau + \psi(\tau)\,\sec^2\tau}{1+\frac{1+\varepsilon}{(1-\varepsilon)^3}\psi^2(\tau)\,\tan^2\tau}\,,$$
$$\dot{r}(\tau) = \frac{\alpha\,\varepsilon\,\dot{\theta}(\tau)\,\sin\theta(\tau)}{(1+\varepsilon\,\cos\,\theta(\tau))^2}\,, \tag{38}$$

and

$$\psi(\tau) = \begin{cases} 1+\frac{\varepsilon^2}{2}\,(\frac{2}{\pi}\,\arctan[\xi(\varepsilon,\tau)]-1)\ \text{ for } \tau\in(0,\frac{\pi}{2})\,, \\ 1\ \text{ for } \tau = 0\,, \\ 1-\varepsilon^2\ \text{ for } \tau = \frac{\pi}{2}\,, \end{cases} \tag{39}$$

$$\dot{\psi}(\tau) = \begin{cases} \frac{\varepsilon^2}{\pi}\,\frac{\dot{\xi}(\tau)}{1+\xi^2(\tau)}\ \text{ for } \tau\in(0,\frac{\pi}{2})\,, \\ 0\ \text{ for } \tau = 0\,, \\ 0\ \text{ for } \tau = \frac{\pi}{2}\,. \end{cases} \tag{40}$$

The function $\xi(\tau)$ defined in the interval $\tau\in(0,\frac{\pi}{2})$ is given by Equation (35) and its derivative is the following:

$$\dot{\xi}(\tau) = -2a_1\tau^{-3} - a_2\tau^{-2} + a_3 - 2b_1\Big(\tau-\frac{\pi}{2}\Big)^{-3} - b_2\Big(\tau-\frac{\pi}{2}\Big)^{-2} + b_3\,. \tag{41}$$

Figure 8 shows a comparison plot between the real orbit, calculated numerically using Equation (7), and the orbit obtained by numerical integration of Equation (37) (where *Method A* was used). Additionally, the velocity vectors are shown at an arbitrary scale at certain points of the trajectory.

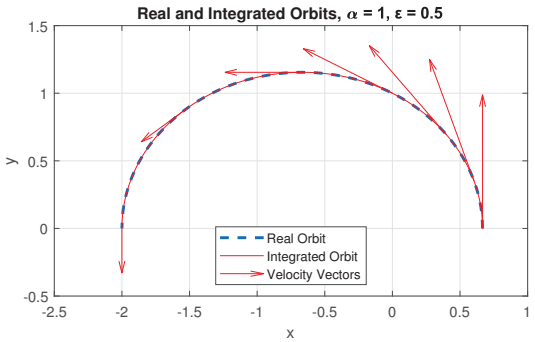


Figure 8. Plot of the orbits, real and integrated, with $\alpha = 1$ and $\varepsilon = 0.5$.

As can be seen in Figure 8, the integrated orbit closely resembles the behavior of the real orbit. This shows that our *Method A* is sufficiently accurate, even for eccentricities of the order of up to $\varepsilon = 0.5$.

4. Solution to Kepler's Equation

As can be seen from Figure 1b, it is trivial to obtain the following expression, which gives the formula for E (the *eccentric anomaly*), appearing in Equation (3) as a function of

the polar coordinates (ρ, θ) with the center in the focus F. In what follows, we set the major semi-axis equal to one ($a = 1$):

$$E = \arccos(F + \rho\cos\theta)\,, \tag{42}$$

where $F = |OF|$ (Figure 1b). As is well known from the theory of conic sections, $F = \varepsilon a = \varepsilon$, since we set $a = 1$. Thus, Equation (42) acquires the following form:

$$E = \arccos(\varepsilon + \rho\cos\theta)\,. \tag{43}$$

Due to the normalization $a = 1$, the expression for ρ is as follows:

$$\rho = \frac{r}{a}\,. \tag{44}$$

The functions $\theta(\tau)$ and $r(\tau)$ are given by our final quasi-analytical solution (Equations (33) and (34)).

Now, in order to obtain the final explicit expression for the solution of KE, $E(M)$, we need to express θ and ρ as functions of M (the *mean anomaly* given by Equation (4)). So far, θ and r are functions of τ (where τ is given by Equation (6)). Thus, by expressing τ as a function of M, θ and r automatically become functions of M. The expression for $\tau(M)$ is the following:

$$\tau(M) = \frac{M}{2}\,. \tag{45}$$

In order to obtain ρ as a function of M, we use Kepler's third law to express a as follows:

$$a = \left(\frac{l}{2\pi\mu\sqrt{1-\varepsilon^2}}\,T\right)^{\frac{1}{2}}. \tag{46}$$

The final solution to KE is given by Equation (43), using the quasi-analytical solution for the motion given by Equations (33) and (34) and taking into account the relations (44), (45) and (46).

5. Remarks and Conclusions

In this work, we obtained a quasi-analytical solution for the motion of the celestial bodies as an explicit function of time in four steps. We called this solution quasi-analytical, due to the dependency on certain numerical coefficients, which in turn themselves depend on the orbital eccentricity. We proposed two methods for evaluating these coefficients: *Method A* and *Method B*. The former gives a higher degree of accuracy, but involves calculations for each value of the eccentricity, while the latter is less accurate (especially for values of $\varepsilon > 0.8$) but more practical, since it works for ranges of the eccentricity. Although there exist methods for solving Kepler's equation up to machine precision, e.g., [11], these methods are completely numerical and require an initial guess. The aim of our work was to find a relatively simple explicit analytical solution with acceptable precision. In future work, we plan to use our results as an initial guess for the iterative procedures mentioned previously. We hypothesize that this approach will lead to highly precise results with fewer iterations.

Author Contributions: The first two authors A.N.B. and V.A.B. developed the theory and worked on the redaction together with R.C.C.-G. and C.O.M. J.L.R. collaborated on the theoretical revision of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

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

Appendix A. Integration of Equation (1)

To solve the integral, we propose the Weierstrass change of variable:

$$u = \tan\frac{\varphi}{2}\,. \tag{A1}$$

From Equation (A1), it follows that

$$\cos\varphi = \frac{1-u^2}{1+u^2}\,, d\varphi = \frac{2\,du}{1+u^2}\,. \tag{A2}$$

Substituting Equation (A2) into Equation (1), and with a little algebra, we obtain the following:

$$F(\varphi) = \frac{2\,\alpha^2\,\mu}{l}\int \frac{1+u^2}{(a+b\,u^2)^2}\,du\,, \tag{A3}$$

which will be left undefined for now and will eventually be evaluated in the original variables. Here, $a = 1+\varepsilon$ and $b = 1-\varepsilon$. To facilitate the integration, the integrand is simplified using partial fractions, leaving the integral as follows:

$$F(\varphi) = \frac{2\,\alpha^2\,\mu}{l}\left[\frac{b-a}{b}\int \frac{du}{(a+b\,u^2)^2} + \frac{1}{b}\int \frac{du}{a+b\,u^2}\right]. \tag{A4}$$

Performing a bit of algebra and making the trigonometric substitution

$$u = c\,\tan v\,,$$
$$du = c\,\sec^2 v\,dv\,,$$

where $c^2 = \frac{a}{b}$. Finally, Equation (A4) reads

$$F(\varphi) = \frac{2\,\alpha^2\,\mu}{l}\left[\frac{b-a}{b^3}\int \frac{c\,\sec^2 v\,dv}{c^4\,\sec^4 v} + \frac{1}{b^2}\int \frac{c\,\sec^2 v\,dv}{c^2\,\sec^2 v}\right] = \frac{2\alpha^2\mu}{l}\left[\frac{b-a}{b^3c^3}\int \cos^2 v\,dv + \frac{1}{b^2c}\int dv\right]. \tag{A5}$$

The latter integral is trivial, and the former is solved using the identity $\cos^2 v = \frac{1}{2}(1+\cos 2v)$, such that $F(\varphi)$ is left as

$$F(\varphi) = \frac{2\alpha^2\mu}{l}\left[\left(\frac{b-a+2bc^2}{2b^3c^3}\right)v + \frac{b-a}{2b^3c^3}\,\sin v\,\cos v\right]. \tag{A6}$$

Returning to the variable u, recalling that $c^2 = \frac{a}{b}$ and performing some algebra, Equation (A6) reads

$$F(\varphi) = \frac{\alpha^2\mu}{l}\left[\frac{a+b}{(ab)^{\frac{3}{2}}}\,\arctan\left(\sqrt{\frac{b}{a}}u\right) + \frac{b-a}{ab}\frac{u}{a+bu^2}\right]. \tag{A7}$$

Finally, recalling Equation (A1), expressing a and b in terms of ε, performing algebra, and evaluating the limits of integration, we arrive at our final result:

$$t(\theta) = \frac{\alpha^2\mu}{l}\left[\frac{2}{(1-\varepsilon^2)^{\frac{3}{2}}}\arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\tan\frac{\theta}{2}\right) - \frac{2\varepsilon}{1-\varepsilon^2}\frac{\tan\frac{\theta}{2}}{1+\varepsilon+(1-\varepsilon)\tan^2\frac{\theta}{2}}\right], \tag{A8}$$

which is in agreement with Equation (5).

Appendix B. Proofs of Limits Equations (18) **and** (19)

We begin by solving for $\varphi(\tau)$ from Equation (14):

$$\varphi(\tau) = \tan\frac{\theta}{2}. \tag{A9}$$

Next, in Equation (15), we substitute τ by the expression Equation (7):

$$\varphi_0(\tau) = \sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\tan\left[\arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\tan\frac{\theta}{2}\right) - \varepsilon\sqrt{1-\varepsilon^2}\frac{\tan\frac{\theta}{2}}{1+\varepsilon+(1-\varepsilon)\tan^2\frac{\theta}{2}}\right]. \tag{A10}$$

We introduce the variable β as follows:

$$\beta = \tan\frac{\theta}{2}. \tag{A11}$$

For the sake of simplicity, we will evaluate the limit $\frac{\varphi_0(\tau)}{\varphi(\tau)}$ instead of $\frac{\varphi(\tau)}{\varphi_0(\tau)}$.

In order to prove limit Equation (18), we will use the Taylor expansion about zero of the following functions:

$$\begin{aligned}\tan x &= x + o(x),\\ \arctan x &= x + o(x).\end{aligned} \tag{A12}$$

Using Equation (A12), we obtain

$$\lim_{\tau\to 0}\frac{\varphi_0(\tau)}{\varphi(\tau)} = \lim_{\beta\to 0}\frac{\varphi_0(\beta)}{\varphi(\beta)} = \lim_{\beta\to 0}\frac{1}{\beta}\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\tan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\beta - \frac{\varepsilon\sqrt{1-\varepsilon^2}\beta}{1+\varepsilon+(1-\varepsilon)\beta^2} + o(\beta)\right)$$

$$= \lim_{\beta\to 0}\frac{1}{\beta}\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\beta - \frac{\varepsilon\sqrt{1-\varepsilon^2}\beta}{1+\varepsilon+o(\beta)} + o(\beta)\right)$$

$$= \lim_{\beta\to 0}\frac{1}{\beta}\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\beta\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}} - \frac{\varepsilon\sqrt{1-\varepsilon^2}}{1+\varepsilon} + O(\beta)\right) = \lim_{\beta\to 0}[(1-\varepsilon+O(\beta)] = 1-\varepsilon,$$

which is equivalent to

$$\lim_{\tau\to 0}\frac{\varphi(\tau)}{\varphi_0(\tau)} = \frac{1}{1-\varepsilon}, \tag{A13}$$

which agrees with Equation (18).

In order to demonstrate Equation (19), we use the following trigonometric identity:

$$\tan(a-b) = \frac{\tan a - \tan b}{1+\tan a\tan b}. \tag{A14}$$

Using Equation (A12) and Equation (A14), we have

$$\lim_{\tau\to\frac{\pi}{2}}\frac{\varphi_0(\tau)}{\varphi(\tau)} = \lim_{\beta\to\infty}\frac{\varphi_0(\beta)}{\varphi(\beta)} = \lim_{\beta\to\infty}\frac{1}{\beta}\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\tan\left[\arctan\left(\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\beta\right) - \frac{\varepsilon\sqrt{1-\varepsilon^2}\beta}{1+\varepsilon+(1-\varepsilon)\beta^2}\right]$$

$$= \lim_{\beta\to\infty}\frac{1}{\beta}\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\frac{\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\beta - \frac{\varepsilon\sqrt{1-\varepsilon^2}\beta}{1+\varepsilon+(1-\varepsilon)\beta^2} + o\left(\frac{1}{\beta}\right)}{1+\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\frac{\varepsilon\sqrt{1-\varepsilon^2}\beta^2}{O\left(\frac{1}{\beta^2}\right)+(1-\varepsilon)\beta^2} + O\left(\frac{1}{\beta}\right)}$$

$$= \lim_{\beta\to\infty}\frac{1}{\beta}\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\frac{\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\beta + o\left(\frac{1}{\beta}\right)}{1+\sqrt{\frac{1-\varepsilon}{1+\varepsilon}}\frac{\varepsilon\sqrt{1-\varepsilon^2}}{1-\varepsilon} + O\left(\frac{1}{\beta}\right)} = \lim_{\beta\to\infty}\frac{1+o\left(\frac{1}{\beta}\right)}{1+\varepsilon+O\left(\frac{1}{\beta}\right)} = \frac{1}{1+\varepsilon},$$

which is equivalent to

$$\lim_{\tau \to \frac{\pi}{2}} \frac{\varphi(\tau)}{\varphi_0(\tau)} = 1 + \varepsilon\,, \tag{A15}$$

which agrees with Equation (19).

References

1. Krisciunas, K. Demonstrating the elliptical orbit of Mars using naked eye data. *Am. J. Phys.* **2019**, *87*, 885–893. [CrossRef]
2. Goldstein, H. *Classical Mechanics*, 2nd ed.; Addison-Wesley: Reading, UK, 2020.
3. Baisheng, W.; Zhou, Y.; Lim, C.; Zhong, H. A new solution approach via analytical approximation of the elliptic kepler equation. *Acta Astronaut.* **2023**, *202*, 303–310. [CrossRef]
4. Colwell, P. *Solving Kepler's Equation over Three Centuries*; Willman-Bell, Inc.: Richmond, VA, USA, 1993.
5. Landau, L.D.; Lifshitz, E.M. *Mechanics*, 3rd ed.; Elsevier Butterworth-Heinemann: Burlington, NJ, USA, 1976.
6. Hagihara, Y. *Celestial Mechanics: Perturbation Theory*, 1st ed.; MIT Press: Cambridge, MA, USA, 1970.
7. Danby, J.M.A. *Fundamentals of Celestial Mechanics: Perturbation Theory*, 2nd ed.; Willmann-Bell: New York, NY, USA, 1988.
8. Odell, A.W.; Gooding, R.H. Procedures for Solving Kepler's Equation. *Cel. Mech.* **1986**, *38*, 307–334. [CrossRef]
9. Sacchetti, A. Francesco Carlini: Kepler's equation and the asymptotic solution to singular differential equations. *Hist. Math.* **2020**, *53*, 1–32. [CrossRef]
10. González-Gaxiola, O.; Hernández-Linares, S. An Efficient Iterative Method for Solving the Elliptical Kepler's Equation. *Int. J. Appl. Comput. Math* **2021**, *7*, 1–14. [CrossRef]
11. Abubekerov, M.K.; Gostev, N.Y. Solution of Kepler's equation with machine precision, *Astr. Rep.* **2020**, *64*, 1060–1066. [CrossRef]
12. Dubinov, A.E.; Galidakis, I.N. Explicit solution of the Kepler equation. *Phys. Part. Nuclei Lett.* **2007**, *4*, 213–216. [CrossRef]
13. Elenin, G.G.; Elenina, T.G. Parametrization of the Solution of the Kepler Problem and New Adaptive Numerical Methods Based on This Parametrization. *Differ. Equ.* **2018**, *54*, 911–918. [CrossRef]
14. Markley, F.L. Kepler Equation solver. *Celest. Mech. Dyn. Astron.* **1995**, *63*, 101–111. [CrossRef]
15. Simha, A. An algebra and trigonometry-based proof of Kepler's first law. *Am. J. Phys.* **2021**, *89*, 1009–1011. [CrossRef]
16. Easton, R.W.; Anderson, R.L.; Lo, M.W. Conic transfer arcs for Kepler's problem. *Am. J. Phys.* **2022**, *90*, 666–671. [CrossRef]
17. Calvo, M.; Elipe, A.; Rández, L. On the integral solution of elliptic Kepler's equation. *Celest. Mech. Dyn. Astron.* **2023**, *135*, 26. [CrossRef]
18. Borghi, R. On the Bessel solution of Kepler's Equation. *Mathematics* **2024**, *12*, 154. [CrossRef]
19. Orlando, F.; de Souza, C.F.; Zarro, C.; Terra, P. Kepler's equation and some of its pearls. *Am. J. Phys.* **2018**, *86*, 849–858. [CrossRef]
20. Zheng, M.; Luo, J.; Dang, Z. *Machine Learning-Based Solution of Kepler's Equation*; Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series; SPIE: Bellingham, WA, USA, 2022. [CrossRef]
21. Marion, J.B. *Classical Dynamics*, 1st ed.; Academic Press Inc.: Cambridge, MA, USA, 1965.
22. Mikkola, S.A. A cubic approximation for Kepler's equation. *Cel. Mech.* **2018**, *86*, 849–858. [CrossRef]

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

mathematics

MDPI

Article

A Time-Series Feature-Extraction Methodology Based on Multiscale Overlapping Windows, Adaptive KDE, and Continuous Entropic and Information Functionals

Antonio Squicciarini [1,*], Elio Valero Toranzo [2] and Alejandro Zarzo [1]

[1] GI-TACA, Departamento de Matemática Aplicada a la Ingeniería Industrial, Escuela Técnica Superior de Ingenieros Industriales, Universidad Politécnica de Madrid, 28006 Madrid, Spain; alejandro.zarzo@upm.es
[2] Departamento de Matemática Aplicada, Escuela Superior de Ciencias Experimentales y Tecnología, Universidad Rey Juan Carlos, 28933 Madrid, Spain; elio.vtoranzo@urjc.es
* Correspondence: a.squicciarini@alumnos.upm.es

Abstract: We propose a new methodology to transform a time series into an ordered sequence of any entropic and information functionals, providing a novel tool for data analysis. To achieve this, a new algorithm has been designed to optimize the Probability Density Function (PDF) associated with a time signal in the context of non-parametric Kernel Density Estimation (KDE). We illustrate the applicability of this method for anomaly detection in time signals. Specifically, our approach combines a non-parametric kernel density estimator with overlapping windows of various scales. Regarding the parameters involved in the KDE, it is well-known that bandwidth tuning is crucial for the kernel density estimator. To optimize it for time-series data, we introduce an adaptive solution based on Jensen–Shannon divergence, which adjusts the bandwidth for each window length to balance overfitting and underfitting. This solution selects unique bandwidth parameters for each window scale. Furthermore, it is implemented offline, eliminating the need for online optimization for each time-series window. To validate our methodology, we designed a synthetic experiment using a non-stationary signal generated by the composition of two stationary signals and a modulation function that controls the transitions between a normal and an abnormal state, allowing for the arbitrary design of various anomaly transitions. Additionally, we tested the methodology on real scalp-EEG data to detect epileptic crises. The results show our approach effectively detects and characterizes anomaly transitions. The use of overlapping windows at various scales significantly enhances detection ability, allowing for the simultaneous analysis of phenomena at different scales.

Keywords: time series; anomaly detection; Jensen–Shannon divergences; kernel method; generalised entropy

MSC: 94A16; 94A17

1. Introduction

In the ever-evolving landscape of data-driven decision-making, time-series analysis stands as a pivotal tool for extracting valuable insights from sequential data points. The increase of sources relying on real-time data has revived interest in time-series analysis methods.

Entropic and information functionals [1–3] have proven to be effective tools for analysing time-series signals across various applications. For example, Gupta and Pachori [4] demonstrated the promising capabilities of combining Rényi permutation entropy with Fourier–Bessel series expansion to train different machine-learning algorithms effectively for seizure detection. Rosso et al. [5] utilized discrete Wavelet information tools for quantitative EEG record analysis, showing that the relative wavelet energy information ap-

proach can capture epileptic rhythm characteristics without applying parametric inference solutions to the EEG.

Mateos et al. [6] combined permutation entropy with permutation Lempel–Ziv complexity to describe different states of consciousness, demonstrating how these transformations are potential tools for quantifying cognitive mental states. Martin et al. [7,8] showed that Shannon entropy, Fisher information, and Tsallis entropy, combined with a discrete non-parametric inference window approach, can detect EEG seizures. In another study, Lerga et al. [9] proposed a solution to classify hand movements based on short-term Réyni entropy over EEG signals. Alkahtani et al. [10] combined statistical feature extraction, including Shannon entropy, approximate entropy, and power spectral entropy, with feature selection solutions like LASSO applied over different machine-learning methods to detect paediatric attention deficit hyperactivity disorder. Bezerianos et al. [11] found that Shannon and Tsallis entropies could discriminate different injury levels during recovery from global ischemia in Wistar rats. Additionally, Kalimeri et al. [12] demonstrated that Tsallis entropy, in combination with symbolic dynamics, provides a quantitative strategy for monitoring states in a focal area leading up to an impending earthquake. Guignard et al. [13] utilized the kernel density estimator method to calculate information measures, particularly the Fisher–Shannon complexity measure, on nonlinear time series of high-frequency wind, successfully describing the time signal evolution. Conejero et al. [14] assessed the effectiveness of different entropy formulations for non-linear signal classification using chaotic mapping. Zhu et al. [15] utilized graph entropy, various node degrees, and support vector machines to detect major depressive disorder from a single-channel EEG signal.

All the previous methods observed in the literature are case-specific applications, in the sense that there is no focus on how the transformation parameters have been optimized, limiting the generalization of results across different application domains. Furthermore, most of the used inference solutions return discrete probability mass functions, which do not allow the computation of differentiable entropic and information measures.

Entropy functionals have shown their capability as feature-extraction tools to describe and detect anomalies inside time signals, providing information about the complexity and dynamic behaviour.

One application of these new feature-extraction methods is to enhance the detectability of anomalies in time series. Broadly speaking, an anomaly within a time series can be characterized as an unusual pattern that deviates from what is considered normal behaviour. Identifying unusual patterns or outliers within time-series data is crucial for maintaining the integrity and reliability of various systems, ranging from industrial processes, and financial markets to healthcare and cybersecurity. Anomalies in time series can be classified into three categories: point anomaly, contextual anomaly, and collective anomaly. A point anomaly refers to a data instance that deviates from the normal range of values. For the second type of anomaly, contextual anomaly, the abnormality is not determined by the absolute value; rather, it is assessed based on its position within the time series. The third category, collective anomaly, pertains to a sequence of instances that diverges from the expected normal behaviour, yet the anomaly is not necessarily associated with any specific data point [16,17].

The main difference is that point anomalies can be detected using upper and lower control limits, whereas contextual and collective anomalies cannot. While there is no universally accepted definition of an anomaly in time series, it is often associated with shifts in frequency content. For example, in the context of machine fault detection, an anomaly may arise from a change in machine stiffness, impacting its modal response [18]. In medical applications, critical information for detecting epileptic seizures is gleaned from variations in specific frequency bands in EEG signals [5].

Entropic techniques for feature extraction first require time localization criteria, when applied to time series. Subsequently, an inference solution is needed to transform the data into a probability function. This approach can be referred to as Time-Dependent Entropy (TDE) [11]. Therefore, the parameters governing this transformation can be divided into

two groups, the time-localization and the inference parameters, the latter specific for each inference solution utilized. Those two sets of parameters are related. Combining the window localization with the inference solution, the result is an ordered set of windowed probability distributions. To accomplish this goal, various inference solutions have been tested, such as symbolic solutions [12,19], non-parametric discrete inference or histogram [7,8,11,20], power spectral density [21], approximate entropy [22], wavelet entropy [5,22], sample and fuzzy entropy [23–25], Kraskov entropy [26], and neural network entropy [14]. Among them, few efforts have been applied to testing continuous non-parametric inference solutions, such as kernel density estimation (also called Parzen–Rosenblatt window) [13]. Moreover, for detecting anomalies, the utilization of different window scales for time localization is crucial for the sensitivity of the transformation [20], yet almost all the cited works only employ one window size.

Among the inference solutions analysed, the only one that returns a probability density function is Kernel Density Estimation (KDE) [27], whose output is a continuous probability density function. This reflects the continuous nature of the underlying real-time-series signal and allows for the application of entropic feature-extraction techniques, such as differential Shannon entropy and non-parametric Fisher information. Moreover, in the discrete case, not all measures are uniquely defined, as is the case of the Fisher information [28]. Therefore, it is important to study continuous entropic formulations since those are not the limit case of the discrete scenario, given the fact that the limit diverges [1,29]. Based on our knowledge, the capability of this solution combined with information/entropic differential measures for time anomaly detection has not yet been thoroughly explored in the literature. Telesca et al. [30] demonstrated the superiority of the Parzen window methodology over the histogram technique to estimate Fisher information and Shannon entropy for the time signal generated with a Gaussian process. To adapt the KDE transformation for specific data, the bandwidth optimization of the kernel is a crucial point. The main optimization solutions in the literature are designed for identically independently distributed (i.i.d.) data. Some works adapted KDE to time-series data [31], with specific inference parameter optimization designed for this type of data. Despite this, to the best of our knowledge, no one has designed an algorithm to select a bandwidth specifically optimized for time-series anomaly detection.

In light of all the above observations, in this work, we devise a novel methodology for estimating the continuous Probability Density Function (PDF) associated with an arbitrary time signal via KDE.

The application of overlapping windows allows for obtaining time-dependent entropy and information measures, freeing the window length scale parameter from the time step, which governs the resolution of the sequences of feature-extraction measures.

For window size selection, given the absence of a defined main periodicity in the time series and the unknown scale of malfunction, our approach employs multiple synchronous scales for window division. The utilization of different synchronous scales is crucial to enhancing sensitivity to malfunctions. Overestimation of the window size could overlook anomalies, while underestimation reduces the ability to capture long-term time dependencies in the time series, both normal and abnormal [20,32].

As an inference algorithm, we utilize KDE, and to select the bandwidth we devised an algorithm utilizing Jensen–Shannon Divergences (JSD), which act as metrics to balance underfitting and overfitting (or the bias-variance trade-off) of the final PDF concerning the data, that can be executed offline over a normal reference time signal. Because the bandwidth value and the window length are highly related, for each window length considered a bandwidth has to be selected.

To test our methodology, we designed synthetic experiments based on a multi-tone signal where, at a specific timestamp, the signal incorporates additional "anomaly harmonics". Afterwards, we also tested the methodology on a seizure-affected time signal from the CHB-MIT open dataset [33]. Our solution demonstrates effectiveness in synthetic

experiments, which makes it suitable to be part of a robust and adaptive framework for real-world applications.

The main contribution can be summarized as follows:

- The proposal of a methodology for feature extraction to transform a time series into a sequence of continuous entropic and information functionals.
- The introduction of a new algorithm to optimize the bandwidth of the Kernel Density Estimation, based on the Jensen–Shannon divergence, to balance the overfitting–underfitting trade-off across multiple PDFs.
- The implementation of the methodology for time-series anomaly detection, including:
  - A synthetic experiment, where a contextual collective anomaly is transformed into a new series where the anomaly can be simply detected with upper and lower control bands.
  - The final application in a real case of seizure detection using Scalp-EEG.

The article is structured as follows: The theory background is introduced in Section 2.1. The methodology is illustrated in Section 2.2. The synthetic experiment and the application in a real scenario are illustrated in Section 3. Finally, the discussion and conclusions are presented in Section 4.

2. Materials and Methods

2.1. Theory Background

2.1.1. Kernel Density Estimation

The transformation of a time-series window into a PDF is a crucial aspect of our methodology. There are various methods available in the literature for inferring PDFs from a time-series window, and each of them has its own drawbacks and/or limitations. Other techniques include frequency polygons, nearest neighbour methods, splines, restricted maximum likelihood estimators, and neural network-based density estimation [34,35]. When dealing with large amounts of data, it is often necessary to employ an automatic technique that does not require the inference parameters to be adjusted manually. In our case, we have chosen a non-parametric inference solution, the Kernel Density Estimation (KDE) method, also known as the Parzen window method [36]. The choice of a non-parametric approach allows one to bypass constraints that can arise from making prior assumptions, unlike a parametric approach. The KDE method involves estimating the PDF at a specific point by summing up the contributions of the experimental data points in its vicinity.

Giving a set of points, $\{x_i\}_{i=1}^{n}$, the equation that describes the KDE transformation is given by

$$\hat{p}(x) = \mathcal{K}_h[\{x_i\}_{i=1}^{n}] = \frac{1}{hn}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right), \tag{1}$$

where $\hat{p}(x)$ is the estimate probability distribution, n is the sample size, $K(x)$ is the kernel, and h is the kernel's bandwidth. The kernel, $K(x)$, is assumed to be an even regular function, with unit variance and zero mean. The method provides the elements needed to build a PDF from a random dataset (for a review on KDE solutions, see [27]). Between the two parameters, the bandwidth exerts a greater influence on the outcome compared to the choice of kernel, thereby reducing the kernel's impact on the results. However, in our specific case, the differentiability of the kernel is pivotal in ensuring the methodology's suitability for various differential entropic and information measures. Thus, in our case, we use the Gaussian one, which consists of a Gaussian distribution with zero mean and unit variance. Since the transform is the scaled summation of centred kernels to data points, the resulting PDF will inherit its main properties, such as infinite differentiability. Nevertheless, as mentioned before, more influential on the result than the chosen kernel is the bandwidth. The influence of varying bandwidth values on the estimation of the PDF using a synthetic dataset is visually depicted in Figure 1.

Numerous studies have been undertaken to determine the optimal bandwidth tailored to specific datasets [27].

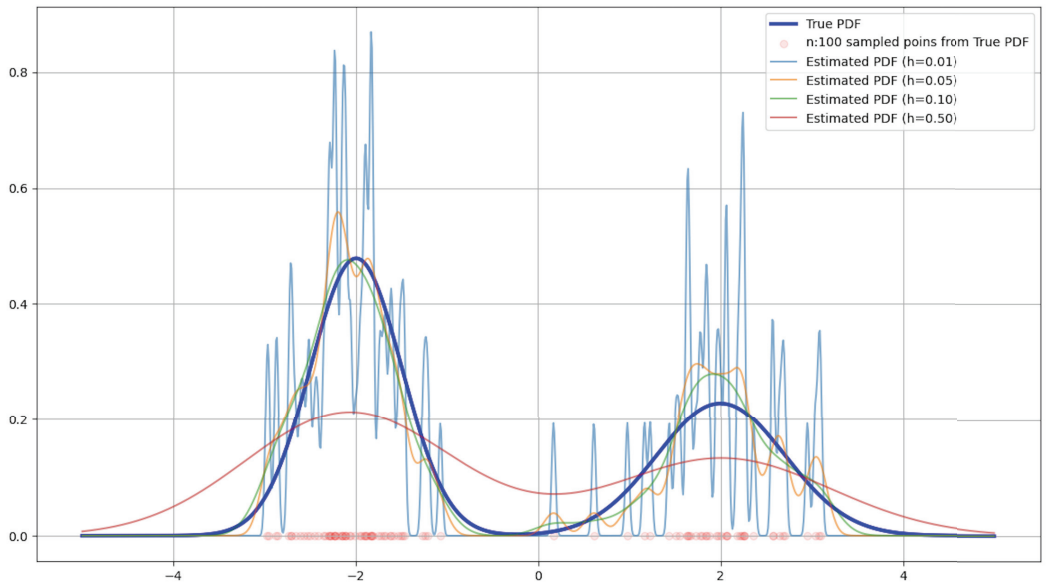


Figure 1. Applying the Kernel Density Estimation (KDE) method with various bandwidths (h) to a known Probability Density Function (PDF) (depicted by the blue line) using finite samplings (illustrated by red points). A low value of h leads to overfitting of the empirical data by the PDF, while a high value of h results in underfitting, leading to an overly smoothed distribution.

The importance of this (hyper)parameter relies on the fact that it is responsible for the delicate balance between overfitting and underfitting the data. Underestimated bandwidth (small h) leads to small bias and large variances, with the increased complexity of the resulting PDF that overfits the training set. Overestimated bandwidth (large h) leads to an increase in bias and small variances (e.g., underfitting). Hence, achieving this balance is crucial for a successful application of the KDE method [37]. Although the problem of the automatic selection of the kernel's bandwidth estimation has been explored by many authors, no procedure has yet been considered the best in every situation [27].

The more common bandwidth optimization solution relies on the minimization of the Mean Integrated Squared Error (MISE):

$$\mathrm{MISE}(h) = \mathbb{E}\left[\int_{-\infty}^{+\infty} |p(x) - \hat{p}_h(x)|^2 \, dx\right], \tag{2}$$

where $p(x)$, the real density, is unknown and $\mathbb{E}[f(x)] = \int_\Omega f(x)\rho(x)\,dx$ denotes the expected value of the continuous function, $f(x)$, with respect to the PDF, $\rho(x)$. Based on a certain assumption (AMISE assumption [37]; $p''(x)$ is continuous, square-integrable, and ultimately monotone independently and identically distributed (i.i.d.) data), the MISE can be rewritten using the Taylor series be expansion after decomposing the MISE into the variance and bias terms:

$$\mathrm{MISE}(h) = \mathrm{AMISE}(h) + o\left(\frac{1}{nh} + h^5\right), \tag{3}$$

where the Asymptotic Mean Integrated Squared Error (AMISE) is composed of

$$\mathrm{AMISE}(h) = \frac{1}{Nh} R(K) + \frac{1}{4} h^4 \mu_2(K)^2 R(p''), \tag{4}$$

and

$$R(g) = \int_{\mathbf{R}} g(x)^2 dx, \quad \mu_2(g) = \int_{\mathbf{R}} x^2 g(x) dx\,. \tag{5}$$

The left component of the AMISE corresponds to the variance; meanwhile, the right term corresponds to the bias. From the previous formulation, the h that minimizes the AMISE is given by

$$h = \left(\frac{R(K)}{n\sigma_K^4 R(p'')} \right)^{1/5}, \tag{6}$$

where σ_K denotes the standard deviation of the Kernel and p'' is the second derivative of p. Different h optimization solutions have been proposed using the AMISE. Based on simplistic assumptions, Silverman (7) and Scott (8) proposed the well-known heuristic rules,

$$h = 1.06 \times \min\left(\hat{\sigma}, \frac{IQR}{1.34} \right) \times n^{-1/5}, \tag{7}$$

$$h = \left(\frac{4\hat{\sigma}^5}{3n} \right)^{1/5}, \tag{8}$$

where $\hat{\sigma}$ is the sample standard deviation, IQR is the interquartile range, and n is the sample size.

Apart from that, more complex solutions are based on plug-in solutions, substituting the real p'' with its approximated version $\hat{p}''$ [38].

Regarding the specific application of KDE in time-series analysis, Harvey et al. [31] proposed the integration of kernel density estimation with weighted schemes derived from time-series analysis theory, where both bandwidth and the scale parameter are optimized together through maximum likelihood or likelihood cross-validation, respectively, for filtering and smoothing applications.

Despite these efforts, the optimization of h remains an open problem. In a recent article, Garcin [39] proposes complexity measures to strike a balance between overfitting and underfitting, particularly when applied to financial time-series data. The measure estimates a pseudo-distance to the maximum overfitting solution (the empirical distribution) and the maximum underfitting solution (a parametric Gaussian applied over the data). In this case, the window is fixed at one year in order to obtain a finance estimator. When dealing with time-series data, we must exercise caution in assuming independence and identical distribution, as inherent temporal dependencies and patterns may not be removable.

Moreover, in time-series anomaly detection applications, instead of focusing on individual instances, we need to evaluate shifts relative to a reference situation. Therefore, a quantitative measure that balances the overfitting–underfitting trade-off with respect to a set of reference PDFs, rather than a single PDF, would be more suitable for this scenario. To the best of the author's knowledge, such a strategy is not available in the literature.

2.1.2. Entropic Functionals and Fisher Non-Parametric Information

In this section, we explore various formulations of entropic functions, including Shannon entropy (9) [1], Tsallis entropy (10) [40], and Rényi entropy (11) [2], denoted by $\mathbb{H}[p]$, $\mathbb{H}_q[p]$, and $\mathbb{H}_\alpha[p]$, respectively. Here, $p(x)$ represents a PDF ($\int_\Omega p(x) = 1$ and $p(x) \geq 0 \ \ \forall x \in \Omega \subseteq \mathbb{R}$).

$$\mathbb{H}[p] = -\int_\Omega p(x) \ln p(x) dx, \tag{9}$$

$$\mathbb{H}_q[p] = \frac{1}{1-q}\left(1 - \int_\Omega p(x)^q dx \right) \quad q \in \mathbb{R}, \tag{10}$$

$$\mathbb{H}_\alpha[p] = \frac{1}{1-\alpha} \ln\left(\int_\Omega p(x)^\alpha dx \right) \quad \alpha \in \mathbb{R}. \tag{11}$$

Both Rényi and Tsallis entropies converge to Shannon entropy as $\alpha \to 1$ and $q \to 1$, respectively. Let us mention that these two entropies belong to the so-called "generalised

entropies" group, composed of entropic formulations that do not fulfil axiom 4 Shannon–Khinchin (see [41]).

The parameter q (or α in the case of Rényi entropy) plays a crucial role in determining the emphasis on the centre of mass or the tails of the probability distribution. When $q > 1$, the entropic puts more weight on frequent events, whereas for $q < 1$ it privileges rare events [42]. Therefore, adjusting the q parameter allows for a flexible exploration of different aspects of the PDF.

For more information about Tsallis entropy, please refer to [3]. For comprehensive insights into Shannon, Rényi, and Tsallis entropies, please consult [41].

Besides those generalised entropic functionals, another complementary measure has been utilised to analyse the signal, the so-called non-parametric Fisher information [1,13,43], defined as:

$$\mathbb{I}[p] = \int_{\Omega} \frac{\left(\frac{d}{dx}p(x)\right)^2}{p(x)}\, dx = \mathbb{E}\left[\left(\frac{\partial}{\partial x}\log p(x)\right)^2\right]. \tag{12}$$

Regarding the parametric version [1], here, θ is a location parameter. For (12) to be well-defined, it is assumed that $p(x)$ is differentiable and both $p(x)$ and $\frac{d}{dx}p(x)$ are quadratically integrable on $\mathbb{R}$ [44].

Fisher information is a non-negative functional that quantifies the average of the proportional change of $p(x)$ per unit change in x. Hence, Fisher information is able to detect the degree of oscillatory character of a given PDF.

It is essential to emphasize that finding an optimal entropic or information measure is not possible before analysing the actual data. Consequently, the most promising approach in order to apply entropic/information measures to analyse a time series involves combining several such quantities that relate to different aspects, whether structural or dynamic properties [19].

2.1.3. Jensen–Shannon Divergences Measure

Lin et al. [45] introduced a new divergence formulation, the discrete JSD. Unlike the well-known Kullback–Leibler divergence, the JSD is symmetric, bounded, and does not require absolute continuity between the distributions. Even more, satisfying the triangle inequality, the square root of the JSD qualifies as an actual mathematical metric [46].

$$\begin{aligned} JS^{\pi}(p_1, p_2) &= \pi_1 KL(p_1||m) + \pi_2 KL(p_2||m) = \\ &= \mathbb{H}[m] - \pi_1 \mathbb{H}[p_1] - \pi_2 \mathbb{H}[p_2], \end{aligned} \tag{13}$$

with $m(x) = \pi_1 p_1(x) + \pi_2 p_2(x)$ and $\boldsymbol{\pi}$ is a discrete Probability Mass Function (PMF). JSD is upper-bounded by the entropy of the weight distribution, $\boldsymbol{\pi}$, as

$$JS^{\pi}(p_1, p_2) \leq \mathbb{H}[\boldsymbol{\pi}]. \tag{14}$$

In the case of uniform weights, $\pi_1 = \pi_2 = 0.5$, $\mathbb{H}[\boldsymbol{\pi}] = \ln(2)$.

Moreover, JSD can be applied to more than two PDFs ($M \geq 2$),

$$\begin{aligned} JS^{\pi}(\{p_j\}_{j=1}^{M}) &= \sum_{j=1}^{M} \pi_j KL(p_j||m) = \\ &= \mathbb{H}\left[\sum_{j=1}^{M} \pi_j p_j\right] - \sum_{j=1}^{M} \pi_j \mathbb{H}[p_j]. \end{aligned} \tag{15}$$

Hence, the JSD can be formulated as the weighted average of Kullback–Leibler divergence of each PDF, $p_i(x)$, to the weighted average distribution $m = \sum_{j=1}^{M} \pi_j p_j$.

The upper bound of the JSD in (13) is given in terms of the entropy of the PDF weight distribution as

$$JS^{\pi}(\{p_j\}_{j=1}^{M}) \leq \mathbb{H}\Big[\{\pi_j\}_{j=1}^{M}\Big]. \tag{16}$$

2.2. Methodology

In this work, a methodology based on multiscale overlapping window division, KDE inference, and differential entropic/information measures is proposed to detect anomalies within a time signal.

For KDE optimization, a specific bandwidth offline algorithm selection has been designed specifically for this application scenario, based on the JSD, providing a metric to balance overfitting and underfitting about the reference time signal.

The multiscale approach is adopted because the scale of the window significantly impacts the transformation sensitivity and cannot be solely optimized without considering the specific anomaly [11]. Therefore, using several windows in parallel allows us to increase the range of scales analysed simultaneously, enhancing the ability to detect unknown anomalous behaviour [20].

An illustration of the methodology can be seen in Figure 2.

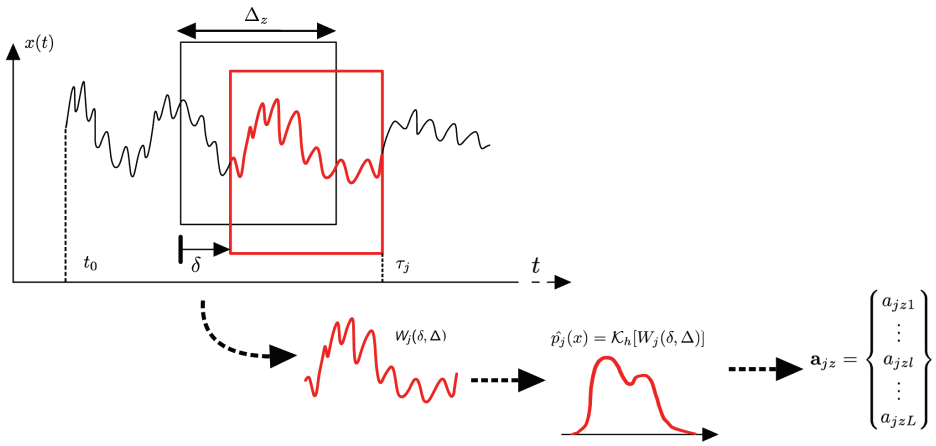


Figure 2. This figure illustrates the methodology employed. By implementing an overlapping window division with a window length of Δ_z and a window step of δ, a segment of the signal is extracted and then transformed into a PDF using a KDE inference solution. Subsequently, by applying various information/entropic measures, we derive an entropic/information measure vector, $\mathbf{a}_{jz}$.

2.2.1. Overlapping Window Divisions

Consider a discrete signal composed of N equispaced samples, $\{x_i\}_{i=1}^{N} = \{x(t_i) = x_i \in \mathbb{R},\ i = 1, \ldots, N\}$. The overlapping window division is defined through a sliding temporal window as:

$$W_j(\delta, \Delta) = \{x_i,\ i = 1 + j\delta - \Delta, \ldots, j\delta\}\,. \tag{17}$$

In this approach, $\Delta \in \mathbb{N}$, where $\Delta \leqslant N$ represents the window length and $\delta \in \mathbb{N}$ is the sliding factor. This separation distinguishes the resolution parameter, δ, from the scale coefficient, Δ. The subscript $j \in \left[0, \left\lfloor \frac{N-\Delta}{\delta} \right\rfloor\right] \cap \mathbb{N}$ denotes the temporal order of the windows, and the window temporal reference is defined as $\tau_j = t_0 + \frac{j\delta}{f_s}$, where f_s is the sampling frequency of the signal. This definition ensures that the temporal reference is independent of Δ, enabling the output of a multiscale synchronous sequence of entropic/information measures. Moreover, placing the time reference at the end of the window aligns with the moment when all information within the window becomes available, as illustrated in Figure 3. The case $\delta < \Delta$, overlapping windows, is the one considered in this work. The maximum resolution achievable by the transformation equals the original sampling frequency, attained when $\delta = 1$. Standardizing each window ensures that the subsequent transformation result is independent of the signal's absolute amplitude, $\tilde{x}_i$. This procedure is reiterated across a range of predefined window lengths, represented by $\mathbf{\Delta}$, to encompass a broad spectrum of scales. The approach described above enables the simultaneous

analysis of the signal with varying synchronous scales, avoiding any prior assumption of the optimal length of Δ, without considering the specific anomaly.

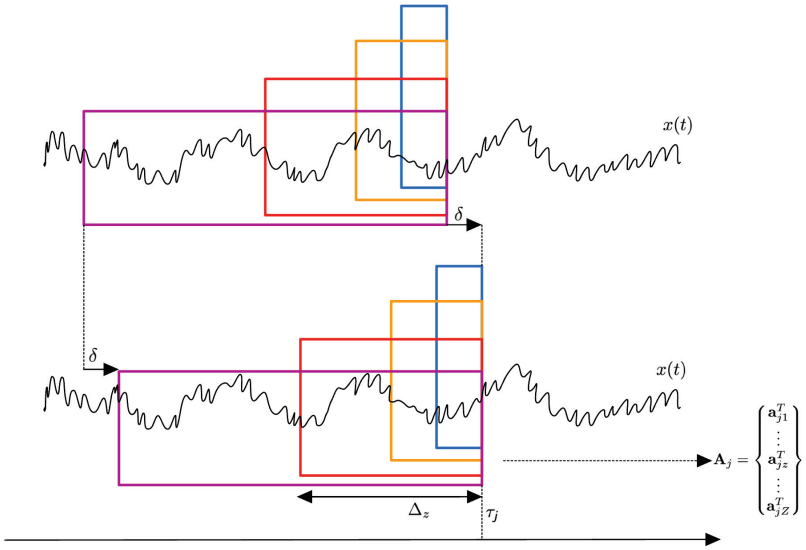


Figure 3. Illustration of the synchronous multiscale feature-extraction solution. At each time instance, τ_j, a matrix, $\mathbf{A}_j$, is constructed, containing L information/entropic measures for each of the Z window scales considered.

The time series is annotated with timestamps t_b and t_e, indicating the beginning and end of the anomaly, respectively. This annotation allows the assignment of a label to each window, indicating whether or not it is affected by the anomaly. A positive label, $y_j = 1$, is assigned if $t_b < \tau_j < t_e$, and a negative label, $y_j = 0$, otherwise .

2.2.2. Jensen–Shannon Divergence H-Selection Algorithm

In order to select the bandwidth, h, with respect to a reference time series, a solution based on the Jensen–Shannon divergence (13) has been designed. The main scope is to look for a balance between overfitting and underfitting over a reference time series not affected by the anomaly, and the JSD serves as an intuitive parameter that controls over- or under-smoothing of the KDE transformation, functioning as a similarity metric between PDFs. The main advantage of the proposed solution is that it does not make any assumptions about the nature of the data. Moreover, it can be executed offline, eliminating the need for online bandwidth optimization, which is a key problem when KDE is applied to time signals.

By using the JSD, we can control the bias-variance trade-off, not only for a single instance but for a set of instances to achieve a bandwidth suitable for all of them, making the transformation adapted to a reference time signal, and then be sensitive when the underlying condition of the system changes. The JSD measure is applied to PDFs obtained from a anomaly-free time signal with $y_j = 0$, recognised as the normal state. This approach, on one hand, allows an offline optimization of the bandwidth, making it unreliable over the malfunction signal and, on the other, reduces the risk of oversmoothing the differences between no anomaly and anomaly windows.

The JSD score utilised to pick the bandwidth, h, is defined as

$$S^{(JS)}(h, \Delta, \delta) = JS^{\pi}\left[\left\{\mathcal{K}_h[W_j(\delta, \Delta)]\right\}_{j=1}^{M^*}\right] = JS^{\pi}\left[\left\{\frac{1}{h\Delta}\sum_{i=\delta\cdot j}^{\delta\cdot j+\Delta} K\left(\frac{x - x_i}{h}\right)\right\}_{j=1}^{M^*}\right], \quad (18)$$

where M^* is the total number of anomaly-free PDFs.

Then, it is analysed how $S^{(JS)}$ changes when acting on one variable while keeping the other two constants. Starting with h, the score decreases monotonically as the bandwidth increases. This occurs because a wider kernel reduces the distance between the inferred PDFs. An illustrative example of this phenomenon can be observed in Figure 4. Specifically, as h approaches 0, the KDE tends to coincide with the empirical distribution, allowing for the computation of the exact maximum JSD score based on the relative frequency distribution. Conversely, as h approaches $+\infty$, the differences between the distributions are annealed, resulting in $S^{(JS)}(h, \Delta, \delta) = 0$. Further information is available in Appendix A.

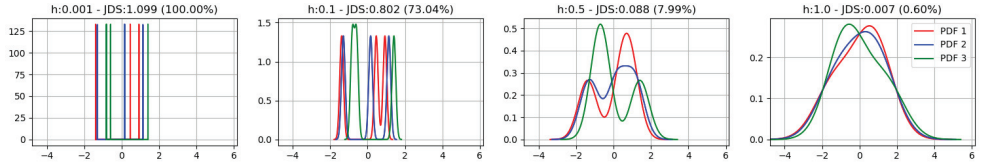


Figure 4. Jensen–Shannon Divergence (JSD) computed between three simple PDFs, represented by the blue, red, and green lines. These PDFs are generated using KDE applied to two distinct sets of points, with varying values of the smoothing parameter h. The maximum JSD in this scenario is $\log(3)$.

Shifting the focus to how $S^{(JS)}$ changes concerning Δ, it tends to decrease as a wider window is considered, accommodating more points in each instance. This trend aligns with results obtained by applying other, more classical techniques. However, this behaviour is not strictly monotonic, as the periodicities of the time signal could interact with the window scale in unpredictable ways. Lastly, the δ parameter has the role of a resolution scale. The optimal value for δ is 1, as it maximizes the number of windows extracted from a predefined time signal. However, to reduce computational costs, δ can be increased. Up to a certain limit, the score remains constant before becoming unstable. This limit value is dependent on the type of data being dealt with, as observed through numerical experiments.

Among the various information divergences, the choice of Jensen–Shannon Divergence (JSD) arises from its defining characteristics. Notably, JSD is non-negative, symmetric, and bounded. Moreover, it serves as a well-defined metric, whose square root satisfies the triangle inequality, and can effectively compute similarities among more than two distributions. This is not the case, for, e.g, the Kullback–Leibler and Jeffrey's divergences [47].

When determining the appropriate parameter for use, the hyperparameter th^{JS} becomes crucial. This hyperparameter governs the chosen value of the JSD, allowing the desired balance between bias and variance. The selection of th^{JS} can be accomplished through cross-validation when employing a classification algorithm, or by visually inspecting the resulting PDFs. The target score is computed as a percentage of the maximum value, which is contingent upon the number of normal state windows generated by the non-anomalous reference signal, M^*. Subsequently, for each Δ within the predefined multiscale vector, $\overline{\Delta}$, h^* is determined by taking into account its monotonically decreasing behaviour, which allows us to use the bisection method. Finally, a parameter, δ, is chosen to ensure score stability,

$$S^{(JS)}(h, \Delta) = th^{JS} \log M^* \rightarrow h^*(\Delta). \tag{19}$$

This approach allows us to control the bandwidth at each scale, Δ, with only one hyperparameter, th^{JS}.

With respect to the convergence to h^*, the main parameter influencing it is the number of windows considered, M^*, in the reference time signal. Additionally, the integration parameters, such as discretization in the case of Simpson's integral, and the minimum error allowed in the bisection method also affect the convergence.

2.2.3. Time Dependent Multiscale Entropy

In this context, we consider z window scales, i.e., $\overline{\Delta} \in \mathbb{N}^Z$. At each instance, τ_j, Z PDFs are generated. Over each PDF, a predetermined set of information/entropic measures is computed. Since determining the optimal information entropic measures in advance is challenging, and each one analyzes a different aspect of the time signal, employing a combination of such measures improves anomaly detection capabilities. Consequently, we obtain an ordered sequence of matrices $\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots, \boldsymbol{A}_M$, where $\boldsymbol{A}_j \in \mathbb{R}^{Z \times L}$. Each $\boldsymbol{a}_j \in \mathbb{R}^L$ comprises L distinct entropic or information outputs. The simultaneous use of various measures reveals unique characteristics of the underlying time signal dynamic [5,13].

3. Results

3.1. Synthetic Signal Generation

To assess the efficacy of the proposed algorithm, a synthetic simulation was devised. Based on the discussions highlighted in the Introduction, where the definition of anomaly can be subject to various interpretations, we observed a common trend in many application scenarios: anomalies are often associated with variations in signal frequency content. Therefore, in our synthetic simulation, the temporal signal has been crafted as a multi-tone time series, with anomalies represented as variations in the signal tones.

The equation describing our synthetic signal is

$$x(t) = g(t) \sum_{k=1}^{K_n} \mathbf{Re}\Big(A_k\, e^{-i(2\pi f_k t + \phi_k)}\Big) + (1 - g(t)) \sum_{k=1}^{K_a} \mathbf{Re}\Big(A_k^{(a)}\, e^{-i\left(2\pi f_k^{(a)} t + \phi_k^{(a)}\right)}\Big) + \epsilon(t), \tag{20}$$

which consists of a normal state signal, comprising K_n tones, and an anomaly signal composed of K_a tones. Each tone is characterized by an amplitude, A_k, a characteristic frequency, f_k, and a phase, ϕ_k. The design of Equation (20) allows the simulation of various types of anomalies. For instance, we can simply generate an anomaly signal by adding new tones, modifying the amplitude of the normal ones, or a combination of both. The transition between the two signals is controlled by a modulation function, $g(t)$, adding flexibility to the experiment. In particular, $g(t)$ can be a function gradually transitioning from the normal state to an anomalous one, or a localized function to pinpoint the anomaly at a specific time spam. The last term in Equation (20), $\epsilon(t) = \mathcal{N}(t|0, \sigma_\epsilon)$, represents white noise applied to the signal.

3.2. Synthetic Experiment Settings

The parameters associated with the methodology explained above are contained in Table 1, whereas the parameters used to generate the synthetic signal are listed in Table 2.

Table 1. Main transformation parameters employed to analyse the synthetic signal.

f_s	4096
δ	256
Δ	$2^{[4,5,\ldots,11]}$
th^{JS}	0.001

Table 2. Synthetic signal parameters. Time in [s] and frequencies in [Hz].

ϕ_k	440.0, 220.0, 22.0
A_k	1.5, 2.0, 1.0
$\phi_k^{(a)}$	440.0, 220.0, 22.0, 50.0, 1000.0
$A_k^{(a)}$	1.0, 1.0, 0.5, 2.0, 0.5
σ_ϵ	0.5

To define the parameters in these two tables, an artificial signal was first created. The anomaly was defined by adding higher and lower frequency harmonics to the signal. All

the parameters were adjusted to create a non-trivial time-series anomaly that could not be detected simply by applying upper and lower control limits.

The sampling frequency f_s was selected to be higher than the Nyquist rate of the signal to avoid aliasing problems. Other parameters were tuned to allow a clear representation of the anomaly. In particular, δ was kept as high as possible to reduce the computational cost, even though this reduced the resolution of the resulting sequence. Meanwhile, Δ was set to achieve the widest spectrum of scales while keeping the simulation within a reasonable duration. A power of 2 was employed for window scale selection. The threshold th^{JS} was determined through visual inspection of both the final time-dependent entropic/information plots.

For the anomaly signals, we devised a simulation in which, during the anomaly period, new tones are added to the original normal state signal, and the amplitude of the different tones is adjusted to maintain a constant total amplitude. The purpose of this approach was to create an anomaly that could not be easily detected using upper or lower threshold criteria. The normal and anomaly signal settings remained constant while different modulation signals $g(t)$ were tested. In this study, we present two cases: a linearly increasing modulation function, Equation (21), and a localized version represented by a normalized Gaussian modulation function, Equation (22).

$$\begin{cases} g(t) = 1 & \text{if } t \leq t_b \\ g(t) = 1 - \frac{t - t_b}{t_f - t_b} & \text{if } t > t_b, \end{cases} \tag{21}$$

$$\begin{cases} g(t) = 1 & \text{if } t \leq t_b \\ g(t) = 1 - \frac{f(t)}{\max(f(t))} & \text{if } t > t_b, \end{cases} \tag{22}$$

where t_b and t_f are the timestamps indicating the beginning and the end of the anomaly, respectively.

The resulting datasets can be observed in Figure 5, with linear anomaly characterized by $t_b = 5$ s and $t_f = 10$ s, and in Figure 6 for a localised anomaly. For the latter, the localization function, $f(t)$, is a normalised Gaussian with mean $\mu = 7.5$ s and standard deviation $\sigma = 0.1$ s.

In Figure 7, all the PDFs generated by the dataset are plotted, both with and without anomalies, at each scale. For each scale, a value, h^*, has been selected using the JSD-h algorithm. The PDFs obtained can exhibit multimodal distributions, as observed for higher Δ values. In this simulation, the selected h^* decreases monotonically with respect to Δ.

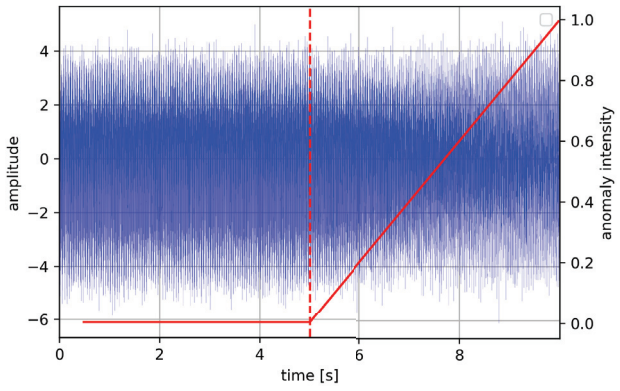


Figure 5. Example of a synthetic signal, where $g(t)$ is depicted with a continuous red line, representing the linear increasing anomaly function.

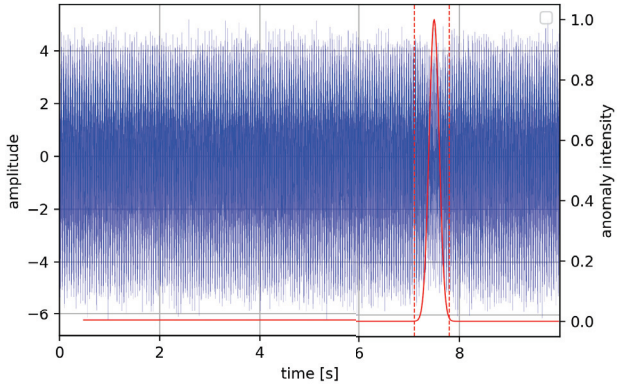


Figure 6. Example of a synthetic signal, where $g(t)$ is depicted with a continuous red line, representing the Gaussian localization anomaly function.

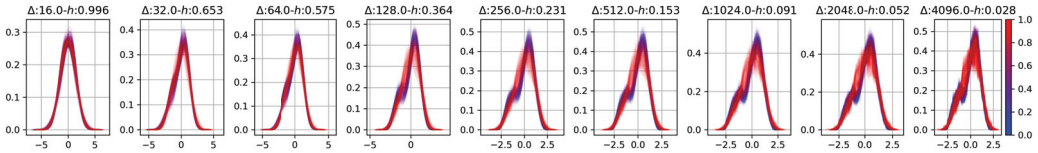


Figure 7. PDFs generated from the sythetic signal at different scales, Δ. A bandwidth, h, has been selected for each using the JSD-h algorithm. The red color gradient indicates the intensity of anomalies in the PDFs.

In Figure 8, the information/entropic time-dependent plots of the linear increasing anomaly case is depicted. Each subplot contains different entropic/information measures. For Tsallis and Rényi entropies, parameters q and α greater and lower than 1, respectively, have been chosen. In the case of Fisher information, the results are standardized to enable a proper graphic representation.

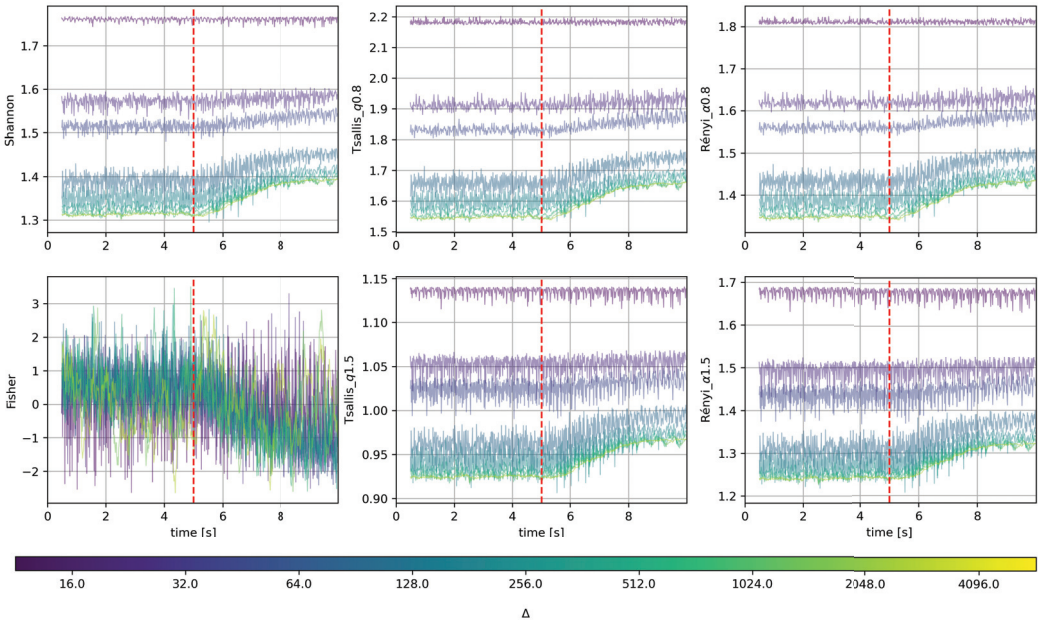


Figure 8. Entropic and information time-dependent plots related to the synthetic experiment. The color gradient indicates the Δ scale of the signal.

To begin with, a complementary behaviour of Fisher information can be observed compared to Shannon entropy. The anomaly generates an increase in the entropy component and a reduction in the information content, showing that the signal tends to enter into

a state of greater uncertainty, consistent with the structure of the underlying real signal, which is generated at that time by the combination of two multi-tonal signals. The non-parametric Fisher information measure method shows a less stable behaviour with respect to the entropic measures. Concerning the effect of the scales on the results, the substantial impact of parameter Δ becomes evident, significantly influencing both the signal's normal condition and its response to anomalies. Regarding the normal condition, an increase in Δ tends to lower the entropic levels and increase the information content of the signal. However, for high values of Δ, this tendency is stabilised at a specific entropic content. For anomaly response, in this specific simulation the measures tend to show a monotonic variation, reflecting the anomaly linear modulation signal, $g(t)$, applied (21). Analyzing the different results of the various measures, Rényi and Tsallis entropies with parameters lower than 1 show a less variable signal compared to the other measures. Nevertheless, in this specific case, all of them detected the anomaly.

Meanwhile, in Figure 9 the time-dependent plots with the localised anomaly are depicted. As in the previous case, the anomaly is detected by wider windows; however, it is possible to notice whether or not the window generates a delay in the detection of the anomaly. An important difference from the former case is that, when the localised anomaly is active, generalized entropies with parameters higher than 1 blur the anomaly detection for lower window scales. As before, the anomaly causes an increase in the uncertainty and a decrease in the information content of the signal. From an analysis of both experiments, it is clear how the methodology transforms the time signal containing the collective contextual anomaly into a new sequence, where the same anomaly can be easily detected using upper and lower control limits. Moreover, as can be seen in both cases, the intensity of the shift is related to the intensity of the anomaly itself.

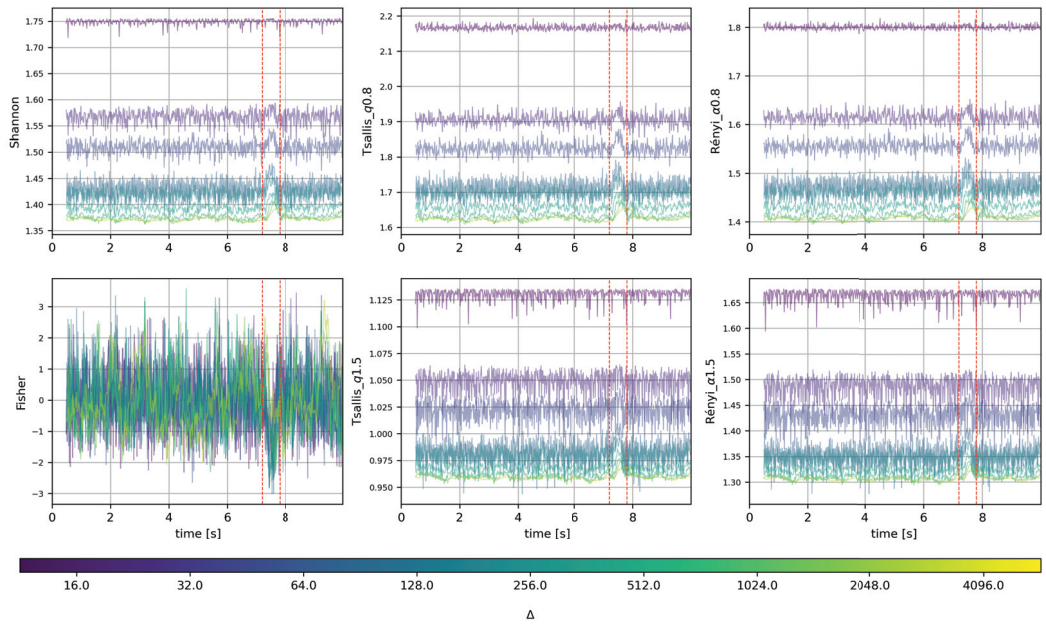


Figure 9. Entropic and information time-dependent plots related to the synthetic experiment. The color gradient indicates the Δ scale of the signal.

3.3. Real-Case Scenario Application: Scalp-EEG Seizure Detection

To evaluate the proposed methodology on real data, a test was conducted over a record of a specific patient from the open dataset Children's Hospital Boston CHB-MIT. This dataset comprises pediatric subjects with intractable seizures and is available at https://physionet.org/content/chbmit/1.0.0/ (accessed on 31 January 2023). According to the International League Against Epilepsy and the International Bureau for Epilepsy [48], *"an epileptic seizure is defined as a transient occurrence of signs and/or symptoms resulting from abnormal excessive or synchronous neuronal activity in the brain"*. Scalp Electroencephalogram

(EEG) is a non-invasive diagnostic tool widely used to assess brain activity, especially in the clinical diagnoses of epilepsy. By monitoring the brain's processes using electrodes attached to the scalp, EEG generates a signal that can detect abnormalities or changes in brain function. The signal in the CHB-MIT dataset is characterized by a sampling frequency of $f_s = 256$ Hz with a 16-bit resolution. For more information about the dataset, refer to [33].

The test was conducted on a single patient operating data from one channel of the EEG. Specifically, a record without a seizure (see Figure 10) was used as the normal state to select the bandwidths. Subsequently, time-dependent entropic/information measures were computed using a record affected by a seizure attack (see Figure 11). The parameters employed for this simulation are reported in Table 3.

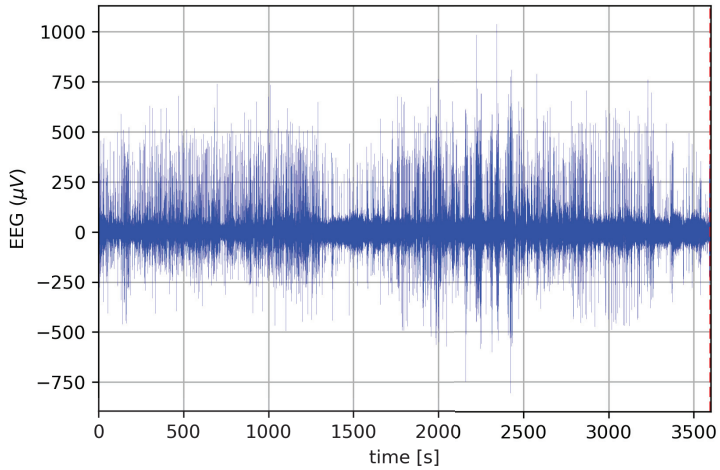


Figure 10. EEG signal: record chb01-01 Channel 1 (FP1-F7).

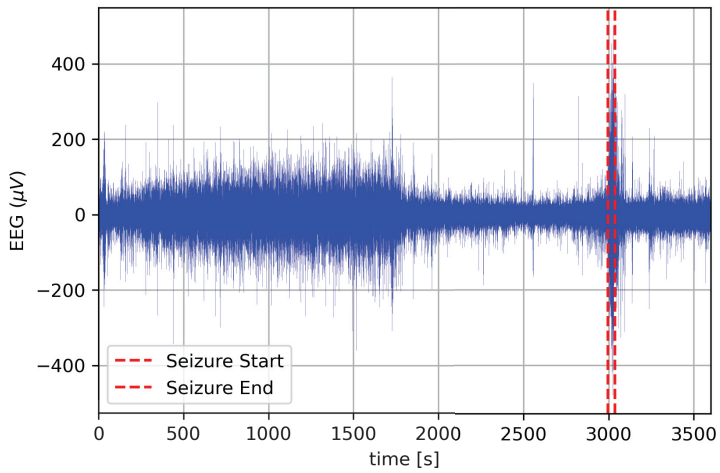


Figure 11. EEG signal: record chb01-03 Channel 1 (FP1-F7).

Table 3. Main transformation parameter employed to analyse chb01 scalp-EEG.

f_s	256
δ	256
Δ	$2^{[4,5,\ldots,13]}$
th^{JS}	0.01

In Figure 12, the selected combinations of $\Delta - h$ parameters are depicted along with all the PDFs generated in a healthy state. In this case, we notice that the selected values of

h do not decrease monotonically with Δ from the beginning, but instead maintain almost a constant value until $\Delta = 1028$ (or 4 s), after which they start to decrease.

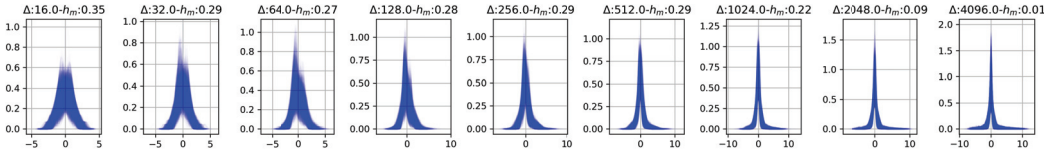


Figure 12. Plots of PDFs generated from the EEG signal at various scales, Δ. A bandwidth, h_m, has been selected for each scale using the JSD-h algorithm.

Observing the time-dependent plots in Figure 13, it can be seen how the largest windows recorded a decrease in entropy and an increase in information, right before and after the epileptic attack.

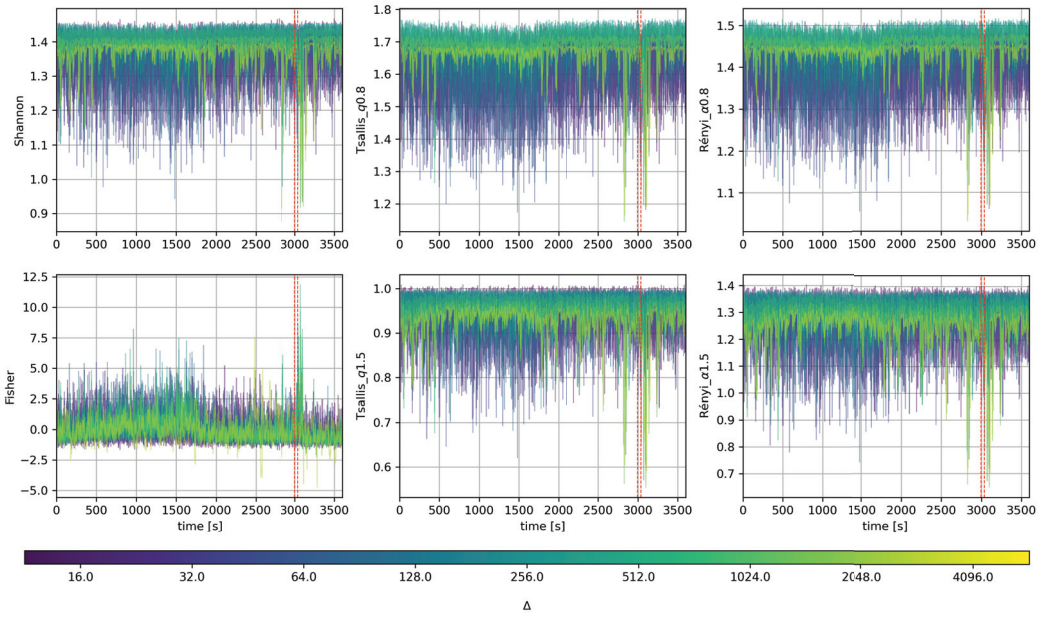


Figure 13. Entropic and information time-dependent plots related to the record chb01-03 Channel 1 (FP1-F7). The colour gradient indicates the Δ scale of the signal.

4. Discussion and Conclusions

The proposed methodology for detecting anomalies in time series represents a novel approach that combines multiscale window division with entropic/information time-dependent measurements. In the inference step, a Kernel Density Transformation (KDE) is utilized alongside a unique bandwidth selection algorithm based on the Jensen–Shannon divergence. This algorithm, tailored for anomaly detection, strikes a balance between bias and variance over a set of instances generated from a reference time signal for offline optimization. The main advantages of the proposed bandwidth optimization algorithm lie in the ability for it to be executed offline, without making assumptions about the nature of the data. This stands in contrast to classical optimization methods for h, which are designed for i.i.d. data. The offline characteristic eliminates the need for an online optimization of a specific bandwidth for each windowed time-series instance.

One of the key strengths of this methodology lies in its flexibility, allowing for the application of any differential entropic/information measures with a multiscale approach to characterize a time series. Unlike previous approaches that were often confined to specific application scenarios, our methodology offers a generalized framework for utilizing these feature-extraction tools across various domains.

Our results show the efficacy of this methodology in describing different types of anomalies through time-dependent differential entropy/information plots. Notably, the importance of scale in anomaly detection is highlighted, although it is acknowledged that wider windows may introduce a delay in detecting localized anomalies. Across the various generalized entropic measures, all proved capable of detecting anomalies with varying window sizes. Particularly noteworthy were the Shannon, Tsallis, and Rényi entropies with parameters lower than 1, which successfully identified localized anomalies, even with smaller windows. In both synthetic and real EEG experiments, the JSD-h algorithm effectively balanced bias and variance, selecting appropriate bandwidths for each scale to accurately characterize both the normal signal state and the anomalies.

5. Limitations and Future Works

The proposed methodology enables the extraction of any continuous entropic, informational, and complexity functionals from a time series, without application limitations when dealing with time signals. It can also be generalized beyond the measures used here.

Regarding potential difficulties with the methodology, there is no established method to determine the optimal set of entropic and informational measures before computing them over the data. For the JSD-h algorithm, one challenge is selecting the threshold th^{JS}, which is currently based on empirical tests and depends on the specific classification algorithm used. A potential solution might involve using a complexity distance instead of a divergence measure to find a specific minimum, though this would shift the challenge to determining which complexity measure to use. Another aspect that could be improved is the convergence speed of the JSD-h algorithm. While this optimization can be performed offline, tuning the parameters that influence convergence could speed up the algorithm's execution. Once optimized, it would eliminate the need for the instance-specific optimization of h, as required by other solutions.

Regarding possible improvements and the next applications of the methodology, the JSD-h algorithm could be improved by incorporating weighted parameters using filter methods, rather than a uniform weight scheme. This adjustment would allow the adaptation of the reference signal in non-stationary time series. Additionally, our methodology will address collective contextual anomalies, where anomalies can be characterized by shifts in frequency content. However, the method could also be applied to contextual point anomalies, as these should reflect significant changes in the PDF obtained without any data filtering or smoothing. Furthermore, while we have considered various entropic and informational functionals, we have not yet explored other types of measures, such as entropic and informational divergences, information planes, and complexities, which could potentially enhance our results. Future research will also aim to integrate this feature-extraction solution with machine-learning classification techniques in real-world scenarios, although this extension is beyond the scope of the current work.

Author Contributions: Conceptualization, A.S., E.V.T. and A.Z.; methodology, A.S., E.V.T. and A.Z.; software, A.S.; validation, A.S., E.V.T. and A.Z.; formal analysis, A.S., E.V.T. and A.Z.; investigation, A.S., E.V.T. and A.Z.; resources, A.S., E.V.T. and A.Z.; data curation, A.S.; writing—original draft preparation, A.S. and E.V.T.; writing—review and editing, A.S., E.V.T. and A.Z.; visualization, A.S.; supervision, E.V.T. and A.Z.; project administration, A.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The code necessary to reproduce all the experiments is available at https://github.com/antosquicciarini/Information_Measurement (accessed on 18 June 2024).

Acknowledgments: We would like to express our gratitude to the "Programa Propio" of the "Universidad Politécnica de Madrid" (UPM) for their support. Additionally, we extend our thanks to the GI-TACA UPM research group for their contributions to this work.

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

Abbreviations

The following abbreviations are used in this manuscript:

i.i.d.	independently and identically distributed
EEG	Electroencephalogram
KDE	Kernel Density Estimation
TDE	Time-Dependent Entropy
PMF	Probability Mass Function
PDF	probability density function
JSD	Jensen–Shannon Divergence
AMISE	Asymptotic Mean Integrated Squared Error

Appendix A. JSD Scores: Relation with h

As the bandwidth, h, tends to 0, the KDE of the data $\{x_i\}_{i=1}^n$, with $x_i \in \mathbb{R}$ for all i, collapses to the empirical density function, i.e.,

$$\lim_{h\to 0} \hat{p}(x) = \lim_{h\to 0} \frac{1}{hn} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) = \frac{1}{n} \sum_{i=1}^{n} \delta(x - x_i), \tag{A1}$$

where $\delta(x)$ denotes the Dirac distribution [49]. In this case, the Shannon entropy of the empirical density distribution is equal to

$$\begin{aligned} \mathbb{H}[\hat{p}(x)] &= -\int_{-\infty}^{\infty} \hat{p}(x) \log \hat{p}(x)\, dx = \\ &= -\int_{\Omega} \frac{1}{n} \sum_{i=1}^{n} \delta(x - x_i) \log \hat{p}(x)\, dx = \\ &= -\frac{1}{n} \sum_{i=1}^{n} \log \hat{p}(x_i) = \\ &= -\frac{1}{n} \log\left(\prod_{i=1}^{n} \hat{p}(x_i)\right). \end{aligned} \tag{A2}$$

Assuming $0 \log 0 = 0$, if the collection $\{x_i\}_{i=1}^n$ is a set (with only unique elements), $\hat{p}(x_i) = \frac{1}{n}$ for all i. Thus, in this specific case, $\mathbb{H}[\hat{p}(x)] = -\frac{1}{n} \log\left(\frac{1}{n}\right)^n = \log n$.

Considering two distinct empirical density functions, $\hat{p}^{(1)}(x)$ and $\hat{p}^{(2)}(x)$, generated by sets $\left\{x_i^{(1)}\right\}_{i=1}^n$ and $\left\{x_i^{(2)}\right\}_{i=1}^n$, each containing the same number of elements, if the union of the two sets is another set, and if the two sets are disjoint, the JSD will be given by

$$\begin{aligned} JSD[\hat{p}^{(1)}(x), \hat{p}^{(2)}(x)] &= \mathbb{H}[\bar{\hat{p}}(x)] - \tfrac{1}{2}\mathbb{H}[\hat{p}^{(1)}(x)] - \tfrac{1}{2}\mathbb{H}[\hat{p}^{(2)}(x)] \\ &= \log 2n - \tfrac{1}{2}\log n - \tfrac{1}{2}\log n = \log(2), \end{aligned} \tag{A3}$$

where, $\bar{\hat{p}}(x) = \frac{\hat{p}^{(1)}(x) + \hat{p}^{(2)}(x)}{2}$ is the empirical distribution of $\left\{x_i^{(1)}\right\}_{i=1}^n \cup \left\{x_i^{(2)}\right\}_{i=1}^n$.

Generalizing this result, $JSD\left[\left\{\hat{p}^{(j)}(x)\right\}_{j=1}^M\right] = \log(M)$. In case the sets overlap, $JSD\left[\left\{\hat{p}^{(j)}(x)\right\}_{j=1}^M\right] < \log(M)$.

In the opposite case, with $h \to +\infty$, one has

$$\begin{aligned} \lim_{h\to+\infty} \hat{p}(x) &= \lim_{h\to+\infty} \frac{1}{hn} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \\ &= K\left(\frac{x - \frac{\sum_{i=1}^n x_i}{n}}{h}\right). \end{aligned} \tag{A4}$$

In this situation, it is conjectured that the estimate retains the shape of the used kernel, centred on the mean of the samples (completely smooth).

Appendix B. Further Experiments

To have a representation of how the bandwidth influences the results of the PDFs, in Figure A1 a grid of values of $\boldsymbol{h} - \boldsymbol{\Delta}$ is shown. Each box displays PDFs generated both with and without anomalies. This plot vividly illustrates how the bandwidth serves as a crucial factor in striking a balance between overfitting and underfitting the data. When the bandwidth, h, is low, the distributions closely align with the empirical one. Conversely, with a larger bandwidth, the output of the kernel density estimator tends to mirror the kernel itself.

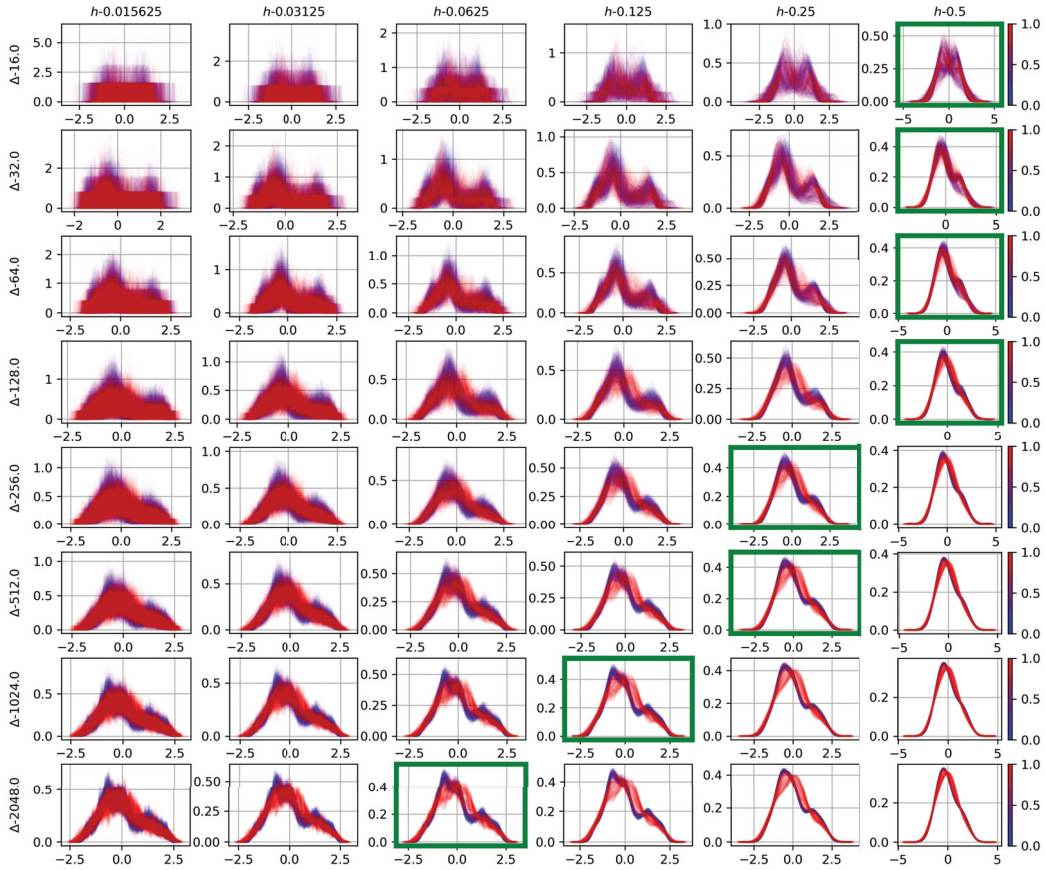


Figure A1. PDFs comparative grid. Each square shows the PDFs generated with that specific $\Delta - h$ pair. Normal state PDFs are depicted in blue, and anomaly PDFs are in red. The green borders show which is the best $\Delta - h$ combination closer to the JSD score utilised.

References

1. Cover, T.M.; Thomas, J.A. *Elements of Information Theory*; Wiley-Interscience: Hoboken, NJ, USA, 2006; p. 774.
2. Rényi, A. On Measures of Entropy and Information. In Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics, Los Angeles, CA, USA, 20 June–30 July 1960; University of California Press: Berkeley, CA, USA, 1961; Volume 4, pp. 547–562.
3. Tsallis, C. Entropic Nonextensivity: A Possible Measure of Complexity. *Chaos Solitons Fractals* **2002**, *13*, 371–391. [CrossRef]
4. Gupta, V.; Pachori, R.B. Epileptic Seizure Identification Using Entropy of FBSE Based EEG Rhythms. *Biomed. Signal Process. Control* **2019**, *53*, 101569. [CrossRef]
5. Rosso, O.; Martin, M.; Figliola, A.; Keller, K.; Plastino, A. EEG Analysis Using Wavelet-Based Information Tools. *J. Neurosci. Methods* **2006**, *153*, 163–182. [CrossRef] [PubMed]
6. Mateos, D.M.; Guevara Erra, R.; Wennberg, R.; Perez Velazquez, J.L. Measures of Entropy and Complexity in Altered States of Consciousness. *Cogn. Neurodyn.* **2018**, *12*, 73–84. [CrossRef] [PubMed]

7. Martin, M.T.; Pennini, F.; Plastino, A. Fisher's Information and the Analysis of Complex Signals. *Phys. Lett. A* **1999**, *256*, 173–180. [CrossRef]
8. Martin, M.; Plastino, A.; Plastino, A. Tsallis-like Information Measures and the Analysis of Complex Signals. *Phys. A Stat. Mech. Its Appl.* **2000**, *275*, 262–271. [CrossRef]
9. Lerga, J.; Saulig, N.; Stanković, L.; Seršić, D. Rule-Based EEG Classifier Utilizing Local Entropy of Time–Frequency Distributions. *Mathematics* **2021**, *9*, 451. [CrossRef]
10. Alkahtani, H.; Aldhyani, T.H.H.; Ahmed, Z.A.T.; Alqarni, A.A. Developing System-Based Artificial Intelligence Models for Detecting the Attention Deficit Hyperactivity Disorder. *Mathematics* **2023**, *11*, 4698. [CrossRef]
11. Bezerianos, A.; Tong, S.; Thakor, N. Time-Dependent Entropy Estimation of EEG Rhythm Changes Following Brain Ischemia. *Ann. Biomed. Eng.* **2003**, *31*, 221–232. [CrossRef]
12. Kalimeri, M.; Papadimitriou, C.; Balasis, G.; Eftaxias, K. Dynamical Complexity Detection in Pre-Seismic Emissions Using Nonadditive Tsallis Entropy. *Phys. A Stat. Mech. Its Appl.* **2008**, *387*, 1161–1172. [CrossRef]
13. Guignard, F.; Laib, M.; Amato, F.; Kanevski, M. Advanced Analysis of Temporal Data Using Fisher-Shannon Information: Theoretical Development and Application in Geosciences. *Front. Earth Sci.* **2020**, *8*, 255. [CrossRef]
14. Conejero, J.A.; Velichko, A.; Garibo-i-Orts, Ò.; Izotov, Y.; Pham, V.T. Exploring the Entropy-Based Classification of Time Series Using Visibility Graphs from Chaotic Maps. *Mathematics* **2024**, *12*, 938. [CrossRef]
15. Zhu, G.; Qiu, T.; Ding, Y.; Gao, S.; Zhao, N.; Liu, F.; Zhou, X.; Gururajan, R. Detecting Depression Using Single-Channel EEG and Graph Methods. *Mathematics* **2022**, *10*, 4177. [CrossRef]
16. Choi, K.; Yi, J.; Park, C.; Yoon, S. Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines. *IEEE Access* **2021**, *9*, 120043–120065. [CrossRef]
17. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly Detection: A Survey. *Acm Comput. Surv.* **2009**, *41*, 1–58. [CrossRef]
18. Bently, D.E.; Hatch'Charles, T. Fundamentals of Rotating Machinery Diagnostics. *Mech. Eng.-CIME* **2003**, *125*, 53–54.
19. Eftaxias, K.; Minadakis, G.; Athanasopoulou, L.; Kalimeri, M.; Potirakis, S.M.; Balasis, G. Are Epileptic Seizures Quakes of the Brain? An Approach by Means of Nonextensive Tsallis Statistics. *arXiv* **2011**, arXiv:1110.2169.
20. Farashi, S. A Multiresolution Time-Dependent Entropy Method for QRS Complex Detection. *Biomed. Signal Process. Control* **2016**, *24*, 63–71. [CrossRef]
21. Zhang, A.; Yang, B.; Huang, L. Feature Extraction of EEG Signals Using Power Spectral Entropy. In Proceedings of the 2008 International Conference on BioMedical Engineering and Informatics, Sanya, China, 27–30 May 2008; Volume 2, pp. 435–439. [CrossRef]
22. Ocak, H. Automatic Detection of Epileptic Seizures in EEG Using Discrete Wavelet Transform and Approximate Entropy. *Expert Syst. Appl.* **2009**, *36*, 2027–2036. [CrossRef]
23. Cao, Z.; Lin, C.T. Inherent Fuzzy Entropy for the Improvement of EEG Complexity Evaluation. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 1032–1035. [CrossRef]
24. Cao, Z.; Ding, W.; Wang, Y.K.; Hussain, F.K.; Al-Jumaily, A.; Lin, C.T. Effects of Repetitive SSVEPs on EEG Complexity Using Multiscale Inherent Fuzzy Entropy. *Neurocomputing* **2020**, *389*, 198–206. [CrossRef]
25. Xiang, J.; Li, C.; Li, H.; Cao, R.; Wang, B.; Han, X.; Chen, J. The Detection of Epileptic Seizure Signals Based on Fuzzy Entropy. *J. Neurosci. Methods* **2015**, *243*, 18–25. [CrossRef] [PubMed]
26. Patidar, S.; Panigrahi, T. Detection of Epileptic Seizure Using Kraskov Entropy Applied on Tunable-Q Wavelet Transform of EEG Signals. *Biomed. Signal Process. Control* **2017**, *34*, 74–80. [CrossRef]
27. Zambom, A.Z.; Dias, R. A Review of Kernel Density Estimation with Applications to Econometrics. *Int. Econom. Rev.* **2013**, *5*, 20–42.
28. Sánchez-Moreno, P.; Yanez, R.; Dehesa, J. Discrete Densities and Fisher Information. In *Difference Equations and Applications*; Bahçesehir University Press: Istanbul, Turkey, 2009.
29. Tabass, M.S.; Borzadaran, G.R.M.; AmiNi, M. Renyi Entropy in Continuous Case Is Not the Limit of Discrete Case. *Math. Sci. Appl. E-Notes* **2016**, *4*, 113–117. [CrossRef]
30. Telesca, L.; Lovallo, M. On the Performance of Fisher Information Measure and Shannon Entropy Estimators. *Phys. A Stat. Mech. Its Appl.* **2017**, *484*, 569–576. [CrossRef]
31. Harvey, A.; Oryshchenko, V. Kernel Density Estimation for Time Series Data. *Int. J. Forecast.* **2012**, *28*, 3–14. [CrossRef]
32. Choi, E.; Schuetz, A.; Stewart, W.F.; Sun, J. Using Recurrent Neural Network Models for Early Detection of Heart Failure Onset. *J. Am. Med. Inform. Assoc.* **2017**, *24*, 361–370. [CrossRef]
33. Shoeb, A.H.; Guttag, J.V. Application of Machine Learning To Epileptic Seizure Detection. In Proceedings of the 36th International Conference on Machine Learning, ICML 2019, Long Beach, CA, USA, 9–15 June 2019.
34. Izenman, A.J. Review Papers: Recent Developments in Nonparametric Density Estimation. *J. Am. Stat. Assoc.* **1991**, *86*, 205–224. [CrossRef]
35. Wang, Z.; Scott, D.W. Nonparametric Density Estimation for High-dimensional Data—Algorithms and Applications. *WIREs Comput. Stat.* **2019**, *11*, e1461. [CrossRef]
36. Parzen, E. On Estimation of a Probability Density Function and Mode. *Ann. Math. Stat.* **1962**, *33*, 1065–1076. [CrossRef]
37. Raykar, V.C.; Duraiswami, R. Fast Optimal Bandwidth Selection for Kernel Density Estimation. In Proceedings of the 2006 SIAM International Conference on Data Mining (SDM), Bethesda, MD, USA, 20–22 April 2006; pp. 524–528. [CrossRef]

38. Sheather, S.J.; Jones, M.C. A Reliable Data-Based Bandwidth Selection Method for Kernel Density Estimation. *J. R. Stat. Soc. Ser. B (Methodol.)* **1991**, *53*, 683–690. [CrossRef]
39. Garcin, M. Complexity Measure, Kernel Density Estimation, Bandwidth Selection, and the Efficient Market Hypothesis. *arXiv* **2023**, arXiv:2305.13123.
40. Tsallis, C.; Baldovin, F.; Cerbino, R.; Pierobon, P. Introduction to Nonextensive Statistical Mechanics and Thermodynamics. *arXiv* **2003**, arXiv:cond-mat/0309093.
41. Amigó, J.M.; Balogh, S.G.; Hernández, S. A Brief Review of Generalized Entropies. *Entropy* **2018**, *20*, 813. [CrossRef] [PubMed]
42. Tsallis, C.; Mendes, R.; Plastino, A. The Role of Constraints within Generalized Nonextensive Statistics. *Phys. A Stat. Mech. Its Appl.* **1998**, *261*, 534–554. [CrossRef]
43. Vignat, C.; Bercher, J.F. Analysis of Signals in the Fisher–Shannon Information Plane. *Phys. Lett. A* **2003**, *312*, 27–33. [CrossRef]
44. Bercher, J.F. On Escort Distributions, Q-gaussians and Fisher Information. *AIP Conf. Proc.* **2011**, *1305*, 208–215. [CrossRef]
45. Lin, J. Divergence Measures Based on the Shannon Entropy. *IEEE Trans. Inf. Theory* **1991**, *37*, 145–151. [CrossRef]
46. Endres, D.; Schindelin, J. A New Metric for Probability Distributions. *IEEE Trans. Inf. Theory* **2003**, *49*, 1858–1860. [CrossRef]
47. Jeffreys, S.H.; Jeffreys, S.H. *The Theory of Probability*, 3rd ed.; Oxford Classic Texts in the Physical Sciences; Oxford University Press: Oxford, NY, USA, 1998.
48. Fisher, R.S.; van Emde Boas, W.; Blume, W.; Elger, C.; Genton, P.; Lee, P.; Engel, J. Epileptic Seizures and Epilepsy: Definitions Proposed by the International League Against Epilepsy (ILAE) and the International Bureau for Epilepsy (IBE). *Epilepsia* **2005**, *46*, 470–472. [CrossRef] [PubMed]
49. Roth, V. Outlier Detection with One-class Kernel Fisher Discriminants. In *Advances in Neural Information Processing Systems*; MIT Press: Cambridge, MA, USA, 2004; Volume 17.

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

mathematics

MDPI

Article

An Improved Three-Term Conjugate Gradient Algorithm for Constrained Nonlinear Equations under Non-Lipschitz Conditions and Its Applications

Dandan Li [1], Yong Li [2] and Songhua Wang [2,*]

[1] School of Artificial Intelligence, Guangzhou Huashang College, Guangzhou 511300, China; liddjq@gdhsc.edu.cn
[2] School of Mathematics, Physics and Statistics, Baise University, Baise 533099, China; liyong@bsuc.edu.cn
* Correspondence: wang6600789@bsuc.edu.cn

Abstract: This paper proposes an improved three-term conjugate gradient algorithm designed to solve nonlinear equations with convex constraints. The key features of the proposed algorithm are as follows: (i) It only requires that nonlinear equations have continuous and monotone properties; (ii) The designed search direction inherently ensures sufficient descent and trust-region properties, eliminating the need for line search formulas; (iii) Global convergence is established without the necessity of the Lipschitz continuity condition. Benchmark problem numerical results illustrate the proposed algorithm's effectiveness and competitiveness relative to other three-term algorithms. Additionally, the algorithm is extended to effectively address the image denoising problem.

Keywords: nonlinear monotone equations; conjugate gradient method; convergence analysis; image denoising

MSC: 65K05; 65H10; 90C30; 90C56

1. Introduction

Consider the following constrained nonlinear monotone equations of the form:

$$E(x) = 0, \quad x \in \mathbb{E}, \tag{1}$$

where $E : \mathbb{R}^n \to \mathbb{R}^n$ is a monotonic and continuous mapping, and $\mathbb{E} \subseteq \mathbb{R}^n$ is a convex set. The monotonic property of the mapping is defined as

$$\langle E(x) - E(y), x - y \rangle \geq 0, \quad \forall x, y \in \mathbb{R}^n. \tag{2}$$

Numerous practical and theoretical problems can be transformed into nonlinear equations, such as those arising from nonlinear mathematical physics [1,2], compressed sensing [3,4], economic equilibrium [5], and optimal power flow control [6]. This broad applicability has driven extensive research into efficient solution methods. Among the various numerical methods that have been developed, derivative-free methods have gained significant attention due to their unique advantages. These methods include spectral gradient methods [7–9], two-term conjugate gradient methods [10–15], and three-term conjugate gradient methods [16–20]. To be specific, these methods leverage the structure of first-order optimization methods, inheriting the advantages of simplicity and low storage requirements, making them highly effective for solving a wide range of practical problems. However, it has been observed that the convergence properties of the aforementioned derivative-free methods often require mapping to satisfy the Lipschitz continuity condition, which is a stringent theoretical requirement. Hence, our goal in this paper stems from the

need to develop a more robust algorithm that operates under the non-Lipschitz continuity condition.

Before presenting our new algorithm, it is essential to review the three-term conjugate gradient method designed for unconstrained optimization problems, specifically those of the form $\min\{f(x) \mid x \in \mathbb{R}^n\}$. Here, $f : \mathbb{R}^n \to \mathbb{R}$ represents a continuously differentiable function, with its gradient at any point $x_k \in \mathbb{R}^n$ denoted by $g_k := \nabla f(x_k)$. The iterative formula for the three-term conjugate gradient method can be formulated as follows:

$$x_{k+1} = x_k + \alpha_k d_k, \quad d_k = -g_k + \tilde{\beta}_k d_{k-1} + \tilde{\theta}_k y_{k-1}, \quad k \geq 1, \quad d_0 = -g_0,$$

where α_k is the step length determined by a specific line search formula, $\tilde{\beta}_k$ and $\tilde{\theta}_k$ are scalar parameters, and $y_{k-1} = g_k - g_{k-1}$. The choice of $\tilde{\beta}_k$ and $\tilde{\theta}_k$ is critical, as different values of these parameters lead to different variants of the three-term conjugate gradient method [21–23]. Recently, leveraging the memoryless BFGS approach, Li [24] developed a three-term Hestense–Stiefel (HS)-type conjugate gradient for unconstrained optimization problems. This method's search direction closely approximates that of the memoryless BFGS method, offering improved performance and robustness. Additionally, Li [25] introduced a three-term Polak–Ribière–Polyak (PRP)-type conjugate gradient method, which modified the search direction by replacing $\langle d_{k-1}, y_{k-1}\rangle$ with $\|g_{k-1}\|^2$, thereby enhancing the efficiency in solving optimization problems. Furthermore, through comprehensive analysis [24,25], Li [26] developed a family of three-term conjugate gradient methods for unconstrained optimization problems. A notable feature of these methods is that their search directions consistently satisfy the sufficient descent property, ensuring reliable and effective convergence. Hence, our goal for this paper was to extend and modify these methods for solving nonlinear monotone equations with constraints.

Drawing inspiration from three-term conjugate gradient methods [24–26] and the projection technique, our goal was to extend these methods and propose an improved three-term conjugate gradient projection algorithm to solve the problem (1) without requiring the Lipschitz continuity condition. The advantages of our proposed algorithm are multifaceted, addressing several key challenges in solving nonlinear equations with convex constraints: minimal requirements, eliminates the need for line search formulas, global convergence without Lipschitz continuity, effective and competitive performance, and extension to image denoising. The remainder of this paper is structured as follows: In Section 2, we detail the process of the proposed algorithm. Section 3 is dedicated to establishing the convergence analysis of the proposed algorithm. Sections 4 and 5 present numerical experiments for nonlinear monotone equations with convex constraints and the image denoising problem, respectively. Finally, the conclusions are given in Section 6. Throughout the paper, the symbols $\|\cdot\|$ and $\langle\cdot,\cdot\rangle$ denote the Euclidean norm and the product of two vectors. For convenience, we abbreviate $E(x_k)$ to E_k.

2. Algorithm

In this section, we detail the formulation of our proposed algorithm, outlining its mathematical foundation and the derivation of key parameters. We start by defining the search direction and associated parameters that ensure efficiency and robustness. In addition, we provide a step-by-step description of the algorithm and discuss the theoretical underpinnings of the designed search direction.

To facilitate our formulation, we define several key parameters as follows: We introduce a notation $\tilde{y}_{k-1}$, which is given by [27]

$$\tilde{y}_{k-1} = y_{k-1} + v_{k-1}\|E_{k-1}\|^{a_1} d_{k-1},$$

where $v_{k-1} = a_2 + \max\{0, -\frac{\langle d_{k-1}, y_{k-1}\rangle}{\|d_{k-1}\|^2}\}\|E_{k-1}\|^{-a_1}$ and $y_{k-1} = E_k - E_{k-1}$ with $a_1, a_2 > 0$. These parameters play a crucial role in the formulation of our proposed search direction. After making a careful modification, we propose the following search direction:

$$d_k = \begin{cases} -E_0, & k = 0, \\ -E_k + \beta_k d_{k-1} + \theta_k \tilde{y}_{k-1} & k \geq 1. \end{cases} \tag{3}$$

Here, the coefficients β_k and θ_k are defined by the following expressions:

$$\beta_k = \frac{\langle E_k, \tilde{y}_{k-1} \rangle}{\varpi_k} - \frac{\|\tilde{y}_{k-1}\|^2 \langle E_k, d_{k-1} \rangle}{\varpi_k^2}$$

and

$$\theta_k = \frac{\delta_k \langle E_k, d_{k-1} \rangle}{\varpi_k},$$

where $\varpi_k = b_1(\|d_{k-1}\| + \|\tilde{y}_{k-1}\|)^2 + b_2 \max\{\|E_{k-1}\|^2, \langle d_{k-1}, \tilde{y}_{k-1} \rangle\}$ with $0 \leq \delta_k \leq \bar{\delta} < 1$ and $b_1, b_2 > 0$. Note that the inclusion of ϖ_k can be mathematically justified by its role in ensuring the sufficient descent property and trust-region characteristics. These properties are essential for the global convergence of the algorithm.

Before detailing our algorithm, it is essential to define the projection operator, which ensures the feasibility of our solutions. The projection operator is defined as follows:

$$T_{\mathbb{E}}[x] = \arg\min\{\|x - y\| \mid y \in \mathbb{E}\}, \quad x \in \mathbb{R}^n.$$

Projecting x onto the closed convex set $\mathbb{E}$ guarantees that the subsequent iterative point determined by our algorithm remains within the set $\mathbb{E}$. Additionally, this operator possesses a well-known non-expansive property, which can be expressed as

$$\|T_{\mathbb{E}}[x] - T_{\mathbb{E}}[y]\| \leq \|x - y\|, \quad \forall x, y \in \mathbb{R}^n. \tag{4}$$

Now, we illustrate the steps of our algorithm designed to efficiently solve nonlinear monotone equations subject to convex constraints. For convenience, Algorithm 1 is referred to as Algorithm ITTCG.

Algorithm 1 Improved Three-Term Conjugate Gradient Algorithm

Step 0. Choose $\sigma, \rho \in (0,1)$, $\xi \in (0,2)$, $a_1, a_2, b_1, b_2, \varepsilon > 0$, $\bar{\delta} \in (0,1)$, and an initial point $x_0 \in \mathbb{R}^n$. Set $k := 0$.
Step 1. Set $d_k = -E_k$.
Step 2. Set the trial point $z_k = x_k + \alpha_k d_k$, where the step length $\alpha = \max\{\rho^i \mid i = 0, 1, \ldots, \}$ satisfies

$$-\langle E(z_k), d_k \rangle \geq \sigma \alpha_k \|E(z_k)\| \|d_k\|^2. \tag{5}$$

Step 3. If $z_k \in \mathbb{E}$ and $\|E(z_k)\| \leq \varepsilon$, $x_{k+1} := z_k$ and stop. Otherwise, continue to Step 4.
Step 4. Compute the next iterative point as

$$x_{k+1} = T_{\mathbb{E}}[x_k - \xi \tau_k E(z_k)], \quad \tau_k = \frac{\langle E(z_k), x_k - z_k \rangle}{||E(z_k)||^2}.$$

Step 5. If $\|E(x_{k+1})\| \leq \varepsilon$, stop. Otherwise, compute the search direction d_{k+1} by (3).
Step 6. Set $k := k + 1$ and go to Step 2.

Remark 1. *Based on the definitions of $\tilde{y}_{k-1}$ and v_{k-1}, we can derive the following expression:*

$$\begin{aligned} \langle d_{k-1}, \tilde{y}_{k-1} \rangle &= \langle d_{k-1}, y_{k-1} \rangle + v_{k-1} \|E_{k-1}\|^{a_1} \|d_{k-1}\|^2 \\ &\geq \langle d_{k-1}, y_{k-1} \rangle + a_2 \|E_{k-1}\|^{a_1} \|d_{k-1}\|^2 - \frac{\langle d_{k-1}, y_{k-1} \rangle}{\|d_{k-1}\|^2} \|E_{k-1}\|^{-a_1} \|E_{k-1}\|^{a_1} \|d_{k-1}\|^2 \\ &= a_2 \|E_{k-1}\|^{a_1} \|d_{k-1}\|^2 > 0. \end{aligned}$$

This derivation shows that $\langle d_{k-1}, \tilde{y}_{k-1} \rangle$ is always positive. Consequently, this implies that the definitions of β_k and θ_k are valid and feasible within the context of our algorithm.

The following lemma indicates that the search direction determined by Algorithm ITTCG meets both the sufficient descent and trust-region properties. These properties are crucial for establishing the global convergence of Algorithm ITTCG.

Lemma 1. *Let the sequences $\{d_k\}$ and $\{E_k\}$ be determined by Algorithm ITTCG. Then, we have the following results:*

$$\langle E_k, d_k\rangle \leq -c_1\|E_k\|^2 \tag{6}$$

and

$$c_1\|E_k\| \leq \|d_k\| \leq c_2\|E_k\|, \tag{7}$$

where $c_1 = 1 - \frac{(1+\bar{\delta})^2}{4}$ *and* $c_2 = 1 + \frac{1+\bar{\delta}}{4b_1} + \frac{1}{16b_2^2}$.

Proof. (i) We will show that (6) holds. For $k = 0$, we have $\langle E_0, d_0\rangle = -\|E_0\|^2 \leq -c_1\|E_0\|^2$. For $k \geq 1$, using the search direction defined in (3), we obtain

$$\begin{array}{rcl} \langle E_k, d_k\rangle & = & -\|E_k\|^2 + \beta_k\langle E_k, d_{k-1}\rangle + \theta_k\langle E_k, \tilde{y}_{k-1}\rangle \\ & = & -\|E_k\|^2 + \frac{\langle E_k,\tilde{y}_{k-1}\rangle\langle E_k,d_{k-1}\rangle}{\omega_k} - \frac{\|\tilde{y}_{k-1}\|^2\langle E_k,d_{k-1}\rangle^2}{\omega_k^2} + \frac{\delta_k\langle E_k,d_{k-1}\rangle\langle E_k,\tilde{y}_{k-1}\rangle}{\omega_k} \\ & = & -\|E_k\|^2 + (1+\delta_k)\frac{\langle E_k,d_{k-1}\rangle\langle E_k,\tilde{y}_{k-1}\rangle}{\omega_k} - \frac{\|\tilde{y}_{k-1}\|^2\langle E_k,d_{k-1}\rangle^2}{\omega_k^2}. \end{array} \tag{8}$$

In addition, using the inequality $2\langle e_k, l_k\rangle \leq \|e_k\|^2 + \|l_k\|^2$ with $e_k = \frac{1+\delta_k}{2}E_k$ and $l_k = \frac{\langle E_k,d_{k-1}\rangle}{\omega_k}\tilde{y}_{k-1}$, we obtain

$$(1+\delta_k)\frac{\langle E_k, d_{k-1}\rangle\langle E_k, \tilde{y}_{k-1}\rangle}{\omega_k} \leq \frac{(1+\delta_k)^2}{4}\|E_k\|^2 + \frac{\langle E_k, d_{k-1}\rangle^2}{\omega_k^2}\|\tilde{y}_{k-1}\|^2. \tag{9}$$

Substituting (9) into (8), we have

$$\langle E_k, d_k\rangle \leq -\|E_k\|^2 + \frac{(1+\delta_k)^2}{4}\|E_k\|^2 \leq -\left(1 - \frac{(1+\bar{\delta})^2}{4}\right)\|E_k\|^2.$$

(ii) We will show that (7) holds. For $k = 0$, we have $c_1\|E_0\| \leq \|d_0\| = \|E_0\| \leq c_2\|E_0\|$. For $k \geq 1$, from the definition of ω_k and using the inequality $(e-l)^2 = e^2 - 2el + l^2 = (e+l)^2 - 4el \geq 0$, we obtain

$$\omega_k \geq 4b_1\|d_{k-1}\|\|\tilde{y}_{k-1}\|.$$

Using this relation and the definitions of β_k and θ_k, we obtain

$$|\beta_k| \leq \frac{\|E_k\|\|\tilde{y}_{k-1}\|}{4b_1\|d_{k-1}\|\|\tilde{y}_{k-1}\|} + \frac{\|\tilde{y}_{k-1}\|^2\|E_k\|\|d_{k-1}\|}{(4b_1\|d_{k-1}\|\|\tilde{y}_{k-1}\|)^2} \leq \left(\frac{1}{4b_1} + \frac{1}{16b_1^2}\right)\frac{\|E_k\|}{\|d_{k-1}\|}$$

and

$$\theta_k \leq \frac{\delta_k\langle E_k, d_{k-1}\rangle}{4b_1\|d_{k-1}\|\|\tilde{y}_{k-1}\|} \leq \frac{\bar{\delta}\|E_k\|\|d_{k-1}\|}{4b_1\|d_{k-1}\|\|\tilde{y}_{k-1}\|} \leq \frac{\bar{\delta}\|E_k\|}{4b_1\|\tilde{y}_{k-1}\|}.$$

Combining these inequalities with the definition of d_k, we obtain

$$\begin{array}{rcl} \|d_k\| & \leq & \|E_k\| + \left(\frac{1}{4b_1} + \frac{1}{16b_1^2}\right)\frac{\|E_k\|}{\|d_{k-1}\|}\|d_{k-1}\| + \frac{\bar{\delta}\|E_k\|}{4b_1\|\tilde{y}_{k-1}\|}\|\tilde{y}_{k-1}\| \\ & \leq & \left(1 + \frac{1+\bar{\delta}}{4b_1} + \frac{1}{16b_2^2}\right)\|E_k\|. \end{array}$$

Additionally, together with (6), we have

$$-\|E_k\|\|d_k\| \leq \langle E_k, d_k\rangle \leq -c_1\|E_k\|^2,$$

which implies that $\|d_k\| \geq c_1\|E_k\|$. □

3. Convergence Analysis

In this section, we analyze the global convergence of the proposed algorithm without assuming the Lipschitz continuity condition. We assume that $E(x) \neq 0$ for any $x \notin \mathbb{E}_*$, where $\mathbb{E}_*$ represents the solution set of problem (1). If $E(x) = 0$ for some $x \in \mathbb{E}_*$, this indicates that the solution to problem (1) has already been achieved.

The following lemma indicates that the line search Formula (5) of the proposed algorithm is well-defined.

Lemma 2. *Let the sequences $\{d_k\}$ and $\{x_k\}$ be generated by Algorithm ITTCG. Then, in each iteration, there exists a step length α_k that satisfies the line search Formula* (5)*.*

Proof. We begin by contradiction and assume that there exists $k_0 \geq 0$ such that the line search formula (5) does not hold for any non-negative integer i, i.e.,

$$-\langle E(x_{k_0} + \rho^i d_{k_0}), d_{k_0}\rangle < \sigma\rho^i \|E(x_{k_0} + \rho^i d_{k_0})\|\|d_{k_0}\|^2.$$

Given the continuity of E and the fact that $0 < \rho < 1$, we take the limit as $i \to \infty$ and obtain the relation $\langle E(x_{k_0}), d_{k_0}\rangle \geq 0$. This contradicts with $\langle E(x_{k_0}), d_{k_0}\rangle \leq -c_1\|E(x_{k_0})\|^2 < 0$ from (6). Therefore, there must be a step length α_k that satisfies the line search formula. □

The following lemma indicates that the sequence $\{x_k\}$ generated by Algorithm ITTCG is monotonic with respect to the solution from the set $\mathbb{E}_*$ of problem (1).

Lemma 3. *Let the sequences $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm ITTCG, then we have*

$$\|x_{k+1} - x_*\|^2 \leq \|x_k - x_*\|^2 - \sigma^2(2\xi - \xi^2)\|x_k - z_k\|^4, \quad x_* \in \mathbb{E}_*. \tag{10}$$

Moreover, the sequence $\{x_k\}$ is bounded.

Proof. From the inequality (2), we have

$$\begin{array}{rcl} \langle E(z_k), x_k - x_*\rangle & = & \langle E(z_k), x_k - z_k\rangle + \langle E(z_k), z_k - x_*\rangle \\ & \geq & \langle E(z_k), x_k - z_k\rangle + \langle E(x_*), z_k - x_*\rangle \\ & = & \langle E(z_k), x_k - z_k\rangle \\ & \geq & \sigma\alpha_k^2\|E(z_k)\|\|d_k\|^2, \end{array} \tag{11}$$

where the second inequality follows from the definition of z_k and the search line Formula (5). Additionally, using the definition of τ_k and the inequality (11), we have

$$\tau_k = \frac{\langle E(z_k), x_k - z_k\rangle}{\|E(z_k)\|^2} \geq \frac{\sigma\alpha_k^2\|E(z_k)\|\|d_k\|^2}{\|E(z_k)\|^2} = \frac{\sigma\alpha_k^2\|d_k\|^2}{\|E(z_k)\|}. \tag{12}$$

Utilizing the inequalities (4), (11), and (12), we have

$$\begin{array}{rcl} \|x_{k+1} - x_*\|^2 & = & \|T_{\mathbb{R}}[x_k - \xi\tau_k E(z_k)] - T_{\mathbb{R}}[x_*]\|^2 \\ & \leq & \|x_k - \xi\tau_k E(z_k) - x_*\|^2 \\ & = & \|x_k - x_*\|^2 - 2\xi\tau_k\langle E(z_k), x_k - x_*\rangle + \xi^2\tau_k^2\|E(z_k)\|^2 \\ & \leq & \|x_k - x_*\|^2 - 2\xi\tau_k\langle E(z_k), x_k - z_k\rangle + \xi^2\tau_k^2\|E(z_k)\|^2 \\ & = & \|x_k - x_*\|^2 - 2\xi\tau_k^2\|E(z_k)\|^2 + \xi^2\tau_k^2\|E(z_k)\|^2 \\ & = & \|x_k - x_*\|^2 - (2\xi - \xi^2)\tau_k^2\|E(z_k)\|^2 \\ & \leq & \|x_k - x_*\|^2 - (2\xi - \xi^2)\left(\frac{\sigma\alpha_k^2\|d_k\|^2}{\|E(z_k)\|}\right)^2\|E(z_k)\|^2 \\ & = & \|x_k - x_*\|^2 - (2\xi - \xi^2)\sigma^2\alpha_k^4\|d_k\|^4 \\ & = & \|x_k - x_*\|^2 - (2\xi - \xi^2)\sigma^2\|x_k - z_k\|^4, \end{array} \tag{13}$$

which implies that the sequence $\{\|x_k - x_*\|\}$ is monotonically non-increasing and convergent. Hence, the sequence $\{x_k\}$ is bounded. □

To be specific, if the sequence $\{x_k\}$ is finite, then the last iterative point is the solution to problem (1). If the sequence $\{x_k\}$ is infinite, we assume this to prove the following result:

Theorem 1. *Let the sequences $\{x_k\}$, $\{z_k\}$, $\{d_k\}$, and $\{E_k\}$ b generated by Algorithm ITTCG, then we have*

$$\lim_{k\to+\infty} \inf \|E_k\| = 0. \tag{14}$$

Proof. We begin by contradiction and assume that there exists a constant $\epsilon_1 > 0$ such that $\|E_k\| > \epsilon_1$ for any $k \geq 0$. This, combined with (7), yields

$$\|d_k\| \geq c_1\|E_k\| > c_1\epsilon_1, \quad \forall k \geq 0. \tag{15}$$

According to the continuity of E and the boundedness of $\{x_k\}$, the sequence $\{E_k\}$ is also bounded. That is, there exists a non-negative constant ϵ_2 such that $\|E_k\| \leq \epsilon_2$ for any $k \geq 0$. This, combined with (7), yields

$$\|d_k\| \leq c_2\|E_k\| \leq c_2\epsilon_2, \quad \forall k \geq 0. \tag{16}$$

The inequalities (15) and (16) imply that the sequence $\{d_k\}$ is bounded.

Moreover, from (10), we deduce that

$$\sum_{k=0}^{\infty} \|x_k - z_k\|^4 \leq \frac{1}{\sigma^2(2\xi - \xi^2)} \sum_{k=0}^{\infty} \Big(\|x_k - x_*\|^2 - \|x_{k+1} - x_*\| \Big) \leq \frac{\|x_0 - x_*\|^2}{\sigma^2(2\xi - \xi^2)},$$

which implies that

$$\lim_{k\to\infty} \|x_k - z_k\| = \lim_{k\to\infty} \alpha_k\|d_k\| = 0.$$

Together with the boundedness of the sequence $\{d_k\}$, it follows that

$$\lim_{k\to\infty} \alpha_k = 0. \tag{17}$$

Given the boundedness of the sequences $\{x_k\}$ and $\{d_k\}$, there exists two convergent subsequences $\{x_{k_n}\}$ and $\{d_{k_n}\}$ such that

$$\lim_{n\to\infty, n\in\mathcal{K}} x_{k_n} = \bar{x}, \quad \lim_{n\to\infty, n\in\mathcal{K}} d_{k_n} = \bar{d},$$

where $\mathcal{K}$ is an infinite index set. The inequality (6) yields

$$-\langle E_{k_n}, d_{k_n}\rangle \geq c_1\|E_{k_n}\|^2.$$

By allowing $n \to \infty$ in the above inequality, the continuity of E shows that

$$-\langle E(\bar{x}), \bar{d}\rangle \geq c_1\|E(\bar{x})\|^2 > c_1\epsilon_1^2 > 0. \tag{18}$$

Next, considering the line search Formula (5), we have

$$-\langle E(x_{k_n} + \rho^{-1}\alpha_{k_n}d_{k_n}), d_{k_n}\rangle < \sigma\rho^{-1}\alpha_{k_n}\|E(x_{k_n} + \rho^{-1}\alpha_{k_n}d_{k_n})\|\|d_{k_n}\|^2.$$

By allowing $n \to \infty$ in the above inequality, the continuity of E implies that

$$-\langle E(\bar{x}), \bar{d}\rangle \leq 0,$$

which contradicts with (18). Therefore, the desired result holds. □

4. Numerical Experiments

In this section, we conducted numerical experiments to demonstrate the effectiveness and competitiveness of Algorithm ITTCG. We compared it with two existing three-term algorithms: Algorithm HTTCGP [18] and Algorithm ZYL [28]. All experiments were performed on an Ubuntu 20.04.2 LTS 64 bit operating system, utilizing an Intel(R) Xeon(R) Gold 5115 CPU at 2.40 GHz.

The parameters for Algorithm ITTCG were configured as follows: $\sigma = 10^{-4}$, $\rho = 0.74$, $\xi = 1.3$, $a_2 = 0.001$, $b_1 = 0.3$, $b_2 = 1$, $\varepsilon = 10^{-6}$, $\bar{\delta} = 0.1$, and τ_k is computed by

$$\tau_k = \min\left\{\bar{\delta}, \max\left\{0, 1 - \frac{\langle y_{k-1}, s_{k-1}\rangle}{\|y_{k-1}\|^2}\right\}\right\}.$$

The parameters for Algorithms HTTCGP and ZYL were set according to their respective references. We selected benchmark problems with dimensions $n = [1000\ 5000\ 10,000\ 50,000\ 100,000]$. The benchmark problems were formulated as $E(x) = (E_1(x), E_2(x), \ldots, E_n(x))^{\mathrm{T}}$ with $x = (x_1, x_2, \ldots, x_n)^{\mathrm{T}}$. For each benchmark problem, we utilized the following initial points: $x_1 = (1, 1, \ldots, 1)^{\mathrm{T}}$, $x_2 = (\frac{1}{3}, \frac{1}{3^2}, \ldots, \frac{1}{3^n})^{\mathrm{T}}$, $x_3 = (\frac{1}{2}, \frac{1}{2^2}, \ldots, \frac{1}{2^n})^{\mathrm{T}}$, $x_4 = (0, \frac{1}{n}, \frac{2}{n}, \ldots, \frac{n-1}{n})^{\mathrm{T}}$, $x_5 = (1, \frac{1}{2}, \ldots, \frac{1}{n})$, $x_6 = (\frac{1}{n}, \frac{2}{n}, \ldots, \frac{n}{n})$, $x_7 = (1 - \frac{1}{n}, 1 - \frac{2}{n}, \ldots, 1 - \frac{n}{n})$, $x_8 = rand(n, 1)$. For each benchmark problem, each algorithm was terminated when $\|E_k\| \leq \epsilon$ or the number of iterations exceeded 2000.

Problem 1. Set

$$\begin{aligned} E_1(x) &= e^{x_1} - 1, \\ E_i(x) &= e^{x_i} + x_i - 1, \text{for } i = 2, 3, \cdots, n, \end{aligned}$$

and $\mathbb{E} = \mathbb{R}^n_+$.

Problem 2. Set

$$E_i(x) = e^{x_i} - 1, \text{ for } i = 1, 2, \cdots, n,$$

and $\mathbb{E} = \mathbb{R}^n_+$. Clearly, this problem has a unique solution $x^* = (0, 0, \cdots, 0)^{\mathrm{T}}$.

Problem 3. Set

$$\begin{aligned} E_1(x) &= 2x_1 + \sin(x_1) - 1, \\ E_i(x) &= 2x_{i-1} + 2x_i + \sin(x_i) - 1, \text{ for } i = 2, 3, \cdots, n-1, \\ E_n(x) &= 2x_n + \sin(x_n) - 1, \end{aligned}$$

and $\mathbb{E} = \mathbb{R}^n_+$.

Problem 4. Set

$$E_i(x) = \frac{i}{n} e^{x_i} - 1, \text{ for } i = 1, 2, \cdots, n,$$

and $\mathbb{E} = \mathbb{R}^n_+$.

Problem 5. Set

$$E_i(x) = 2x_i - \sin(x_i), \text{ for } i = 1, 2, \cdots, n,$$

and $\mathbb{E} = [-2, +\infty)$.

Problem 6. Set

$$E_i(x) = (e^{x_i})^2 + 3\sin(x_i)\cos(x_i) - 1, \text{ for } i = 1, 2, \cdots, n,$$

and $\mathbb{E} = \mathbb{R}^n_+$.

Problem 7. Set

$$\begin{aligned} E_1(x) &= x_1 - e^{\cos(\frac{x_1 + x_2}{2})}, \\ E_i(x) &= x_i - e^{\cos(\frac{x_{i-1} + x_i + x_{i+1}}{i})}, \text{ for } i = 2, 3, \cdots, n-1, \\ E_n(x) &= x_n - e^{\cos(\frac{x_{n-1} + x_n}{n})}, \end{aligned}$$

and $\mathbb{E} = \mathbb{R}^n_+$.

Problem 8. Set

$$\begin{aligned} E_1(x) &= x_1 + \sin(x_1) - 1, \\ E_i(x) &= -x_{i-1} + 2x_i + \sin(x_i) - 1, \text{ for } i = 2, 3, \cdots, n-1, \\ E_n(x) &= x_n + \sin(x_n) - 1, \end{aligned}$$

and $\mathbb{E} = \{x \in \mathbb{R}^n : x \geq -3\}$.

The numerical results of benchmark problems solved by Algorithms ITTCG, HTTCGP, and ZYL are presented in Tables 1–8. In these tables, "Init(n)" refers to the initial points and the dimension multiplied by 1000. The detailed results are formatted as Time/Nfunc /Niter/Norm, where "Time" represents the CPU time in seconds, "Nfunc" represents the number of function evaluations, "Niter" represents the number of iterations, and "Norm" represents the norm of the function at the approximate optimal point. These tables illustrate that all three algorithms were capable of solving the benchmark problems across various initial points and dimensions. Notably, Algorithm ITTCG exhibited superior performance in most cases. To clearly demonstrate the performance of Algorithm ITTCG, we utilized the performance profiles developed by Dolan and Moré [29]. These profiles visually compared the performance in terms of CPU time, Nfunc, and Niter, as shown in Figures 1–3. From these figures, we can observe that Algorithm ITTCG won about 59%, 77%, and 80% of the experiments in terms of CPU time, Nfunc, and Niter, respectively. The results indicate that Algorithm ITTCG outperformed Algorithm HTTCGP and ZYL on the given benchmark problems.

Table 1. Numerical results for Problem 1.

Init (n)	**ITTCG** **Time/Nfunc/Niter/Norm**	**HTTCGP** **Time/Nfunc/Niter/Norm**	**ZYL** **Time/Nfunc/Niter/Norm**
x1(1)	$8.30 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$2.89 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$3.77 \times 10^{-3}/90/18/9.93 \times 10^{-7}$
x2(1)	$7.84 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$1.11 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$5.27 \times 10^{-4}/25/8/1.84 \times 10^{-7}$
x3(1)	$1.53 \times 10^{-3}/78/16/5.89 \times 10^{-7}$	$1.56 \times 10^{-3}/84/17/4.60 \times 10^{-7}$	$4.51 \times 10^{-3}/261/62/6.28 \times 10^{-7}$
x4(1)	$1.16 \times 10^{-3}/53/11/3.86 \times 10^{-7}$	$1.44 \times 10^{-3}/69/15/2.46 \times 10^{-7}$	$1.84 \times 10^{-3}/100/20/5.05 \times 10^{-7}$
x5(1)	$1.93 \times 10^{-3}/69/14/6.42 \times 10^{-7}$	$8.06 \times 10^{-4}/38/8/6.62 \times 10^{-7}$	$6.76 \times 10^{-3}/288/68/8.66 \times 10^{-7}$
x6(1)	$1.49 \times 10^{-3}/53/11/3.94 \times 10^{-7}$	$1.52 \times 10^{-3}/69/15/3.14 \times 10^{-7}$	$2.03 \times 10^{-3}/100/20/5.09 \times 10^{-7}$
x7(1)	$1.35 \times 10^{-3}/53/11/3.44 \times 10^{-7}$	$1.78 \times 10^{-3}/69/15/1.28 \times 10^{-7}$	$2.96 \times 10^{-3}/100/20/5.60 \times 10^{-7}$
x8(1)	$1.28 \times 10^{-3}/53/11/8.69 \times 10^{-7}$	$2.57 \times 10^{-3}/108/21/9.53 \times 10^{-7}$	$1.95 \times 10^{-3}/100/20/4.68 \times 10^{-7}$
x1(5)	$1.35 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$8.77 \times 10^{-4}/7/1/0.00 \times 10^{0}$	$1.23 \times 10^{-2}/90/18/8.21 \times 10^{-7}$
x2(5)	$5.94 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$5.96 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$4.38 \times 10^{-3}/25/8/1.84 \times 10^{-7}$
x3(5)	$6.18 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$4.95 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$3.91 \times 10^{-3}/25/8/2.00 \times 10^{-7}$
x4(5)	$5.79 \times 10^{-3}/44/9/3.64 \times 10^{-7}$	$8.06 \times 10^{-3}/69/15/2.23 \times 10^{-7}$	$1.18 \times 10^{-2}/105/21/4.48 \times 10^{-7}$
x5(5)	$8.99 \times 10^{-3}/69/14/6.60 \times 10^{-7}$	$5.43 \times 10^{-3}/38/8/6.39 \times 10^{-7}$	$3.46 \times 10^{-2}/288/68/8.67 \times 10^{-7}$
x6(5)	$5.24 \times 10^{-3}/44/9/3.72 \times 10^{-7}$	$8.65 \times 10^{-3}/69/15/4.01 \times 10^{-7}$	$1.04 \times 10^{-2}/105/21/4.49 \times 10^{-7}$
x7(5)	$5.48 \times 10^{-3}/44/9/3.98 \times 10^{-7}$	$7.85 \times 10^{-3}/69/15/3.15 \times 10^{-7}$	$1.16 \times 10^{-2}/105/21/4.60 \times 10^{-7}$
x8(5)	$6.42 \times 10^{-3}/53/11/5.88 \times 10^{-7}$	$8.85 \times 10^{-3}/69/15/9.92 \times 10^{-9}$	$1.16 \times 10^{-2}/105/21/4.40 \times 10^{-7}$
x1(10)	$1.28 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$1.25 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$1.50 \times 10^{-2}/90/18/8.80 \times 10^{-7}$
x2(10)	$8.37 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$7.26 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$5.07 \times 10^{-3}/25/8/1.84 \times 10^{-7}$
x3(10)	$7.53 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$7.21 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$5.80 \times 10^{-3}/25/8/2.00 \times 10^{-7}$
x4(10)	$8.52 \times 10^{-3}/44/9/1.54 \times 10^{-7}$	$1.28 \times 10^{-2}/69/15/3.77 \times 10^{-7}$	$1.82 \times 10^{-2}/105/21/6.34 \times 10^{-7}$
x5(10)	$1.33 \times 10^{-2}/69/14/6.62 \times 10^{-7}$	$6.92 \times 10^{-3}/38/8/6.36 \times 10^{-7}$	$5.57 \times 10^{-2}/288/68/8.67 \times 10^{-7}$
x6(10)	$9.67 \times 10^{-3}/44/9/1.58 \times 10^{-7}$	$1.39 \times 10^{-2}/69/15/5.06 \times 10^{-7}$	$1.91 \times 10^{-2}/105/21/6.34 \times 10^{-7}$
x7(10)	$9.01 \times 10^{-3}/44/9/1.69 \times 10^{-7}$	$1.37 \times 10^{-2}/69/15/4.45 \times 10^{-7}$	$1.92 \times 10^{-2}/105/21/6.42 \times 10^{-7}$
x8(10)	$1.15 \times 10^{-2}/53/11/2.05 \times 10^{-8}$	$1.39 \times 10^{-2}/69/15/2.94 \times 10^{-8}$	$1.83 \times 10^{-2}/105/21/6.17 \times 10^{-7}$
x1(50)	$5.84 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$3.71 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$5.84 \times 10^{-2}/95/19/5.74 \times 10^{-7}$
x2(50)	$2.75 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$2.56 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.56 \times 10^{-2}/25/8/1.84 \times 10^{-7}$

Table 1. *Cont.*

Init (n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x3(50)	$2.46 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$2.31 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.60 \times 10^{-2}/25/8/2.00 \times 10^{-7}$
x4(50)	$5.49 \times 10^{-2}/81/17/5.07 \times 10^{-7}$	$4.94 \times 10^{-2}/69/15/9.57 \times 10^{-7}$	$6.51 \times 10^{-2}/110/22/5.60 \times 10^{-7}$
x5(50)	$4.53 \times 10^{-2}/69/14/6.64 \times 10^{-7}$	$2.60 \times 10^{-2}/38/8/6.34 \times 10^{-7}$	$1.84 \times 10^{-1}/288/68/8.67 \times 10^{-7}$
x6(50)	$5.33 \times 10^{-2}/76/16/8.44 \times 10^{-7}$	$5.10 \times 10^{-2}/74/16/6.93 \times 10^{-8}$	$6.71 \times 10^{-2}/110/22/5.60 \times 10^{-7}$
x7(50)	$5.21 \times 10^{-2}/76/16/4.33 \times 10^{-7}$	$4.64 \times 10^{-2}/69/15/9.88 \times 10^{-7}$	$6.74 \times 10^{-2}/110/22/5.62 \times 10^{-7}$
x8(50)	$3.59 \times 10^{-2}/48/10/0.00 \times 10^{0}$	$5.82 \times 10^{-2}/78/17/7.67 \times 10^{-7}$	$6.84 \times 10^{-2}/110/22/5.55 \times 10^{-7}$
x1(100)	$9.61 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$9.45 \times 10^{-3}/7/1/0.00 \times 10^{0}$	$1.29 \times 10^{-1}/95/19/7.76 \times 10^{-7}$
x2(100)	$6.76 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$5.55 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$3.12 \times 10^{-2}/25/8/1.84 \times 10^{-7}$
x3(100)	$5.52 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$5.36 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$3.02 \times 10^{-2}/25/8/2.00 \times 10^{-7}$
x4(100)	$7.97 \times 10^{-2}/52/11/0.00 \times 10^{0}$	$1.12 \times 10^{-1}/74/16/6.18 \times 10^{-8}$	$1.42 \times 10^{-1}/110/22/7.92 \times 10^{-7}$
x5(100)	$9.69 \times 10^{-2}/69/14/6.64 \times 10^{-7}$	$5.44 \times 10^{-2}/38/8/6.33 \times 10^{-7}$	$3.61 \times 10^{-1}/288/68/8.67 \times 10^{-7}$
x6(100)	$7.86 \times 10^{-2}/52/11/0.00 \times 10^{0}$	$1.10 \times 10^{-1}/74/16/8.59 \times 10^{-8}$	$1.40 \times 10^{-1}/110/22/7.92 \times 10^{-7}$
x7(100)	$8.00 \times 10^{-2}/52/11/0.00 \times 10^{0}$	$1.15 \times 10^{-1}/74/16/7.44 \times 10^{-8}$	$1.82 \times 10^{-1}/110/22/7.94 \times 10^{-7}$
x8(100)	$1.02 \times 10^{-1}/62/13/5.59 \times 10^{-7}$	$1.25 \times 10^{-1}/78/17/9.11 \times 10^{-7}$	$1.43 \times 10^{-1}/110/22/7.88 \times 10^{-7}$

Table 2. Numerical results for Problem 2.

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(1)	$3.05 \times 10^{-3}/5/1/0.00 \times 10^{0}$	$1.26 \times 10^{-4}/5/1/0.00 \times 10^{0}$	$6.58 \times 10^{-4}/36/11/1.82 \times 10^{-7}$
x2(1)	$9.14 \times 10^{-5}/4/1/0.00 \times 10^{0}$	$1.24 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$4.13 \times 10^{-4}/25/8/1.84 \times 10^{-7}$
x3(1)	$4.95 \times 10^{-4}/22/8/9.13 \times 10^{-7}$	$4.36 \times 10^{-4}/23/8/1.61 \times 10^{-7}$	$1.46 \times 10^{-3}/94/31/5.23 \times 10^{-7}$
x4(1)	$1.06 \times 10^{-3}/47/15/2.08 \times 10^{-7}$	$1.23 \times 10^{-3}/59/19/9.26 \times 10^{-7}$	$2.11 \times 10^{-3}/117/38/7.52 \times 10^{-7}$
x5(1)	$5.70 \times 10^{-4}/24/8/6.49 \times 10^{-8}$	$5.69 \times 10^{-4}/28/9/6.46 \times 10^{-8}$	$1.43 \times 10^{-3}/81/26/9.56 \times 10^{-7}$
x6(1)	$9.49 \times 10^{-4}/40/13/5.57 \times 10^{-7}$	$7.91 \times 10^{-4}/38/13/8.06 \times 10^{-9}$	$2.14 \times 10^{-3}/121/39/4.72 \times 10^{-7}$
x7(1)	$1.06 \times 10^{-3}/47/15/2.08 \times 10^{-7}$	$1.15 \times 10^{-3}/59/19/9.26 \times 10^{-7}$	$2.06 \times 10^{-3}/117/38/7.52 \times 10^{-7}$
x8(1)	$1.03 \times 10^{-3}/39/14/1.88 \times 10^{-7}$	$9.06 \times 10^{-4}/40/13/4.75 \times 10^{-7}$	$1.42 \times 10^{-3}/76/24/8.46 \times 10^{-7}$
x1(5)	$5.79 \times 10^{-4}/5/1/0.00 \times 10^{0}$	$4.40 \times 10^{-4}/5/1/0.00 \times 10^{0}$	$4.36 \times 10^{-3}/36/11/4.07 \times 10^{-7}$
x2(5)	$3.91 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$4.16 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$2.82 \times 10^{-3}/25/8/1.84 \times 10^{-7}$
x3(5)	$5.10 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$4.63 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$3.11 \times 10^{-3}/25/8/2.00 \times 10^{-7}$
x4(5)	$6.11 \times 10^{-3}/45/15/3.16 \times 10^{-7}$	$6.60 \times 10^{-3}/55/18/1.08 \times 10^{-12}$	$1.27 \times 10^{-2}/126/41/8.02 \times 10^{-7}$
x5(5)	$2.69 \times 10^{-3}/24/8/9.08 \times 10^{-8}$	$2.99 \times 10^{-3}/28/9/8.18 \times 10^{-8}$	$8.06 \times 10^{-3}/81/26/9.57 \times 10^{-7}$
x6(5)	$4.58 \times 10^{-3}/44/14/5.94 \times 10^{-7}$	$4.47 \times 10^{-3}/43/15/6.72 \times 10^{-8}$	$1.00 \times 10^{-2}/106/34/9.43 \times 10^{-7}$
x7(5)	$4.87 \times 10^{-3}/45/15/3.16 \times 10^{-7}$	$6.44 \times 10^{-3}/55/18/1.08 \times 10^{-12}$	$1.11 \times 10^{-2}/126/41/8.02 \times 10^{-7}$
x8(5)	$6.10 \times 10^{-3}/50/17/7.67 \times 10^{-8}$	$4.00 \times 10^{-3}/38/13/7.39 \times 10^{-7}$	$1.15 \times 10^{-2}/123/40/6.51 \times 10^{-7}$
x1(10)	$6.70 \times 10^{-4}/5/1/0.00 \times 10^{0}$	$6.54 \times 10^{-4}/5/1/0.00 \times 10^{0}$	$4.92 \times 10^{-3}/36/11/5.75 \times 10^{-7}$
x2(10)	$5.04 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$5.16 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$3.03 \times 10^{-3}/25/8/1.84 \times 10^{-7}$
x3(10)	$5.41 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$6.07 \times 10^{-4}/4/1/0.00 \times 10^{0}$	$3.99 \times 10^{-3}/25/8/2.00 \times 10^{-7}$
x4(10)	$9.40 \times 10^{-3}/52/16/7.66 \times 10^{-7}$	$8.07 \times 10^{-3}/56/18/5.41 \times 10^{-10}$	$1.80 \times 10^{-2}/129/42/6.00 \times 10^{-7}$
x5(10)	$4.74 \times 10^{-3}/24/8/9.43 \times 10^{-8}$	$4.19 \times 10^{-3}/28/9/8.42 \times 10^{-8}$	$1.18 \times 10^{-2}/81/26/9.57 \times 10^{-7}$
x6(10)	$8.57 \times 10^{-3}/51/16/8.19 \times 10^{-7}$	$8.58 \times 10^{-3}/58/19/2.93 \times 10^{-7}$	$1.59 \times 10^{-2}/112/36/4.86 \times 10^{-7}$
x7(10)	$1.06 \times 10^{-2}/52/16/7.66 \times 10^{-7}$	$9.62 \times 10^{-3}/56/18/5.41 \times 10^{-10}$	$1.94 \times 10^{-2}/129/42/6.00 \times 10^{-7}$
x8(10)	$1.30 \times 10^{-2}/50/17/5.50 \times 10^{-7}$	$9.49 \times 10^{-3}/58/19/1.09 \times 10^{-7}$	$2.12 \times 10^{-2}/120/39/8.17 \times 10^{-7}$
x1(50)	$3.13 \times 10^{-3}/5/1/0.00 \times 10^{0}$	$1.94 \times 10^{-3}/5/1/0.00 \times 10^{0}$	$2.00 \times 10^{-2}/39/12/2.06 \times 10^{-7}$
x2(50)	$1.69 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.39 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.03 \times 10^{-2}/25/8/1.84 \times 10^{-7}$
x3(50)	$1.71 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.64 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.12 \times 10^{-2}/25/8/2.00 \times 10^{-7}$
x4(50)	$3.51 \times 10^{-2}/60/19/4.83 \times 10^{-7}$	$3.23 \times 10^{-2}/57/18/4.68 \times 10^{-14}$	$6.65 \times 10^{-2}/129/42/9.70 \times 10^{-7}$
x5(50)	$1.48 \times 10^{-2}/24/8/9.72 \times 10^{-8}$	$1.68 \times 10^{-2}/28/9/8.62 \times 10^{-8}$	$4.04 \times 10^{-2}/81/26/9.57 \times 10^{-7}$
x6(50)	$3.64 \times 10^{-2}/54/18/1.53 \times 10^{-7}$	$2.29 \times 10^{-2}/39/14/3.40 \times 10^{-7}$	$5.72 \times 10^{-2}/117/38/5.84 \times 10^{-7}$

Table 2. *Cont.*

Init(n)	**ITTCG** Time/Nfunc/Niter/Norm	**HTTCGP** Time/Nfunc/Niter/Norm	**ZYL** Time/Nfunc/Niter/Norm
x7(50)	$3.77 \times 10^{-2}/60/19/4.83 \times 10^{-7}$	$3.44 \times 10^{-2}/57/18/4.68 \times 10^{-14}$	$6.29 \times 10^{-2}/129/42/9.70 \times 10^{-7}$
x8(50)	$3.82 \times 10^{-2}/58/18/2.83 \times 10^{-7}$	$2.89 \times 10^{-2}/41/14/4.10 \times 10^{-7}$	$6.26 \times 10^{-2}/129/42/8.15 \times 10^{-7}$
x1(100)	$4.75 \times 10^{-3}/5/1/0.00 \times 10^{0}$	$3.72 \times 10^{-3}/5/1/0.00 \times 10^{0}$	$3.23 \times 10^{-2}/39/12/2.91 \times 10^{-7}$
x2(100)	$3.49 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$2.62 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.60 \times 10^{-2}/25/8/1.84 \times 10^{-7}$
x3(100)	$3.94 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$3.03 \times 10^{-3}/4/1/0.00 \times 10^{0}$	$1.60 \times 10^{-2}/25/8/2.00 \times 10^{-7}$
x4(100)	$4.69 \times 10^{-2}/41/15/3.94 \times 10^{-8}$	$3.74 \times 10^{-2}/41/14/3.81 \times 10^{-7}$	$1.06 \times 10^{-1}/132/43/8.32 \times 10^{-7}$
x5(100)	$2.63 \times 10^{-2}/24/8/9.75 \times 10^{-8}$	$2.77 \times 10^{-2}/28/9/8.64 \times 10^{-8}$	$6.34 \times 10^{-2}/81/26/9.57 \times 10^{-7}$
x6(100)	$5.50 \times 10^{-2}/47/18/3.93 \times 10^{-7}$	$4.14 \times 10^{-2}/43/15/4.35 \times 10^{-8}$	$1.04 \times 10^{-1}/129/42/5.76 \times 10^{-7}$
x7(100)	$4.78 \times 10^{-2}/41/15/3.94 \times 10^{-8}$	$4.06 \times 10^{-2}/41/14/3.81 \times 10^{-7}$	$1.10 \times 10^{-1}/132/43/8.32 \times 10^{-7}$
x8(100)	$6.30 \times 10^{-2}/48/17/3.01 \times 10^{-7}$	$5.47 \times 10^{-2}/45/16/4.28 \times 10^{-15}$	$1.07 \times 10^{-1}/129/42/7.87 \times 10^{-7}$

Table 3. Numerical results for Problem 3.

Init(n)	**ITTCG** **Time/Nfunc/Niter/Norm**	**HTTCGP** **Time/Nfunc/Niter/Norm**	**ZYL** **Time/Nfunc/Niter/Norm**
x1(1)	$5.36 \times 10^{-3}/250/33/5.87 \times 10^{-7}$	$5.72 \times 10^{-3}/293/36/9.74 \times 10^{-7}$	$8.26 \times 10^{-3}/439/73/9.64 \times 10^{-7}$
x2(1)	$3.93 \times 10^{-3}/180/23/7.99 \times 10^{-7}$	$5.81 \times 10^{-3}/307/38/8.37 \times 10^{-7}$	$7.07 \times 10^{-3}/367/61/9.94 \times 10^{-7}$
x3(1)	$4.19 \times 10^{-3}/208/27/8.37 \times 10^{-7}$	$6.63 \times 10^{-3}/349/43/8.55 \times 10^{-7}$	$7.49 \times 10^{-3}/385/64/8.74 \times 10^{-7}$
x4(1)	$4.51 \times 10^{-3}/222/29/6.50 \times 10^{-7}$	$8.57 \times 10^{-3}/433/54/9.46 \times 10^{-7}$	$7.66 \times 10^{-3}/391/65/9.70 \times 10^{-7}$
x5(1)	$4.73 \times 10^{-3}/237/31/6.04 \times 10^{-7}$	$6.64 \times 10^{-3}/337/42/8.37 \times 10^{-7}$	$7.42 \times 10^{-3}/391/65/9.80 \times 10^{-7}$
x6(1)	$4.61 \times 10^{-3}/222/29/6.49 \times 10^{-7}$	$8.58 \times 10^{-3}/410/51/9.15 \times 10^{-7}$	$8.01 \times 10^{-3}/397/66/9.25 \times 10^{-7}$
x7(1)	$5.08 \times 10^{-3}/250/33/5.41 \times 10^{-7}$	$6.35 \times 10^{-3}/330/41/8.54 \times 10^{-7}$	$7.97 \times 10^{-3}/421/70/9.17 \times 10^{-7}$
x8(1)	$6.45 \times 10^{-3}/308/41/5.09 \times 10^{-7}$	$8.47 \times 10^{-3}/413/51/9.89 \times 10^{-7}$	$9.79 \times 10^{-3}/480/80/9.21 \times 10^{-7}$
x1(5)	$3.76 \times 10^{-2}/243/32/6.03 \times 10^{-7}$	$3.64 \times 10^{-2}/299/37/8.62 \times 10^{-7}$	$5.50 \times 10^{-2}/445/74/8.19 \times 10^{-7}$
x2(5)	$2.20 \times 10^{-2}/173/22/8.68 \times 10^{-7}$	$3.98 \times 10^{-2}/338/42/1.00 \times 10^{-6}$	$4.94 \times 10^{-2}/403/67/9.85 \times 10^{-7}$
x3(5)	$2.43 \times 10^{-2}/201/26/8.32 \times 10^{-7}$	$3.75 \times 10^{-2}/319/39/9.75 \times 10^{-7}$	$4.98 \times 10^{-2}/415/69/9.21 \times 10^{-7}$
x4(5)	$2.67 \times 10^{-2}/229/30/8.15 \times 10^{-7}$	$4.77 \times 10^{-2}/418/52/3.03 \times 10^{-7}$	$5.21 \times 10^{-2}/421/70/7.23 \times 10^{-7}$
x5(5)	$2.62 \times 10^{-2}/216/28/5.92 \times 10^{-7}$	$3.86 \times 10^{-2}/316/39/7.17 \times 10^{-7}$	$5.17 \times 10^{-2}/415/69/8.21 \times 10^{-7}$
x6(5)	$2.92 \times 10^{-2}/229/30/8.14 \times 10^{-7}$	$5.12 \times 10^{-2}/417/52/9.34 \times 10^{-7}$	$5.52 \times 10^{-2}/415/69/8.50 \times 10^{-7}$
x7(5)	$3.48 \times 10^{-2}/271/36/7.63 \times 10^{-7}$	$7.09 \times 10^{-2}/560/70/9.12 \times 10^{-7}$	$5.42 \times 10^{-2}/439/73/7.95 \times 10^{-7}$
x8(5)	$4.21 \times 10^{-2}/309/41/5.34 \times 10^{-7}$	$5.09 \times 10^{-2}/413/51/5.52 \times 10^{-7}$	$6.69 \times 10^{-2}/522/87/9.37 \times 10^{-7}$
x1(10)	$4.56 \times 10^{-2}/236/31/6.73 \times 10^{-7}$	$6.37 \times 10^{-2}/338/42/3.92 \times 10^{-7}$	$8.78 \times 10^{-2}/451/75/9.13 \times 10^{-7}$
x2(10)	$3.39 \times 10^{-2}/187/24/9.18 \times 10^{-7}$	$6.63 \times 10^{-2}/338/42/9.20 \times 10^{-7}$	$7.80 \times 10^{-2}/415/69/9.61 \times 10^{-7}$
x3(10)	$3.83 \times 10^{-2}/208/27/9.16 \times 10^{-7}$	$6.19 \times 10^{-2}/335/42/6.78 \times 10^{-7}$	$7.91 \times 10^{-2}/427/71/9.89 \times 10^{-7}$
x4(10)	$4.54 \times 10^{-2}/236/31/7.88 \times 10^{-7}$	$6.21 \times 10^{-2}/359/45/4.72 \times 10^{-7}$	$6.89 \times 10^{-2}/397/66/6.98 \times 10^{-7}$
x5(10)	$4.12 \times 10^{-2}/223/29/5.94 \times 10^{-7}$	$5.37 \times 10^{-2}/302/37/5.30 \times 10^{-7}$	$7.13 \times 10^{-2}/409/68/9.04 \times 10^{-7}$
x6(10)	$4.33 \times 10^{-2}/236/31/7.88 \times 10^{-7}$	$6.24 \times 10^{-2}/369/46/3.53 \times 10^{-7}$	$6.95 \times 10^{-2}/397/66/7.00 \times 10^{-7}$
x7(10)	$5.05 \times 10^{-2}/264/35/8.21 \times 10^{-7}$	$9.77 \times 10^{-2}/568/71/5.46 \times 10^{-7}$	$7.02 \times 10^{-2}/397/66/9.17 \times 10^{-7}$
x8(10)	$5.65 \times 10^{-2}/309/41/7.30 \times 10^{-7}$	$6.72 \times 10^{-2}/387/48/8.03 \times 10^{-7}$	$8.77 \times 10^{-2}/486/81/7.25 \times 10^{-7}$
x1(50)	$1.75 \times 10^{-1}/236/31/7.21 \times 10^{-7}$	$2.01 \times 10^{-1}/292/36/6.60 \times 10^{-7}$	$3.00 \times 10^{-1}/427/71/9.67 \times 10^{-7}$
x2(50)	$1.31 \times 10^{-1}/180/23/9.41 \times 10^{-7}$	$2.50 \times 10^{-1}/350/44/9.18 \times 10^{-7}$	$2.97 \times 10^{-1}/427/71/9.94 \times 10^{-7}$
x3(50)	$1.44 \times 10^{-1}/209/27/4.76 \times 10^{-7}$	$2.80 \times 10^{-1}/412/52/8.85 \times 10^{-7}$	$3.02 \times 10^{-1}/415/69/7.96 \times 10^{-7}$
x4(50)	$1.60 \times 10^{-1}/229/30/8.95 \times 10^{-7}$	$2.65 \times 10^{-1}/394/50/8.74 \times 10^{-7}$	$2.90 \times 10^{-1}/416/69/9.13 \times 10^{-7}$
x5(50)	$1.62 \times 10^{-1}/237/31/6.63 \times 10^{-7}$	$1.95 \times 10^{-1}/288/36/9.73 \times 10^{-7}$	$3.12 \times 10^{-1}/439/73/8.44 \times 10^{-7}$
x6(50)	$1.52 \times 10^{-1}/229/30/8.95 \times 10^{-7}$	$2.64 \times 10^{-1}/387/49/8.40 \times 10^{-7}$	$2.90 \times 10^{-1}/416/69/9.17 \times 10^{-7}$
x7(50)	$1.80 \times 10^{-1}/264/35/8.82 \times 10^{-7}$	$2.63 \times 10^{-1}/381/48/7.31 \times 10^{-7}$	$2.79 \times 10^{-1}/410/68/9.11 \times 10^{-7}$
x8(50)	$2.24 \times 10^{-1}/324/43/6.03 \times 10^{-7}$	$3.16 \times 10^{-1}/451/56/7.36 \times 10^{-7}$	$3.41 \times 10^{-1}/481/80/9.57 \times 10^{-7}$

Table 3. *Cont.*

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(100)	$4.19 \times 10^{-1}/222/29/8.67 \times 10^{-7}$	$5.07 \times 10^{-1}/261/32/7.78 \times 10^{-7}$	$8.75 \times 10^{-1}/433/72/8.25 \times 10^{-7}$
x2(100)	$3.24 \times 10^{-1}/180/23/9.00 \times 10^{-7}$	$5.38 \times 10^{-1}/265/33/7.48 \times 10^{-7}$	$8.16 \times 10^{-1}/415/69/8.73 \times 10^{-7}$
x3(100)	$3.84 \times 10^{-1}/216/28/4.70 \times 10^{-7}$	$6.33 \times 10^{-1}/320/40/4.66 \times 10^{-7}$	$7.70 \times 10^{-1}/415/69/9.78 \times 10^{-7}$
x4(100)	$4.77 \times 10^{-1}/243/32/9.84 \times 10^{-7}$	$5.49 \times 10^{-1}/378/48/8.73 \times 10^{-7}$	$7.55 \times 10^{-1}/416/69/7.31 \times 10^{-7}$
x5(100)	$4.32 \times 10^{-1}/230/30/7.85 \times 10^{-7}$	$6.88 \times 10^{-1}/344/43/9.43 \times 10^{-7}$	$9.14 \times 10^{-1}/451/75/9.94 \times 10^{-7}$
x6(100)	$4.56 \times 10^{-1}/243/32/9.84 \times 10^{-7}$	$6.98 \times 10^{-1}/347/44/9.61 \times 10^{-7}$	$8.11 \times 10^{-1}/416/69/7.31 \times 10^{-7}$
x7(100)	$4.67 \times 10^{-1}/244/32/5.07 \times 10^{-7}$	$6.55 \times 10^{-1}/317/40/6.90 \times 10^{-7}$	$8.56 \times 10^{-1}/422/70/9.36 \times 10^{-7}$
x8(100)	$6.35 \times 10^{-1}/345/46/8.22 \times 10^{-7}$	$9.66 \times 10^{-1}/475/59/7.73 \times 10^{-7}$	$9.91 \times 10^{-1}/493/82/8.45 \times 10^{-7}$

Table 4. Numerical results for Problem 4.

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(1)	$2.21 \times 10^{-3}/89/20/2.89 \times 10^{-7}$	$2.83 \times 10^{-3}/119/27/8.70 \times 10^{-8}$	$2.78 \times 10^{-3}/118/38/8.33 \times 10^{-7}$
x2(1)	$1.57 \times 10^{-3}/51/19/1.61 \times 10^{-7}$	$1.56 \times 10^{-3}/56/21/9.69 \times 10^{-7}$	$2.28 \times 10^{-3}/96/33/6.03 \times 10^{-7}$
x3(1)	$1.23 \times 10^{-3}/39/15/8.04 \times 10^{-7}$	$1.53 \times 10^{-3}/54/18/3.13 \times 10^{-7}$	$2.20 \times 10^{-3}/93/32/7.55 \times 10^{-7}$
x4(1)	$4.91 \times 10^{-3}/223/39/4.76 \times 10^{-7}$	$4.59 \times 10^{-3}/237/35/7.19 \times 10^{-7}$	$2.80 \times 10^{-3}/123/39/8.83 \times 10^{-7}$
x5(1)	$9.47 \times 10^{-4}/29/12/4.29 \times 10^{-7}$	$1.40 \times 10^{-3}/52/19/6.57 \times 10^{-8}$	$2.37 \times 10^{-3}/102/35/8.01 \times 10^{-7}$
x6(1)	$4.47 \times 10^{-3}/212/36/8.45 \times 10^{-7}$	$5.37 \times 10^{-3}/279/43/1.69 \times 10^{-7}$	$2.77 \times 10^{-3}/123/39/8.74 \times 10^{-7}$
x7(1)	$1.12 \times 10^{-3}/37/14/1.06 \times 10^{-7}$	$1.41 \times 10^{-3}/53/19/8.14 \times 10^{-7}$	$2.05 \times 10^{-3}/87/30/7.34 \times 10^{-7}$
x8(1)	$6.41 \times 10^{-3}/328/44/6.49 \times 10^{-7}$	$6.04 \times 10^{-3}/320/44/7.16 \times 10^{-7}$	$4.16 \times 10^{-3}/207/50/6.30 \times 10^{-7}$
x1(5)	$2.23 \times 10^{-2}/231/36/1.93 \times 10^{-7}$	$1.77 \times 10^{-2}/216/33/5.67 \times 10^{-7}$	$1.50 \times 10^{-2}/132/42/7.97 \times 10^{-7}$
x2(5)	$5.03 \times 10^{-3}/39/16/7.13 \times 10^{-7}$	$6.34 \times 10^{-3}/50/18/4.49 \times 10^{-7}$	$1.02 \times 10^{-2}/102/35/6.85 \times 10^{-7}$
x3(5)	$5.63 \times 10^{-3}/39/16/8.49 \times 10^{-7}$	$7.93 \times 10^{-3}/68/26/5.31 \times 10^{-7}$	$1.03 \times 10^{-2}/96/33/6.96 \times 10^{-7}$
x4(5)	$3.01 \times 10^{-2}/373/49/6.04 \times 10^{-7}$	$4.30 \times 10^{-2}/567/65/2.84 \times 10^{-7}$	$1.33 \times 10^{-2}/133/42/8.46 \times 10^{-7}$
x5(5)	$4.56 \times 10^{-3}/36/14/2.40 \times 10^{-7}$	$7.41 \times 10^{-3}/62/23/9.87 \times 10^{-7}$	$9.95 \times 10^{-3}/99/34/6.30 \times 10^{-7}$
x6(5)	$3.07 \times 10^{-2}/376/50/1.06 \times 10^{-7}$	$4.29 \times 10^{-2}/570/62/1.23 \times 10^{-7}$	$1.31 \times 10^{-2}/141/45/7.46 \times 10^{-7}$
x7(5)	$6.42 \times 10^{-3}/45/17/3.98 \times 10^{-7}$	$6.82 \times 10^{-3}/61/22/7.17 \times 10^{-7}$	$1.05 \times 10^{-2}/96/33/5.87 \times 10^{-7}$
x8(5)	$4.89 \times 10^{-2}/637/69/9.82 \times 10^{-7}$	$4.17 \times 10^{-2}/562/63/6.41 \times 10^{-8}$	$1.70 \times 10^{-2}/189/44/9.43 \times 10^{-7}$
x1(10)	$2.08 \times 10^{-2}/93/24/3.59 \times 10^{-7}$	$3.40 \times 10^{-2}/265/34/3.37 \times 10^{-7}$	$2.11 \times 10^{-2}/118/38/7.24 \times 10^{-7}$
x2(10)	$1.13 \times 10^{-2}/42/16/1.55 \times 10^{-7}$	$6.60 \times 10^{-3}/35/13/3.54 \times 10^{-7}$	$2.09 \times 10^{-2}/108/37/5.11 \times 10^{-7}$
x3(10)	$1.12 \times 10^{-2}/45/17/3.65 \times 10^{-7}$	$1.33 \times 10^{-2}/55/20/9.85 \times 10^{-7}$	$1.98 \times 10^{-2}/108/37/5.70 \times 10^{-7}$
x4(10)	$6.45 \times 10^{-2}/464/57/8.15 \times 10^{-8}$	$1.00 \times 10^{-1}/811/87/5.54 \times 10^{-7}$	$2.51 \times 10^{-2}/138/44/7.43 \times 10^{-7}$
x5(10)	$1.15 \times 10^{-2}/37/15/9.00 \times 10^{-7}$	$9.85 \times 10^{-3}/51/19/2.68 \times 10^{-7}$	$1.91 \times 10^{-2}/102/35/5.79 \times 10^{-7}$
x6(10)	$6.39 \times 10^{-2}/461/56/9.13 \times 10^{-7}$	$1.17 \times 10^{-1}/1002/101/2.35 \times 10^{-7}$	$2.36 \times 10^{-2}/138/44/9.78 \times 10^{-7}$
x7(10)	$1.23 \times 10^{-2}/43/17/4.03 \times 10^{-7}$	$1.32 \times 10^{-2}/53/21/7.96 \times 10^{-7}$	$2.08 \times 10^{-2}/109/37/8.60 \times 10^{-7}$
x8(10)	$8.19 \times 10^{-2}/633/66/5.97 \times 10^{-7}$	$1.11 \times 10^{-1}/1005/92/2.54 \times 10^{-7}$	$4.14 \times 10^{-2}/276/54/8.07 \times 10^{-7}$
x1(50)	$1.73 \times 10^{-1}/401/46/1.44 \times 10^{-7}$	$2.90 \times 10^{-1}/724/73/8.61 \times 10^{-7}$	$9.25 \times 10^{-2}/186/46/7.62 \times 10^{-7}$
x2(50)	$4.29 \times 10^{-2}/57/20/2.84 \times 10^{-7}$	$4.22 \times 10^{-2}/65/22/4.43 \times 10^{-7}$	$6.37 \times 10^{-2}/108/37/8.56 \times 10^{-7}$
x3(50)	$3.75 \times 10^{-2}/57/19/3.18 \times 10^{-7}$	$3.22 \times 10^{-2}/46/18/8.01 \times 10^{-7}$	$6.17 \times 10^{-2}/108/37/8.41 \times 10^{-7}$
x4(50)	$3.34 \times 10^{-1}/788/76/5.24 \times 10^{-7}$	$8.21 \times 10^{-1}/2118/166/6.17 \times 10^{-7}$	$8.29 \times 10^{-2}/139/43/9.48 \times 10^{-7}$
x5(50)	$3.79 \times 10^{-2}/45/17/2.58 \times 10^{-7}$	$5.17 \times 10^{-2}/70/24/4.35 \times 10^{-7}$	$6.62 \times 10^{-2}/108/37/8.69 \times 10^{-7}$
x6(50)	$3.40 \times 10^{-1}/805/78/9.22 \times 10^{-7}$	$6.01 \times 10^{-1}/1542/130/6.18 \times 10^{-7}$	$7.45 \times 10^{-2}/133/41/7.02 \times 10^{-7}$
x7(50)	$4.43 \times 10^{-2}/58/21/1.15 \times 10^{-7}$	$4.87 \times 10^{-2}/75/26/3.42 \times 10^{-8}$	$6.55 \times 10^{-2}/112/38/9.59 \times 10^{-7}$
x8(50)	$4.03 \times 10^{-1}/968/91/8.15 \times 10^{-7}$	$7.75 \times 10^{-1}/2002/154/3.01 \times 10^{-7}$	$1.63 \times 10^{-1}/371/61/9.71 \times 10^{-7}$
x1(100)	$3.19 \times 10^{-1}/422/51/5.34 \times 10^{-8}$	$6.89 \times 10^{-1}/943/93/3.29 \times 10^{-7}$	$2.17 \times 10^{-1}/267/58/9.06 \times 10^{-7}$
x2(100)	$5.56 \times 10^{-2}/42/16/6.16 \times 10^{-7}$	$8.42 \times 10^{-2}/75/25/3.72 \times 10^{-7}$	$1.19 \times 10^{-1}/118/40/4.98 \times 10^{-7}$
x3(100)	$5.52 \times 10^{-2}/42/16/6.54 \times 10^{-8}$	$1.17 \times 10^{-1}/103/33/7.65 \times 10^{-7}$	$1.17 \times 10^{-1}/118/40/5.08 \times 10^{-7}$
x4(100)	$5.75 \times 10^{-1}/771/77/2.88 \times 10^{-7}$	$1.34 \times 10^{0}/1932/156/7.33 \times 10^{-7}$	$1.24 \times 10^{-1}/133/39/7.09 \times 10^{-7}$
x5(100)	$8.06 \times 10^{-2}/66/23/7.79 \times 10^{-8}$	$7.31 \times 10^{-2}/58/20/1.15 \times 10^{-7}$	$1.17 \times 10^{-1}/118/40/5.29 \times 10^{-7}$
x6(100)	$5.64 \times 10^{-1}/774/80/6.88 \times 10^{-7}$	$1.28 \times 10^{0}/1883/150/7.31 \times 10^{-8}$	$1.25 \times 10^{-1}/133/39/7.55 \times 10^{-7}$
x7(100)	$8.95 \times 10^{-2}/76/24/2.56 \times 10^{-7}$	$8.48 \times 10^{-2}/71/25/6.40 \times 10^{-7}$	$1.10 \times 10^{-1}/109/37/9.37 \times 10^{-7}$
x8(100)	$8.61 \times 10^{-1}/1225/105/7.94 \times 10^{-7}$	$1.92 \times 10^{0}/2868/204/7.36 \times 10^{-7}$	$2.37 \times 10^{-1}/320/49/9.81 \times 10^{-7}$

Table 5. Numerical results for Problem 5.

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(1)	$3.96 \times 10^{-4}/12/5/5.48 \times 10^{-8}$	$5.80 \times 10^{-4}/24/10/3.72 \times 10^{-7}$	$5.63 \times 10^{-4}/31/10/6.08 \times 10^{-7}$
x2(1)	$2.51 \times 10^{-4}/10/4/1.65 \times 10^{-8}$	$3.69 \times 10^{-4}/18/7/5.57 \times 10^{-8}$	$3.90 \times 10^{-4}/25/8/3.43 \times 10^{-7}$
x3(1)	$5.97 \times 10^{-4}/20/7/8.62 \times 10^{-7}$	$8.13 \times 10^{-4}/33/11/8.36 \times 10^{-7}$	$1.04 \times 10^{-3}/52/17/5.45 \times 10^{-7}$
x4(1)	$1.06 \times 10^{-3}/42/16/6.01 \times 10^{-7}$	$8.85 \times 10^{-4}/44/14/1.09 \times 10^{-7}$	$1.57 \times 10^{-3}/88/29/8.47 \times 10^{-7}$
x5(1)	$6.14 \times 10^{-4}/26/9/6.17 \times 10^{-8}$	$1.25 \times 10^{-3}/61/19/6.07 \times 10^{-7}$	$1.39 \times 10^{-3}/79/26/9.94 \times 10^{-7}$
x6(1)	$1.04 \times 10^{-3}/40/16/4.36 \times 10^{-7}$	$7.10 \times 10^{-4}/35/11/4.13 \times 10^{-7}$	$1.69 \times 10^{-3}/88/29/9.04 \times 10^{-7}$
x7(1)	$1.08 \times 10^{-3}/42/16/6.01 \times 10^{-7}$	$9.01 \times 10^{-4}/44/14/1.09 \times 10^{-7}$	$1.56 \times 10^{-3}/88/29/8.47 \times 10^{-7}$
x8(1)	$7.76 \times 10^{-4}/32/11/5.11 \times 10^{-7}$	$1.13 \times 10^{-3}/56/17/3.91 \times 10^{-7}$	$1.45 \times 10^{-3}/79/26/7.43 \times 10^{-7}$
x1(5)	$2.17 \times 10^{-3}/12/5/1.39 \times 10^{-7}$	$3.04 \times 10^{-3}/24/10/8.33 \times 10^{-7}$	$3.11 \times 10^{-3}/34/11/2.17 \times 10^{-7}$
x2(5)	$1.18 \times 10^{-3}/10/4/1.65 \times 10^{-8}$	$1.87 \times 10^{-3}/18/7/5.57 \times 10^{-8}$	$2.36 \times 10^{-3}/25/8/3.43 \times 10^{-7}$
x3(5)	$9.94 \times 10^{-4}/8/3/2.00 \times 10^{-7}$	$1.70 \times 10^{-3}/14/5/7.19 \times 10^{-8}$	$2.78 \times 10^{-3}/25/8/4.88 \times 10^{-7}$
x4(5)	$8.32 \times 10^{-3}/37/13/5.39 \times 10^{-7}$	$5.83 \times 10^{-3}/47/16/5.29 \times 10^{-7}$	$9.94 \times 10^{-3}/91/30/9.14 \times 10^{-7}$
x5(5)	$3.12 \times 10^{-3}/26/9/6.15 \times 10^{-8}$	$3.98 \times 10^{-3}/39/14/5.16 \times 10^{-7}$	$7.61 \times 10^{-3}/79/26/9.95 \times 10^{-7}$
x6(5)	$4.18 \times 10^{-3}/37/13/5.25 \times 10^{-7}$	$5.90 \times 10^{-3}/56/18/6.52 \times 10^{-7}$	$8.74 \times 10^{-3}/91/30/9.26 \times 10^{-7}$
x7(5)	$4.40 \times 10^{-3}/37/13/5.39 \times 10^{-7}$	$4.87 \times 10^{-3}/47/16/5.29 \times 10^{-7}$	$1.31 \times 10^{-2}/91/30/9.14 \times 10^{-7}$
x8(5)	$5.04 \times 10^{-3}/37/13/5.20 \times 10^{-7}$	$4.02 \times 10^{-3}/40/13/3.87 \times 10^{-8}$	$8.56 \times 10^{-3}/91/30/7.14 \times 10^{-7}$
x1(10)	$2.64 \times 10^{-3}/12/5/2.15 \times 10^{-7}$	$5.83 \times 10^{-3}/27/11/4.48 \times 10^{-8}$	$4.55 \times 10^{-3}/34/11/3.08 \times 10^{-7}$
x2(10)	$1.68 \times 10^{-3}/10/4/1.65 \times 10^{-8}$	$2.94 \times 10^{-3}/18/7/5.57 \times 10^{-8}$	$3.62 \times 10^{-3}/25/8/3.43 \times 10^{-7}$
x3(10)	$1.27 \times 10^{-3}/8/3/2.00 \times 10^{-7}$	$2.51 \times 10^{-3}/14/5/7.19 \times 10^{-8}$	$3.17 \times 10^{-3}/25/8/4.88 \times 10^{-7}$
x4(10)	$7.97 \times 10^{-3}/40/14/2.70 \times 10^{-7}$	$8.08 \times 10^{-3}/49/17/7.74 \times 10^{-7}$	$1.51 \times 10^{-2}/97/32/4.85 \times 10^{-7}$
x5(10)	$4.56 \times 10^{-3}/26/9/6.15 \times 10^{-8}$	$9.67 \times 10^{-3}/42/16/1.01 \times 10^{-7}$	$1.26 \times 10^{-2}/79/26/9.95 \times 10^{-7}$
x6(10)	$1.00 \times 10^{-2}/40/14/2.69 \times 10^{-7}$	$7.89 \times 10^{-3}/45/16/4.43 \times 10^{-7}$	$1.55 \times 10^{-2}/97/32/4.88 \times 10^{-7}$
x7(10)	$8.85 \times 10^{-3}/40/14/2.70 \times 10^{-7}$	$9.63 \times 10^{-3}/49/17/7.74 \times 10^{-7}$	$1.61 \times 10^{-2}/97/32/4.85 \times 10^{-7}$
x8(10)	$7.50 \times 10^{-3}/37/13/9.78 \times 10^{-7}$	$1.09 \times 10^{-2}/59/20/4.61 \times 10^{-8}$	$1.50 \times 10^{-2}/91/30/9.04 \times 10^{-7}$
x1(50)	$8.56 \times 10^{-3}/12/5/6.49 \times 10^{-7}$	$2.03 \times 10^{-2}/27/11/1.00 \times 10^{-7}$	$1.79 \times 10^{-2}/34/11/6.88 \times 10^{-7}$
x2(50)	$5.63 \times 10^{-3}/10/4/1.65 \times 10^{-8}$	$9.17 \times 10^{-3}/18/7/5.57 \times 10^{-8}$	$9.69 \times 10^{-3}/25/8/3.43 \times 10^{-7}$
x3(50)	$4.62 \times 10^{-3}/8/3/2.00 \times 10^{-7}$	$6.75 \times 10^{-3}/14/5/7.19 \times 10^{-8}$	$9.82 \times 10^{-3}/25/8/4.88 \times 10^{-7}$
x4(50)	$2.99 \times 10^{-2}/40/14/3.65 \times 10^{-7}$	$3.13 \times 10^{-2}/47/17/4.20 \times 10^{-8}$	$5.09 \times 10^{-2}/100/33/8.67 \times 10^{-7}$
x5(50)	$1.85 \times 10^{-2}/26/9/6.15 \times 10^{-8}$	$2.43 \times 10^{-2}/39/13/7.07 \times 10^{-8}$	$4.00 \times 10^{-2}/79/26/9.95 \times 10^{-7}$
x6(50)	$3.18 \times 10^{-2}/40/14/3.65 \times 10^{-7}$	$2.60 \times 10^{-2}/42/14/8.30 \times 10^{-8}$	$5.30 \times 10^{-2}/100/33/8.68 \times 10^{-7}$
x7(50)	$2.55 \times 10^{-2}/40/14/3.65 \times 10^{-7}$	$3.04 \times 10^{-2}/47/17/4.20 \times 10^{-8}$	$5.10 \times 10^{-2}/100/33/8.67 \times 10^{-7}$
x8(50)	$2.91 \times 10^{-2}/40/14/3.61 \times 10^{-7}$	$3.75 \times 10^{-2}/63/20/7.11 \times 10^{-8}$	$4.94 \times 10^{-2}/97/32/9.80 \times 10^{-7}$
x1(100)	$1.97 \times 10^{-2}/14/6/1.09 \times 10^{-8}$	$3.18 \times 10^{-2}/27/11/1.42 \times 10^{-7}$	$3.09 \times 10^{-2}/34/11/9.72 \times 10^{-7}$
x2(100)	$9.06 \times 10^{-3}/10/4/1.65 \times 10^{-8}$	$1.52 \times 10^{-2}/18/7/5.57 \times 10^{-8}$	$1.74 \times 10^{-2}/25/8/3.43 \times 10^{-7}$
x3(100)	$7.64 \times 10^{-3}/8/3/2.00 \times 10^{-7}$	$1.07 \times 10^{-2}/14/5/7.19 \times 10^{-8}$	$1.55 \times 10^{-2}/25/8/4.88 \times 10^{-7}$
x4(100)	$4.34 \times 10^{-2}/40/14/7.05 \times 10^{-7}$	$4.65 \times 10^{-2}/45/16/3.74 \times 10^{-7}$	$8.37 \times 10^{-2}/103/34/5.77 \times 10^{-7}$
x5(100)	$2.98 \times 10^{-2}/26/9/6.15 \times 10^{-8}$	$4.12 \times 10^{-2}/41/14/4.50 \times 10^{-7}$	$6.93 \times 10^{-2}/79/26/9.95 \times 10^{-7}$
x6(100)	$4.34 \times 10^{-2}/40/14/7.05 \times 10^{-7}$	$4.32 \times 10^{-2}/41/14/6.54 \times 10^{-7}$	$8.85 \times 10^{-2}/103/34/5.78 \times 10^{-7}$
x7(100)	$5.00 \times 10^{-2}/40/14/7.05 \times 10^{-7}$	$4.74 \times 10^{-2}/45/16/3.74 \times 10^{-7}$	$9.16 \times 10^{-2}/103/34/5.77 \times 10^{-7}$
x8(100)	$4.77 \times 10^{-2}/40/14/7.04 \times 10^{-7}$	$6.10 \times 10^{-2}/54/19/6.43 \times 10^{-7}$	$8.39 \times 10^{-2}/103/34/5.47 \times 10^{-7}$

Table 6. Numerical results for Problem 6.

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(1)	$2.71 \times 10^{-4}/6/1/0.00 \times 10^{0}$	$2.30 \times 10^{-4}/6/1/0.00 \times 10^{0}$	$1.25 \times 10^{-3}/57/9/6.00 \times 10^{-7}$
x2(1)	$1.63 \times 10^{-4}/9/1/0.00 \times 10^{0}$	$1.48 \times 10^{-4}/9/1/0.00 \times 10^{0}$	$6.92 \times 10^{-4}/43/7/1.85 \times 10^{-7}$
x3(1)	$1.63 \times 10^{-3}/96/12/3.75 \times 10^{-15}$	$2.41 \times 10^{-3}/151/19/2.45 \times 10^{-7}$	$2.70 \times 10^{-3}/174/28/9.86 \times 10^{-7}$
x4(1)	$2.26 \times 10^{-3}/114/14/5.42 \times 10^{-7}$	$3.11 \times 10^{-3}/163/20/0.00 \times 10^{0}$	$4.31 \times 10^{-3}/219/35/5.14 \times 10^{-7}$
x5(1)	$1.66 \times 10^{-3}/81/11/7.63 \times 10^{-7}$	$2.88 \times 10^{-3}/152/19/3.07 \times 10^{-8}$	$3.36 \times 10^{-3}/182/29/4.94 \times 10^{-7}$
x6(1)	$2.33 \times 10^{-3}/115/15/0.00 \times 10^{0}$	$3.59 \times 10^{-3}/200/24/0.00 \times 10^{0}$	$4.57 \times 10^{-3}/237/38/2.96 \times 10^{-7}$
x7(1)	$2.25 \times 10^{-3}/114/14/5.42 \times 10^{-7}$	$3.02 \times 10^{-3}/163/20/0.00 \times 10^{0}$	$4.16 \times 10^{-3}/219/35/5.14 \times 10^{-7}$
x8(1)	$2.19 \times 10^{-3}/95/12/1.03 \times 10^{-7}$	$3.90 \times 10^{-3}/185/23/2.29 \times 10^{-7}$	$4.09 \times 10^{-3}/186/30/5.18 \times 10^{-7}$

Table 6. *Cont.*

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(5)	$1.31 \times 10^{-3}/6/1/0.00 \times 10^{0}$	$1.12 \times 10^{-3}/6/1/0.00 \times 10^{0}$	$7.51 \times 10^{-3}/63/10/1.25 \times 10^{-7}$
x2(5)	$1.05 \times 10^{-3}/9/1/0.00 \times 10^{0}$	$8.05 \times 10^{-4}/9/1/0.00 \times 10^{0}$	$4.20 \times 10^{-3}/43/7/1.85 \times 10^{-7}$
x3(5)	$3.70 \times 10^{-4}/3/1/0.00 \times 10^{0}$	$4.07 \times 10^{-4}/3/1/0.00 \times 10^{0}$	$3.86 \times 10^{-4}/3/1/0.00 \times 10^{0}$
x4(5)	$1.65 \times 10^{-2}/179/22/3.86 \times 10^{-7}$	$1.36 \times 10^{-2}/163/21/9.04 \times 10^{-7}$	$2.00 \times 10^{-2}/244/39/4.27 \times 10^{-7}$
x5(5)	$1.26 \times 10^{-2}/136/18/0.00 \times 10^{0}$	$1.13 \times 10^{-2}/143/18/2.74 \times 10^{-7}$	$1.59 \times 10^{-2}/194/31/8.94 \times 10^{-7}$
x6(5)	$1.58 \times 10^{-2}/185/23/4.39 \times 10^{-7}$	$1.13 \times 10^{-2}/141/18/2.33 \times 10^{-7}$	$2.13 \times 10^{-2}/255/41/4.07 \times 10^{-7}$
x7(5)	$1.55 \times 10^{-2}/179/22/3.86 \times 10^{-7}$	$1.34 \times 10^{-2}/163/21/9.04 \times 10^{-7}$	$1.94 \times 10^{-2}/244/39/4.27 \times 10^{-7}$
x8(5)	$7.83 \times 10^{-3}/81/10/0.00 \times 10^{0}$	$1.40 \times 10^{-2}/154/20/8.32 \times 10^{-7}$	$1.93 \times 10^{-2}/225/36/4.26 \times 10^{-7}$
x1(10)	$9.42 \times 10^{-4}/6/1/0.00 \times 10^{0}$	$8.92 \times 10^{-4}/6/1/0.00 \times 10^{0}$	$6.75 \times 10^{-3}/63/10/1.76 \times 10^{-7}$
x2(10)	$8.31 \times 10^{-4}/9/1/0.00 \times 10^{0}$	$8.03 \times 10^{-4}/9/1/0.00 \times 10^{0}$	$4.85 \times 10^{-3}/43/7/1.85 \times 10^{-7}$
x3(10)	$5.54 \times 10^{-4}/3/1/0.00 \times 10^{0}$	$4.58 \times 10^{-4}/3/1/0.00 \times 10^{0}$	$3.31 \times 10^{-4}/3/1/0.00 \times 10^{0}$
x4(10)	$1.20 \times 10^{-2}/105/13/1.67 \times 10^{-9}$	$1.65 \times 10^{-2}/157/20/5.74 \times 10^{-7}$	$2.66 \times 10^{-2}/244/39/4.95 \times 10^{-7}$
x5(10)	$1.61 \times 10^{-2}/122/16/0.00 \times 10^{0}$	$1.56 \times 10^{-2}/154/19/0.00 \times 10^{0}$	$1.93 \times 10^{-2}/182/29/4.08 \times 10^{-7}$
x6(10)	$1.10 \times 10^{-2}/90/11/2.02 \times 10^{-9}$	$1.44 \times 10^{-2}/134/17/2.39 \times 10^{-7}$	$2.34 \times 10^{-2}/226/36/7.42 \times 10^{-7}$
x7(10)	$1.21 \times 10^{-2}/105/13/1.67 \times 10^{-9}$	$1.60 \times 10^{-2}/157/20/5.74 \times 10^{-7}$	$2.61 \times 10^{-2}/244/39/4.95 \times 10^{-7}$
x8(10)	$1.21 \times 10^{-2}/98/12/0.00 \times 10^{0}$	$1.66 \times 10^{-2}/131/17/0.00 \times 10^{0}$	$2.84 \times 10^{-2}/255/41/8.67 \times 10^{-7}$
x1(50)	$3.23 \times 10^{-3}/6/1/0.00 \times 10^{0}$	$2.94 \times 10^{-3}/6/1/0.00 \times 10^{0}$	$2.81 \times 10^{-2}/63/10/3.94 \times 10^{-7}$
x2(50)	$2.65 \times 10^{-3}/9/1/0.00 \times 10^{0}$	$2.75 \times 10^{-3}/9/1/0.00 \times 10^{0}$	$1.46 \times 10^{-2}/43/7/1.85 \times 10^{-7}$
x3(50)	$1.44 \times 10^{-3}/3/1/0.00 \times 10^{0}$	$1.66 \times 10^{-3}/3/1/0.00 \times 10^{0}$	$1.39 \times 10^{-3}/3/1/0.00 \times 10^{0}$
x4(50)	$4.61 \times 10^{-2}/103/13/1.41 \times 10^{-7}$	$6.50 \times 10^{-2}/157/20/3.17 \times 10^{-7}$	$1.04 \times 10^{-1}/250/40/5.94 \times 10^{-7}$
x5(50)	$8.27 \times 10^{-2}/186/24/9.60 \times 10^{-8}$	$6.28 \times 10^{-2}/148/19/3.83 \times 10^{-7}$	$7.85 \times 10^{-2}/194/31/5.49 \times 10^{-7}$
x6(50)	$4.52 \times 10^{-2}/103/13/1.43 \times 10^{-7}$	$5.55 \times 10^{-2}/134/17/2.22 \times 10^{-16}$	$1.00 \times 10^{-1}/238/38/5.59 \times 10^{-7}$
x7(50)	$4.50 \times 10^{-2}/103/13/1.41 \times 10^{-7}$	$6.59 \times 10^{-2}/157/20/3.17 \times 10^{-7}$	$1.06 \times 10^{-1}/250/40/5.94 \times 10^{-7}$
x8(50)	$4.76 \times 10^{-2}/96/12/1.33 \times 10^{-7}$	$5.49 \times 10^{-2}/117/15/8.48 \times 10^{-7}$	$1.03 \times 10^{-1}/238/38/6.93 \times 10^{-7}$
x1(100)	$8.14 \times 10^{-3}/6/1/0.00 \times 10^{0}$	$6.15 \times 10^{-3}/6/1/0.00 \times 10^{0}$	$4.80 \times 10^{-2}/63/10/5.57 \times 10^{-7}$
x2(100)	$5.83 \times 10^{-3}/9/1/0.00 \times 10^{0}$	$4.69 \times 10^{-3}/9/1/0.00 \times 10^{0}$	$2.41 \times 10^{-2}/43/7/1.85 \times 10^{-7}$
x3(100)	$3.14 \times 10^{-3}/3/1/0.00 \times 10^{0}$	$2.90 \times 10^{-3}/3/1/0.00 \times 10^{0}$	$2.63 \times 10^{-3}/3/1/0.00 \times 10^{0}$
x4(100)	$1.27 \times 10^{-1}/166/21/2.91 \times 10^{-7}$	$1.31 \times 10^{-1}/173/22/5.65 \times 10^{-7}$	$1.81 \times 10^{-1}/250/40/9.26 \times 10^{-7}$
x5(100)	$1.04 \times 10^{-1}/146/19/0.00 \times 10^{0}$	$1.04 \times 10^{-1}/148/19/5.90 \times 10^{-7}$	$1.18 \times 10^{-1}/175/28/1.08 \times 10^{-7}$
x6(100)	$1.25 \times 10^{-1}/166/21/3.23 \times 10^{-7}$	$9.50 \times 10^{-2}/134/17/2.71 \times 10^{-15}$	$1.71 \times 10^{-1}/238/38/9.99 \times 10^{-7}$
x7(100)	$1.19 \times 10^{-1}/166/21/2.91 \times 10^{-7}$	$1.23 \times 10^{-1}/173/22/5.65 \times 10^{-7}$	$1.76 \times 10^{-1}/250/40/9.26 \times 10^{-7}$
x8(100)	$9.91 \times 10^{-2}/111/14/9.76 \times 10^{-9}$	$1.06 \times 10^{-1}/126/16/3.64 \times 10^{-15}$	$2.06 \times 10^{-1}/273/44/9.01 \times 10^{-7}$

Table 7. Numerical results for Problem 7.

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(1)	$9.76 \times 10^{-3}/90/19/1.33 \times 10^{-7}$	$1.35 \times 10^{-2}/120/24/3.94 \times 10^{-7}$	$3.80 \times 10^{-2}/381/90/9.43 \times 10^{-7}$
x2(1)	$8.69 \times 10^{-3}/84/18/8.04 \times 10^{-7}$	$1.28 \times 10^{-2}/133/27/6.50 \times 10^{-7}$	$4.29 \times 10^{-2}/434/103/6.56 \times 10^{-7}$
x3(1)	$1.10 \times 10^{-2}/98/21/6.30 \times 10^{-7}$	$1.32 \times 10^{-2}/132/26/9.64 \times 10^{-7}$	$4.17 \times 10^{-2}/426/101/6.50 \times 10^{-7}$
x4(1)	$9.65 \times 10^{-3}/102/22/8.39 \times 10^{-7}$	$1.07 \times 10^{-2}/109/22/5.40 \times 10^{-7}$	$4.01 \times 10^{-2}/418/99/5.99 \times 10^{-7}$
x5(1)	$7.44 \times 10^{-3}/76/16/9.56 \times 10^{-7}$	$9.36 \times 10^{-3}/96/19/6.21 \times 10^{-7}$	$4.00 \times 10^{-2}/401/95/7.48 \times 10^{-7}$
x6(1)	$1.01 \times 10^{-2}/101/22/9.38 \times 10^{-7}$	$1.17 \times 10^{-2}/117/23/2.55 \times 10^{-7}$	$4.17 \times 10^{-2}/418/99/5.98 \times 10^{-7}$
x7(1)	$8.83 \times 10^{-3}/89/19/3.19 \times 10^{-7}$	$1.24 \times 10^{-2}/129/25/2.52 \times 10^{-7}$	$3.56 \times 10^{-2}/357/84/5.14 \times 10^{-7}$
x8(1)	$8.88 \times 10^{-3}/91/20/2.65 \times 10^{-7}$	$1.71 \times 10^{-2}/167/33/4.74 \times 10^{-7}$	$4.15 \times 10^{-2}/417/99/9.79 \times 10^{-7}$
x1(5)	$4.95 \times 10^{-2}/94/20/5.32 \times 10^{-7}$	$5.72 \times 10^{-2}/111/22/5.98 \times 10^{-7}$	$1.94 \times 10^{-1}/369/87/5.58 \times 10^{-7}$
x2(5)	$5.82 \times 10^{-2}/113/25/1.86 \times 10^{-7}$	$6.32 \times 10^{-2}/129/27/5.34 \times 10^{-7}$	$2.32 \times 10^{-1}/458/109/6.29 \times 10^{-7}$
x3(5)	$5.97 \times 10^{-2}/112/25/8.38 \times 10^{-7}$	$6.88 \times 10^{-2}/137/29/4.35 \times 10^{-7}$	$2.40 \times 10^{-1}/458/109/6.29 \times 10^{-7}$
x4(5)	$5.91 \times 10^{-2}/111/24/2.77 \times 10^{-7}$	$4.96 \times 10^{-2}/94/19/7.52 \times 10^{-7}$	$2.31 \times 10^{-1}/446/106/5.82 \times 10^{-7}$
x5(5)	$5.37 \times 10^{-2}/103/23/1.59 \times 10^{-7}$	$8.01 \times 10^{-2}/166/31/5.77 \times 10^{-7}$	$2.40 \times 10^{-1}/458/109/6.07 \times 10^{-7}$
x6(5)	$6.43 \times 10^{-2}/122/27/6.59 \times 10^{-7}$	$4.75 \times 10^{-2}/94/19/6.85 \times 10^{-7}$	$2.28 \times 10^{-1}/446/106/5.81 \times 10^{-7}$
x7(5)	$5.11 \times 10^{-2}/96/21/3.29 \times 10^{-7}$	$5.95 \times 10^{-2}/115/23/8.71 \times 10^{-7}$	$2.29 \times 10^{-1}/454/108/6.09 \times 10^{-7}$
x8(5)	$5.29 \times 10^{-2}/103/22/5.52 \times 10^{-7}$	$5.48 \times 10^{-2}/111/23/7.31 \times 10^{-7}$	$2.06 \times 10^{-1}/405/96/7.94 \times 10^{-7}$

Table 7. *Cont.*

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(10)	$1.14 \times 10^{-1}/118/26/7.92 \times 10^{-7}$	$1.24 \times 10^{-1}/132/27/4.22 \times 10^{-7}$	$4.45 \times 10^{-1}/474/113/7.04 \times 10^{-7}$
x2(10)	$1.04 \times 10^{-1}/108/24/5.78 \times 10^{-7}$	$1.06 \times 10^{-1}/112/23/4.30 \times 10^{-7}$	$3.78 \times 10^{-1}/395/94/6.07 \times 10^{-7}$
x3(10)	$1.04 \times 10^{-1}/109/24/5.61 \times 10^{-7}$	$9.55 \times 10^{-2}/101/21/4.62 \times 10^{-7}$	$3.72 \times 10^{-1}/395/94/6.07 \times 10^{-7}$
x4(10)	$1.15 \times 10^{-1}/123/27/4.37 \times 10^{-7}$	$1.13 \times 10^{-1}/122/25/6.26 \times 10^{-7}$	$4.19 \times 10^{-1}/446/106/7.30 \times 10^{-7}$
x5(10)	$1.29 \times 10^{-1}/135/30/4.75 \times 10^{-7}$	$1.22 \times 10^{-1}/132/27/1.79 \times 10^{-7}$	$3.48 \times 10^{-1}/370/88/9.63 \times 10^{-7}$
x6(10)	$1.24 \times 10^{-1}/130/29/5.79 \times 10^{-7}$	$1.19 \times 10^{-1}/129/27/7.05 \times 10^{-7}$	$4.25 \times 10^{-1}/446/106/7.30 \times 10^{-7}$
x7(10)	$1.07 \times 10^{-1}/112/25/2.50 \times 10^{-7}$	$1.20 \times 10^{-1}/127/27/4.10 \times 10^{-7}$	$4.28 \times 10^{-1}/444/106/9.47 \times 10^{-7}$
x8(10)	$1.05 \times 10^{-1}/109/24/3.00 \times 10^{-7}$	$1.24 \times 10^{-1}/133/27/9.07 \times 10^{-7}$	$4.46 \times 10^{-1}/466/111/6.03 \times 10^{-7}$
x1(50)	$6.01 \times 10^{-1}/118/26/8.42 \times 10^{-7}$	$5.36 \times 10^{-1}/106/22/8.97 \times 10^{-7}$	$2.45 \times 10^{0}/468/112/7.75 \times 10^{-7}$
x2(50)	$5.22 \times 10^{-1}/102/23/3.61 \times 10^{-7}$	$6.67 \times 10^{-1}/130/27/2.68 \times 10^{-7}$	$2.42 \times 10^{0}/464/111/8.61 \times 10^{-7}$
x3(50)	$5.24 \times 10^{-1}/102/23/3.92 \times 10^{-7}$	$6.79 \times 10^{-1}/133/27/9.03 \times 10^{-7}$	$2.39 \times 10^{0}/464/111/8.57 \times 10^{-7}$
x4(50)	$5.01 \times 10^{-1}/98/22/6.48 \times 10^{-7}$	$7.64 \times 10^{-1}/150/31/2.72 \times 10^{-7}$	$2.35 \times 10^{0}/452/108/8.91 \times 10^{-7}$
x5(50)	$5.68 \times 10^{-1}/111/25/4.20 \times 10^{-7}$	$7.90 \times 10^{-1}/155/32/7.32 \times 10^{-7}$	$2.41 \times 10^{0}/464/111/8.34 \times 10^{-7}$
x6(50)	$5.04 \times 10^{-1}/98/22/6.48 \times 10^{-7}$	$7.17 \times 10^{-1}/141/29/5.02 \times 10^{-7}$	$2.34 \times 10^{0}/452/108/8.91 \times 10^{-7}$
x7(50)	$6.01 \times 10^{-1}/115/26/4.87 \times 10^{-7}$	$6.63 \times 10^{-1}/128/27/9.58 \times 10^{-7}$	$2.40 \times 10^{0}/460/110/9.28 \times 10^{-7}$
x8(50)	$5.11 \times 10^{-1}/98/22/7.88 \times 10^{-7}$	$6.79 \times 10^{-1}/131/28/5.48 \times 10^{-7}$	$2.30 \times 10^{0}/432/103/7.76 \times 10^{-7}$
x1(100)	$9.81 \times 10^{-1}/97/22/8.53 \times 10^{-7}$	$1.08 \times 10^{0}/112/23/5.11 \times 10^{-7}$	$4.53 \times 10^{0}/456/109/9.55 \times 10^{-7}$
x2(100)	$1.19 \times 10^{0}/118/27/8.34 \times 10^{-7}$	$1.33 \times 10^{0}/129/26/4.22 \times 10^{-7}$	$4.91 \times 10^{0}/497/119/6.39 \times 10^{-7}$
x3(100)	$1.25 \times 10^{0}/127/29/2.67 \times 10^{-7}$	$1.38 \times 10^{0}/143/30/9.45 \times 10^{-7}$	$4.77 \times 10^{0}/481/115/6.26 \times 10^{-7}$
x4(100)	$1.06 \times 10^{0}/107/24/3.28 \times 10^{-7}$	$1.17 \times 10^{0}/122/25/7.53 \times 10^{-7}$	$4.52 \times 10^{0}/456/109/8.83 \times 10^{-7}$
x5(100)	$9.93 \times 10^{-1}/101/23/6.39 \times 10^{-7}$	$1.14 \times 10^{0}/119/24/4.93 \times 10^{-7}$	$4.92 \times 10^{0}/493/118/6.21 \times 10^{-7}$
x6(100)	$1.06 \times 10^{0}/107/24/3.29 \times 10^{-7}$	$1.25 \times 10^{0}/128/26/3.82 \times 10^{-8}$	$4.56 \times 10^{0}/456/109/8.83 \times 10^{-7}$
x7(100)	$9.94 \times 10^{-1}/98/22/9.49 \times 10^{-8}$	$1.25 \times 10^{0}/128/26/3.81 \times 10^{-7}$	$4.85 \times 10^{0}/485/116/6.04 \times 10^{-7}$
x8(100)	$1.02 \times 10^{0}/102/23/3.43 \times 10^{-7}$	$1.09 \times 10^{0}/113/23/5.86 \times 10^{-7}$	$4.82 \times 10^{0}/485/116/6.34 \times 10^{-7}$

Table 8. Numerical results for Problem 8.

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(1)	$4.04 \times 10^{-3}/44/14/6.03 \times 10^{-7}$	$1.97 \times 10^{-3}/26/8/4.64 \times 10^{-8}$	$2.25 \times 10^{-3}/29/7/5.31 \times 10^{-7}$
x2(1)	$1.03 \times 10^{-2}/159/23/6.90 \times 10^{-7}$	$1.31 \times 10^{-2}/205/28/9.90 \times 10^{-7}$	$1.02 \times 10^{-2}/160/28/5.49 \times 10^{-7}$
x3(1)	$9.97 \times 10^{-3}/164/24/8.26 \times 10^{-7}$	$1.26 \times 10^{-2}/215/30/7.18 \times 10^{-7}$	$9.64 \times 10^{-3}/159/28/5.57 \times 10^{-7}$
x4(1)	$7.67 \times 10^{-3}/128/19/9.28 \times 10^{-7}$	$8.69 \times 10^{-3}/140/20/5.23 \times 10^{-7}$	$9.64 \times 10^{-3}/159/30/4.97 \times 10^{-7}$
x5(1)	$9.63 \times 10^{-3}/159/23/7.20 \times 10^{-7}$	$1.32 \times 10^{-2}/213/30/6.66 \times 10^{-7}$	$9.20 \times 10^{-3}/155/27/4.53 \times 10^{-7}$
x6(1)	$7.92 \times 10^{-3}/128/19/9.28 \times 10^{-7}$	$8.39 \times 10^{-3}/144/20/5.73 \times 10^{-7}$	$1.03 \times 10^{-2}/159/30/4.88 \times 10^{-7}$
x7(1)	$7.57 \times 10^{-3}/128/19/7.06 \times 10^{-7}$	$9.54 \times 10^{-3}/155/22/5.75 \times 10^{-7}$	$9.33 \times 10^{-3}/155/29/9.48 \times 10^{-7}$
x8(1)	$1.15 \times 10^{-2}/187/27/5.73 \times 10^{-7}$	$1.32 \times 10^{-2}/221/30/5.32 \times 10^{-7}$	$7.11 \times 10^{-3}/120/20/9.25 \times 10^{-7}$
x1(5)	$2.10 \times 10^{-2}/47/15/3.88 \times 10^{-7}$	$1.00 \times 10^{-2}/26/8/1.04 \times 10^{-7}$	$1.21 \times 10^{-2}/33/8/6.70 \times 10^{-8}$
x2(5)	$5.09 \times 10^{-2}/165/24/7.30 \times 10^{-7}$	$6.85 \times 10^{-2}/230/32/9.16 \times 10^{-7}$	$4.22 \times 10^{-2}/144/25/8.58 \times 10^{-7}$
x3(5)	$5.39 \times 10^{-2}/177/26/9.37 \times 10^{-7}$	$6.29 \times 10^{-2}/213/30/9.98 \times 10^{-7}$	$4.46 \times 10^{-2}/148/26/4.02 \times 10^{-7}$
x4(5)	$4.28 \times 10^{-2}/133/20/9.06 \times 10^{-7}$	$4.55 \times 10^{-2}/147/21/7.30 \times 10^{-7}$	$4.27 \times 10^{-2}/137/26/4.54 \times 10^{-7}$
x5(5)	$5.68 \times 10^{-2}/183/27/7.21 \times 10^{-7}$	$6.18 \times 10^{-2}/197/27/5.19 \times 10^{-7}$	$4.93 \times 10^{-2}/155/27/7.52 \times 10^{-7}$
x6(5)	$4.07 \times 10^{-2}/133/20/9.04 \times 10^{-7}$	$5.31 \times 10^{-2}/175/25/7.40 \times 10^{-7}$	$4.29 \times 10^{-2}/137/26/4.73 \times 10^{-7}$
x7(5)	$4.01 \times 10^{-2}/133/20/5.99 \times 10^{-7}$	$5.52 \times 10^{-2}/182/26/5.60 \times 10^{-7}$	$4.15 \times 10^{-2}/135/26/9.26 \times 10^{-7}$
x8(5)	$6.15 \times 10^{-2}/206/30/5.21 \times 10^{-7}$	$7.30 \times 10^{-2}/239/33/2.79 \times 10^{-7}$	$4.02 \times 10^{-2}/126/21/5.03 \times 10^{-7}$
x1(10)	$3.57 \times 10^{-2}/47/15/5.49 \times 10^{-7}$	$1.98 \times 10^{-2}/26/8/1.47 \times 10^{-7}$	$2.13 \times 10^{-2}/33/8/9.47 \times 10^{-8}$
x2(10)	$9.53 \times 10^{-2}/165/24/8.88 \times 10^{-7}$	$1.21 \times 10^{-1}/215/30/5.80 \times 10^{-7}$	$8.14 \times 10^{-2}/144/25/6.81 \times 10^{-7}$
x3(10)	$9.90 \times 10^{-2}/172/25/5.14 \times 10^{-7}$	$1.15 \times 10^{-1}/202/28/3.43 \times 10^{-7}$	$8.20 \times 10^{-2}/136/24/8.97 \times 10^{-7}$
x4(10)	$7.54 \times 10^{-2}/133/20/4.98 \times 10^{-7}$	$7.96 \times 10^{-2}/144/21/3.17 \times 10^{-7}$	$8.38 \times 10^{-2}/141/27/4.63 \times 10^{-7}$
x5(10)	$9.17 \times 10^{-2}/165/24/9.01 \times 10^{-7}$	$1.07 \times 10^{-1}/193/27/7.56 \times 10^{-7}$	$8.60 \times 10^{-2}/150/26/5.92 \times 10^{-7}$
x6(10)	$7.73 \times 10^{-2}/133/20/4.98 \times 10^{-7}$	$1.12 \times 10^{-1}/206/29/3.42 \times 10^{-7}$	$8.49 \times 10^{-2}/141/27/4.68 \times 10^{-7}$
x7(10)	$7.29 \times 10^{-2}/126/19/8.04 \times 10^{-7}$	$1.17 \times 10^{-1}/204/29/4.02 \times 10^{-7}$	$8.39 \times 10^{-2}/140/27/8.91 \times 10^{-7}$
x8(10)	$1.13 \times 10^{-1}/200/29/7.31 \times 10^{-7}$	$1.32 \times 10^{-1}/222/30/3.73 \times 10^{-7}$	$7.25 \times 10^{-2}/126/21/6.66 \times 10^{-7}$

Table 8. *Cont.*

Init(n)	ITTCG Time/Nfunc/Niter/Norm	HTTCGP Time/Nfunc/Niter/Norm	ZYL Time/Nfunc/Niter/Norm
x1(50)	$1.83 \times 10^{-1}/50/16/3.53 \times 10^{-7}$	$9.11 \times 10^{-2}/26/8/3.28 \times 10^{-7}$	$1.12 \times 10^{-1}/33/8/2.12 \times 10^{-7}$
x2(50)	$5.21 \times 10^{-1}/171/25/8.48 \times 10^{-7}$	$5.41 \times 10^{-1}/177/25/7.99 \times 10^{-7}$	$4.51 \times 10^{-1}/143/25/5.60 \times 10^{-7}$
x3(50)	$5.16 \times 10^{-1}/171/25/8.84 \times 10^{-7}$	$7.61 \times 10^{-1}/251/35/5.12 \times 10^{-7}$	$4.13 \times 10^{-1}/131/23/8.83 \times 10^{-7}$
x4(50)	$4.21 \times 10^{-1}/138/21/5.04 \times 10^{-7}$	$4.63 \times 10^{-1}/150/22/8.39 \times 10^{-7}$	$4.68 \times 10^{-1}/150/29/9.54 \times 10^{-7}$
x5(50)	$5.25 \times 10^{-1}/171/25/8.61 \times 10^{-7}$	$7.06 \times 10^{-1}/231/33/7.37 \times 10^{-7}$	$4.45 \times 10^{-1}/143/25/4.76 \times 10^{-7}$
x6(50)	$4.20 \times 10^{-1}/138/21/5.04 \times 10^{-7}$	$4.90 \times 10^{-1}/164/24/3.00 \times 10^{-7}$	$4.77 \times 10^{-1}/150/29/9.67 \times 10^{-7}$
x7(50)	$4.06 \times 10^{-1}/131/20/6.58 \times 10^{-7}$	$4.65 \times 10^{-1}/152/22/7.21 \times 10^{-7}$	$5.08 \times 10^{-1}/156/30/4.61 \times 10^{-7}$
x8(50)	$5.92 \times 10^{-1}/195/28/8.22 \times 10^{-7}$	$8.47 \times 10^{-1}/281/39/2.74 \times 10^{-7}$	$4.11 \times 10^{-1}/132/22/6.17 \times 10^{-7}$
x1(100)	$3.58 \times 10^{-1}/50/16/4.99 \times 10^{-7}$	$1.80 \times 10^{-1}/26/8/4.64 \times 10^{-7}$	$2.12 \times 10^{-1}/33/8/3.00 \times 10^{-7}$
x2(100)	$1.05 \times 10^{0}/177/26/5.90 \times 10^{-7}$	$1.42 \times 10^{0}/244/34/4.53 \times 10^{-7}$	$8.12 \times 10^{-1}/137/24/6.64 \times 10^{-7}$
x3(100)	$1.04 \times 10^{0}/178/26/5.25 \times 10^{-7}$	$9.33 \times 10^{-1}/158/22/9.92 \times 10^{-7}$	$7.95 \times 10^{-1}/131/23/7.50 \times 10^{-7}$
x4(100)	$7.89 \times 10^{-1}/131/20/5.53 \times 10^{-7}$	$1.33 \times 10^{0}/226/33/7.44 \times 10^{-7}$	$9.26 \times 10^{-1}/149/29/6.70 \times 10^{-7}$
x5(100)	$1.06 \times 10^{0}/177/26/5.21 \times 10^{-7}$	$1.47 \times 10^{0}/242/34/5.15 \times 10^{-7}$	$8.52 \times 10^{-1}/137/24/7.67 \times 10^{-7}$
x6(100)	$7.95 \times 10^{-1}/131/20/5.53 \times 10^{-7}$	$1.26 \times 10^{0}/206/30/2.16 \times 10^{-7}$	$9.38 \times 10^{-1}/149/29/6.70 \times 10^{-7}$
x7(100)	$7.34 \times 10^{-1}/124/19/8.94 \times 10^{-7}$	$8.96 \times 10^{-1}/151/22/7.80 \times 10^{-7}$	$9.80 \times 10^{-1}/158/31/6.50 \times 10^{-7}$
x8(100)	$1.20 \times 10^{0}/202/29/5.27 \times 10^{-7}$	$1.35 \times 10^{0}/231/32/9.96 \times 10^{-7}$	$7.90 \times 10^{-1}/132/22/9.92 \times 10^{-7}$

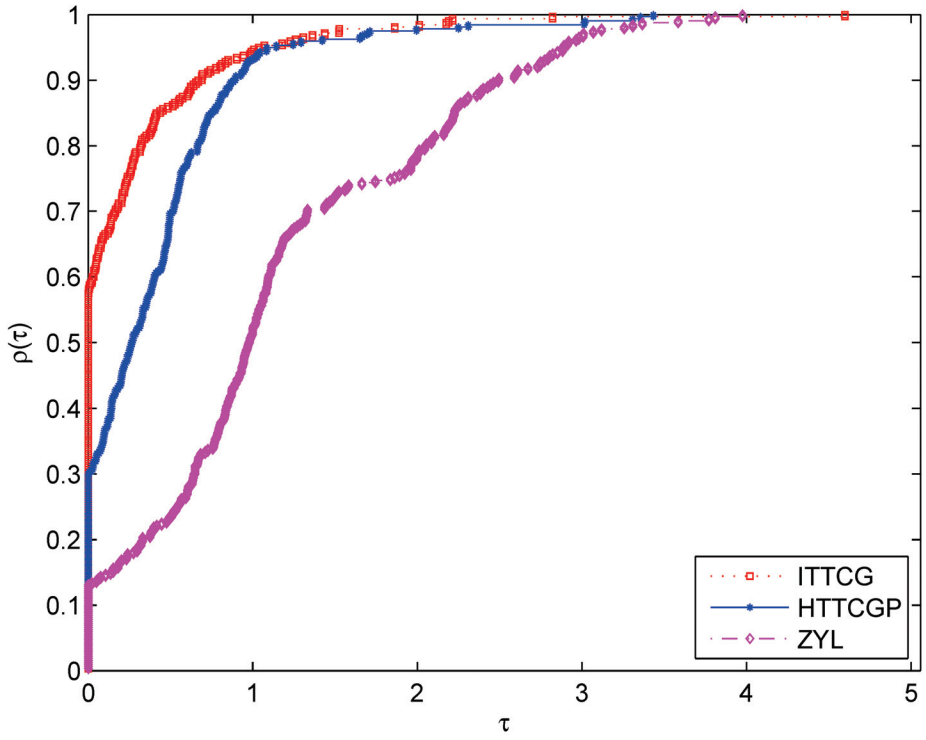


Figure 1. Performance profiles for time.

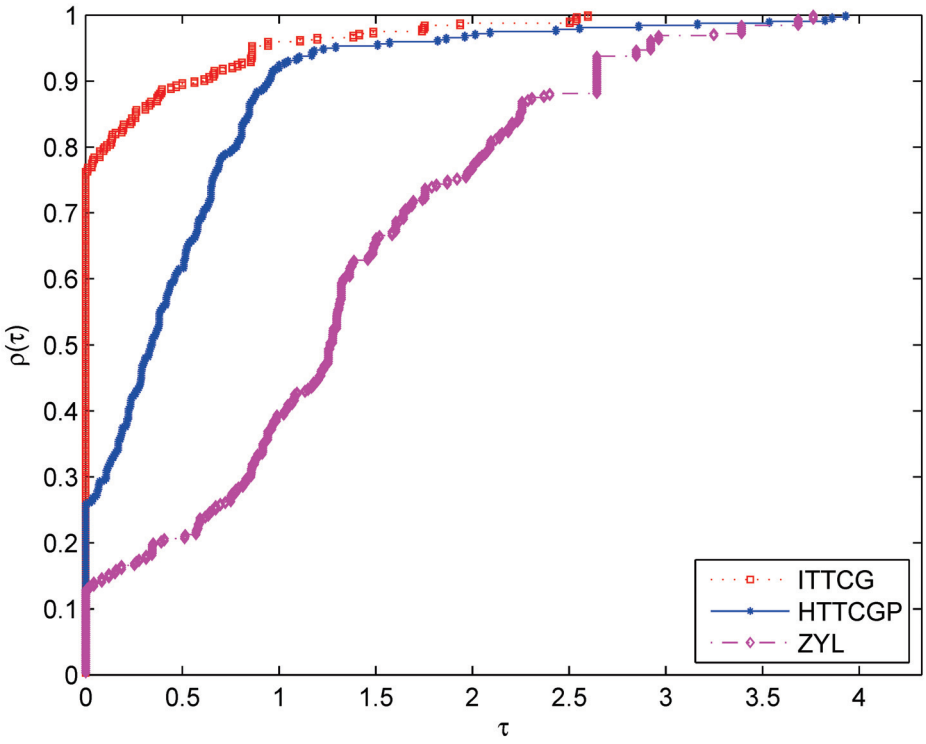


Figure 2. Performance profiles for Nfunc.

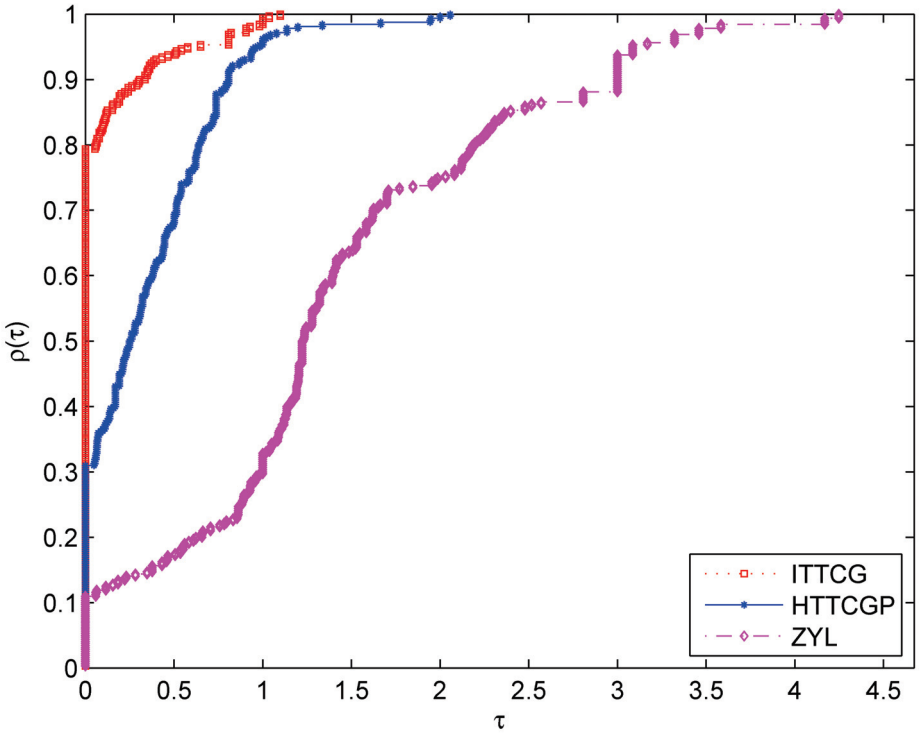


Figure 3. Performance profiles for Niter.

5. Applications in Image Denoising

Image denoising, a well-known inverse problem in the field of compressive sensing, poses significant challenges due to various sources of image noise. This noise can originate from faulty pixels in camera sensors, errors in hardware storage locations, or transmission through noisy channels. Some pixels in the image are contaminated by Gaussian noise, known as additive white Gaussian noise (AWGN), or impulse noise, known as salt-and-pepper noise. Our primary focus is on images affected by salt-and-pepper noise. This type of noise is particularly challenging because it can obscure important image details and edges, which are critical for various image processing applications such as medical imaging, remote sensing, and object recognition. In the works [30,31], a robust two-phase scheme was proposed to detect and remove salt-and-pepper noise. The first stage involves using an adaptive median filter to identify noisy pixels. The adaptive median filter is effective because it can handle varying noise densities and preserve image edges better than standard median filters. Once the noisy pixels have been detected, the second stage employs variational methods to restore the image. Variational methods are advantageous because they formulate image restoration as an optimization problem, balancing between data fidelity and the smoothness of the image. To enhance readability and comprehensiveness, we now provide an in-depth and concise explanation of this method.

Given an original image x with dimensions $m \times n$, let $x_{i,j}$ represent the grayscale level at the pixel location $(i,j) \in \mathcal{A} = \{1,2,\ldots,m\} \times \{1,2,\ldots,n\}$. To facilitate image processing and analysis, we often consider the neighborhood of each pixel. Let $\mathcal{V}_{i,j}$ denote the neighborhood of (i,j), defined as $\mathcal{V}_{i,j} = \{(i,j-1),(i,j+1),(i-1,j),(i+1,j)\}$. This represents the four direct neighbors of the pixel at (i,j): left, right, up, and down. A common type of noise is salt-and-pepper noise, which randomly alters the pixel values to either the minimum or maximum grayscale level, creating a "salt-and-pepper" appearance. When the image x is corrupted by salt-and-pepper noise, the observed noisy image is presented by y. The grayscale level at pixel location (i,j) in the noisy image y is given by the following probabilistic model:

$$y_{i,j} = \begin{cases} x_{i,j}, & \text{with probability } 1-r, \\ s_{min}, & \text{with probability } p, \\ s_{max} & \text{with probability } q, \end{cases}$$

where $[s_{min}, s_{max}]$ is the range of $x_{i,j}$, and $r = q + p$ represents the overall noise level. To obtain the denoised image u^*, we employ a comprehensive two-phase scheme. In the first stage, we apply an adaptive median filter to the noisy image y. This process results in an intermediate image, denoted as $\tilde{y}$. Based on the differences between the noisy image y and the filtered image $\tilde{y}$, we define the noise candidate set as follows:

$$\mathcal{N} = \{(i,j) \in \mathcal{A} : \tilde{y}_{i,j} \neq y_{i,j} \text{ and } y_{i,j} = s_{min} \text{ or } s_{max}\}.$$

In the second stage, we proceed with the recovery of the noisy pixels identified in the set $\mathcal{N}$. For each pixel $(i,j) \in \mathcal{N}$, if it is not contaminated by noise, we retain its original value, i.e., $u^*_{i,j} = y_{i,j}$. For noisy pixels $y_{i,j}$, $(i,j) \in \mathcal{N}$, we need to perform recovery. We set $u^*_{m,n} = y_{m,n}$ for $(m,n) \in \mathcal{V}_{i,j} \setminus \mathcal{N}$, ensuring that neighboring non-noisy pixels are preserved. For the pixels $(m,n) \in \mathcal{V}_{i,j} \cap \mathcal{N}$, which are in the neighborhood and are also candidates for noise, we must also recover their values. To restore the image, we aim to minimize the following function:

$$\min_u E(u) = \sum_{(i,j)\in\mathcal{N}} \left\{ \sum_{(m,n)\in\mathcal{V}_{i,j}\setminus\mathcal{N}} 2\phi_\alpha(u_{i,j} - y_{m,n}) + \sum_{(m,n)\in\mathcal{V}_{i,j}\cap\mathcal{N}} \phi_\alpha(u_{i,j} - u_{m,n}) \right\},$$

where ϕ_α is an even edge-preserving potential function with parameter α. We know from [11] that if ϕ_α is convex, then $\nabla E(u)$ is monotone.

We utilized the well-known grayscale test images: lighthouse (512 × 512), peppers (256 × 256), boat (512 × 512), Kiel (512 × 512), fruits (256 × 256), brain (256 × 256), clown (512 × 512), couple (512 × 512), trucks (512 × 512), baboon (256 × 256), Barbara (512 × 512), and cameraman (256 × 256). Each image was affected by 30% salt-and-pepper noise, and the experiments were repeated 10 times with different noise samples. The detailed numerical results are presented in Table 9, where $\overline{\text{Niter}}$, $\overline{\text{Time}}$, $\overline{\text{PSNR}}$, and $\overline{\text{SSIM}}$ represent the number of average iterations, the average CPU time in seconds, the average peak signal-to-noise ratio, and the average structural similarity index, respectively. Additionally, we display the noisy images with 30% salt-and-pepper noise and the images restored using the ITTCG, HTTCGP, and ZYL algorithms (see Figures 4 and 5). From the results in Table 9, and Figures 4 and 5, we can draw the following conclusions: (i) All images affected by 30% salt-and-pepper noise were successfully recovered by the ITTCG, HTTCGP, and ZYL algorithms. (ii) With a similar average structural similarity index, Algorithm ITTCG generally required less CPU time, fewer iterations, and achieved a lower peak signal-to-noise-ratio than the HTTCGP and ZYL algorithms, indicating that Algorithm ITTCG was efficient and competitive in image denoising.

Table 9. Efficiency comparison for different algorithms.

Image	**ITTCG** $\overline{Niter}/\overline{Time}/\overline{PSNR}/\overline{SSIM}$	**HTTCGP** $\overline{Niter}/\overline{Time}/\overline{PSNR}/\overline{SSIM}$	**ZYL** $\overline{Niter}/\overline{Time}/\overline{PSNR}/\overline{SSIM}$
lighthouse	28.2/5.73/30.85/0.97	50.5/11.51/31.32/0.97	58.7/14.24/31.12/0.97
peppers	19.4/1.18/33.24/0.96	51.5/3.19/33.77/0.96	29.4/1.94/33.37/0.96
boat	14.4/3.19/33.96/0.98	48.0/11.00/34.46/0.98	24.4/5.98/34.08/0.98
kiel	25.5/5.25/27.81/0.97	49.0/10.93/27.94/0.97	47.5/11.37/27.88/0.97
fruits	21.3/1.33/30.01/0.94	69.3/4.34/30.25/0.95	27.1/1.80/30.01/0.94
brain	14.2/0.90/31.07/0.87	23.9/1.51/31.13/0.87	18.7/1.28/31.10/0.87
clown	14.1/3.15/36.62/0.99	42.4/9.68/37.17/0.99	24.9/6.02/36.80/0.99
couple	14.4/3.15/34.26/0.99	36.1/8.38/34.60/0.99	24.2/5.90/34.33/0.99
trucks	12.1/2.77/34.07/0.98	25.0/5.89/34.13/0.98	18.4/4.66/34.08/0.98
baboon	32.9/1.72/24.68/ 0.87	35.8/1.85/24.67/0.87	72.1/4.47/24.68/0.87
barbara	15.6/3.31/29.03/0.96	46.1/10.39/29.07/0.96	27.2/6.47/29.04/0.96
Cameraman	19.0/1.13/29.95/0.96	37.6/2.33/30.14/0.96	38.9/2.54/30.13/0.96

Figure 4. *Cont*.

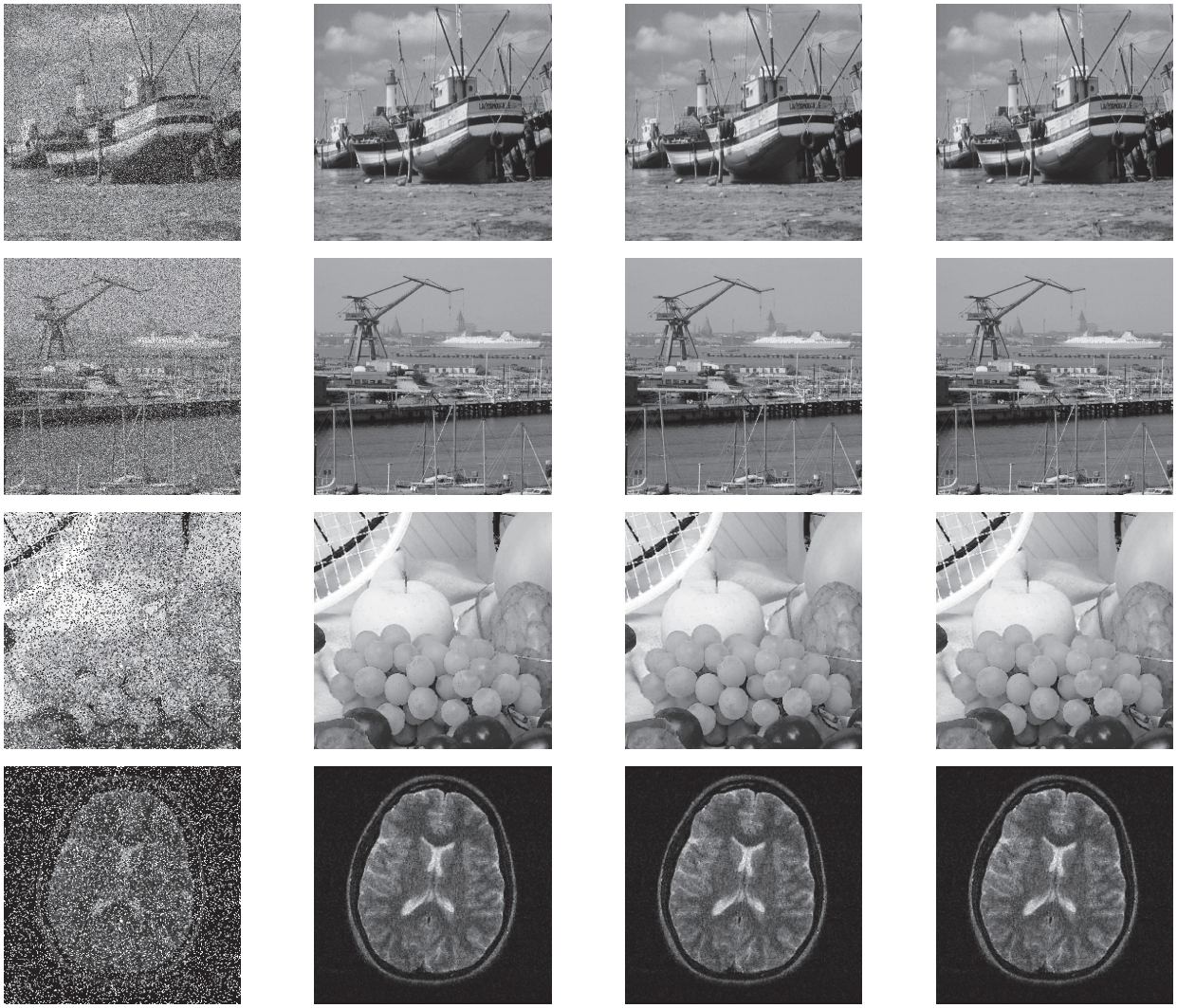


Figure 4. The noise images for lighthouse, peppers, boat, Kiel, fruits, and brain with 30% salt and pepper noise (first column) and the images recovered by Algorithms ITTCG (second column), HTTCGP (third column), and ZYL (forth column).

Figure 5. *Cont*.

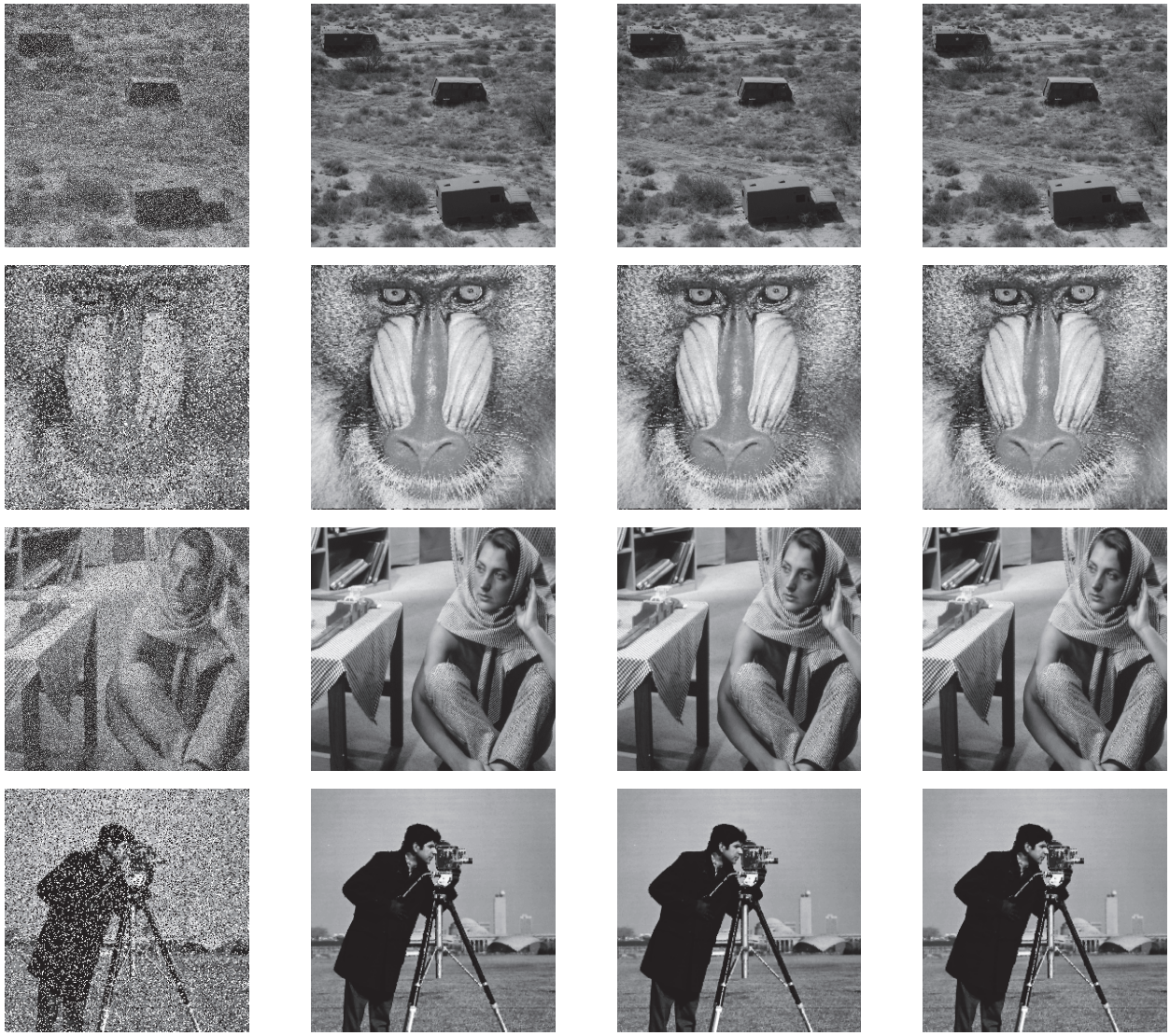


Figure 5. The noise images for clown, couple, trucks, baboon, Barbara, and cameraman with 30% salt and pepper noise (first column) and the images recovered by Algorithms ITTCG (second column), HTTCGP (third column), and ZYL (forth column).

6. Conclusions

In this paper, we proposed a projection-based improved three-term conjugate gradient algorithm for solving constrained nonlinear monotone equations. Its search direction automatically satisfies the sufficient descent and trust-region properties. The global convergence of the proposed algorithm is established under the assumption that the mapping is continuous and monotonic. A notable theoretical advantage of the proposed algorithm is that it does not require Lipschitz continuity of the mapping, unlike traditional algorithms for similar problems. Numerical results on benchmark problems demonstrated the effectiveness and competitiveness of the proposed algorithm. Furthermore, the proposed algorithm could successfully recover noise images.

Author Contributions: Conceptualization, D.L. and S.W.; Formal analysis, Y.L.; Funding acquisition, Y.L. and S.W.; Methodology, D.L.; Resources, Y.L.; Software, D.L.; Validation, D.L., Y.L. and S.W.; Writing—original draft, D.L.; Writing—review and editing, Y.L. and S.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation in China (grant number 11661009), the Natural Science Foundation in Guangxi Province, PR China (grant number

2024GXNSFAA010478; 2020GXNSFAA159069), the Special projects in key areas of ordinary universities in Guangdong Province (grant number 2023ZDZX4069), and the Research Team Project of Guangzhou Huashang University (grant number 2021HSKT01).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

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

References

1. Garcke, H.; Hüttl, P.; Knopf, P. Shape and topology optimization involving the eigenvalues of an elastic structure: A multi-phase-field approach. *Adv. Nonlinear Anal.* **2021**, *11*, 159–197. [CrossRef]
2. Garcke, H.; Knopf, P.; Yayla, S. Long-time dynamics of the Cahn–Hilliard equation with kinetic rate dependent dynamic boundary conditions. *Nonlinear Anal.* **2022**, *215*, 112619. [CrossRef]
3. Liu, J.; Du, X. A gradient projection method for the sparse signal reconstruction in compressive sensing. *Appl. Anal.* **2018**, *97*, 2122–2131. [CrossRef]
4. Xiao, Y.; Zhu, H. A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing. *J. Math. Anal. Appl.* **2013**, *405*, 310–319. [CrossRef]
5. Dirkse, S.; Ferris, M. MCPLIB: A collection of nonlinear mixed complementarity problems. *Optim. Methods Softw.* **1995**, *5*, 319–345. [CrossRef]
6. Wood, A.; Wollenberg, B. *Power Generation, Operation, and Control*; Wiley: New York, NY, USA, 1996.
7. Li, D.; Wang, S.; Li, Y.; Wu, J. A convergence analysis of hybrid gradient projection algorithm for constrained nonlinear equations with applications in compressed sensing. *Numer. Algorithms* **2024**, *95*, 1325–1345. [CrossRef]
8. Li, D.; Wu, J.; Li, Y.; Wang, S. A modified spectral gradient projection-based algorithm for large-scale constrained nonlinear equations with applications in compressive sensing. *J. Comput. Appl. Math.* **2023**, *424*, 115006. [CrossRef]
9. Liu, J.; Duan, Y. Two spectral gradient projection methods for constrained equations and their linear convergence rate. *J. Inequal. Appl.* **2015**, *2015*, 8. [CrossRef]
10. Sulaiman, I.M.; Awwal, A.M.; Malik, M. A derivative-free mzprp projection method for convex constrained nonlinear equations and its application in compressive sensing. *Mathematics* **2022**, *10*, 2884. [CrossRef]
11. Li, D.; Wang, S.; Li, Y.; Wu, J. A projection-based hybrid PRP-DY type conjugate gradient algorithm for constrained nonlinear equations with applications. *Appl. Numer. Math.* **2024**, *195*, 105–125. [CrossRef]
12. Yin, J.; Jian, J.; Jiang, X. A generalized hybrid CGPM-based algorithm for solving large-scale convex constrained equations with applications to image restoration. *J. Comput. Appl. Math.* **2021**, *391*, 113423. [CrossRef]
13. Liu, P.; Jian, J.; Jiang, X. A new conjugate gradient projection method for convex constrained nonlinear equations. *Complexity* **2020**, *2020*, 8323865. [CrossRef]
14. Yuan, G.; Li, T.; Hu, W. A conjugate gradient algorithm for large-scale nonlinear equations and image restoration problems. *Appl. Numer. Math.* **2020**, *147*, 129–141. [CrossRef]
15. Ali E.; Mahdi S. Adaptive hybrid mixed two-point step size gradient algorithm for solving non-linear systems. *Mathematics* **2023**, *11*, 2102. [CrossRef]
16. Kumam, P.; Abubakar, A.B.; Malik, M. A hybrid HS-LS conjugate gradient algorithm for unconstrained optimization with applications in motion control and image recovery. *J. Comput. Appl. Math.* **2023**, *433*, 115304. [CrossRef]
17. Ullah, N.; Shah, A.; Sabi'u, J. A one-parameter memoryless DFP algorithm for solving system of monotone nonlinear equations with application in image processing. *Mathematics* **2023**, *11*, 1221. [CrossRef]
18. Yin, J.; Jian, J.; Jiang, X. A hybrid three-term conjugate gradient projection method for constrained nonlinear monotone equations with applications. *Numer. Algorithms* **2021**, *88*, 389–418. [CrossRef]
19. Gao, P.; He, C. An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints. *Calcolo* **2018**, *55*, 53. [CrossRef]
20. Yuan, G.; Zhang, M. A three-terms Polak–Ribière–Polyak conjugate gradient algorithm for large-scale nonlinear equations. *J. Comput. Appl. Math.* **2015**, *286*, 186–195. [CrossRef]
21. Jiang, X.; Liao, W.; Yin, J. A new family of hybrid three-term conjugate gradient methods with applications in image restoration. *Numer. Algorithms* **2022**, *91*, 161–191. [CrossRef]
22. Liu, Y.; Zhu, Z.; Zhang, B. Two sufficient descent three-term conjugate gradient methods for unconstrained optimization problems with applications in compressive sensing. *J. Appl. Math. Comput.* **2022**, 1–30. [CrossRef]
23. Kim, H.; Wang, C.; Byun, H. Variable three-term conjugate gradient method for training artificial neural networks. *Neural Networks* **2023**, *159*, 125–136. [CrossRef]
24. Li, M. A modified Hestense-Stiefel conjugate gradient method close to the memoryless BFGS quasi-Newton method. *Optim. Methods Softw.* **2018**, *33*, 336–353. [CrossRef]
25. Li, M. A three-term polak-ribière-polyak conjugate gradient method close to the memoryless BFGS quasi-Newton mthod. *J. Ind. Manag. Optim.* **2017**, *13*, 1–16.

26. Li, M. A family of three-term nonlinear conjugate gradient methods close to the memoryless BFGS method. *Optim. Lett.* **2018**, *12*, 1911–1927. [CrossRef]
27. Ding, Y.; Xiao, Y., Li, J. A class of conjugate gradient methods for convex constrained monotone equations. *Optimization* **2017**, *66*, 2309–2328. [CrossRef]
28. Zheng, L.; Yang, L.; Liang, Y. A conjugate gradient projection method for solving equations with convex constraints. *J. Comput. Appl. Math.* **2020**, *375*, 112781. [CrossRef]
29. Dolan, E.; Moré, J.; Benchmarking optimization software with performance profiles. *Math. Program.* **2002**, *91*, 201–213. [CrossRef]
30. Chan, R.; Ho, C.; Nikolova, M. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. *IEEE Trans. Image Process.* **2005**, *14*, 1479–1485. [CrossRef]
31. Cai, J.; Chan, R.; Fiore, D. Minimization of a detail-preserving regularization functional for impulse noise removal. *J. Math. Imaging Vis.* **2007**, *29*, 79–91. [CrossRef]

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

mathematics

MDPI

Article

Identification of Time-Wise Thermal Diffusivity, Advection Velocity on the Free-Boundary Inverse Coefficient Problem

M. S. Hussein [1], Taysir E. Dyhoum [2,3,*], S. O. Hussein [4,*] and Mohammed Qassim [5]

[1] Department of Mathematics, College of Science, University of Baghdad, Baghdad 10071, Iraq; mmmsh@sc.uobaghdad.edu.iq
[2] Department of Computing and Mathematics, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M15 6BX, UK
[3] Department of Mathematics, Misurata University, Misurata P.O. Box 2478, Libya
[4] Department of Mathematics, College of Science, University of Sulaymaniyah, Sulaymaniyah 46001, Iraq
[5] Department of Energy, College of Engineering Al-Musayab, University of Babylon, Babylon 51002, Iraq; mq63582@gmail.com
* Correspondence: t.dyhoum@mmu.ac.uk (T.E.D.); shilan.husen@univsul.edu.iq (S.O.H.)

Abstract: This paper is concerned with finding solutions to free-boundary inverse coefficient problems. Mathematically, we handle a one-dimensional non-homogeneous heat equation subject to initial and boundary conditions as well as non-localized integral observations of zeroth and first-order heat momentum. The direct problem is solved for the temperature distribution and the non-localized integral measurements using the Crank–Nicolson finite difference method. The inverse problem is solved by simultaneously finding the temperature distribution, the time-dependent free-boundary function indicating the location of the moving interface, and the time-wise thermal diffusivity or advection velocities. We reformulate the inverse problem as a non-linear optimization problem and use the *lsqnonlin* non-linear least-square solver from the MATLAB optimization toolbox. Through examples and discussions, we determine the optimal values of the regulation parameters to ensure accurate, convergent, and stable reconstructions. The direct problem is well-posed, and the Crank–Nicolson method provides accurate solutions with relative errors below 0.006% when the discretization elements are $M = N = 80$. The accuracy of the forward solutions helps to obtain sensible solutions for the inverse problem. Although the inverse problem is ill-posed, we determine the optimal regularization parameter values to obtain satisfactory solutions. We also investigate the existence of inverse solutions to the considered problems and verify their uniqueness based on established definitions and theorems.

Keywords: parabolic heat equation; finite-difference method (FDM); Crank–Nicolson method; inverse coefficient identification problem; optimization tool; MATLAB; free-boundary problem

MSC: 65K05; 65K10; 65R30; 65R32; 65Y15; 65N12

1. Introduction

Partial differential equations (PDEs) subject to various non-local initial and boundary conditions are common expressions of mathematical models that arise when solving real-world problems. These real-world applications emerge in several scientific and engineering disciplines and fields, including geology, hydrodynamics, biological fluid dynamics, vibration materials, heat transfer, control theory, and thermoelastic problems [1–7]. Recent work on flow, heat, and thermal conductivity considers essential physical aspects, including Thompson and Troian slip effects on ternary hybrid nanofluid flow across a porous plate with a chemical reaction [8].

Due to the difficulty of obtaining analytical solutions, researchers employ various mathematical, statistical, and computer vision techniques to generate numerical approxi-

mated values to determine PDE systems' direct and inverse solutions. The sought solutions are represented by various physical quantities and medium properties, such as potential and damping parameters [9], the force source function [4], constant voltage and values of contact impedance [10], and reaction coefficients [5]. Such quantities appear as unknown time- and space-dependent coefficients or functions in the model, turning the problem into inverse coefficient problems (ICPs). Many empirical and theoretical studies focused on adapting and applying numerical techniques to solve ICPs. These include implicit finite difference methods [11,12], lattice-free finite difference methods [13], Fourier regularization to solve one-dimensional non-local coefficient heat problems [14], the collocation method [15], and iterative boundary element methods [16,17].

In this work, we consider solving ICPs with free-boundary (non-local) conditions. These conditions can mathematically represent phase-changing processes such as the freezing of water or the ground, solidifying of metals, melting of ice, forming of crystals, evaporation of chemicals, and so on, in which the heat associated with the phase change is either generated or absorbed [18–20]. Finding the solutions means determining the domain's temperature distribution, the location of the movable boundaries and dynamic interface, and the unknown functions of time-wise thermal diffusivity or time- and space-dependent diffusion. This process poses a significant computational challenge, requiring numerical strategies to accurately estimate free boundaries and complex energetic interfacing. Because these inverse problems are ill-posed, we ensure that the solutions exist and are unique (locally) by aligning the considered cases with previous theoretical studies [21–23]. To investigate the inverse problem with non-localized conditions, we structure the model as a non-homogeneous one-dimensional heat equation subject to a set of initial and boundary conditions plus over-determined conditions of the zeroth and first-order heat momentum.

In this study, we apply the Crank–Nicolson (CN) finite difference method to solve the free-boundary (non-local) problem. We then utilize Tikhonov regularization techniques to stabilize the inverse problem and sort out the non-linearity issue by using the MATLAB R2023a optimization toolbox *lsqnonlin*. We find the time-dependent free-boundary function, which indicates the location of the moving interface, the temperature distribution at the boundaries, and the time-wise thermal diffusivity or advection velocities simultaneously. There are many alternative techniques to solve similar problems. For example, Martín-Vaquero and Sajavičius [24] used the two-level finite difference method (FDM) to solve one-dimensional parabolic equations subject to initial conditions represented in non-local discrete integrals and other homogeneous boundary conditions. A minimal surface equation, a two-dimensional nonlinear elliptic equation subject to additional boundary non-local integral conditions, has been solved iteratively using a system of difference equation approximations [2]. A novel iteration scheme based on the domain decomposition method is applied to determine the time-dependent coefficients in heat and Volterra integral equations, as presented in [7,25]. Recently, Huntul and Lesnic [26,27] used multilevel finite difference approximations to retrieve unknown time-dependent intensity and convection coefficients in free-boundary two-dimensional heat problems. We have previously used this numerical approach to identify the temperature distribution and other time-dependent parameters, such as the intensity of reaction, perfusion, and radioactive coefficients, based on over-specified conditions of Stefan-type, zeroth-order heat momentum [28,29].

This paper is organized as follows. In Section 2, the mathematical formulations of the problem are set up, including ensuring that the existence and uniqueness requirements are satisfied. The use of the CN technique to identify the problem's forward solutions is demonstrated in Section 3. We calculate the inverse solutions in Section 4; this section covers the CN solver, Tikhonov's regularization method, and the *lsqnonlin* MATLAB solver. A couple of numerical examples (simulations) are discussed and investigated in Section 5. Section 6 summarizes the findings and suggests further research.

2. Mathematical Formulation

Consider the domain $D_T = \{(x,\tau) : 0 < x < s(\tau),\ 0 < \tau < T\}$ for the following mathematical problem. The primary goal of this research is to find the free boundary $s(\tau)$, and the time-wise thermal diffusivity $a(\tau)$ or advection velocity $b(x,\tau)$. Thermal diffusivity is the heat transfer property of a medium; the advection velocity refers to the flow of molecules in the examined medium.

$$\frac{\partial}{\partial\tau}u(x,\tau) = a(\tau)\frac{\partial^2}{\partial x^2}u(x,\tau) + b(x,\tau)\frac{\partial}{\partial x}u(x,\tau) + c(x,\tau)u(x,\tau) + f(x,\tau), \quad \text{in } D_T, \quad (1)$$

is subject to the initial and non-homogeneous Dirichlet boundary conditions

$$u(x,0) = \varphi(x), \quad 0 \le x \le s(\tau), \quad (2)$$

$$u(0,\tau) = \gamma_1(\tau), \quad u(s(\tau),\tau) = \gamma_2(\tau), \quad 0 \le \tau \le T, \quad (3)$$

where $\{c(x,\tau), f(x,\tau), \varphi(x), \gamma_1(\tau), \gamma_2(\tau)\}$ are given functions and $\{u(x,\tau), s(\tau), a(\tau), b(x,\tau)\}$ are unknown functions that will be numerically approximated.

If the functions $\{s(\tau), a(\tau), b(x,\tau)\}$ are given, Equations (1)–(3) form a direct well-posed problem. If some or all of the function terms $(s(\tau), a(\tau))$ or $(s(\tau), b(x,\tau))$ of Equations (1)–(3) are not defined, the above set of equations is insufficient to determine them uniquely. Such a situation leads to handling a solution of the inverse ill-posed problem [21]. In this case, we must impose additional data to retain uniqueness:

$$\int_0^{s(\tau)} x^\ell u(x,\tau)dx = \gamma_{3+\ell}(\tau), \qquad \tau \in [0,T], \quad \ell \in \{0,1\}. \quad (4)$$

Equation (4) represents the zeroth ($\ell = 0$) and first-order ($\ell = 1$) heat momentum. To solve the inverse ill-posed problem in Equations (1)–(4), we first convert the free domain function $s(\tau)$ to a fixed domain by setting $\eta = \frac{x}{s(\tau)}$ and $\tau = \tau$. This implies that $u(x,\tau) = u(\eta s(\tau),\tau) = v(\eta,\tau)$ and $Q_T = \{(\eta,\tau) : 0 < \eta < 1, \quad 0 < \tau < T\}$. Therefore, using the previous transformation, Equations (1)–(4) can be rewritten in compact notation as

$$v_\tau = \frac{a(\tau)}{s^2(\tau)}v_{\eta\eta} + \frac{b(\eta s(\tau),\tau) + \eta s'(\tau)}{s(\tau)}v_\eta + c(\eta s(\tau),\tau)v + f(\eta s(\tau),\tau),\ (\eta,\tau) \in Q_T, \quad (5)$$

$$v(\eta,0) = \varphi(s(0)\eta), \qquad \eta \in [0,1], \quad (6)$$

$$v(0,\tau) = \gamma_1(\tau), \quad v(1,\tau) = \gamma_2(\tau), \quad \tau \in [0,T], \quad (7)$$

$$s^{\ell+1}(\tau)\int_0^1 v(\eta,\tau)d\eta = \gamma_{3+\ell}(\tau), \qquad \tau \in [0,T], \quad \ell \in \{0,1\}. \quad (8)$$

Based on well-established theories on the uniqueness of this inverse problem [21–23], we assume the problem in Equations (5)–(8) requires the existence and uniqueness criteria as follows.

Definition 1. *The solution of the inverse problem in Equations* (5)–(8) *can be:*
Case 1. *When $b(\eta s(\tau),\tau)$ is known, it is the triplet class $(a(\tau), s(\tau), v(\eta,\tau)) \in C[0,T] \times C^1[0,T] \times C^{2,1}(\overline{Q}_T)$.*
Case 2. *If $a(\tau)$ is given and $b(\eta s(\tau),\tau)$ only depends on time $(b(\eta s(\tau),\tau) = b(\tau))$, it is the triplet class $(s(\tau), b(\tau), v(\eta,\tau)) \in C^1[0,T] \times C[0,T] \times C^{2,1}(\overline{Q}_T)$, where $a(\tau) > 0$ and $s(\tau) > 0$ for $\tau \in [0,T]$.*

Theorem 1. *Consider the case where $b(\eta s(\tau),\tau)$ is known (***Case 1***) and assume the input data for the problem in Equations* (5)–(8) *satisfy the following three conditions:*

1. $\gamma_i \in C^1[0,T], \gamma_i(\tau) > 0$, *for* $i = \overline{1,4}, \gamma_4'(\tau) > 0, s(\tau)u_x(0,\tau) - \gamma_2(\tau) + \gamma_1(\tau) > 0$, $b(0,\tau)\gamma_1(\tau) + \gamma_3'(\tau) \le 0$, *for* $\tau \in [0,T]$.

2. $\varphi \in C^2[0, s(0)]$, $\varphi(x) > 0$, $\varphi'(x) > 0$, *for* $x \in [0, s(0)]$, *where* $s_0 = s(0) > 0$ *by the solution of* $\int_0^{s_0} h(0)\varphi(\eta s(0))d\eta = \gamma_3(0)$.
3. $b, c, f \in C^{1,0}([0, H_1] \times [0, T])$, $f(x,\tau) \geq 0, b(x,\tau) \geq 0, c(x,\tau) - b_x(x,\tau) \geq 0$, *for* $(x,\tau) \in [0, H_1] \times [0, T]$ *where,* $H_1 = \max_{\tau\in[0,T]} \gamma_3(\tau) \left(\min \left(\min_{x\in[0,s_0]} \varphi(x), \min_{\tau\in[0,T]} \gamma_1(\tau), \min_{\tau\in[0,T]} \gamma_2(\tau) \right) \right)^{-1}$.
4. $\varphi(0) = \gamma_1(0), \varphi(s(0)) = \gamma_2(0)$ *and* $s^2(0) \int_0^1 \eta\varphi(s(0)d\eta = \gamma_4(0)$.

Then, there exists a unique solution for the inverse problem in Equations (5)–(8) *where* $\tau_0 \in [0, T]$ *is defined as input data for this problem.*

To solve the inverse problem in Equations (5)–(8) in **Case 1**, with given $b(\eta s(\tau), \tau)$, we start by finding the initial values of the unknown quantities $a(0)$ and $s'(0)$. This step is essential to find stable numerical reconstructions later. Then, we derive the derivative of the integral equations of the over-determination condition in Equation (4) concerning time:

$$\gamma'_{3+\ell}(\tau) = s^\ell(\tau)\gamma_2(\tau)s'(\tau) + \int_0^{s(\tau)} x^\ell u_\tau(x,\tau)dx. \quad \ell \in \{0, 1\}. \tag{9}$$

To obtain the second term on the right-hand side of Equation (9), we integrate the one-dimensional parabolic governing Equation (1) over the interval $[0, s(\tau)]$ with respect to space x. To get the second term on the right-hand side of Equation (9) when $\ell = 1$, we multiply Equation (1) by x and integrate over the same period

$$\int_0^{s(\tau)} u_\tau dx = a(\tau)[u_x(s(\tau),\tau) - u_x(0,\tau)] + \int_0^{s(\tau)} \Big[b(x,\tau)u_x + c(x,\tau)u + f(x,\tau) \Big] dx, \tag{10}$$

multiplying by x

$$\begin{aligned} \int_0^{s(\tau)} xu_\tau dx &= a(\tau)[s(\tau)u_x(s(\tau),\tau) - \gamma_2(\tau) + \gamma_1(\tau)] \\ &+ \int_0^{s(\tau)} x\Big[b(x,\tau)u_x + c(x,\tau)u + f(x,\tau) \Big] dx. \end{aligned} \tag{11}$$

Finally, by substituting Equation (10) into (9) ($\ell = 0$), Equation (11) into (9) ($\ell = 1$), and conducting some re-arrangements taking into account that $s'(\tau)$ and $a(\tau)$ are unknown functions, we obtain

$$\gamma_2(\tau)s'(\tau) + [u_x(s(\tau),\tau) - u_x(0,\tau)]a(\tau) = \gamma'_3(\tau) - \delta_1(\tau) := L_3(\tau),$$

$$s(\tau)\gamma_2(\tau)s'(\tau) + [s(\tau)u_x(s(\tau),\tau) - \gamma_2(\tau) + \gamma_1(\tau)]a(\tau) = \gamma'_4(\tau) - \delta_2(\tau) := L_4(\tau).$$

The above equations can be written in matrix form as

$$\begin{bmatrix} \gamma_2(\tau) & u_x(s(\tau),\tau) - u_x(0,\tau) \\ s(\tau)\gamma_2(\tau) & s(\tau)u_x(s(\tau),\tau) - \gamma_2(\tau) + \gamma_1(\tau) \end{bmatrix} \begin{bmatrix} s'(\tau) \\ a(\tau) \end{bmatrix} = \begin{bmatrix} L_3(\tau) \\ L_4(\tau) \end{bmatrix} \tag{12}$$

where

$$\delta_1(\tau) = \int_0^{s(\tau)} \Big(b(x,\tau)u_x + c(x,\tau)u + f(x,\tau) \Big) dx,$$

$$\delta_2(\tau) = \int_0^{s(\tau)} x\Big(b(x,\tau)u_x + c(x,\tau)u + f(x,\tau) \Big) dx.$$

To obtain a unique solution of the above 2×2 system, the determinant must not vanish in $[0, T]$,

$$\begin{aligned}\Delta_1(\tau) &= \begin{vmatrix} \gamma_2(\tau) & u_x(s(\tau),\tau) - u_x(0,\tau) \\ s(\tau)\gamma_2(\tau) & s(\tau)u_x(s(\tau),\tau) - \gamma_2(\tau) + \gamma_1(\tau) \end{vmatrix} \\ &= \gamma_2(\tau)[s(\tau)u_x(s(\tau),\tau) - \gamma_2 + \gamma_1] - s(\tau)\gamma_2[u_x(s(\tau),\tau) - u_x(0,\tau)] \\ &= s(\tau)\gamma_2(\tau)u_x(s(\tau),\tau) - \gamma_2^2(\tau) + \gamma_2(\tau)\gamma_1(\tau) - s(\tau)\gamma_2(\tau)u_x(s(\tau),\tau) + s(\tau)\gamma_2(\tau)u_x(0,\tau) \\ &= -\gamma_2^2(\tau) + \gamma_2(\tau)\gamma_1(\tau) + s(\tau)\gamma_2(\tau)\gamma_2(\tau)u_x(0,\tau).\end{aligned}$$

Therefore,

$$s'(\tau) = \frac{L_4(\tau)u_x(0,\tau) + s(\tau)L_3(\tau)u_x(s(\tau),\tau) - L_4(\tau)u_x(s(\tau),\tau) + L_3(\tau)\gamma_1(\tau) - L_3(\tau)\gamma_2(\tau)}{\Delta_1(\tau)},$$

$$a(\tau) = \frac{\gamma_2(\tau)L_4(\tau) - \gamma_2(\tau)s(\tau)L_3(\tau)}{\Delta_1(\tau)},$$

and making $\tau = 0$ in the last two expressions results in

$$s'(0) = \frac{L_4(0)u_x(0,0) + h(0)L_3(0)u_x(h_0,0) - L_4(0)u_x(h_0,0) + L_3(0)\gamma_1(0) - L_3(0)\gamma_2(0)}{\Delta_1(0)}, \tag{13}$$

$$a(0) = \frac{\gamma_2(0)L_4(0) - \gamma_2(0)h_0L_3(0)}{\Delta_1(0)}. \tag{14}$$

Theorem 2. *Consider the case where $a(\tau) = 1$, the function $b(\eta s(\tau),\tau) = b(\tau)$ is unknown (***Case 2***), and the following conditions are satisfied:*

1. *$\gamma_i(\tau) \in C^1[0,1]$, $\gamma_i(\tau) > 0$, $i = \overline{1,4}$ and $f \in C([0,\infty)\times[0,T])$, $f(x,\tau) \geq 0$ for $x \in [0,+\infty)$, $\tau \in [0,T]$. Also, $\varphi \in C^2[0,s(0)]$, $\varphi'(x) > 0$ for $x \in [0,h(0)]$.*
2. *The compatibility conditions are*

$$\begin{aligned}&\varphi(0) = \gamma_1(0), \qquad \varphi(s(0)) = \gamma_2(0), \\ &\gamma_1'(0) = \frac{1}{s^2(0)}\varphi''(0) + \frac{b(0)}{s(0)}\varphi'(0) + c(0,0)\varphi(0) + f(0,0), \\ &\gamma_2'(0) = \frac{1}{s^2(0)}\varphi''(s(0)) + \left[\frac{b(0)}{s(0)} + \frac{s'(0)}{h(0)}\right]\varphi'(h(0)) + c(s(0),0)\varphi(s(0)) + f(s(0),0).\end{aligned}$$

Then, we can determine $T_1 \in (0,T]$ such that there exists a local solution to the inverse problem in Equations (1)–(4) *or* (5)–(8) *for $(y,\tau) \in Q_{T_1}$.*

Theorem 3. *Assume the following conditions hold for the previous case:*

1. *$f, c \in C^{1,0}([0,+\infty)\times[0,T])$,*
2. *$\varphi(x) \geq \varphi_0$ and $f(x,\tau) \geq 0$, for $x \in ([0,+\infty)\times[0,T])$,*
3. *$\gamma_i(\tau) > 0, i = \overline{1,4}$ for $\tau \in [0,T]$ and $\varphi'(x) > 0$ for $x \in [0,h_0]$.*

Then, the problem in Equations (5)–(8) *cannot have more than one solution in the domain Q_{T_1}.*

It is necessary to calculate the values of $s'(0)$ and $b(0)$ in **Case 2** to find the inverse solution of Equations (5)–(8). We apply the same approach used for **Case 1**. We find the equations corresponding to Equation (9) when $\ell \in \{0,1\}$ and substitute $a(\tau) = 1, b(x,\tau) = b(\tau)$ into the equations corresponding to Equations (10) and (11), respectively. This yields

$$\int_0^{s(\tau)} u_\tau dx = \int_0^{s(\tau)} \Big[u_{xx} + c(x,\tau)u + f(x,\tau)\Big]dx + b(\tau)\int_0^{s(\tau)} u_x dx,$$
$$\int_0^{s(\tau)} xu_\tau dx = \int_0^{s(\tau)} x\Big[u_{xx} + c(x,\tau)u + f(x,\tau)\Big]dx + b(\tau)\int_0^{s(\tau)} xu_x dx.$$

Then, we apply integration by parts to calculate the exact values of the latest τ terms in the previous equations, which leads to the following equations. In addition, we replace them with Equation (9) ($\ell \in \{0,1\}$):

$$\gamma_2(\tau)s'(\tau) + [\gamma_2(\tau) - \gamma_1(\tau)]b(\tau) = \gamma_3'(\tau) - \int_0^{s(\tau)} [u_{xx} + c(x,\tau)u + f(x,\tau)]dx = L_1(\tau),$$
$$s(\tau)\gamma_2(\tau)s'(\tau) + [s(\tau)\gamma_2(\tau) - \gamma_3(\tau)]b(\tau) = \gamma_4'(\tau) - \int_0^{s(\tau)} x[u_{xx} + c(x,\tau)u + f(x,\tau)]dx = L_2(\tau).$$

To join the previous differential equations, we express them in the following matrix form:

$$\begin{bmatrix} \gamma_2(\tau) & \gamma_2(\tau) - \gamma_1(\tau) \\ \gamma_2(\tau)s(\tau) & s(\tau)\gamma_2(\tau) - \gamma_3(\tau) \end{bmatrix} \begin{bmatrix} s'(\tau) \\ b(\tau) \end{bmatrix} = \begin{bmatrix} L_1(\tau) \\ L_2(\tau) \end{bmatrix},$$

where

$$s'(\tau) = \frac{(\gamma_1(\tau) - \gamma_2(\tau))L_2(\tau) + L_1(\tau)(s(\tau)\gamma_2(\tau) - \gamma_3(\tau))}{\gamma_2(s(\tau)\gamma_1(\tau) - \gamma_3(\tau))} \quad \tau \in [0,T],$$
$$b(\tau) = \frac{-s(\tau)L_1(\tau) + L_2(\tau)}{s(\tau)\gamma_1(\tau) - \gamma_3(\tau)} \quad \tau \in [0,T],$$

and setting $\tau = 0$ results in

$$s'(0) = \frac{(\gamma_1(0) - \gamma_2(0))L_2(0) + L_1(0)(s(0)\gamma_2(0) - \gamma_3(0))}{\gamma_2(0)(s(0)\gamma_1(0) - \gamma_3(0))}, \tag{15}$$
$$b(0) = \frac{-s(0)L_1(0) + L_2(0)}{s(0)\gamma_1(0) - \gamma_3(0)}. \tag{16}$$

Equations (15) and (16) are both required for compatibility with Condition 2 of Theorem 2 to prove the existence of the solutions to the problem in Equations (5)–(8).

3. Applied CN Method to Obtain the Direct Solutions to the Problem

In this section, we take into account the initial boundary value problem in Equations (5)–(7), where $\{a(\tau), b(\eta s(\tau),\tau), c(\eta s(\tau),\tau), \varphi(\eta s(0)), \gamma_i(\tau)\}$ with $i = 1,2$ are known functions that meet the existence and uniqueness conditions in Theorems 1–3. We seek to compute the direct solution $v(\eta,\tau)$. Furthermore, we can figure out the numerical values of Equation (8) ($\ell \in \{0,1\}$) by employing the CN finite difference method. This method is unconditionally stable, and the solutions have second-order accuracy in the time and spatial dimensions.

To discretize the domain $Q_T = (0,1)\times(0,T)$, we divide it into small M and N intervals of equally spaced length $\Delta\eta$ and $\Delta\tau$, denoting the uniform space and time increments by $\Delta\eta = \frac{1}{M}$ and $\Delta\tau = \frac{T}{N}$, respectively. We refer to the solution at the node point (i,j) as $v_{i,j} = v(\eta_i,\tau_j)$, $a(\tau_j) = a_j$, $b(\eta_i,\tau_j) = b_{i,j}$, $c(\eta_i,\tau_j) = c_{i,j}$, and $f(\eta_i,\tau_j) = f_{i,j}$, where $\eta_i = i\Delta\eta$, $\tau_j = j\Delta\tau$ for $i = \overline{0,M}$, $j = \overline{0,N}$ [4,30].

We rename the right-hand side of Equation (5) as $\Theta(\tau,\eta,v,v_\eta,v_{\eta\eta})$, i.e.,

$$\Theta(\tau,\eta,v,v_\eta,v_\eta) = \frac{a(\tau)}{s^2(\tau)}v_{\eta\eta} + \left(\frac{b(\eta s(\tau),\tau) + s'(\tau)\eta}{s(\tau)}\right)v_\eta + c(\eta s(\tau),\tau)v + f(\eta s(\tau),\tau),\ (\eta,\tau) \in Q_T.$$

By discretizing the previous equation using the FDM, we obtain

$$\Theta_{i,j} = \left(\frac{a(\tau_j)}{s^2(\tau_j)}\right)\frac{v_{i+1,j} - 2v_{i,j} + v_{i-1,j}}{\Delta\eta^2} + \left(\frac{b(\eta_i s(\tau_j)) + s'(\tau_j)y_i}{s(\tau_j)}\right)\frac{v_{i+1,j} - v_{i-1,j}}{2\Delta\eta} + c_{i,j}v_{i,j} + f_{i,j}, \quad i = \overline{0,M}, \quad j = \overline{0,N}.$$

Benefiting from the fact that CN techniques are unconditionally stable and provide convergence of the second order in time and space for such problems [12,31,32], we apply them to Equations (5)–(7) to obtain

$$\frac{v_{i,j+1} - v_{i,j}}{\Delta\tau} = \frac{1}{2}(\Theta_{i,j+1} + \Theta_{i,j}), \tag{17}$$

$$v(\eta_i, 0) = \varphi(\eta_i s(0)), \quad i = \overline{0,M}, \tag{18}$$

$$v(0,\tau_j) = \gamma_1(\tau_j), \quad v(1,\tau_j) = \gamma_2(\tau_j), \quad j = \overline{0,N}. \tag{19}$$

By substituting $\Theta_{i,j}$ and $\Theta_{i,j+1}$ into Equation (17), we obtain the system of equations

$$\begin{aligned} &- A_{i,j+1}v_{i-1,j+1} + [1 - B_{i,j+1}]v_{i,j+1} - C_{i,j+1}v_{i+1,j+1} = \\ &A_{i,j}v_{i-1,j} + [1 + B_{i,j+1}]v_{i,j} + C_{i,j}v_{i+1,j} + \frac{\Delta\tau}{2}(f_{i,j} + f_{i,j+1}), \end{aligned} \tag{20}$$

with the matrices $A_{i,j} = \frac{\Delta\tau}{2\Delta\eta^2}\frac{a_j}{s_j^2} - \frac{\Delta\tau}{4\Delta\eta}\frac{b_{i,j}+s'_j\eta_i}{s_j}$, $B_{i,j} = \frac{\Delta\tau}{2}c_{i,j} - \frac{\Delta\tau}{\Delta\eta^2}\frac{a_j}{s_j^2}$, and $C_{i,j} = \frac{\Delta\tau}{2\Delta\eta^2}\frac{a_j}{s_j^2} + \frac{\Delta\tau}{4\Delta\eta}\frac{b_{i,j}+s'_j\eta_i}{s_j}$.

There are three values on the right-hand side of Equation (20): $v_{i-1,j}$, $v_{i,j}$, and $v_{i+1,j}$. Conversely, the values on the left-hand side remain unknown.

For $j = 0$ (i.e, the initial time) and $i = \overline{1, M-1}$, Equation (20) represents a linear system of $M-1$ equations with $M-1$ unknown variables, namely $v_{1,1}, v_{2,1}, \ldots, v_{M-1,1}$. The first time steps in terms of the initial values $v_{0,0}, v_{1,0}, \ldots, v_{n,0}$ and from the Dirichlet boundaries $v_{0,1}$ and $v_{M,1}$ have specific values $\gamma_1(\tau_0)$ and $\gamma_2(\tau_0)$, respectively. We perform a similar procedure for the following iteration time step (τ_j) with $j = \overline{1, N-1}$; that is, for each time step τ_j for $j = \overline{1, N-1}$.

We rewrite Equation (20) in metric form as a $(M-1) \times (M-1)$ system of algebraic linear equations (that can be solved by the Gaussian elimination method) as follows:

$$\mathbf{A}v^{n+1} = \mathbf{B}v^n + d, \tag{21}$$

where $v^{n+1} = (v_{1,j+1}, v_{2,j+1}, ..., v_{M-1,j+1})^t$, $v^n = (v_{1,j}, v_{2,j}, ..., v_{M-1,j})^t$, and $\mathbf{A}$ and $\mathbf{B}$ are $(M-1) \times (M-1)$ matrices as follows:

$$A = \begin{bmatrix} 1 - B_{1,j+1} & -C_{1,j+1} & 0 \ldots \cdots 0 & 0 & 0 \\ -A_{2,j+1} & 1 - B_{2,j+1} & -C_{2,j+1} & \ldots 0 & 0 \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & 0 \ldots - A_{M-2,j+1} & 1 - B_{M-2,j+1} & -C_{M-2,j+1} \\ 0 & 0 & 0 \ldots 0 & -A_{M-1,j+1} & 1 - B_{M-1,j+1} \end{bmatrix}$$

$$B = \begin{bmatrix} 1 + B_{1,j} & C_{1,j} & 0 \ldots 0 & 0 & 0 \\ A_{2,j} & 1 + B_{2,j} & C_{2,j} \ldots 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \\ 0 & 0 & 0 \ldots A_{M-2,j} & 1 + B_{M-2,j} & C_{M-2,j} \\ 0 & 0 & 0 \ldots 0 & A_{M-1,j} & 1 + B_{M-1,j} \end{bmatrix}$$

$$d = \begin{bmatrix} A_{1,j+1}v_{0,j+1} + A_{1,j}v_{0,j} + \frac{\Delta\tau}{2}(f_{1,j} + f_{1,j+1}) \\ \frac{\Delta\tau}{2}(f_{2,j} + f_{2,j+1}) \\ \vdots \\ \frac{\Delta\tau}{2}(f_{M-2,j} + f_{M-2,j+1}) \\ C_{M-1,j+1}v_{M,j+1} + C_{M-1,j}v_{M,j} + \frac{\Delta\tau}{2}(f_{M-1,j} + f_{M-1,j+1}) \end{bmatrix}.$$

The trapezium rule (numerical integration) is applied to discretize Equation (8) ($\ell \in \{0, 1\}$) into different equations:

$$\gamma_3(\tau_j) = \frac{s(\tau_j)}{2N}\left(v_{0,j} + v_{M,j} + 2\sum_{i=1}^{M-1} v_{i,j}\right), \quad j = \overline{1, N}, \tag{22}$$

$$\gamma_4(\tau_j) = \frac{s^2(\tau_j)}{2N}\left(\eta_0 v_{0,j} + \eta_M v_{M,j} + 2\sum_{i=1}^{M-1} \eta_i v_{i,j}\right), \quad j = \overline{1, N}. \tag{23}$$

4. Numerical Approximations of the Inverse Problems

In this section, we find the approximated solutions of different quantities of the inverse problem in Equations (1)–(4) in **Case 1** to obtain $\{(u(x,\tau), a(\tau), s(\tau))\}$ when $b(x,\tau)$ is explicitly is given. Then, we find the corresponding solution of the inverse problem in Equations (1)–(4) in **Case 2**, where $\{(u(x,\tau), b(\tau), s(\tau))\}$ require identification when $a(\tau)$ is given.

Handling these inverse problems means solving non-linear optimization problems that minimize the gap between measured data and computed solutions. The minimization of the objective function, subject to the straightforward physical lower-bound constraint $s > 0$, can be achieved by using the *lsqnonlin* non-linear least-square solver from the MATLAB optimization toolbox, which applies the trust region reflective algorithm (TRR) [33,34]. The *lsqnonlin* solver aims to determine the minimum sum of squares by starting from initial guesses. This toolbox routine does not require a supplement of the gradient of the objective function. It uses the TRR algorithm [33,35,36], so it effectively relies on the interior-reflective Newton method. Each iteration results in a large system of linear equations, which we solve by applying the preconditioned conjugate gradient method [37,38].

As we mentioned earlier, **Case 1** concerns finding the thermal diffusivity $a(\tau)$, the free-boundary condition $s(\tau) > 0$ of one-dimensional heat in Equation (1), and the temperature distribution $u(x,\tau)/v(\eta,\tau)$. Equations (14) and (13) are used to calculate $a(0)$ and $s'(0)$, respectively, when the initial time is $\tau = 0$. Given the ill-posed nature of the problem, Tikhonov regularization (ridge regression) can be applied to ensure the suitability and accuracy of the solution [39,40].

From the over-determination conditions in Equation (8) ($\ell \in \{0, 1\}$), we reconstruct Tikhonov's regularization as follows:

$$\begin{aligned} J(a,s) := &\left\| s(\tau)\int_0^1 v(\eta,\tau)d\eta - \gamma_3(\tau)\right\|^2 + \left\| s^2(\tau)\int_0^1 \eta v(\eta,\tau)d\eta - \gamma_4(\tau)\right\|^2 \\ &+ \beta_1\|s(\tau)\|^2 + \beta_2\|a(\tau)\|^2. \end{aligned} \tag{24}$$

The previous Tikhonov regularization functional reconstruction can be expanded and rewritten in the following form:

$$\begin{aligned} J(\underline{a},\underline{s}) = &\sum_{j=1}^{N}\left(s_j\int_0^1 v(\eta,\tau_j)d\eta - \gamma_3(\tau_j)\right)^2 + \sum_{j=1}^{N}\left(s_j^2\int_0^1 \eta v(\eta,\tau_j)d\eta - \gamma_4(\tau_j)\right)^2 \\ &+ \beta_1\sum_{j=1}^{N} s_j^2 + \beta_2\sum_{j=1}^{N} a_j^2. \end{aligned} \tag{25}$$

$J(\underline{a}, \underline{s})$, which is subject to the physical constraints $\underline{s} > \underline{0}$ (free boundary) and $\underline{a} > \underline{0}$ (thermal diffusivity), is minimized using the optimization package *lsqnonlin*.

We apply the same procedure for **Case 2**, where $a(\tau) = 1$, and we need to identify $b(x, \tau) = b(\tau)$ as well as the free boundary $s(\tau)$ and the temperature $u(x, \tau)/v(\eta, \tau)$. Again, we consider the first initial value for the time $\tau = 0$, which helps to calculate $b(0)$ in Equation (16) and $s'(0)$ in Equation (15). Moreover, we use the over-determination conditions in Equation (8) ($\ell \in \{0, 1\}$) to form the corresponding Tikhonov regularization:

$$G(s, b) := \left\| s(\tau) \int_0^1 v(\eta, \tau) d\eta - \gamma_3(\tau) \right\|^2 + \left\| s^2(\tau) \int_0^1 \eta v(\eta, \tau) d\eta - \gamma_4(\tau) \right\|^2 + \beta_1 \|s(\tau)\|^2 + \beta_2 \|b(\tau)\|^2, \tag{26}$$

which can be rewritten as

$$G(\underline{s}, \underline{b}) = \sum_{j=1}^{N} \left(s_j \int_0^1 v(\eta, \tau_j) d\eta - \gamma_3(\tau_j) \right)^2 + \sum_{j=1}^{N} \left(s_j^2 \int_0^1 \eta v(\eta, \tau_j) d\eta - \gamma_4(\tau_j) \right)^2 + \beta_1 \sum_{j=1}^{N} s_j^2 + \beta_2 \sum_{j=1}^{N} b_j^2. \tag{27}$$

Then, $G(\underline{s}, \underline{b})$ is minimized using the *lsqnonlin* solver. In both the examined cases, $\beta_i \geq 0$ and $i = 1, 2$ are the regularization parameters identified and regulated according to a specific selection procedure, and the norm is taken in the space $L^2[0, T]$.

To ensure the stability of the inverse solutions, we include random errors (noise) in the input data for Equation (8) ($\ell \in \{0, 1\}$) and monitor the effect of the change.

$$\gamma_3^{\epsilon_1}(\tau_j) = \gamma_3(\tau_j) + \epsilon_{1,j}; \qquad \gamma_4^{\epsilon_2}(\tau_j) = \gamma_4(\tau_j) + \epsilon_{2,j}, \qquad j = \overline{0, N}, \tag{28}$$

where $\underline{\epsilon}_1$ and $\underline{\epsilon}_2$ are arbitrary vectors engendered from a Gaussian normal distribution that has mean zero and standard deviations denoted as σ_1 and σ_2, respectively:

$$\sigma_1 = p \times \max_{\tau \in [0,T]} |\gamma_3(\tau)|, \quad \sigma_2 = p \times \max_{\tau \in [0,T]} |\gamma_4(\tau)|. \tag{29}$$

The quantity p refers to the percentage of added noise. The MATLAB bulletin function *normrnd* was used to generate the random variables $\underline{\epsilon}_1 = (\epsilon_{1,j})$ and $\underline{\epsilon}_2 = (\epsilon_{2,j})$ for $j = \overline{0, N}$ as follows:

$$\underline{\epsilon}_1 = normrnd(0, \sigma_1, N), \qquad \underline{\epsilon}_2 = normrnd(0, \sigma_2, N).$$

5. Discussions and Numerical Examples for Cases 1 and 2

In this section, we calculate, discuss, and interpret the numerical results of the time-dependent coefficients $a(\tau)$ and $b(\tau)$ along with $s(\tau) > 0$ and the temperature distribution $v(\eta, \tau)$. We compare the obtained direct solutions with the analytical ones. Because finding the exact solutions to such a problem is not always possible, we run simulations after applying a trim level of noise to the measurements of the direct solver. Then, we seek the best value for the regulation parameters to ensure the accuracy, convergence, and stability of the obtained inverse solutions (reconstructions). We also consider the root mean square error (RMSE), which is given as follows:

$$RMSE(a) = \left[\frac{T}{N} \sum_{j=1}^{N} (a^{numerical}(\tau_j) - a^{exact}(\tau_j))^2 \right]^{\frac{1}{2}}, \tag{30}$$

$$RMSE(b) = \left[\frac{T}{N}\sum_{j=1}^{N}(b^{numerical}(\tau_j) - b^{exact}(\tau_j))^2\right]^{\frac{1}{2}}, \tag{31}$$

$$RMSE(s) = \left[\frac{T}{N}\sum_{j=1}^{N}(s^{numerical}(\tau_j) - s^{exact}(\tau_j))^2\right]^{\frac{1}{2}}. \tag{32}$$

We use the RMSE regression method to understand the relationships between $a^{numerical}$, $b^{numerical}$, and $s^{numerical}$ (the predicted values) and a^{exact}, b^{exact}, and s^{exact} (the observed values), respectively, for the j^{th} observation. For simplicity, we fix $T = 1$ throughout the simulations.

5.1. Numerical Example for Case 1

Considering the data inputs in the work of Hussein and Lesnic [41], we define

$$a(\tau) = \sqrt{1+\tau}, \quad c(x,\tau) = 0, \quad b(x,\tau) = 0, \quad s(\tau) = \sqrt{2-\tau}, \quad f(x,\tau) = 8 - 2\sqrt{1+\tau}, \tag{33}$$
$$u(x,\tau) = 8\tau + (1+x)^2.$$

Inserting the exact value of $u(x,\tau)$ into the integral in Equation (4) ($\ell \in \{0,1\}$) helps to analytically compute $\gamma_3(\tau)$ and $\gamma_4(\tau)$. Thus,

$$\gamma_3(\tau) = \int_0^{s(\tau)} u(x,\tau)dx = \sqrt{2-\tau}\left(\frac{5}{3} + \sqrt{2-\tau} + \frac{23\tau}{3}\right), \tag{34}$$

$$\gamma_4(\tau) = \int_0^{s(\tau)} xu(x,\tau)dx = (2-\tau)\left(1 + \frac{2\sqrt{2-\tau}}{3} + \frac{15\tau}{4}\right), \tag{35}$$

and using the earlier transformation

$$\eta = \frac{x}{s(\tau)} = \frac{x}{\sqrt{2-\tau}}$$

allows us to analytically calculate the exact values of $v(y,\tau)$ and $f(y,\tau)$. This leads to

$$v(\eta,\tau) = 8\tau + (1+\eta\sqrt{2-\tau})^2, \quad f(\eta,\tau) = 8 - 2\sqrt{1+\tau}, \tag{36}$$

which involves the inchoate and boundary conditions in Equations (6) and (7) and results in the following defined functions:

$$\varphi(\eta) = (1+\sqrt{2}\eta)^2; \;\; \gamma_1(\tau) = u(0,\tau) = 1 + 8\tau; \;\; \gamma_2(\tau) = u(s(\tau),\tau) = (1+\sqrt{2-\tau})^2.$$

For the transformed direct problem in Equations (5)–(8), $\gamma_3(\tau)$ and $\gamma_4(\tau)$ can be calculated numerically using the trapezium rule as shown in Equations (22) and (23). Tables 1 and 2 compare the exact values of $\gamma_3(\tau)$ and $\gamma_4(\tau)$, which are defined in Equations (34) and (35) respectively, and the corresponding numerical values approximated via CN techniques (Equations (22) and (23)) at equally-spaced time steps in the interval $\tau \in (0,1)$.

We focus on solving the inverse problem in Equations (5)–(8) in **Case 1**. When $b(x,\tau)$ is given, the functions $s(\tau)$ and $a(\tau)$ must be detected using the previous data inputs. We set initial guesses for $s(\tau)$ and $a(\tau)$ at $\tau = 0$ to start the optimization procedure. We achieve this by using Equations (13) and (14), respectively. Therefore, $\underline{s}(0) = \underline{s}_0 = \sqrt{2}$ and $\underline{a}(0) = 1$. We work with this particular example because all of the conditions in Theorem 1 are met, which ensures the existence and uniqueness of the inverse solutions.

Table 1. Real and numerically approximated values of $\gamma_3(\tau)$ at various times and mesh sizes.

τ	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
$M = N = 10$	2.2585	6.0973	6.8756	7.5906	8.2391	8.8176	9.3217	9.7466	10.0867
(Relative error)	(57.01452%)	(0.06565%)	(0.05384%)	(0.04481%)	(0.03643%)	(0.03176%)	(0.02683%)	(0.02155%)	(0.01884%)
$M = N = 20$	5.2552	6.0943	6.8728	7.5881	8.2369	8.8155	9.3198	9.745	10.0853
(Relative error)	(0.02093%)	(0.01641%)	(0.01309%)	(0.01186%)	(0.00971%)	(0.00794%)	(0.00644%)	(0.00513%)	(0.00496%)
$M = N = 40$	5.2544	6.0935	6.8721	7.5875	8.2363	8.815	9.3194	9.7446	10.0849
(Relative error)	(0.00571%)	(0.00328%)	(0.00291%)	(0.00395%)	(0.00243%)	(0.00227%)	(0.00215%)	(0.00103%)	(0.00099%)
$M = N = 80$	5.2542	6.0933	6.872	7.5873	8.2361	8.8149	9.3193	9.7445	10.0848
(Relative error)	(0.00190%)	(0%)	(0.00146%)	(0.00132%)	(0%)	(0.00113%)	(0.00107%)	(0%)	(0%)
Exact	5.2541	6.0933	6.8719	7.5872	8.2361	8.8148	9.3192	9.7445	10.0848

Table 2. Real and numerically approximated values of $\gamma_4(\tau)$ at various times and mesh sizes.

τ	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
$M = N = 10$	4.3762	4.7761	5.1048	5.3624	5.549	5.6647	5.7098	5.6843	5.5885
(Relative error)	(0.40610%)	(0.33824%)	(0.28683%)	(0.24677%)	(0.21310%)	(0.18393%)	(0.15963%)	(0.13917%)	(0.12362%)
$M = N = 20$	4.3629	4.764	5.0938	5.3525	5.5402	5.6569	5.7029	5.6783	5.5833
(Relative error)	(0.10095%)	(0.08403%)	(0.07072%)	(0.06169%)	(0.05418%)	(0.04598%)	(0.03859%)	(0.03347%)	(0.0305%)
$M = N = 40$	4.3592	4.761	5.0911	5.3501	5.538	5.655	5.7012	5.6769	5.5821
(Relative error)	(0.01606%)	(0.02101%)	(0.01769%)	(0.01682%)	(0.01445%)	(0.01238%)	(0.00877%)	(0.00881%)	(0.00896%)
$M = N = 80$	4.3588	4.7602	5.0904	5.3494	5.5374	5.6545	5.7008	5.6765	5.5817
(Relative error)	(0.00689%)	(0.00420%)	(0.00393%)	(0.00374%)	(0.00361%)	(0.00354%)	(0.00175%)	(0.00176%)	(0.00179%)
Exact	4.3585	4.76	5.0902	5.3492	5.5372	5.6543	5.7007	5.6764	5.5816

We use the same data inputs above (Equation (33)) and consider the numerical estimations of γ_3 and γ_4 when there is no noise applied to Equation (28). Then, we visualize the minimized objective function in Equation (25) against the number of iterations when the regularization parameters β_1 and β_2 are set to zero. Figure 1 shows a fast convergence on the measured minimized objective function as the number of iterations rises, reaching a monotonic stage in 31 iterations. The non-regularized objective function's curve diminishes rapidly in the first five iterations and then reaches a steady stage with a high order of accuracy of $O(10^{-9})$.

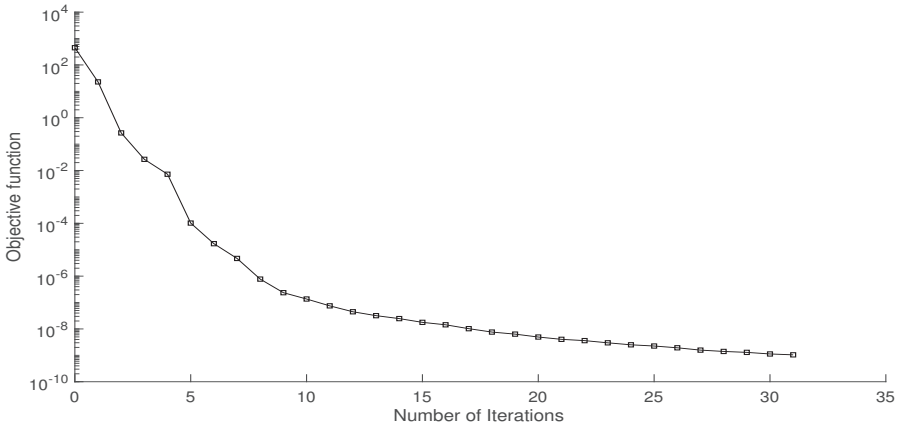
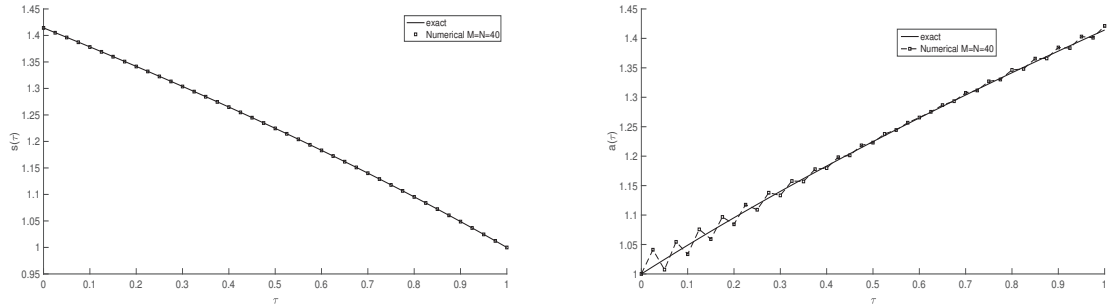


Figure 1. Visualization of the minimized objective function defined in Equation (25) when no noise is imposed and no regularization is applied.

The associated numerical solutions for the unknown functions $s(\tau)$ and $a(\tau)$ are calculated simultaneously and plotted in Figure 2a and b, respectively.



(**a**) (**b**)

Figure 2. Exact solutions (solid lines) and numerical solutions (squares) for (**a**) $s(\tau)$ and (**b**) $a(\tau)$ when noise and regularization are not applied.

We successfully retrieve an accurate and steady solution for the free-boundary function $s(\tau)$. Figure 2 shows minor instability in the thermal diffusivity values of the function $a(\tau)$ close to both edges. The oscillations are more evident on the left-hand side of the approximated $a(\tau)$, increasing as the time gets closer to zero. Consequently, $s(\tau)$ does not need to be regularized. Hence, we fix $\beta_1 = 0$ in Equation (25) and use the Tikhonov regularization method for $a(\tau)$.

Next, we find the inverse solution for **Case 1** when a small level of noise of $\epsilon = 0.01\%$ is included in the over-determination conditions $\gamma_3(\tau)$ and $\gamma_4(\tau)$, as in Equation (28). We emphasize that the regularization procedure has not yet been used to solve the problem. Figure 3 shows the objective minimization function against the number of iterations when noise is applied. The figure illustrates that the non-regularized objective function's convergence is fast in the first few iterations, settled in the next few, and then becomes steady. The objective function reaches a stationary stage in 140 iterations, with a high order of accuracy of $O(10^{-7})$. Not considering the exact solutions of $\gamma_3(\tau)$ and $\gamma_4(\tau)$ and applying some noise to them results in a slower convergence and a lower level of accuracy, as seen by comparing the minimized objective functions shown in Figures 1 and 3.

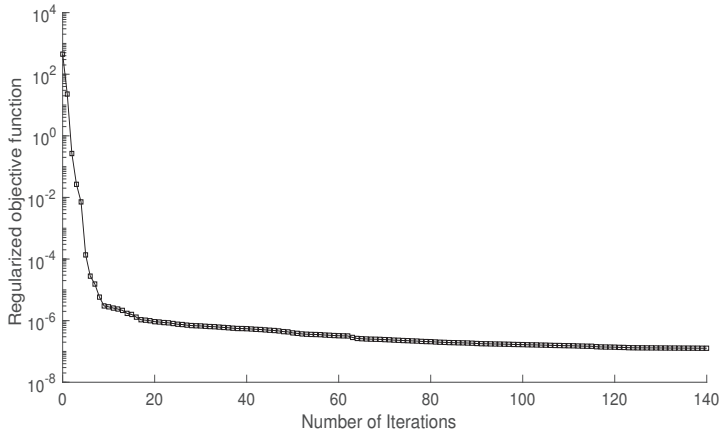
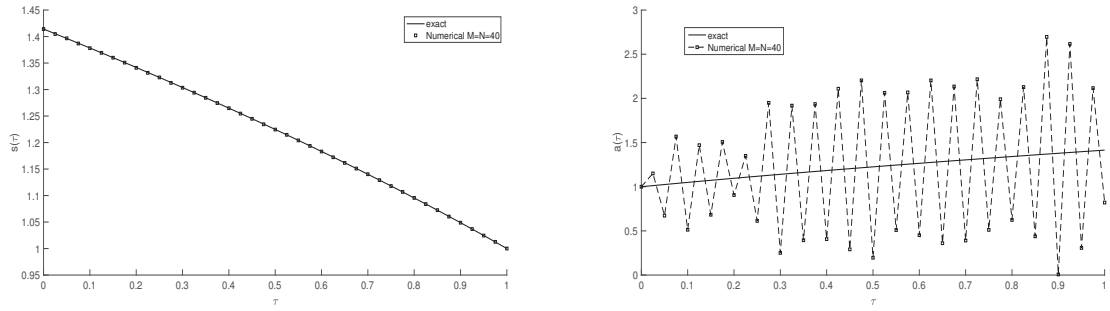


Figure 3. Visualization of the minimized objective function defined in Equation (25) when the noise level $\epsilon = 0.01\%$ is imposed and no regularization is applied.

Exploring the associated numerical results in Figure 4 illustrates that the free boundary maintains stability, while the thermal diffusivity shows more severe oscillatory behavior compared to Figure 2b.

Finally, we apply the Tikhonov regularization method to obtain a stable, accurate, and efficient reconstruction for the unknown function $a(\tau)$. The L-curve method, the RMSE

curve, and trial and error are used to identify the most appropriate regularization parameter β_2 [15,42–44].



(**a**) (**b**)

Figure 4. Real and numerically approximated values for (**a**) $s(\tau)$ and (**b**) $a(\tau)$ with noise level $\epsilon = 0.01\%$ and no regularization applied.

Finding the optimal value of the regularization parameter using the L-curve method is impossible since we cannot see the L-shaped curve in the line graph in Figure 5. Instead, we apply the RMSE regression method; as shown in Figure 6, the curve's minimum value occurs at $\beta_2 = 10^{-4}$. Thus, β_2 is considered an optimal value of the regularization parameter to obtain the best numerical values for $a(\tau)$.

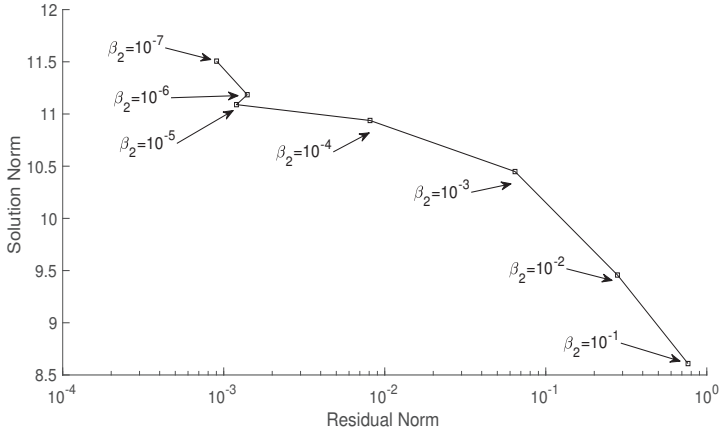


Figure 5. L-curve line graph where potential values for β_2 are tested and the noise level $\epsilon = 0.01\%$ is imposed.

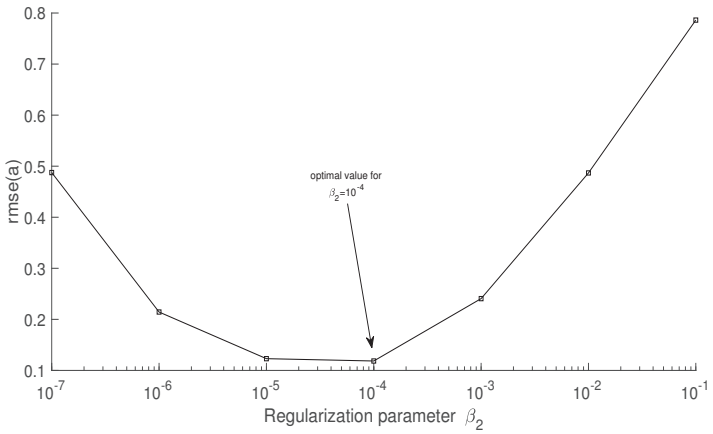


Figure 6. Minimum RMSE line graph where potential values for β_2 are tested and the noise level $\epsilon = 0.01\%$ is imposed.

Figure 7 shows the calculated minimized objective functions using Equation (25) against the number of iterations when the regularization parameter β_2 is set to 10^{-3}, 10^{-4}, and 10^{-5}. The objective function with $\beta_2 = 10^{-4}$ converges faster than the others and reaches a steady distribution in 10 iterations, taking 380 s. The objective functions with $\beta_2 = 10^{-3}$ and $\beta_2 = 10^{-5}$ have slower convergences and reach their stationary distributions in 12 iterations after 447 and 449 s, respectively.

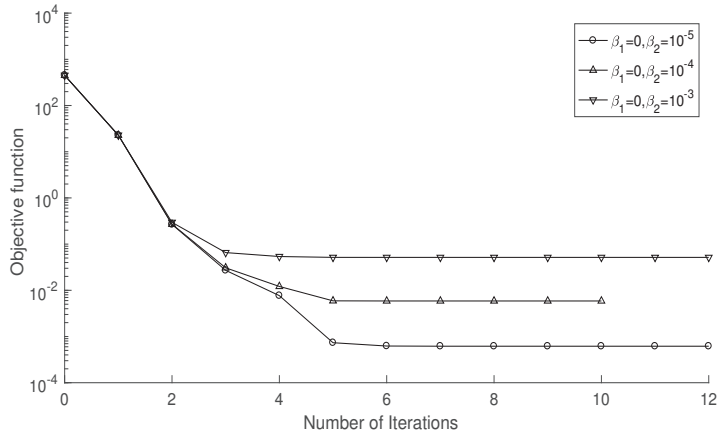
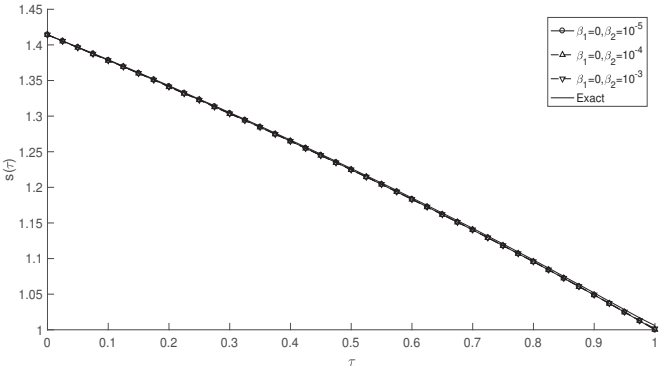
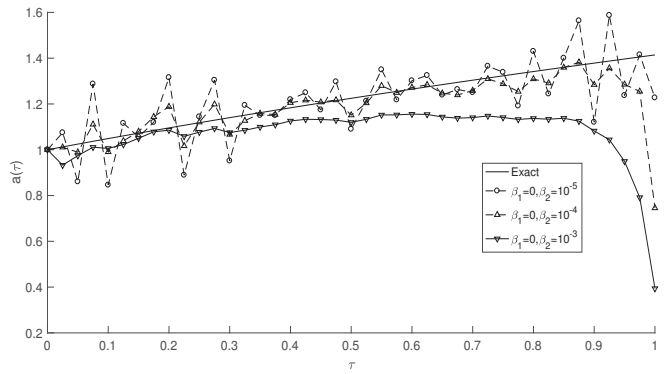


Figure 7. Visualization of the minimized objective function defined in Equation (25) when the noise level $\epsilon = 0.01\%$ is imposed and various regularization parameters are considered.

Figure 8 illustrates the reconstructions of the inverse solutions $s(\tau)$ and $a(\tau)$ in **Case 1**, taking into account various regularization parameters, including the optimal value.



(**a**)



(**b**)

Figure 8. Real and numerically approximated values for (**a**) $s(\tau)$ and (**b**) $a(\tau)$ with $p = 0.01\%$ noise and various regularization parameters for Case 1.

The free-boundary function $s(\tau)$ is estimated very well even when its corresponding regularization parameter is set to zero, $\beta_1 = 0$. Since the reconstruction is performed simultaneously for all model parameters, selecting the best regularization parameter for the thermal diffusivity $a(\tau)$ positively impacts the obtained free-boundary values; this is evident in the right-hand side of the curve in Figure 8a. Moreover, $\beta_2 = 10^{-4}$ has significantly smoothed $a(\tau)$ and increased the solutions' accuracy compared to the other examined regularization parameters.

5.2. Numerical Example for Case 2

In this section, we solve the inverse problem stated in Equations (5)–(8) for **Case 2**, where $s(\tau)$ and $b(\tau)$ are unknown functions and the temperature is $v(\eta, \tau)$. We solve this inverse problem with fed-in data:

$$\begin{aligned}
&\varphi(\eta) = (1+\eta)^2, \quad \gamma_1(\tau) = 1 + 10\tau, \quad \gamma_2(\tau) = 10t + (2+\tau)^2 \\
&f(\eta, \tau) = 8 - 2(-1-\tau)(1 + (1+\tau)\eta), \\
&\gamma_3(\tau) = (1+\tau)\left(\frac{7}{3} + \frac{35\tau}{3} + \frac{\tau^2}{3}\right), \quad \tau \in [0, T] \\
&\gamma_4(\tau) = \int_0^{s(\tau)} xu(x,\tau)dx = \frac{1}{12}(1+\tau)^2(17 + 74\tau + 3\tau^2), \quad \tau \in [0, T].
\end{aligned}$$

The exact and numerical values for input data γ_3 and γ_4 with their relative errors are listed in Tables 3 and 4, respectively. The conditions in Theorems 2 and 3 concerning the uniqueness and existence of the solution hold. Therefore, the local existence and uniqueness of the solution are guaranteed. The analytical solution of this problem is provided as follows:

$$u(x,\tau) = 10\tau + (1+x)^2, \quad b(\tau) = -1 - \tau, \quad s(t) = 1 + \tau,$$

and the transformed solution is

$$v(\eta,\tau) = 10\tau + (1 + (1+\tau)\eta)^2, \quad b(\tau) = -1 - \tau, \quad s(\tau) = 1 + \tau. \tag{37}$$

At the beginning of our investigation, we started with a noise-free case, i.e., $p = 0$ in Equation (28). Figure 9 shows the objective function in Equation (27) as a function of the number of iterations where no regularization is applied, i.e., $\beta_1 = \beta_2 = 0$. The figure shows the speedy convergence of the minimization problem toward local minima with a meagre value of order $O(10^{-9})$ in 19 iterations. The corresponding numerical results are presented in Figure 10. From this figure, we can see the overlap between the exact and numerical solutions of the unknown functions $s(\tau)$ and $b(\tau)$, which indicates an excellent agreement with $RMSE(b) = 7.9 \times 10^{-4}$ and $RMSE(h) = 4.9 \times 10^{-5}$ from Equations (31) and (32), respectively.

Table 3. Real and numerically approximated solutions of $\gamma_3(\tau)$ at various times and mesh sizes for the direct problem.

τ	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
$N = M = 10$	3.8559	5.6189	7.626	9.8792	12.3806	15.1322	18.1359	21.3937	24.9078
(Relative error)	(0.05709%)	(0.05164%)	(0.04854%)	(0.04557%)	(0.04525%)	(0.04562%)	(0.04523%)	(0.045361%)	(0.04619%)
$N = M = 20$	3.8542	5.6167	7.6232	9.8758	12.3764	15.127	18.1297	21.3864	24.8992
(Relative error)	(0.01297%)	(0.01246%)	(0.01181%)	(0.01114%)	(0.01131%)	(0.01124%)	(0.01103%)	(0.01122%)	(0.01165%)
$N = M = 40$	3.8538	5.6162	7.6226	9.875	12.3754	15.1258	18.1282	21.3846	24.897
(Relative error)	(0.00259%)	(0.00356%)	(0.00394%)	(0.00304%)	(0.00323%)	(0.00331%)	(0.00276%)	(0.00281%)	(0.00282%)
$N = M = 80$	3.8537	5.616	7.6224	9.8747	12.3751	15.1254	18.1278	21.3842	24.8965
(Relative error)	(0%)	(0%)	(0.00131%)	(0%)	(0.00081%)	(0.00066%)	(0.00055%)	(0.00094%)	(0.00080%)
Exact	3.8537	5.616	7.6223	9.8747	12.375	15.1253	18.1277	21.384	24.8963

Table 4. Real and numerically approximated solutions of $\gamma_4(\tau)$ at various times and mesh sizes for the direct problem.

τ	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
$N = M = 10$	2.4715	3.8413	5.5732	7.7085	10.2895	13.3591	16.9606	21.1381	25.9361
(Relative error)	(0.32881%)	(0.28457%)	(0.26085%)	(0.24448%)	(0.23282%)	(0.22507%)	(0.21981%)	(0.21667%)	0.21406%)
$N = M = 20$	2.4654	3.8331	5.5623	7.6944	10.2716	13.3366	16.9327	21.1038	25.8946
(Relative error)	(0.08119%)	(0.07049%)	(0.06476%)	(0.06112%)	(0.05844%)	(0.05627%)	(0.05495%)	(0.05405%)	(0.05371%)
$N = M = 40$	2.4639	3.8311	5.5596	7.6909	10.2671	13.3309	16.9257	21.0953	25.8842
(Relative error)	(0.02029%)	(0.01827%)	(0.01619%)	(0.01561%)	(0.01461%)	(0.01350%)	(0.01359%)	(0.01375%)	(0.01352%)
$N = M = 80$	2.4635	3.8306	5.5589	7.69	10.266	13.3295	16.9239	21.0931	25.8816
(Relative error)	(0.0041%)	(0.00522%)	(0.00359%)	(0.00390%)	(0.00389%)	(0.00300%)	(0.00295%)	(0.00332%)	(0.00348%)
Exact	2.4634	3.8304	5.5587	7.6897	10.2656	13.3291	16.9234	21.0924	25.8807

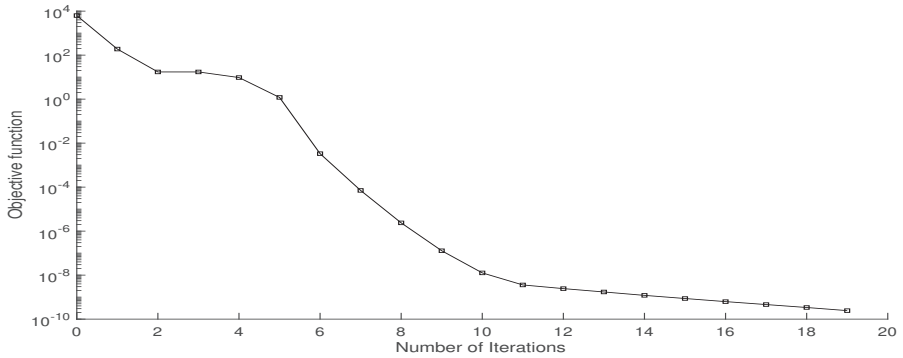
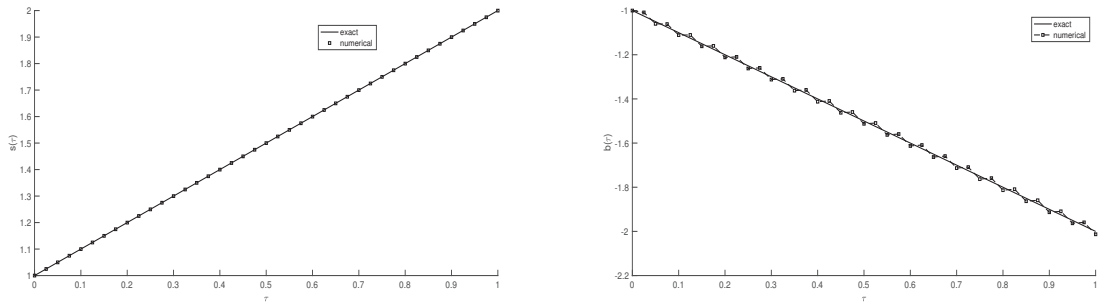


Figure 9. The objective function in Equation (27) when noise and regularization are not applied in Case 2.



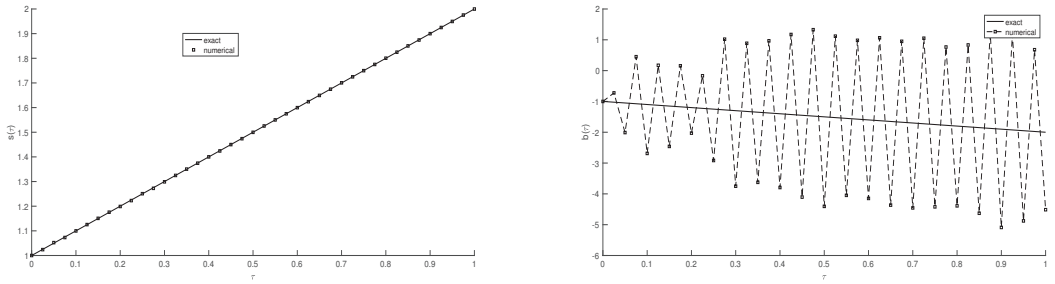
(**a**) (**b**)

Figure 10. Actual and numerically approximated solutions for (**a**) $s(\tau)$ and (**b**) $b(\tau)$ when noise and regularization are not applied in Case 2.

Figure 10b displays the numerical solution of $s(\tau)$, which nearly follows their corresponding precise solutions, with some noticeable small instability despite not yet applying any errors/noise in the inputs. When we add $p = 0.01\%$ noise to the input data in Equation (28), the solutions often follow the same pattern as in Case 1. As shown in Figure 11, we obtain an accurate and stable solution for $s(\tau)$ and an unstable solution for $b(\tau)$, indicating that regularization is necessary.

We expect such unusable behavior of the calculated solution because we are investigating an ill-posed problem. A small error in the input data (γ_3, γ_4) leads to major errors in the output solutions ($s(\tau)$, $b(\tau)$). Regularization should be applied to overcome this difficulty. We apply Tikhonov regularization by adding a penalty term $(\beta_1\|s\|^2 + \beta_2\|b\|^2)$ to the objective function in Equation (27).

Similar to the corresponding case in Case 1, noise does not affect $s(\tau)$. By contrast, $b(\tau)$ applies regularization on $b(\tau)$ only and fixes $\beta_1 = 0$. To obtain the optimal regularization parameter β_2, which gives accurate and stable results, different selection methods were considered. These include the L-curve method, minimum RMSE, and trial and error using Equations (31) and (32). Figures 12 and 13 present the L-curve plot and the minimum RMSE values as a function of the regularization parameter β_2, respectively. From these figures, it can be concluded that the best choice for β_2 is 10^{-3}, which has the lowest value of $RMSE(b)$. The objective function in Equation (27) is plotted for some $\beta_2 \in \{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$ in Figure 14.



(**a**) (**b**)

Figure 11. Actual and numerically approximated solutions for (**a**) $s(\tau)$ and (**b**) $b(\tau)$ where $p = 0.01\%$ noise is included and no regularization is applied in Case 2.

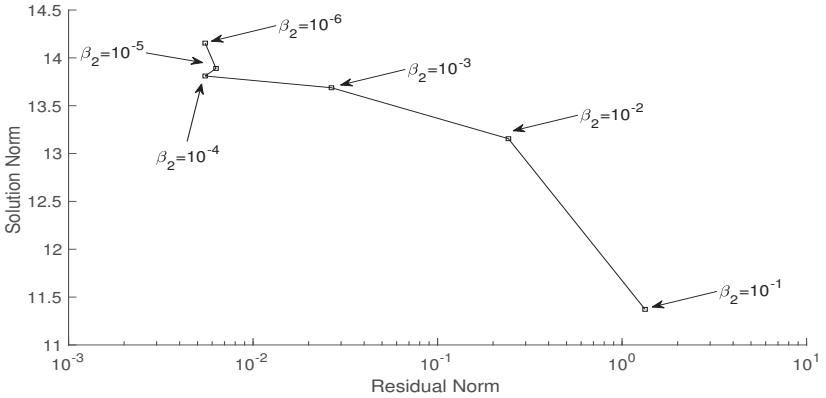


Figure 12. L-curve plot for the second inverse problem with $p = 0.01\%$ noise and regularization for Case 2.

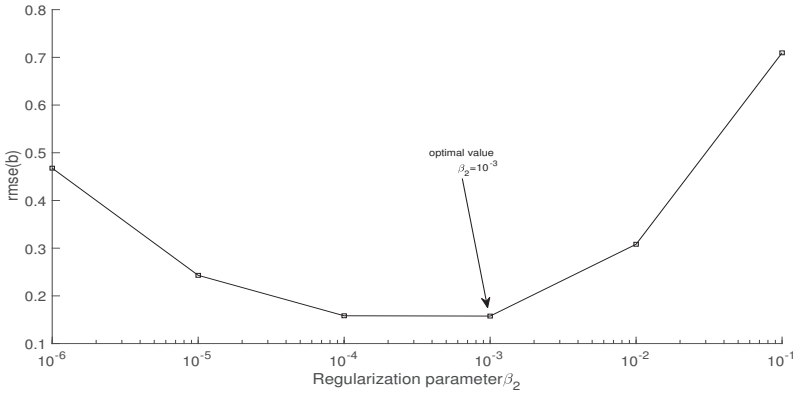


Figure 13. Minimum RMSE plot for the second inverse problem with $p = 0.01\%$ noise (potential measurement errors) and regularization for Case 2.

From Figures 12 and 13, one can conclude that the optimal choice for β_2 is 10^{-3}; this is also clear in Figure 15.

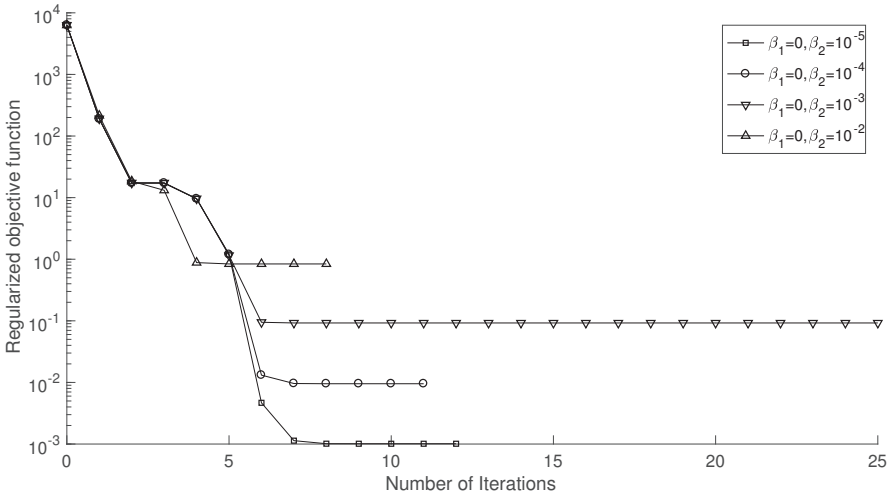
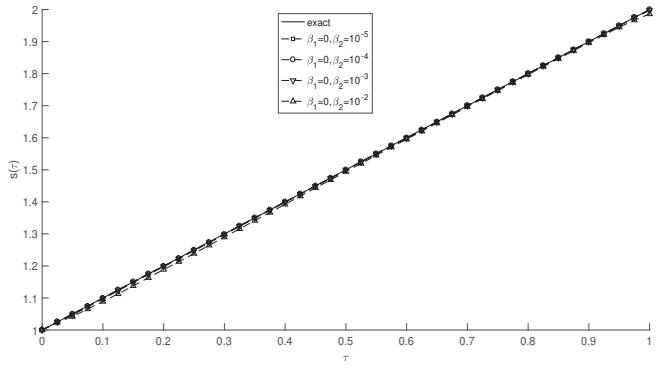
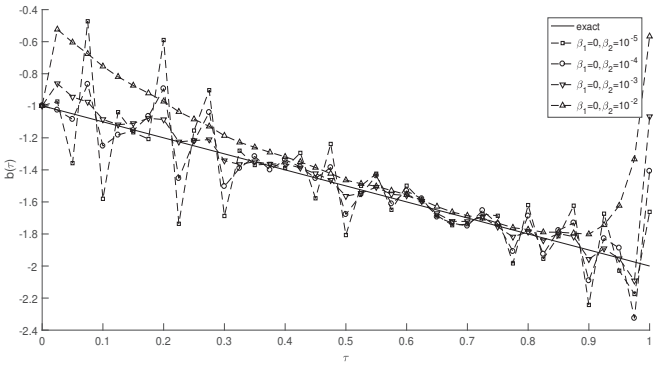


Figure 14. The regularized objective function in Equation (27) for the second inverse problem with $p = 0.01\%$ noise (potential measurement errors) and regularization for Case 2.



(**a**)



(**b**)

Figure 15. Actual and numerically approximated solutions for (**a**) $s(\tau)$ and (**b**) $b(\tau)$ with $p = 0.01\%$ noise (potential measurement errors) and regularization for Case 2.

6. Conclusions

This research describes a successful approach to finding the numerical solutions (temperature distributions, free boundary, and thermal diffusivity or advection velocities) to time-dependent free-boundary inverse coefficient problems while ensuring the approximations' existence, uniqueness, and reliability. First, we converted the moving boundary function to a fixed domain function by choosing a simple transformation. Then, due to the unconditional stability and convergence of the Crank–Nicholson finite difference scheme, we used it to solve the forward problem (an initial boundary value problem). The obtained numerical values of non-localized integral observations, $\gamma_3(\tau)$ and $\gamma_4(\tau)$ in the over-determined conditions, are used to generate and feed in the reconstruction code, which uses the *lsqnonlin* non-linear least-square optimization routine. This MATLAB toolbox uses the trust region reflective algorithm based on the inner-reflective Newton technique and does not call for an additional gradient for the objective function. We used the Tikhonov regularization approach (ridge regression) to overcome the problem's ill-posed nature, ensuring the solution's applicability and correctness. We also used the root mean square error and L-curve to test and select the optimal values for the regularization parameters to obtain excellent approximations, as the numerical examples show. The numerical approach in this paper could be extended to two- or three-dimensional problems. Additionally, future studies could consider time- and spatial-dependent coefficient identification problems. Moreover, deep learning techniques could be integrated into the mathematical methods used in this work to increase the speed and accuracy of solutions for such inverse problems.

Author Contributions: Conceptualization, M.S.H. and S.O.H.; methodology, T.E.D.; software, M.S.H.; validation, T.E.D. and S.O.H.; formal analysis, M.S.H. and M.Q.; investigation, M.S.H.; resources, T.E.D.; data curation, M.S.H.; writing—original draft, M.S.H., T.E.D., S.O.H. and M.Q.; writing—review and editing, T.E.D. and S.O.H.; visualization, M.Q.; supervision, M.S.H. and T.E.D.; project administration, T.E.D.; funding acquisition, T.E.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Department of Computing and Mathematics, Faculty of Science and Engineering, Manchester Metropolitan University, Manchester, UK (grant number).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding authors.

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

Abbreviations

The following abbreviations are used in this manuscript:

CN	Crank–Nicolson finite difference method
L^2	Euclidean norm
FDM	Finite difference method
ICP	Initial condition problem
lsqnonlin	Optimization tool
PDE	Partial differential equation
RMSE	Root mean square error
C^2	The *C* refers to a continuous function, and 2 the times its derivative is continuous
TRR	Trust region reflective

References

1. Glotov, D.; Hames, W.E.; Meirc, A.J.; Ngoma, S. An integral constrained parabolic problem with applications in their thermochronology. *Comput. Math. Appl.* **2016**, *71*, 2301–2312. [CrossRef]
2. Čiupaila, R.; Sapagovas, M.; Štikonienė, O. Numerical solution of nonlinear elliptic equation with nonlocal condition. *Non-Linear Anal. Model. Control* **2013**, *18*, 412–426. [CrossRef]
3. Hazanee, A.; Lesnic, D. Determination of a time-dependent coefficient in the bioheat equation. *Int. J. Mech. Sci.* **2014**, *88*, 259–266. [CrossRef]

4. Hussein, S.O.; Dyhoum, T.E. Solutions for non-homogeneous wave equations subject to unusual and Neumann boundary conditions. *J. Appl. Math.* **2022**, *430*, 127–285. [CrossRef]
5. Cao, K.; Lesnic, D. Simultaneous reconstruction of the spatially-distributed reaction coefficient, initial temperature and heat source from temperature measurements at different times. *Comput. Math. Appl.* **2019**, *78*, 3237–3249. [CrossRef]
6. Liu, W.; Wang, B. A local meshless method for two classes of parabolic inverse problems. *J. Appl. Math. Phys.* **2018**, *6*, 968–978. [CrossRef]
7. Huang, D.; Li, Y.; Pei, D. Identification of a time-dependent coefficient in heat conduction problem by new iteration method. *Adv. Math. Phys.* **2018**, *2018*, 4918256. [CrossRef]
8. Mishra, A. Thompson and Troian slip effects on ternary hybrid nanofluid flow over a permeable plate with chemical reaction. *Numer. Heat Transf. Part B Fundam.* **2024**, 1–29. [CrossRef]
9. Hussein, S.O.; Lesnic, D.; Yamamoto, M. Reconstruction of space-dependent potential and/or damping coefficients in the wave equation. *Comput. Math. Appl.* **2017**, *74*, 1435–1454. [CrossRef]
10. Dyhoum, T.E.; Aykroyd, R.G.; Lesnic, D. Detection of multiple rigid inclusions from ERT data using the complete-electrode model. *Int. J. Tomogr. Simul.* **2017**, *30*, 64–86.
11. Dehghan, M. Implicit collocation technique for the heat equation with the non-classic initial condition. *Int. J. Nonlinear Sci. Numer.* **2006**, *7*, 461–466. [CrossRef]
12. Mazraeh, H.D.; Pourgholi, R.; Tavana, S. The fully-implicit finite difference method for solving nonlinear inverse parabolic problems with unknown source term. *Int. J. Comput. Sci.* **2018**, *9*, 405–418. [CrossRef]
13. Iijima, K. Numerical solution of backward heat conduction problems by a high order lattice-free finite difference method. *J. Chin. Inst. Eng.* **2004**, *27*, 611–620. [CrossRef]
14. Fu, C.-L.; Xiong, X.-T.; Zhi, Q. Fourier regularization for a backward heat equation. *J. Math. Anal. Appl.* **2007**, *331*, 472–480. [CrossRef]
15. Damirchi, J.; Yazdanian, A.R.; Shamami, T.R.; Hasanpour, M. Numerical investigation of an inverse problem based on regularization method. *Math. Sci.* **2019**, *13*, 193–199. [CrossRef]
16. Lesnic, D.; Elliott, L.; Ingham, D.B. An iterative boundary element method for solving the backward heat conduction problem using an elliptic approximation. *Inverse Probl. Sci. Eng.* **1998**, *6*, 255–279. [CrossRef]
17. Mera, N.S.; Elliott, L.; Ingham, D.B.; Lesnic, D. An iterative boundary element method for solving the one-dimensional backward heat conduction problem. *Int. J. Heat Mass Transf.* **2001**, *44*, 1937–1946. [CrossRef]
18. Friedman, A. Free boundary problems in science and technology. *Not. Am. Math. Soc.* **2000**, *47*, 854–861.
19. Słota, D. Direct and inverse one-phase Stefan problem solved by the variational iteration method. *Comput. Math. Appl.* **2007**, *54*, 1139–1146. [CrossRef]
20. Wang, S.; Perdikaris, P. Deep learning of free boundary and Stefan problems. *J. Comput. Phys.* **2020**, *428*, 109914. [CrossRef]
21. Hryntsiv, N. The inverse problem with free boundary for a weakly degenerate parabolic equation. *J. Math. Sci.* **2012**, *183*, 779–795. [CrossRef]
22. Huzyk, N. Inverse problem of determining the coefficients in a degenerate parabolic equation, Electron. *J. Differ. Equ.* **2014**, *2014*, 1–11.
23. Huzyk, N. Determination of the lower coefficient in a parabolic equation with substantial degeneration. *Ukr. Math. J.* **2016**, *68*, 1049–1061. [CrossRef]
24. Martín-Vaquero, J.; Sajavičius, S. The two-level finite difference schemes for the heat equation with nonlocal initial condition. *Appl. Math. Comput.* **2019**, *342*, 166–177. [CrossRef]
25. Daftardar-Gejji, V.; Jafari, H. An iterative method for solving nonlinear functional equations. *J. Math. Anal. Appl.* **2006**, *316*, 753–763. [CrossRef]
26. Huntul, M.J.; Lesnic, D. Determination of a time-dependent free boundary in a two-dimensional parabolic problem. *Int. J. Appl. Comput.* **2019**, *3*, 118–132. [CrossRef]
27. Huntul, M.J.; Lesnic, D. Determination of the time-dependent convection coefficient in two-dimensional free boundary problems. *Eng. Comput.* **2021**, *38*, 3694–3709. [CrossRef]
28. Qassim, M.; Hussein, M.S. Numerical solution to recover time-dependent coefficient and free boundary from nonlocal and Stefan type overdetermination conditions in heat equation. *Iraqi J. Sci.* **2022**, *63*, 147–155.
29. Adil, Z.; Hussein, M.S.; Lesnic, D. Determination of time-dependent coefficients in moving boundary problems under nonlocal and heat moment observations. *Int. J. Comput. Methods Eng. Sci. Mech.* **2021**, *22*, 500–513. [CrossRef]
30. Dyhoum, T.E. Finite Difference Methods for Solving One and Two-Dimensional Heat Equation. Master's Thesis, Misurata University, Misurata, Libya, 2008.
31. Lin, Y. Analytical and numerical solutions for a nonlocal nonlinear parabolic differential equations class. *SIAM J. Math. Anal.* **1994**, *25*, 577–1594. [CrossRef]
32. Liu, J.; Hao, Y. Crank–Nicolson method for solving uncertain heat equation. *Soft. Comput.* **2022**, *26*, 937–945. [CrossRef] [PubMed]
33. Coleman, T.F.; Li, Y. On the convergence of interior-reflective Newton methods for nonlinear minimization subject to bounds. *Math. Program.* **1994**, *67*, 189–224. [CrossRef]
34. Coleman, T.F.; Verma, A. A preconditioned conjugate gradient approach to linear equality constrained minimization. *Comput. Optim. Appl.* **2001**, *20*, 61–72. [CrossRef]

35. Coleman, T.F.; Li, Y. An interior trust region approach for nonlinear minimization subject to bounds. *SIAM J. Optim.* **1996**, *6*, 418–445. [CrossRef]
36. Yuan, Y.X. Recent advances in trust region algorithms. *Math. Program.* **2015**, *151*, 249–281. [CrossRef]
37. Mathwoks, Documentation Optimization Toolbox. 2019. Available online: www.mathworks.com/help/toolbox/optim/ug/brnoybu.html (accessed on 16 September 2019).
38. Manasi, T.; Sathish, V. Pipelined preconditioned conjugate gradient methods for real and complex linear systems for distributed memory architectures. *J. Parallel Distrib. Comput.* **2022**, *163*, 147–155.
39. Kress, R. Tikhonov Regularization. In *Linear Integral Equations*; Applied Mathematical Sciences; Springer: New York, NY, USA, 2014; Volume 82. [CrossRef]
40. Shraddha, M.N.; Prasad, K.J.R.; Venkatanareshbabu, K. Fractional Tikhonov regularization to improve the performance of extreme learning machines. *Phys. A Stat. Mech. Appl.* **2020**, *551*, 124034.
41. Hussein, M.S.; Lesnic, D. Determination of a time-dependent thermal diffusivity and free boundary in heat conduction. *Int. Commun. Heat Mass Transf.* **2014**, *53*, 154–163. [CrossRef]
42. Belge, M.; Kilmer, M.E.; Miller, E.L. Efficient determination of multiple regularization parameters in a generalized L-curve framework. *Inverse Probl.* **2002**, *18*, 1161–1183. [CrossRef]
43. Engl, H.; Grever, W. Using the L-curve for determining optimal regularization parameters. *Numer. Math.* **1994**, *69*, 25–31. [CrossRef]
44. Liu, S.; Zhang, J. Machine-learning-based prediction of regularization parameters for seismic inverse problems. *Acta Geophys.* **2021**, *69*, 809–820. [CrossRef]

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

mathematics

MDPI

Article

Euler Method for a Class of Linear Impulsive Neutral Differential Equations

Gui-Lai Zhang *, Yang Sun, Ya-Xin Zhang and Chao Liu

College of Sciences, Northeastern University, Shenyang 110819, China; 2101906@stu.neu.edu.cn (Y.S.); 202115109@stu.neu.edu.cn (Y.-X.Z.); liuchao@neuq.edu.cn (C.L.)
* Correspondence: zhangguilai@neuq.edu.cn

Abstract: This paper presents a new numerical scheme for a class of linear impulsive neutral differential equations with constant coefficients based on the Euler method. We rigorously establish the first-order convergence of the proposed numerical approach. Additionally, the asymptotical stability of the exact solutions and numerical solutions of impulsive neutral differential equations are studied. To substantiate our findings, two illustrative examples are provided, demonstrating the theoretical conclusions of this paper.

Keywords: impulsive neutral differential equations; convergence; asymptotical stability; Euler method

MSC: 65L03; 65L05; 65L20

1. Introduction

Impulsive delay differential equations (IDDEs) have garnered significant attention due to their applicability across various domains, such as in neural networks [1–3], dynamics [4], control theory [5,6], engineering [7], etc. In particular, the theoretical exploration of impulsive neutral delay differential equations (INDDEs) has been enriched by numerous researchers, focusing on aspects like existence [8–12], oscillation [13–15], and stability: in [16], Xiaodi Li et al. apply the Razumikhin method for impulsive functional differential equations of the neutral type to analyze the stability; Bainov Drumi Dimitrov et al., in [17], studied the uniform asymptotic stability of impulsive differential-difference equations of the neutral type via Lyapunov's direct method; and Refs. [18,19] discussed the exponential stability of INDDEs.

In general, it is difficult or even impossible to obtain explicit solutions for INDDEs. Therefore, it is necessary to study numerical solutions of INDDEs. But there is very little research on the numerical solutions of INDDEs. The stability and asymptotical stability of numerical methods for linear and nonlinear INDDEs with special fixed impulsive moments are studied by applying the method of transformation in [20,21]. The convergence of a numerical format of the Euler method for INDDEs is studied in [22]. However, in Ref. [22], the authors ignore the fact that the exact solution of an INDDE is continuous everywhere except the points at the moments of impulsive effect. Hence, our present paper introduces a new numerical method based on the Euler method for INDDEs, addressing this oversight in the aforementioned study.

The structure of this paper is as follows. Section 2 details the construction of the numerical method and its convergence proof. In Section 3, according to the distribution of the roots of a characteristic INDDE, the conditions of stability, asymptotical stability, and instability for the exact solution of INDDEs are given. Moreover, according to the distribution of the roots of the characteristic equation of the discrete equation obtained from the Euler method for INDDEs, the conditions of stability, asymptotical stability, and instability for the numerical solution of INDDEs are provided. Section 4 presents

two examples to validate the main results. Finally, Section 5 concludes the paper with a summary of the findings and suggestions for future research directions.

2. Convergence of Euler's Method for INDDEs

In this paper, we investigate the following impulsive neutral differential equation:

$$(x(t) + cx(t-\sigma))' = ax(t) + bx(t-\tau),\ t \geq 0, t \neq \tau_k, \tag{1}$$

$$\Delta x(\tau_k) = l_k,\ k \in \mathbb{Z}^+ = \{1, 2, \cdots\}, \tag{2}$$

$$x(t) = \phi(t),\ -r \leq t \leq 0, \tag{3}$$

where σ and τ are positive constants and $r = \max\{\sigma, \tau\}$; a, b, c, and l_k are real constants; $x'(t)$ denotes the right-hand derivative of $x(t)$; and $\Delta x(t) = x(t^+) - x(t^-)$. The impulse times τ_k satisfy $0 < \tau_1 < \cdots < \tau_k < \tau_{k+1} < \cdots$ and $\lim\limits_{k\to\infty} \tau_k = \infty$. The initial function $\phi : [-r, 0] \to \mathbb{R}$ is a given continuous function.

Definition 1 ([23]). *A real valued right continuous function x(t) is said to be the solution of the initial value problem* (1)–(3) *if the following conditions are satisfied:*

(a) *$x(t)$ is continuous everywhere except the points τ_k, $k \in \mathbb{Z}^+$;*

(b) *the function $x(t) + cx(t-\sigma)$ is continuously differentiable for $t \geq 0$ and τ_k, $k \in \mathbb{Z}^+$;*

(c) *$x(\tau_k^+)$ and $x(\tau_k^-)$ exist and $x(\tau_k^+) = x(\tau_k)$, $k \in \mathbb{Z}^+$;*

(d) *$x(t)$ satisfies the differential Equation* (1) *for $t \geq 0$, satisfies the impulsive conditions at $t = \tau_k$, and satisfies* (3).

Based on Euler's method, the new numerical scheme for Equations (1)–(3) is constructed as follows:

$$\begin{cases} X_{n+1} = X_n + ahX_n + bhX_{n-m_2} - cX_{n-m_1+1} + cX_{n-m_1}, n \neq \eta_k - 1, n - m_1 \neq \eta_j - 1, \\ Y_k = X_n + ahX_n + bhX_{n-m_2} - cX_{n-m_1+1} + cX_{n-m_1}, n = \eta_k - 1, \\ X_{n+1} = X_n + ahX_n + bhX_{n-m_2} - cY_j + cX_{n-m_1}, n - m_1 = \eta_j - 1, \\ X_{\eta_k} = Y_k + l_k, \\ X_i = \phi_i = \phi(ih), ih \in [-r, 0], i = -m, \cdots, -1, 0, \end{cases} \tag{4}$$

where $h = \frac{\sigma}{m_1}$, $m_2 = \lfloor \frac{\tau}{h} \rfloor$, $m_1, m_2, j, k \in \mathbb{Z}^+$, h is a stepsize, and $0 < h < \min\{\sigma, \tau\}$; the floor function $\lfloor \frac{\tau}{h} \rfloor$ denotes the largest integer less than or equal to $\frac{\tau}{h}$. Let $\eta_0 = 0$ and

$$\eta_k = \begin{cases} \frac{\tau_k}{h}, & \frac{\tau_k}{h} \in \mathbb{Z}^+, \\ \lfloor \frac{\tau_k}{h} \rfloor + 1, & otherwise. \end{cases} \tag{5}$$

X_n is an approximation of exact solutions $x(t_n)$ for arbitrary $t_n = nh$, $n \in \mathbb{Z}^+$. Y_k is an approximation of exact solutions $x(\tau_k^-)$ if $\eta_k = \frac{\tau_k}{h}$; otherwise, Y_k is a virtual value obtained from the Euler method. Here, Y_k is a virtual value, meaning that Y_k is not an approximation of the exact solution at any given time. The reason why Y_k, $k \in \mathbb{N}$ are calculated is so that the numerical solution does not make additional jumps, but instead jumps only once in the vicinity of each moment of impulsive effect.

In order to study the convergence of INDDEs, Equations (1)–(3) are considered on the finite interval $[-r, T]$, where T is a given positive constant. For convenience, we assume that there exist $p, N \in \mathbb{Z}^+$ such that $T = pmh$ and $0 < \tau_1 < \cdots < \tau_k < \cdots < \tau_N \leq T < \tau_{N+1}$, where $m = \max\{m_1, m_2\}$. From [9], we know that $x(t)$ and $x'(t)$ are bounded. Therefore, we assume that there exists an $M > 0$, such that the solution $x(t)$ of Equations (1) and (2) satisfies $|x(t)| \leq M$ and $|x'(t)| \leq M$ for $t \in [-r, T]$. For the sake of simplicity, we also assume that

$$|\phi(t) - \phi(s)| \leq M|t - s| \tag{6}$$

and

$$|\phi'(t) - \phi'(s)| \leq M|t - s|. \tag{7}$$

Let $e_n = |x(nh) - X_n|$, which denotes the global error. The following theorem will demonstrate that the convergence order of the Euler method for Equations (1)–(2) is 1 by analyzing the global error e_n.

Theorem 1. *The convergence order of Euler scheme* (4) *is 1—that is, there exists a $C > 0$, such that $e_n \leq Ch$, $1 \leq n \leq pm$.*

Proof. We shall show that there exists a $C_k > 0$, such that

$$e_n \leq C_k h, n \in I_k = [\eta_{k-1} + 1, \eta_k] \bigcap \mathbb{Z}, k = 1, 2, 3, \cdots, N. \tag{8}$$

First, we show there exists a $C_1 > 0$ such that $e_n \leq C_1 h$, $n \in I_1$. For the sake of simplicity, we only consider the following situation, but others can be proved similarly. Assume that $\tau = m_2 h + \delta$, $0 \leq \delta < 1$, $r = \max\{\sigma, \tau\} = \sigma$, $m = m_1$ and $0 < \tau < \sigma < \tau_1$.

When $0 \leq t_n \leq \tau < \sigma < \tau_1$, we have

$$\begin{aligned}
&e_n = |x(nh) - X_n| \\
&= |x((n-1)h) + \int_{(n-1)h}^{nh} ax(t) + bx(t-\tau) - cx'(t-\sigma)dt - X_{n-1} - ahX_{n-1} \\
&\quad - bhX_{n-m_2-1} + cX_{n-m_1} - cX_{n-m_1-1}| \\
&\leq e_{n-1} + |\int_0^h ax((n-1)h + t)dt - ahX_{n-1} + \int_0^h bx((n - m_2 - \delta - 1)h + t)dt \\
&\quad - bhX_{n-m_2-1} - \int_0^h cx'((n - m_1 - 1)h + t)dt + cX_{n-m_1} - cX_{n-m_1-1}| \\
&\leq e_{n-1} + \int_0^h |a||x((n-1)h + t) - X_{n-1}| + |b||x((n - m_2 - \delta - 1)h + t) \\
&\quad - X_{n-m_2-1}|dt + |c||\int_0^h -x'((n - m_1 - 1)h + t)dt + X_{n-m_1} - X_{n-m_1-1}|,
\end{aligned} \tag{9}$$

where

$$\begin{aligned}
&|x((n-1)h + t) - X_{n-1}| \\
&= |x((n-1)h) + \int_{(n-1)h}^{t+(n-1)h} ax(u) + bx(u-\tau) - cx'(u-\sigma)du - X_{n-1}| \\
&\leq e_{n-1} + \int_0^t |a||x((n-1)h + u)| + |b||x((n - m_2 - \delta - 1)h + u)| \\
&\quad + |c||x'((n - m_1 - 1)h + u)|du \\
&\leq e_{n-1} + (|a| + |b| + |c|)Mh,
\end{aligned} \tag{10}$$

based on Equation (6)

$$\begin{aligned}
&|x((n - m_2 - \delta - 1)h + t) - X_{n-m_2-1}| \\
&= |\phi(t + (n - m_2 - \delta - 1)h) - \phi((n - m_2 - 1)h)| \\
&\leq M|t - \delta h| \\
&\leq Mh
\end{aligned} \tag{11}$$

and

$$\begin{aligned}
&|\int_0^h -x'((n-m_1-1)h+t)dt + X_{n-m_1} - X_{n-m_1-1}| \\
&= |x((n-m_1-1)h) - x((n-m_1)h) + X_{n-m_1} - X_{n-m_1-1}| \\
&= |\phi((n-m_1-1)h) - \phi((n-m_1)h) + \phi((n-m_1)h) - \phi((n-m_1-1)h)| \\
&= 0.
\end{aligned} \tag{12}$$

Substituting Equations (10)–(12) into Equation (9), we find that

$$\begin{aligned}
&e_n \le e_{n-1} + |a|h[e_{n-1} + (|a|+|b|+|c|)Mh] + |b|Mh^2 + |c| \cdot 0 \\
&= e_{n-1}(1+|a|h) + Mh^2[|a|(|a|+|b|+|c|) + |b|] \\
&\le C_{1,1,1}h.
\end{aligned} \tag{13}$$

Let $e_0 = 0$, and, according to Gronwall inequality, we can calculate

$$C_{1,1,1} = \frac{[|a|(|a|+|b|+|c|)+|b|]M}{|a|}e^{|a|T}.$$

When $\tau < t_n \le 2\tau < \sigma < \tau_1$, we have

$$\begin{aligned}
&e_n = |x(nh) - X_n| \\
&\le e_{n-1} + \int_0^h |a||x((n-1)h+t) - X_{n-1}| + |b||x((n-m_2-\delta-1)h+t) \\
&\quad - X_{n-m_2-1}|dt + |c||\int_0^h -x'((n-m_1-1)h+t)dt + X_{n-m_1} - X_{n-m_1-1}|.
\end{aligned} \tag{14}$$

As discussed in Equation (10), for $t \in [0,h]$, we find that

$$|x((n-1)h+t) - X_{n-1}| \le e_{n-1} + (|a|+|b|+|c|)Mh, \tag{15}$$

$$\begin{aligned}
&|x((n-m_2-\delta-1)h+t) - X_{n-m_2-1}| \\
&= |x((n-m_2-1)h) + \int_{(n-m_2-1)h}^{(n-m_2-\delta-1)h+t} ax(u) + bx(u-\tau) - cx'(u-\sigma)du \\
&\quad - X_{n-m_2-1}| \\
&\le e_{n-m_2-1} + |\int_0^{t-\delta h} ax(u+(n-m_2-1)h) + bx(u+(n-2m_2-\delta-1)h) \\
&\quad - cx'(u+(n-m_1-m_2-1)h)du| \\
&\le e_{n-m_2-1} + (|a|+|b|+|c|)Mh.
\end{aligned} \tag{16}$$

Similar to Equation (12), we obtain

$$|\int_0^h -x'((n-m_1-1)h+t)dt + X_{n-m_1} - X_{n-m_1-1}| = 0. \tag{17}$$

Substituting Equations (15)–(17) into Equation (14) yields

$$\begin{aligned}
&e_n \le e_{n-1} + |a|h(e_{n-1} + (|a|+|b|+|c|)Mh) + |b|h(e_{n-m_2-1} + (|a|+|b|+|c|)Mh) \\
&\le e_{n-1}(1+|a|h) + Mh^2(|a|+|b|)(|a|+|b|+|c|) + |b|C_{1,1,1}h^2 \\
&\le C_{1,1,2}h,
\end{aligned}$$

where

$$C_{1,1,2} = \frac{M(|a|+|b|)(|a|+|b|+|c|)+|b|C_{1,1,1}}{|a|}e^{|a|T}.$$

When $(k-1)\tau < t_n \leq k\tau$ for some $k \in [3, \lfloor \frac{\sigma}{\tau} \rfloor]$, similarly to the discussion above, we can obtain

$$C_{1,1,k} = \frac{M(|a|+|b|)(|a|+|b|+|c|)+|b|C_{1,1,k-1}}{|a|} e^{|a|T},$$

such that $e_n \leq C_{1,1,k}h$.

When $\lfloor \frac{\sigma}{\tau} \rfloor \tau < t_n < \sigma$, an analogous calculation can be performed to calculate $C_{1,1,\lfloor \frac{\sigma}{\tau} \rfloor+1}$, such that the inequality $e_n \leq C_{1,1,\lfloor \frac{\sigma}{\tau} \rfloor+1}h$ holds. Taking $C_{1,1} = max\{C_{1,1,1}, C_{1,1,2}, \cdots, C_{1,1,\lfloor \frac{\sigma}{\tau} \rfloor+1}\}$, we have $e_n \leq C_{1,1}h$ for $1 \leq n < m_1 < \eta_1$.

When $\sigma \leq t_n < 2\sigma < \tau_1$, $m_1 \leq n < 2m_1 < \eta_1$ is satisfied, we prioritize $\sigma \leq t_n < \sigma + \tau < 2\sigma < \tau_1$, which gives us

$$\begin{aligned} e_n &= |x(nh) - X_n| \\ &\leq e_{n-1} + \int_0^h |a||x((n-1)h+t) - X_{n-1}| + |b||x((n-m_2-\delta-1)h+t) \\ &\quad - X_{n-m_2-1}|dt + |c||\int_0^h -x'((n-m_1-1)h+t)dt + X_{n-m_1} - X_{n-m_1-1}|. \end{aligned} \tag{18}$$

For $t \in [0, h]$, as discussed in Equation (10) and Equation (3), we have

$$|x((n-1)h+t) - X_{n-1}| \leq e_{n-1} + (|a|+|b|+|c|)Mh, \tag{19}$$

$$\begin{aligned} &|x((n-m_2-\delta-1)h+t) - X_{n-m_2-1}| \\ &\leq e_{n-m_2-1} + (|a|+|b|+|c|)Mh, \\ &\leq C_{1,1}h + (|a|+|b|+|c|)Mh \end{aligned} \tag{20}$$

and

$$\begin{aligned} &|\int_0^h -x'((n-m_1-1)h+t)dt + X_{n-m_1} - X_{n-m_1-1}| \\ &= |x((n-m_1-1)h) - x((n-m_1)h) + X_{n-m_1} - X_{n-m_1-1}| \\ &= |e_{n-m_1-1} - e_{n-m_1}| \\ &\leq e_{n-m_1-1}|a|h + Mh^2(|a|+|b|)(|a|+|b|+|c|) + |b|C_{1,1}h^2 \\ &= (|a|+|b|)C_{1,1}h^2 + Mh^2(|a|+|b|)(|a|+|b|+|c|). \end{aligned} \tag{21}$$

Substituting Equations (19)–(21) into Equation (18) yields

$$\begin{aligned} e_n &\leq e_{n-1} + |a|h(e_{n-1} + (|a|+|b|+|c|)Mh) + |b|h(C_{1,1}h + (|a|+|b|+|c|)Mh) + |c| \\ &\quad ((|a|+|b|)C_{1,1}h^2 + Mh^2(|a|+|b|)(|a|+|b|+|c|)) \\ &\leq (1+|a|h)e_{n-1} + Mh^2(|a|+|b|+|c|)(|a|+|b|)(1+|c|) + (|b|+|c|(|a|+|b|))C_{1,1}h^2 \\ &\leq C_{1,2,1}h, \end{aligned}$$

where $C_{1,2,1} = \frac{M(|a|+|b|+|c|)(|a|+|b|)(1+|c|)+(|b|+|c|(|a|+|b|))C_{1,1}}{|a|} e^{|a|T}$.

When $\sigma + \tau \leq t_n < \sigma + 2\tau$, similarly to the discussion above, we can conclude that

$$\begin{aligned} e_n &\leq e_{n-1} + |a|h(e_{n-1} + (|a|+|b|+|c|)Mh) + |b|h(C_{1,2,1}h + (|a|+|b|+|c|)Mh) \\ &\quad + |c|((|a|+|b|)C_{1,1}h^2 + Mh^2(|a|+|b|)(|a|+|b|+|c|)) \\ &\leq e_{n-1}(1+|a|h) + Mh^2(|a|+|b|+|c|)(|a|+|b|)(1+|c|) + (|b|+|c|(|a|+|b|))C_{1,2,1}h^2 \\ &\leq C_{1,2,2}h, \end{aligned}$$

where $C_{1,2,2} = \frac{M(|a|+|b|+|c|)(|a|+|b|)(1+|c|)+C_{1,2,1}(|b|+|c|(|a|+|b|))}{|a|} e^{|a|T}$.

When $\sigma + (k-1)\tau \le t_n < \sigma + k\tau$ for some $k \in [3, \lfloor \frac{\sigma}{\tau} \rfloor]$, similarly to the discussion above, we can calculate $C_{1,2,k} = \frac{M(|a|+|b|+|c|)(|a|+|b|)(1+|c|)+C_{1,2,k-1}(|b|+|c|(|a|+|b|))}{|a|} e^{|a|T}$.

When $\sigma + \lfloor \frac{\sigma}{\tau} \rfloor \tau \le t_n < 2\sigma$, an analogous calculation can be performed to obtain $C_{1,2,\lfloor \frac{\sigma}{\tau} \rfloor + 1}$, such that the inequality $e_n \le C_{1,2,\lfloor \frac{\sigma}{\tau} \rfloor + 1} h$ holds.

Taking $C_{1,2} = max\{C_{1,2,1}, C_{1,2,2}, \cdots, C_{1,2,\lfloor \frac{\sigma}{\tau} \rfloor + 1}\}$, we have $e_n \le C_{1,2} h$ for $m_1 \le n < 2m_1 < \eta_1$.

Let $B_{i+1} = \lfloor \frac{\eta_{i+1} - \eta_i}{m_1} \rfloor$. When $(j-1)\sigma < t_n \le j\sigma$, $(j-1)m_1 < n \le jm_1$, $3 \le j \le B_1$, and $e_n \le C_{1,j} h$, where $C_{1,j} = max\{C_{1,j,1}, C_{1,j,2}, \cdots, C_{1,j,\lfloor \frac{\sigma}{\tau} \rfloor + 1}\}$, and $C_{1,j,k}$ can thus be obtained as follows:

$$\begin{aligned} e_n &\le e_{n-1}(1 + |a|h) + Mh^2(|a| + |b| + |c|)(|a| + |b|)(1 + |c|) \\ &\quad + (|b| + |c|(|a| + |b|))C_{1,j,k-1}h^2 \\ &\le C_{1,j,k} h. \end{aligned}$$

When $B_1\sigma \le t_n < \tau_1$, an analogous calculation can be performed to obtain C_{1,B_1+1}, such that the inequality $e_n \le C_{1,B_1+1} h$ holds.

If we take $\tilde{C}_1 = max\{C_{1,1}, C_{1,2}, \cdots, C_{1,B_1+1}\}$, we have $e_n \le \tilde{C}_1 h$ for $1 \le n < \eta_1$.

If $t_n = \tau_1$, $n = \eta_1$, it is possible to derive

$$\begin{aligned} &e_n = |x(\eta_1 h) - X_{\eta_1}| = |x(\eta_1 h) - Y_1 - l_1| \\ &= |x(\tau_1) + \int_{\tau_1}^{\eta_1 h} ax(t) + bx(t-\tau) - cx'(t-\sigma)dt - X_{\eta_1}| \\ &= |l_1 + x(\tau_1^-) + \int_{\tau_1}^{\eta_1 h} ax(t) + bx(t-\tau) - cx'(t-\sigma)dt - (l_1 + X_{\eta_1 - 1} \\ &\quad + ahX_{\eta_1-1} + bhX_{\eta_1-m_2-1} - cX_{\eta_1-m_1} + cX_{\eta_1-m_1-1})| \\ &\le |x((\eta_1 - 1)h) + \int_{(\eta_1-1)h}^{\tau_1} ax(t) + bx(t-\tau) - cx'(t-\sigma)dt + \int_{\tau_1}^{\eta_1 h} ax(t) \\ &\quad + bx(t-\tau) - cx'(t-\sigma)dt - (X_{\eta_1-1} + ahX_{\eta_1-1} + bhX_{\eta_1-m_2-1} \\ &\quad - cX_{\eta_1-m_1} + cX_{\eta_1-m_1-1})| \\ &\le e_{\eta_1-1} + |\int_{(\eta_1-1)h}^{\tau_1} ax(t) + bx(t-\tau) - cx'(t-\sigma)dt + \int_{\tau_1}^{(\eta_1-1)h} ax(t) \\ &\quad + bx(t-\tau) - cx'(t-\sigma)dt + \int_{(\eta_1-1)h}^{\eta_1 h} ax(t) + bx(t-\tau) - cx'(t-\sigma)dt \\ &\quad - ahX_{\eta_1-1} - bhX_{\eta_1-m_2-1} + cX_{\eta_1-m_1} - cX_{\eta_1-m_1-1}| \\ &\le e_{\eta_1-1} + \int_0^h |a||x(t + (\eta_1 - 1)h) - X_{\eta_1-1}| + |b||x(t + (\eta_1 - m_2 - \delta - 1)h \\ &\quad - X_{\eta_1-m_2-1})dt + |\int_0^h -cx'(t + (\eta_1 - m_1 - 1)h)dt + cX_{\eta_1-m_1} - cX_{\eta_1-m_1-1}| \\ &\le e_{\eta_1-1} + |a|h(e_{\eta_1-1} + (|a| + |b| + |c|)Mh) + |b|h(e_{\eta_1-m_2-1} + (|a| + |b| \\ &\quad + |c|)Mh) + |c||e_{\eta_1-m_1-1} - e_{\eta_1-m_1}| \\ &\le (1 + |a|h)e_{\eta_1-1} + Mh^2(|a| + |b|)(1 + |c|)(|a| + |b| + |c|) + \tilde{C}_1 h^2(|b| + |c| \\ &\quad (|a| + |b|)) \\ &\le C_1 h. \end{aligned} \tag{22}$$

If we let $C_1 = \frac{M(|a|+|b|)(1+|c|)(|a|+|b|+|c|)+\tilde{C}_1(|b|+|c|(|a|+|b|))}{|a|} e^{|a|T}$, we can obtain the inequality $e_n \le C_1 h$, which holds for $n \in I_1$.

When $n \in I_2$, an analogous calculation can be performed to obtain $e_n \leq C_{2,1}h$ for $\eta_1 < n < \eta_1 + m_1 - 1$, but in this paper we omit the proof of this calculation. Next, for $n = \eta_1 + m_1$,

$$\begin{aligned}
&e_n = |x(nh) - X_n| \\
&= |x((n-1)h) + \int_{(n-1)h}^{nh} \big[ax(t) + bx(t-\tau) - cx'(t-\sigma)\big]dt - X_{n-1} - ahX_{n-1} \\
&\quad - bhX_{n-m_2-1} + cY_1 - cX_{n-m_1-1}| \\
&\leq e_{n-1} + |a| \int_0^h |x((n-1)h + t) - X_{n-1}|dt + |b| \int_0^h |x((n - m_2 - \delta - 1)h + t) \\
&\quad - X_{n-m_2-1}|dt + |c|| \int_{(n-1)h}^{nh} -x'(t-\sigma)dt + Y_1 - X_{n-m_1-1}|.
\end{aligned} \tag{23}$$

As discussed in Equations (11) and (17) for $t \in [0, h]$, we have

$$|x((n-1)h + t) - X_{n-1}| \leq e_{n-1} + (|a| + |b| + |c|)Mh \tag{24}$$

and

$$\begin{aligned}
&|x((n - m_2 - \delta - 1)h + t) - X_{n-m_2-1}| \\
&\leq e_{n-m_2-1} + (|a| + |b| + |c|)Mh \\
&\leq C_{2,1}h + (|a| + |b| + |c|)Mh
\end{aligned} \tag{25}$$

and

$$\begin{aligned}
&|\int_{(n-1)h}^{nh} -x'(t-\sigma)dt + Y_1 - X_{n-m_1-1}| \\
&= |\int_{(\eta_1-1)h}^{\eta_1 h} -x'(s)ds + Y_1 - X_{n-m_1-1}| \\
&= |-\int_{(\eta_1-1)h}^{\tau_1} x'(s)ds - \int_{\tau_1}^{\eta_1 h} x'(s)ds + Y_1 - X_{n-m_1-1}| \\
&= |\int_{(\eta_1-1)h}^{\tau_1} \big[-ax(s) - bx(s-\tau) + cx'(s-\sigma)\big]ds + \int_{\tau_1}^{\eta_1 h} [-ax(s) - bx(s-\tau) \\
&\quad + cx'(s-\sigma)]ds + ahX_{\eta_1-1} + bhX_{\eta_1-m_1-1} - cX_{\eta_1-m_1} + cX_{\eta_1-m_1-1}| \\
&\leq |a| \int_0^{\tau_1-(\eta_1-1)h} |x(s + (\eta_1 - 1)h) - X_{\eta_1-1}|ds + |b| \int_0^{\tau_1-(\eta_1-1)h} |x(s + (\eta_1 - m_2 - \delta - 1)h) \\
&\quad - X_{\eta_1-m_2-1}|ds + |a| \int_{\tau_1-(\eta_1-1)h}^{h} |x(s + (\eta_1 - 1)h) - X_{\eta_1-1}|ds \\
&\quad + |b| \int_{\tau_1-(\eta_1-1)h}^{h} |x(s + (\eta_1 - m_2 - \delta - 1)h) - X_{\eta_1-m_2-1}|ds + |\int_{(\eta_1-1)h}^{\tau_1} cx'(s-\sigma)ds \\
&\quad + \int_{\tau_1}^{\eta_1 h} cx'(s-\sigma)ds - cX_{\eta_1-m_1} + cX_{\eta_1-m_1-1}| \\
&\leq |a|h\big(e_{\eta_1-1} + (|a| + |b| + |c|)Mh\big) + |b|h\big(e_{\eta_1-m_2-1} + (|a| + |b| + |c|)Mh\big) \\
&\quad + |c||e_{\eta_1-m_2} - e_{\eta_1-m_2-1}| \\
&\leq Mh^2(|a| + |b|)(1 + |c| + |c|^2)(|a| + |b| + |c|) + C_1h^2(|a| + |b|)(1 + |c| + |c|^2).
\end{aligned} \tag{26}$$

Substituting Equations (24)–(26) into Equation (23), we find that

$$\begin{aligned} e_n &\le e_{n-1} + |a|h[e_{n-1} + (|a| + |b| + |c|)Mh] + |b|h[C_{2,1}h + (|a| + |b| + |c|)Mh] \\ &\quad + |c|Mh^2(|a| + |b|)(1 + |c| + |c|^2)(|a| + |b| + |c|) \\ &\quad + |c|C_1h^2(|a| + |b|)(1 + |c| + |c|^2) \\ &= e_{n-1}(1 + |a|h) + Mh^2(|a| + |b| + |c|)(|a| + |b|)(1 + |c| + |c|^2 + |c|^3) \\ &\quad + |b|C_{2,1}h^2 + |c|C_1h^2(|a| + |b|)(1 + |c| + |c|^2) \\ &\le D_1h, \end{aligned} \tag{27}$$

where $D_1 = \frac{M(|a|+|b|+|c|)(|a|+|b|)(1+|c|+|c|^2+|c|^3)+C_{2,1}|b|+|c|C_1(|a|+|b|)(1+|c|+|c|^2)}{|a|} \cdot \mathrm{e}^{|a|T}$.

The same as before, we take $\tilde{C}_2 = \max\{C_{2,1}, C_{2,2}, \cdots, C_{2,B_2+1}, D_1\}$. So, $e_n \le \tilde{C}_2h$ holds for $n \in I_2$.

Assume that Equation (10) holds for $n \in I_{s-1}$—that is, $e_n \le C_{s-1}h$ holds for $n \in I_{s-1}$. Now, we will show that Equation (8) holds for $n \in I_s$.

When $\tau_{s-1} \le t_n \le \tau_{s-1} + \tau < \tau_{s-1} + \sigma$,

$$\begin{aligned} e_n &= |x(nh) - X_n| \\ &\le e_{n-1} + \int_0^h |a||x((n-1)h + t) - X_{n-1}| + |b||x((n - m_2 - \delta - 1)h + t) \\ &\quad - X_{n-m_2-1}|dt + |c||\int_0^h -x'((n - m_1 - 1)h + t)dt + X_{n-m_1} - X_{n-m_1-1}|. \end{aligned}$$

According to Equation (10), Equation (3), and Equation (21) and the related discussions above, for $t \in [0, h]$, we have

$$|x((n-1)h + t) - X_{n-1}| \le e_{n-1} + (|a| + |b| + |c|)Mh,$$

$$|x((n - m_2 - \delta - 1)h + t) - X_{n-m_2-1}| \le e_{n-m_2-1} + (|a| + |b| + |c|)Mh,$$

and

$$\begin{aligned} &|\int_0^h -x'((n - m_1 - 1)h + t)dt + X_{n-m_1} - X_{n-m_1-1}| \\ &\le e_{n-m_1-1}|a|h + Mh^2(|a| + |b|)(|a| + |b| + |c|) + |b|C_{s-1}h^2. \end{aligned}$$

Then, we find

$$\begin{aligned} e_n &\le (1 + |a|h)e_{n-1} + Mh^2(|a| + |b| + |c|)(|a| + |b|)(1 + |c|) \\ &\quad + (|b| + |c|(|a| + |b|))C_{s-1}h^2 \\ &\le C_{s,1,1}h. \end{aligned}$$

Just as in the discussion of $C_{1,1,k}$, we can obtain $C_{s,1,k}$, such that $e_n \le C_{s,1,k}h$, for $k = 1, 2, \cdots, \lfloor \frac{\sigma}{\tau} \rfloor$. When $\tau_{s-1} + \lfloor \frac{\sigma}{\tau} \rfloor \tau \le t_n < \tau_{s-1} + \sigma$, we also find that $C_{s,1,\lfloor \frac{\sigma}{\tau} \rfloor + 1}$ satisfies $e_n \le C_{s,1,\lfloor \frac{\sigma}{\tau} \rfloor + 1}h$.

Taking $C_{s,1} = \max\{C_{s,1,1}, C_{s,1,2}, \cdots, C_{s,1,\lfloor \frac{\sigma}{\tau} \rfloor + 1}\}$, we have $e_n \le C_{s,1}h$, for $\eta_{s-1} + 1 \le n < \eta_{s-1} + m_2$. Then, there is a finite number of $C_{s,j}$, and the number of $C_{s,j}$ does not depend on the stepsize h.

Also, when $n - m_1 = \eta_s$, similarly to Equation (27), we have

$$\begin{aligned} e_n &\le e_{n-1} + |a|h[e_{n-1} + (|a| + |b| + |c|)Mh] + |b|h[C_{s,1}h + (|a| + |b| + |c|)Mh] \\ &\quad + |c|Mh^2(|a| + |b|)(1 + |c| + |c|^2)(|a| + |b| + |c|) \\ &\quad + |c|C_{s-1}h^2(|a| + |b|)(1 + |c| + |c|^2) \\ &= e_{n-1}(1 + |a|h) + Mh^2(|a| + |b| + |c|)(|a| + |b|)(1 + |c| + |c|^2 + |c|^3) \\ &\quad + |b|C_{s,1}h^2 + |c|C_{s-1}h^2(|a| + |b|)(1 + |c| + |c|^2) \\ &\le D_{s-1}h, \end{aligned} \tag{28}$$

where $D_{s-1} = \frac{M(|a|+|b|+|c|)(|a|+|b|)(1+|c|+|c|^2+|c|^3)+C_{s,1}|b|+|c|C_{s-1}(|a|+|b|)(1+|c|+|c|^2)}{|a|} \cdot \mathrm{e}^{|a|T}$.

Taking

$$\tilde{C}_s = max\{C_{s,1}, C_{s,2}, \cdots, C_{s,B_s+1}, D_{s-1}\},$$

we have

$$e_n \le \tilde{C}_s h, \eta_{s-1} + 1 \le n < \eta_s.$$

When $t_n = \tau_s$ and $n = \eta_s$, performing a calculation analogous to that in Equation (22) yields the following:

$$\begin{aligned} e_n &= |x(\eta_s h) - X_{\eta_s}| \\ &\le (1 + |a|h)e_{\eta_s - 1} + Mh^2(|a| + |b|)(1 + |c|)(|a| + |b| + |c|) \\ &\quad + \tilde{C}_s h^2(|b| + |c|(|a| + |b|)) \\ &\le C_s h. \end{aligned}$$

If we let $C_s = \frac{M(|a|+|b|)(1+|c|)(|a|+|b|+|c|)+\tilde{C}_s(|b|+|c|(|a|+|b|))}{|a|} e^{|a|T}$, we obtain $e_n \le C_s h$ for $n \in I_s$.

When $\eta_N < n \le pm = \frac{T}{h}$, the same as $n \in [\eta_{s-1} + 1, \eta_s)$, we can calculate that $C_{N+1,1}$ satisfies $e_n \le C_{N+1,1}h$.

Taking

$$C = max\{C_1, C_2, \cdots, C_N, C_{N+1,1}\},$$

we find that $e_n \le Ch$ holds for $1 \le n \le pm$. This completes the proof. □

3. Asymptotical Stability of INDDEs

In this section, we study the asymptotical stability not only of the exact solutions of INDDEs but also of the numerical solutions of INDDEs.

3.1. Asymptotical Stability of the Exact Solutions of INDDEs

In order to study the asymptotical stability of INDDE (1) and (2), we consider the same equation with another initial function:

$$(\tilde{x}(t) + c\tilde{x}(t - \sigma))' = a\tilde{x}(t) + b\tilde{x}(t - \tau),\ t \ge 0, t \ne \tau_k, \tag{29}$$

$$\Delta\tilde{x}(\tau_k) = l_k,\ k \in \mathbb{Z}^+, \tag{30}$$

$$\tilde{x}(t) = \tilde{\phi}(t),\ -r \le t \le 0, \tag{31}$$

where σ and τ are positive constants and $r = \max\{\sigma, \tau\}$, a, b, c and l_k are real constants, $\tilde{x}'(t)$ denotes the right-hand derivative of $\tilde{x}(t)$, and $\Delta\tilde{x}(t) = \tilde{x}(t^+) - \tilde{x}(t^-)$. The impulse times τ_k satisfy $0 < \tau_1 < \cdots < \tau_k < \tau_{k+1} < \cdots$ and $\lim_{k\to\infty} \tau_k = \infty$. The initial function $\tilde{\phi} : [-r, 0] \to \mathbb{R}$ is a given continuous function.

Definition 2. *The solutions $x(t)$ of INDDE* (1)–(3) *and $\tilde{x}(t)$ of* (29)–(31) *are said to be stable if for every $\epsilon > 0$, there exists a number $\delta = \delta(\epsilon) > 0$, such that*

$$\|\phi - \tilde{\phi}\| = \max_{-r \le t \le 0} |\phi(t) - \tilde{\phi}(t)| < \delta,$$

which implies that

$$\|x(t) - \tilde{x}(t)\| < \epsilon, \quad \text{for all} \quad t \ge 0.$$

The solutions $x(t)$ of INDDE (1)–(3) *and $\tilde{x}(t)$ of* (29)–(31) *are said to be asymptotically stable if they are stable and there exists a number $\delta_0 > 0$ such that $\|\phi - \tilde{\phi}\| < \delta_0$ implies*

$$\lim_{t \to \infty} \|x(t) - \tilde{x}(t)\| = 0.$$

Assume that $x(t)$ is the solution of INDDE (1)–(3) and $\tilde{x}(t)$ is the solution of (29)–(31). Then, $y(t) = x(t) - \tilde{x}(t)$, $t \ge -r$, satisfying the following NDDE without impulsive perturbations:

$$\begin{cases} (y(t) + cy(t-\sigma))' = ay(t) + by(t-\tau), & t \ge 0, \\ y(t) = \varphi(t), & -r \le t \le 0, \end{cases} \tag{32}$$

where a, b, c, σ, and τ are real constants; $\tau > 0$, $\sigma > 0$, $\varphi(t) = \phi(t) - \tilde{\phi}(t)$ for all $t \in [-r, 0]$; and $\varphi \in C([-r, 0], \mathbb{R})$ is the initial function.

Definition 3 ([24,25])**.** *The zero solution of* (32) *is stable if for every $\epsilon > 0$, there exists a number $\delta = \delta(\epsilon) > 0$ such that*

$$\|\varphi\| = \max_{-r \le t \le 0} |\varphi(t)| = \max_{-r \le t \le 0} |\phi(t) - \tilde{\phi}(t)| < \delta$$

implies $|y(t)| < \epsilon$.

The zero solution of (32) *is asymptotically stable if the zero solution of* (32) *is stable and there exists a number $\delta_0 > 0$ such that $\|\varphi\| < \delta_0$ implies*

$$\lim_{t \to \infty} |y(t)| = 0.$$

Due to Definitions 2 and 3, we can easily reveal the following theorem.

Theorem 2. *The solutions $x(t)$ of INDDE* (1)–(3) *and $\tilde{x}(t)$ of* (29)–(31) *are stable if and only if the zero solution of* (32) *is stable.*

Moreover, the solutions $x(t)$ of INDDE (1)–(3) *and $\tilde{x}(t)$ of* (29)–(31) *are asymptotically stable if and only if the zero solution of* (32) *is asymptotically stable.*

The characteristic equation for an NDDE (i.e., the first of (32)) is as follows:

$$\lambda(1 + ce^{-\lambda\sigma}) = a + be^{-\lambda\tau}. \tag{33}$$

According to Refs. [24,25], the following lemma is a special case of their main results; the associated proof is omitted from this paper for brevity.

Lemma 1. *Assume that λ_0 is a real root of characteristic Equation* (33) *and satisfies*

$$\mu(\lambda_0) = |b|\tau e^{-\lambda_0\tau} + |c|e^{-\lambda_0\sigma}(1 + |\lambda_0|\sigma) < 1. \tag{34}$$

Then, the solution $y(t)$ of (32) *satisfies*

$$\lim_{t \to \infty} [e^{-\lambda_0 t} y(t)] = \frac{L(\lambda_0, \varphi)}{1 + \beta(\lambda_0)}, \tag{35}$$

where

$$L(\lambda_0, \varphi) = \varphi(0) + c\varphi(-\sigma) + b\mathrm{e}^{-\lambda_0\tau}\int_{-\tau}^{0}\mathrm{e}^{-\lambda_0 s}\varphi(s)ds - c\lambda_0\mathrm{e}^{-\lambda_0\sigma}\int_{-\sigma}^{0}\mathrm{e}^{-\lambda_0 s}\varphi(s)ds$$

and

$$\beta(\lambda_0) = b\tau\mathrm{e}^{-\lambda_0\tau} + c\mathrm{e}^{-\lambda_0\sigma}(1-\lambda_0\sigma). \tag{36}$$

Based on this, we can utilize the following statement to illustrate the stability and asymptotic stability of (32).

Theorem 3. *Assume that λ_0 is a real root of characteristic Equation* (32) *and satisfies $\mu(\lambda_0) < 1$. Let $\beta(\lambda_0)$ be defined by* (36) *and set*

$$R(\lambda_0;\varphi) = \max\left\{1, \max_{-r\le t\le 0}|\varphi(t)|, \max_{-r\le t\le 0}\mathrm{e}^{\lambda_0 t}|\varphi(t)|\right\}.$$

Then, the solution $y(t)$ of (32) *satisfies*

$$|y(t)| \le N(\lambda_0)R(\lambda_0;\varphi)\mathrm{e}^{\lambda_0 t}, \forall t \ge 0,$$

where

$$N(\lambda_0) = \mu(\lambda_0) + k(\lambda_0)\left(\frac{1+\mu(\lambda_0)}{1+\beta(\lambda_0)}\right)$$

and

$$k(\lambda_0) = 1 + |b|\tau\mathrm{e}^{-\lambda_0\tau} + |c|(1+|\lambda_0|\sigma\mathrm{e}^{-\lambda_0\sigma}).$$

Moreover, the zero solution of (32) *is described as follows:*

(i) The solution is stable if $\lambda_0 = 0$, or, equivalently, if the following conditions are satisfied:

$$a + b = 0, \ \ |b|\tau + |c| < 1;$$

(ii) The solution is asymptotically stable if $\lambda_0 < 0$;
(iii) The solution is unstable if $\lambda_0 > 0$.

Proof. Assume that

$$z(t) = \mathrm{e}^{-\lambda_0 t}y(t), \ \ \hat{z}(t) = z(t) - \frac{L(\lambda_0;\varphi)}{1+\beta(\lambda_0)}.$$

We can show that for $t \ge 0$,

$$z(t) \le \mu(\lambda_0)H(\lambda_0;\varphi) + \frac{|L(\lambda_0;\varphi)|}{1+\beta(\lambda_0)}.$$

Obviously, we can affirm that

$$\begin{aligned}
|L(\lambda_0;\varphi)| &\le |\varphi(0)| + |c||\varphi(-\sigma)| + |b|\mathrm{e}^{-\lambda_0\tau}\int_{-\tau}^{0}\mathrm{e}^{-\lambda_0 s}|\varphi(s)|ds \\
&\quad + |c||\lambda_0|\mathrm{e}^{-\lambda_0\sigma}\int_{-\sigma}^{0}\mathrm{e}^{-\lambda_0 s}|\varphi(s)|ds \\
&\le \left(1 + |c| + |b|\tau\mathrm{e}^{-\lambda_0\tau} + |c||\lambda_0|\sigma\mathrm{e}^{-\lambda_0\sigma}\right)R(\lambda_0;\varphi) \\
&= k(\lambda_0)R(\lambda_0;\varphi).
\end{aligned}$$

We also can find that

$$\begin{aligned} |H(\lambda_0;\varphi)| &\leq \max\left\{1, R(\lambda_0;\varphi) + \frac{L(\lambda_0;\varphi)}{1+\beta(\lambda_0)}\right\} \\ &\leq R(\lambda_0;\varphi) + \frac{L(\lambda_0;\varphi)}{1+\beta(\lambda_0)} \\ &\leq R(\lambda_0;\varphi) + \frac{k(\lambda_0)R(\lambda_0;\varphi)}{1+\beta(\lambda_0)} \\ &= \left(1 + \frac{k(\lambda_0)}{1+\beta(\lambda_0)}\right)R(\lambda_0;\varphi). \end{aligned}$$

So, for $t \geq 0$, we have

$$\begin{aligned} |z(t)| &\leq \mu(\lambda_0)\left(1 + \frac{k(\lambda_0)}{1+\beta(\lambda_0)}\right)R(\lambda_0;\varphi) + \frac{k(\lambda_0)R(\lambda_0;\varphi)}{1+\beta(\lambda_0)} \\ &= \left[\mu(\lambda_0)\left(1 + \frac{k(\lambda_0)}{1+\beta(\lambda_0)}\right) + \frac{k(\lambda_0)}{1+\beta(\lambda_0)}\right]R(\lambda_0;\varphi) \\ &= N(\lambda_0)R(\lambda_0;\varphi). \end{aligned}$$

Finally, from the definition of z, we calculate

$$|y(t)| = N(\lambda_0)R(\lambda_0;\varphi)\mathrm{e}^{-\lambda_0 t}, \ \ t \geq 0. \tag{37}$$

When $\lambda_0 = 0$,

$$|y(t)| = N(0)R(0;\varphi), \ \ t \geq 0.$$

Obviously, $\|\varphi\| = \max_{-r\leq t\leq 0}|\varphi(t)| \leq R(0;\varphi)$, and thus it follows that

$$|y(t)| = N(0)R(0;\varphi), \ \ t \geq -r.$$

For arbitrary $\epsilon > 0$, there exists a constant $\delta = \frac{\epsilon}{N(0)}$ such that $\|\varphi\| \leq R(0;\varphi) \leq \delta$. Then,

$$|y(t)| = N(0)R(0;\varphi) < N(0)\delta = \epsilon.$$

In addition, for $\lambda_0 < 0$, the inequality (37) guarantees that

$$\lim_{t\to\infty} y(t) = 0.$$

Hence, the zero solution of (32) is asymptotically stable when $\lambda_0 < 0$. □

According to Theorem 2 and Theorem 1, we can obtain the following theorem.

Theorem 4. *Assume that λ_0 is a real root of characteristic Equation* (33) *and satisfies* (34)*. Then, the solutions $x(t)$ of* (1)–(3) *and $\tilde{x}(t)$ of* (29)–(31) *satisfy*

$$\lim_{t\to\infty}\left[\mathrm{e}^{-\lambda_0 t}(x(t) - \tilde{x}(t))\right] = \frac{L(\lambda_0;\phi - \tilde{\phi})}{1+\beta(\lambda_0)}.$$

Using Theorem 2 and Theorem 3, we can obtain the following theorem.

Theorem 5. *Assume that λ_0 is a real root of the characteristic Equation* (32) *and satisfies $\mu(\lambda_0) < 1$. Then, the solution $x(t)$ of* (1)–(3) *and $\tilde{x}(t)$ of* (29)–(31) *satisfies*

$$|x(t) - \tilde{x}(t)| \leq N(\lambda_0)R(\lambda_0;\phi - \tilde{\phi})\mathrm{e}^{\lambda_0 t}, \forall t \geq 0,$$

where

$$N(\lambda_0) = \mu(\lambda_0) + k(\lambda_0)\left(\frac{1+\mu(\lambda_0)}{1+\beta(\lambda_0)}\right)$$

and

$$k(\lambda_0) = 1 + |b|\tau e^{-\lambda_0\tau} + |c|(1 + |\lambda_0|\sigma e^{-\lambda_0\sigma}).$$

Moreover, the solution of (1)–(3) *is described as follows:*

(i) The solution is stable if $\lambda_0 = 0$, or, equivalently, if the following conditions are satisfied:

$$a + b = 0, \quad |b|\tau + |c| < 1; \tag{38}$$

(ii) The solution is asymptotically stable if $\lambda_0 < 0$;
(iii) The solution is unstable if $\lambda_0 > 0$.

3.2. Asymptotical Stability of Euler's Method for IDDEs

To analyze the stability of the numerical method (4) for (1)–(3), we also consider the Euler method for IDDE (29)–(31), as follows:

$$\begin{cases} \tilde{X}_{n+1} = \tilde{X}_n + ah\tilde{X}_n + bh\tilde{X}_{n-m_2} - c\tilde{X}_{n-m_1+1} + c\tilde{X}_{n-m_1}, n \neq \eta_k - 1, n - m_1 \neq \eta_j - 1 \\ \tilde{Y}_k = \tilde{X}_n + ah\tilde{X}_n + bh\tilde{X}_{n-m_2} - c\tilde{X}_{n-m_1+1} + c\tilde{X}_{n-m_1}, n = \eta_k - 1, \\ \tilde{X}_{n+1} = \tilde{X}_n + ah\tilde{X}_n + bh\tilde{X}_{n-m_2} - c\tilde{Y}_j + c\tilde{X}_{n-m_1}, n - m_1 = \eta_j - 1, \\ \tilde{X}_{\eta_k} = \tilde{Y}_k + l_k, \\ \tilde{X}_i = \tilde{\phi}_i = \tilde{\phi}(ih), ih \in [-r, 0], i \in [-m, 0]. \end{cases} \tag{39}$$

Definition 4. *The Euler method for INDDE* (1)–(3) *is said to be stable if for every $\epsilon > 0$, there exists a number $\delta = \delta(\epsilon) > 0$ such that*

$$\max_{-m \leq i \leq 0} |\phi_i - \tilde{\phi}_i| < \delta,$$

which implies that

$$|X_n - \tilde{X}_n| < \epsilon, \quad \text{for all} \quad n \geq 0,$$

where X_n and $\tilde{X}_n$ are obtained from (4) *and* (39)*, respectively.*

The Euler method for INDDE (1)–(3) *is said to be asymptotically stable if it is stable and there exists a number $\delta_1 > 0$ such that $\max_{-m \leq i \leq 0} |\phi(ih) - \tilde{\phi}(ih)| < \delta_1$ implies*

$$\lim_{n \to \infty} |X_n - \tilde{X}_n| = 0.$$

Denote $y_n = \delta X_n = X_n - \tilde{X}_n$,$n \geq -m$, $\delta\phi_i = \phi(ih) - \tilde{\phi}(ih)$, $i = -m, \cdots, 0$. It is very interesting that we can obtain the following neat difference equation from (4) and (39):

$$y_{n+1} = y_n + ahy_n + bhy_{n-m_2} - cy_{n-m_1+1} + cy_{n-m_1}, \forall n \in \mathbb{N}, \tag{40}$$

$$y_i = \delta\phi_i, \ i = -m, \cdots, 0. \tag{41}$$

The characteristic equation of (40) is

$$(\lambda - 1)(1 + c\lambda^{-m_1}) = ha + hb\lambda^{-m_2}. \tag{42}$$

Applying [26] (Theorem 1) to the differential Equations (40) and (41), we can obtain the following theorem of the Euler method for INDDEs.

Theorem 6. *Assume that λ_1 is a positive root of characteristic Equation* (42) *and satisfies*

$$\mu_{\lambda_1} = |c| - m_1\lambda_1^{-m_1}|c||1 - \frac{1}{\lambda_1}| + \frac{1}{\lambda_1}hm_2|b|\lambda_1^{-m_2} < 1. \tag{43}$$

Then, the solutions X_n obtained from (4) *and $\tilde{X}_n$ obtained from* (39) *satisfy*

$$\lim_{n\to\infty} \lambda_1^{-n}|X_n - \tilde{X}_n| = \frac{L_{\lambda_1}(\phi)}{1+\gamma_{\lambda_1}},$$

where

$$L_{\lambda_1}(\phi) = \phi_0 + c\phi_{-m_1} - (1 - \frac{1}{\lambda_1})c\lambda_1^{-m_1}\sum_{s=-m_1}^{-1}\lambda_1^{-s}\phi_s + \frac{1}{\lambda_1}hb\lambda_1^{-m_2}\sum_{s=-m_2}^{-1}\lambda_1^{-s}\phi_s$$

and

$$\gamma_{\lambda_1} = \sum c\left(1 - (1 - \frac{1}{\lambda_1})m_1\right)\lambda_1^{-m_1} + \frac{1}{\lambda_1}bm_2\lambda_1^{-m_2}.$$

Similarly, applying [26] (Theorem 2) to (40) and (41), we can obtain the following theorem.

Theorem 7. *Assume that λ_1 is a real root of characteristic Equation* (42) *and satisfies $\mu_{\lambda_1} < 1$. Then, the solutions X_n obtained from* (4) *and $\tilde{X}_n$ obtained from* (39) *satisfy*

$$|X_n - \tilde{X}_n| \le N_{\lambda_1}\|\phi\|\lambda_1^n, \ \ \forall n \ge 0,$$

where

$$N_{\lambda_1} = \frac{1+\mu_{\lambda_1}}{1+\gamma_{\lambda_1}} + \left(1 + \frac{1+\mu_{\lambda_1}}{1+\gamma_{\lambda_1}}\right)\mu_{\lambda_1}\max\{1, \lambda_1^r\}.$$

Moreover, the Euler method (4) *for INDDE* (1)–(3) *is described as follows:*

(i) The solution is stable if $\lambda_1 = 1$, or, equivalently, if the following conditions are satisfied:

$$a + b = 0, \ \ |c| + h|b|m_2 < 1; \tag{44}$$

(ii) The solution is asymptotically stable if $\lambda_1 < 1$;
(iii) The solution is unstable if $\lambda_1 > 1$.

Corollary 1. *If the condition* (38) *holds, the Euler method* (4) *preserves the stable property of INDDE* (1)–(3) *without additional restrictions on the stepsize.*

Proof. Using Theorem 5, we find that the solution $x(t)$ of INDDE (1)–(3) is stable. Because $m_2 = \lfloor \frac{\tau}{h} \rfloor$, it is easy to conclude that $m_2 h \le \tau$. Hence, the condition (38) holds, implying that (44) holds. Based on Theorem 7, we can affirm that the Euler method (4) for INDDE (1)–(3) is also stable. □

4. Numerical Examples

In this section, two examples are given to illustrate the conclusions of this paper.

Example 1. *Consider the following INDDE:*

$$\left(x(t) - \frac{1}{3\mathrm{e}}x(t - \frac{1}{2})\right)' = \frac{1}{3}x(t) - \frac{1}{\mathrm{e}}x(t - \frac{1}{5}), t \ge 0, t \ne k, \tag{45}$$

$$\Delta x(\tau_k) = \left(-\frac{1}{\mathrm{e}}\right)^k, \ \tau_k = k, \ k \in \mathbb{Z}^+, \tag{46}$$

$$x(t) = \phi(t), \ -\frac{1}{2} \leq t \leq 0. \tag{47}$$

The characteristic equation of (45) *is*

$$\lambda\left(1 - \frac{1}{3\mathrm{e}}\right) = \frac{1}{3} - \frac{1}{\mathrm{e}}\mathrm{e}^{-\frac{\lambda}{5}}. \tag{48}$$

Solving (48)*, we find that* $\lambda_1 \approx -3.94$ *and* $\lambda_2 \approx -0.043$*. Let* $\lambda_0 = \lambda_2$*, which implies that* $\mu(\lambda_0) < 1$*. Using Theorem 5, we can conclude that the exact solution of* (45)–(47) *is asymptotically stable in the sense of Definition 2.*

Let $h = \frac{1}{i}$*,* $i \in \mathbb{Z}^+$*, with i being divisible by* 10*. The characteristic Equation* (42) *of* (40) *for* (45) *is then changed into*

$$(\lambda - 1)(1 - \frac{\lambda^{-\frac{i}{2}}}{3\mathrm{e}}) = \frac{1}{3i} - \frac{\lambda^{-\frac{i}{5}}}{i\mathrm{e}}. \tag{49}$$

When $i = 10$*,* (49) *is changed into* $f(\lambda) = 0$*, where*

$$f(\lambda) = (\lambda - 1)(1 - \frac{\lambda^{-5}}{3\mathrm{e}}) - \frac{1}{30} + \frac{\lambda^{-2}}{10\mathrm{e}}. \tag{50}$$

A root of $f(\lambda) = 0$ *is* $\lambda_1 \approx 0.995685180437323 < 1$*. It is easy to verify that* $\mu_{\lambda_1} < 1$*. Applying Theorem 7, we can conclude that the Euler method* (4) *for* (45)–(47) *is asymptotically stable for* $h = \frac{1}{10}$ *(see Figure 1).*

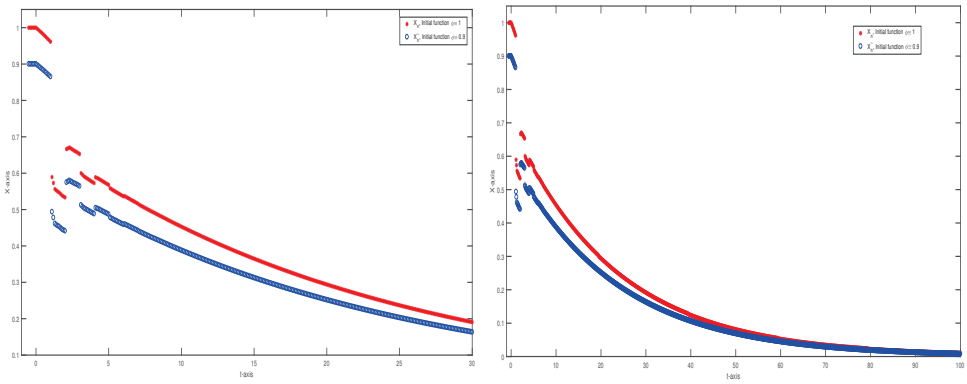


Figure 1. The Euler method (4) for (45)–(47), when $h = \frac{1}{10}$.

In Table 1, the global errors at $t = \frac{6}{5}$ *and* $t = \frac{7}{5}$ *of the Euler method* (4) *for* (45)–(47) *are represented by* $e_{6/5}$ *and* $e_{7/5}$*, respectively. As can be seen from the ratios in this table, when the stepsizes are halved, the global errors become about half of the originals, which roughly shows that the Euler method* (4) *for* (45)–(47) *is convergent of order* 1*.*

Table 1. The global errors of the Euler scheme for INDDE (45)–(47).

Stepsize	$e_{6/5}$	**Ratio**	$e_{7/5}$	**Ratio**
1/20	0.0066590383		$6.0532924880 \times 10^{-4}$	
1/40	0.0033576411	0.5042231270	$3.3600485518 \times 10^{-4}$	0.5550778454
1/80	0.0016859400	0.5021203734	$1.7654071490 \times 10^{-4}$	0.52541120220
1/160	$8.4476115143 \times 10^{-4}$	0.5010624028	$9.0430888108 \times 10^{-5}$	0.5122381438
1/320	$4.2282978351 \times 10^{-4}$	0.5005317572	$4.5758856992 \times 10^{-5}$	0.5060091519

According to Figure 2, the numerical solution obtained from (4) *for* (45)–(47) *does not make additional jumps, but rather only jumps near* $t = k$*,* $k \in \mathbb{N}$*, which is consistent with the nature of the exact solution (see (a) of Definition 1).*

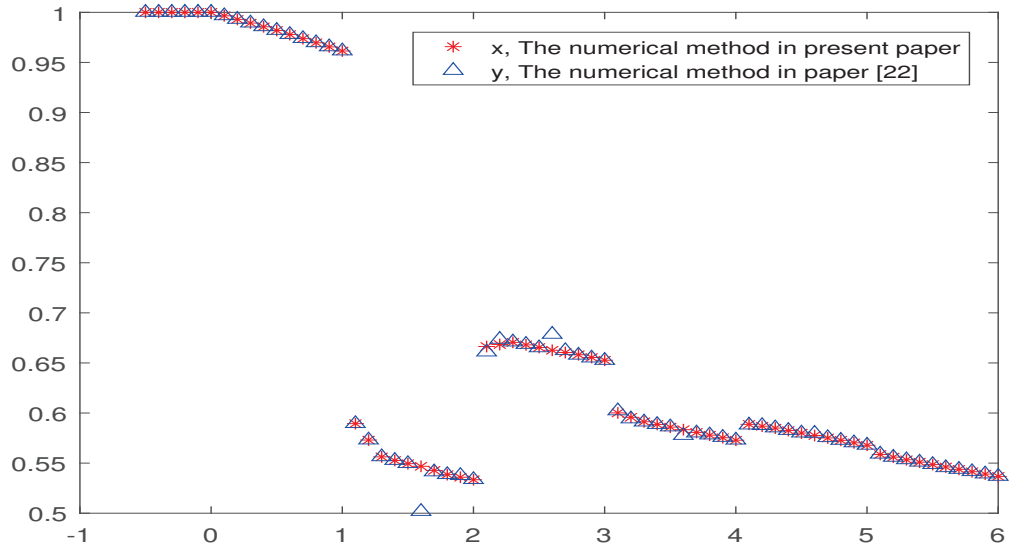


Figure 2. Comparison of the numerical solution obtained from (4) for (45)–(47) in this paper with that obtained from the numerical format constructed in Ref. [22], when $h = \frac{1}{10}$.

Example 2. *Consider the following INDDE:*

$$\left(x(t) + \frac{1}{2}x(t-1)\right)' = x(t) + \frac{1}{2}x(t-1), t \geq 0, t \neq 2k \tag{51}$$

$$\Delta x(\tau_k) = k,\ \tau_k = 2k,\ k \in \mathbb{Z}^+, \tag{52}$$

$$x(t) = \phi(t),\ -1 \leq t \leq 0. \tag{53}$$

The characteristic equation of (51) *is*

$$\lambda\left(2 + \mathrm{e}^{-\lambda}\right) = 2 + \mathrm{e}^{-\lambda}, \tag{54}$$

which implies that $\lambda = 1$ *is the unique real root of* (54). *For* $\lambda_0 = \lambda = 1$,

$$\mu(\lambda_0) = \frac{1}{2\mathrm{e}} + \frac{1}{\mathrm{e}} < 1.$$

Consequently, the exact solution of (51)–(53) *is unstable, based on Theorem 3.*

Let $h = \frac{1}{i}$, $i \in \mathbb{Z}^+$, *with i being divisible by* 10. *The characteristic Equation* (42) *of* (40) *for* (51) *is then changed into*

$$(\lambda - 1)(1 + \frac{\lambda^{-i}}{2}) = \frac{1}{i} + \frac{\lambda^{-i}}{2i}. \tag{55}$$

When $i = 10$, (55) *is changed into* $g(\lambda) = 0$, *where*

$$g(\lambda) = (\lambda - 1)(1 + \frac{\lambda^{-10}}{2}) - \frac{1}{10} - \frac{\lambda^{-10}}{20}. \tag{56}$$

A root of $g(\lambda) = 0$ *is* $\lambda_1 \approx 1.1 > 1$. *It is easy to verify that* $\mu_{\lambda_1} < 1$. *Applying Theorem 7, we can affirm that the Euler method* (4) *for* (51)–(53) *is unstable for* $h = \frac{1}{10}$ *(see Figure 3).*

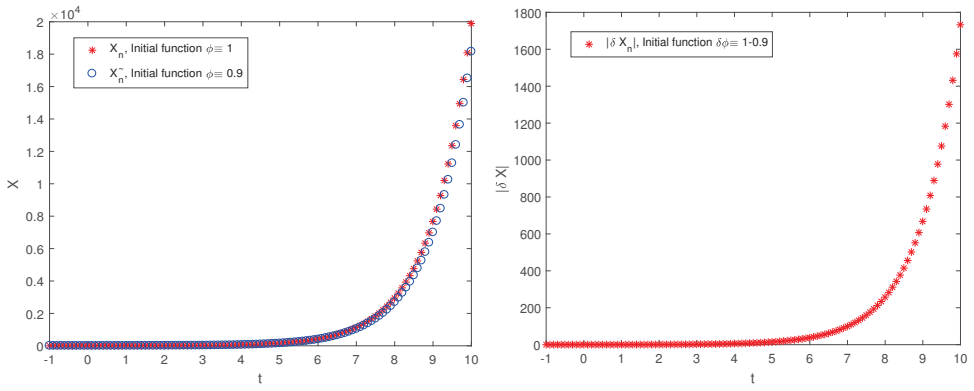


Figure 3. The Euler method (4) for (45)–(47), when $h = \frac{1}{10}$.

In Table 2, the global errors at $t = 3$ and $t = 4$ of the Euler method (4) *for* (51)–(53) *are represented by e_3 and e_4, respectively. As can be seen from the ratios in this table, when the stepsizes are halved, the global errors decrease to about half of the originals, which roughly shows that the Euler method* (4) *for* (51)–(53) *is convergent of order* 1.

Table 2. The global errors of the Euler scheme for INDDE (51)–(53).

Stepsize	e_3	**Ratio**	e_4	**Ratio**
1/100	0.0134679990		0.0730382471	
1/200	0.0067647055	0.5022799238	0.0367309910	0.5029007739
1/400	0.0033900841	0.5011428898	0.0184189152	0.5014543502
1/800	0.0016969818	0.5005721798	0.0092228697	0.5007281696
1/1600	$8.4897669001 \times 10^{-4}$	0.5002862740	0.0046147951	0.5003643339

5. Conclusions and Future Work

In this research, we have introduced a new numerical scheme based on the Euler method for solving linear impulsive neutral differential equations. It is rigorously proven that the constructed numerical method is convergent of order 1. Additionally, we have determined the conditions under which the numerical solutions maintain the asymptotic stability of the exact solutions.

Overall, we find that the numerical methods constructed in this article are only convergent of order 1. Future work will focus on developing higher-order numerical methods for INDDEs.

Author Contributions: Methodology, G.-L.Z.; Writing—original draft, G.-L.Z., Y.S. and Y.-X.Z.; Writing—review & editing, G.-L.Z. and C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (No. 11701074) and Hebei Natural Science Foundation (No. A2020501005).

Data Availability Statement: The datasets generated during the current study are available from the corresponding author on reasonable request.

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

References

1. Faria, T.; Oliveira, J.J. General criteria for asymptotic and exponential stabilities of neural network models with unbounded delays. *Appl. Math. Comput.* **2011**, *217*, 9646–9658. [CrossRef]
2. Song, X.; Xin, X.; Huang, W. Exponential stability of delayed and impulsive cellular neural networks with partially Lipschitz continuous activation functions. *Neural Netw.* **2012**, *29–30*, 80–90. [CrossRef] [PubMed]
3. Yang, Z.; Xu, D. Stability analysis of delay neural networks with impulsive effects. *IEEE Trans. Circuits Syst.-II Express Briefs* **2005**, *52*, 517–521. [CrossRef]

4. Li, W.; Huo, H. Global attractivity of positive periodic solutions for an impulsive delay periodic model of respiratory dynamics. *J. Comput. Appl. Math.* **2005**, *174*, 227–238. [CrossRef]
5. Lee, H.J.; Park, J.B.; Joo, Y.H. Robust control for uncertain Takagi-Sugeno fuzzy with time-varying input delay. *ASME J. Dyn. Syst. Meas. Control* **2005**, *127*, 302–306. [CrossRef]
6. Li, C.; Sun, J. Stability analysis of nonlinear stochastic differential delay systems under impulsive control. *Phys. Lett. A* **2010**, *374*, 1154–1158. [CrossRef]
7. Wu, K.; Ding, X. Stability and stabilization of impulsive stochastic delay differential equations. *Math. Probl. Eng.* **2012**, 176375. [CrossRef]
8. Hernández, E. Global solutions for abstract impulsive neutral differential equations. *Math. Comput. Model.* **2011**, *53*, 196–204. [CrossRef]
9. Lakrib, M. Existence of solutions for impulsive neutral functional differential equations with multiple delays. *Electron. J. Differ. Equ.* **2008**, *2008*, 1–7.
10. Li, M. Existence results for nondensely defined impulsive neutral functional differential equations with infinite delay. *Nonlinear Anal. Hybrid Syst.* **2011**, *5*, 502–512. [CrossRef]
11. Ye, R. Existence of solutions for impulsive partial neutral functional differential equation with infinite delay. *Nonlinear Anal. Theory Methods Appl.* **2010**, *73*, 155–162. [CrossRef]
12. Sun, X.; Huo, H.; Ma, C. Periodic solutions of a class of impulsive neutral delay differential equation. *Appl. Math. Comput.* **2012**, *219*, 3947–3955. [CrossRef]
13. Duan, Y.; Tian, P.; Zhang, S. Oscillation and stability of nonlinear neutral impulsive delay differential equations. *J. Comput. Appl. Math.* **2003**, *11*, 243–253. [CrossRef]
14. Graef, J.R.; Shen, J.H.; Stavroulakis, I.P. Oscillation of impulsive neutral delay differential equations. *J. Math. Anal. Appl.* **2002**, *268*, 310–333. [CrossRef]
15. Luo, Z.; Shen, J. Oscillation for solutions of nonlinear neutral differential equations with impulses. *Comput. Math. Appl.* **2001**, *42*, 1285–1292. [CrossRef]
16. Li, X.; Deng, F. Razumikhin method for impulsive functional differential equations of neutral type. *Chaos Solitons Fractals* **2017**, *101*, 41–49. [CrossRef]
17. Bainov, D.D.; Stamova, I.M. Uniform asymptotic stability of impulsive differential-difference equations of neutral type by Lyapunov's direct method. *J. Comput. Appl. Math.* **1995**, *62*, 359–369. [CrossRef]
18. Xu, D.; Yang, Z.; Yang, Z. Exponential stability of nonlinear impulsive neutral differential equations with delays. *Nonlinear Anal. Theory Methods Appl.* **2007**, *67*, 1426–1439. [CrossRef]
19. Xu, L.; Xu, D. Exponential stability of nonlinear impulsive neutral integro-differential equations. *Nonlinear Anal.* **2008**, *69*, 2910–2923. [CrossRef]
20. Zhang, G.; Wang, Z.; Sun, Y.; Liu, T. Asymptotical stability criteria for exact solutions and numerical solutions of nonlinear impulsive neutral delay differential equations. *Axioms* **2023**, *12*, 988. [CrossRef]
21. Zhang, G.; Sun, Y.; Wang, Z. Asymptotical stability of the exact solutions and the numerical solutions for impulsive neutral differential equations. *Comput. Appl. Math.* **2024**, *43*, 8. [CrossRef]
22. Sun, Y.; Zhang, G.; Wang, Z.; Liu, T. Convergence of the Euler method for impulsive neutral delay differential equations. *Mathematics* **2023**, *11*, 4684. [CrossRef]
23. Yeniçerioğlu, A.F. Stability of linear impulsive neutral differential equations with constant coefficients. *J. Math. Anal. Appl.* **2019**, *479*, 2196–2213. [CrossRef]
24. Philos, C.G.; Purnaras, I.K. Purnaras, Periodic first order linear neutral delay differential equations. *Appl. Math. Comput.* **2001**, *117*, 203–222.
25. Philos, C.G.; Purnaras, I.K. On the behavior of the solutions for certain first order linear autonomous functional differential equations. *Rocky Mountain J. Math.* **2006**, *36*, 1999–2019. [CrossRef]
26. Kordonis, I.G.; Philos, C.G. On the behavior of the solutions for linear autonomous neutral delay difference equations. *J. Differ. Equ. Appl.* **1999**, *5*, 219–233. [CrossRef]

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

mathematics

MDPI

Article

Nonlinear Complex Wave Excitations in (2+1)-Dimensional Klein–Gordon Equation Investigated by New Wave Transformation

Guojiang Wu [1], Yong Guo [2,*] and Yanlin Yu [1,*]

[1] School of Science, Kaili University, Kaili 556000, China; gjwu5221@163.com
[2] Institute of Plasma Physics, Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, China
* Correspondence: yguo@ipp.ac.cn (Y.G.); yuyanlin@kluniv.edu.cn (Y.Y.)

Abstract: The Klein–Gordon equation plays an important role in mathematical physics, such as plasma and, condensed matter physics. Exploring its exact solution helps us understand its complex nonlinear wave phenomena. In this paper, we first propose a new extended Jacobian elliptic function expansion method for constructing rich exact periodic wave solutions of the (2+1)-dimensional Klein–Gordon equation. Then, we introduce a novel wave transformation for constructing nonlinear complex waves. To demonstrate the effectiveness of this method, we numerically simulated several sets of complex wave structures, which indicate new types of complex wave phenomena. The results show that this method is simple and effective for constructing rich exact solutions and complex nonlinear wave phenomena to nonlinear equations.

Keywords: (2+1)-dimensional Klein–Gordon equation; Jacobian elliptic function; auxiliary equation; nonlinear evolution equation; complex wave structure

MSC: 37N15; 37N30; 35Q40; 35Q51

1. Introduction

Under specific approximate conditions, many nonlinear wave phenomena can be expressed as nonlinear mathematical problems in the form of nonlinear evolution equations (NLEEs). In research in fields such as physics and engineering, many well-known NLEEs have been developed to explain the dynamics of some nonlinear waves [1–7]. Therefore, it is very important to obtain the exact solutions of these NLEEs for understanding the spatiotemporal dynamics of physics phenomena. In order to obtain the exact traveling wave solutions of the NLEEs, a number of methods have been proposed, such as the F-expansion method [8,9], tanh–sech method and its extension [10,11], Jacobi elliptic function method [12–15], auxiliary equation method in refs. [15–19], (G′/G)-expansion method and its extension [20–23] and so on. These methods can basically provide a large number of exact solutions when dealing with certain types of NLEEs. However, there are still some issues that need further research in the field of nonlinear science. For example, developing a simple and universal method to construct complex and diverse exact analytical solutions and nonlinear wave structures for NLEEs is a development direction in nonlinear physics research. In ref. [24], an extended Jacobi elliptic function method is proposed to solve the (2+1)-dimensional asymmetric Nizhnik–Novikov–Veselov (aNNV) equation. A large number of new types of exact Jacobian elliptic function solutions have been obtained and various nonlinear wave structures are constructed through different arbitrary wave transformations. However, this type of arbitrary wave transformation only exists as a traveling wave solution in most nonlinear systems, and there are still other forms of wave transformations and wave structures in these nonlinear systems, which is the creative purpose of this paper.

Among the NLEEs, the Klein–Gordon equation plays an important role in nonlinear mathematical physics, such as plasma electromagnetic interactions, the relativistic hydrogen spectrum, coulomb scattering, nonlinear optics, solid state physics, quantum field theory, etc. [25–27]. In this paper, we consider the following (2+1)-dimensional Klein–Gordon equation [28,29]:

$$u_{xx} + u_{yy} - u_{tt} + \alpha u - \beta u^3 = 0, \tag{1}$$

Equation (1) and its general formula have been solved by various methods, and a large number of exact solutions have been obtained [15,28–31]. There are still some other solutions applied to Klein–Gordon-type equations, such as the solitonic, rogue wave and lump wave solutions in Ref. [32], the doubly periodic function solutions in ref. [33] and the hyperbolic, trigonometric and rational functions solutions in ref. [34]. Although these methods are effective in solving (2+1)-dimensional Klein–Gordon equations, there are still some new types of solutions and nonlinear wave structures to be explored. The main work of this paper is to apply a new extended Jacobian elliptic function expansion method to solve the (2+1)-dimensional Klein–Gordon equation. As a result, we have obtained a large number of new and more general solutions of the (2+1)-dimensional Klein–Gordon equation, which may provide useful help for physicists studying more complicated physical phenomena. Then, we obtained a large number of complex wave structures through a new nonlinear wave construction method.

The structure of the manuscript is as follows: In Section 2, many new solutions of the Jacobian elliptic equation were constructed through two transformation relationships. In Section 3, a detailed introduction is given on how to use this scheme to handle the (2+1)-dimensional Klein–Gordon equation, and general expressions of the solutions were obtained. In Section 4, a set of Jacobian elliptic function solutions is used as an example to prove that this method is powerful and effective. Multiple complex wave structures have been constructed using a new nonlinear wave construction method in Section 5. Finally, Section 6 is the conclusion and discussion.

2. New Jacobian Elliptic Function Solutions of Elliptic Equation

We consider the following form of elliptic equation,

$$\left[g'(\xi)\right]^2 = pg^4(\xi) + qg^2(\xi) + r, \tag{2}$$

where p, q and r are undetermined constants. Equation (2) has Jacobian elliptic function solutions as shown in Table 1, where $i^2 = -1$. It should be pointed out that in the first column of the table, $sn(\xi)$, $cd(\xi) = cn(\xi)/dn(\xi)$ mean $g(\xi) = sn(\xi)$, and $g(\xi) = cd(\xi) = cn(\xi)/dn(\xi)$.

Table 1. The Jacobi elliptic function solutions of Equation (2).

$g(\xi)$	p	q	r
$sn(\xi)$, $cd(\xi) = cn(\xi)/dn(\xi)$	m^2	$-(1+m^2)$	1
$cn(\xi)$	$-m^2$	$-1+2m^2$	$1-m^2$
$dn(\xi)$	-1	$2-m^2$	$-1+m^2$
$ns(\xi) = \frac{1}{sn(\xi)}$, $dc(\xi) = dn(\xi)/cn(\xi)$	1	$-(1+m^2)$	m^2
$nc(\xi) = 1/\ cn(\xi)$	$1-m^2$	$-1+2m^2$	$-m^2$
$nd(\xi) = 1/\ dn(\xi)$	$-1+m^2$	$2-m^2$	-1
$cs(\xi) = cn(\xi)/sn(\xi)$	1	$2-m^2$	$1-m^2$
$sc(\xi) = sn(\xi)/cn(\xi)$	$1-m^2$	$2-m^2$	1

Table 1. *Cont.*

$sd(\xi) = sn(\xi)/dn(\xi)$	$m^2(-1+m^2)$	$-1+2m^2$	1
$ds(\xi) = dn(\xi)/sn(\xi)$	1	$-1+2m^2$	$m^2(-1+m^2)$
$m\,cn(\xi) \pm dn(\xi)$	$-1/4$	$(1+m^2)/2$	$-(1-m^2)^2/4$
$ns(\xi) \pm cs(\xi)$, $cn(\xi)/\left(\sqrt{1-m^2}sn(\xi) \pm dn(\xi)\right)$ $msn(\xi) \pm idn(\xi)$, $sn(\xi)/(1 \pm cn(\xi)$	$1/4$	$(1-2m^2)/2$	$1/4$
$nc(\xi) \pm sc(\xi)$, $cn(\xi)/(1 \pm sn(\xi))$	$(1-m^2)/4$	$(1+m^2)/2$	$(1-m^2)/4$
$ns(\xi) \pm ds(\xi)$	$1/4$	$(-2+m^2)/2$	$m^4/4$
$sn(\xi) \pm icn(\xi)$, $dn(\xi)/\left(\sqrt{m^2-1}sn(\xi) \pm cn(\xi)\right)$	$m^2/4$	$(-2+m^2)/2$	$m^2/4$
$dn(\xi)/(\sqrt{\frac{m^2-1}{m^2}} \pm cn(\xi))$	$\frac{1}{4m^2}$	$(1-2m^2)/2$	$m^2/4$
$sn(\xi)/(1 \pm dn(\xi))$	$m^2/4$	$(-2+m^2)/2$	$1/4$
$dn(\xi)/(1 \pm msn(\xi))$	$(-1+m^2)/4$	$(1+m^2)/2$	$(-1+m^2)/4$
$sn(\xi)/(cn(\xi) \pm dn(\xi))$	$(1-m^2)^2/4$	$(1+m^2)/2$	$1/4$
$cn(\xi)/\left(\sqrt{1-m^2} \pm dn(\xi)\right)$	$m^4/4$	$(-2+m^2)/2$	$1/4$
$sn(\xi)/(cn(\xi)dn(\xi))$	$(1-m^2)^2$	$2(1+m^2)$	1
$cn(\xi)dn(\xi)/\,sn(\xi)$	1	$2(1+m^2)$	$(1-m^2)^2$
$\frac{cn(\xi)}{sn(\xi)dn(\xi)}$, $\mathrm{sn}(\xi)dn(\xi)/cn(\xi)$	1	$2(1-2m^2)$	1
$sn(\xi)cn(\xi)/dn(\xi)$	m^4	$2(-2+m^2)$	1
$dn(\xi)/(sn(\xi)cn(\xi))$	1	$2(-2+m^2)$	m^4

To construct new solutions for Equation (2), we introduce the elliptic equation shown below,

$$\left[f'(\xi)\right]^2 = p_1 f^4(\xi) + q_1 f^2(\xi) + r_1, \tag{3}$$

where p_1, q_1 and r_1 are undetermined constants. Next, we will solve Equation (3) through two different transformation relationships. Firstly, we assume that $f(\xi)$ and $g(\xi)$ satisfy the following relationship,

Case 1

$$f(\xi) = \frac{g(\xi)}{a_0 g^2(\xi) + a_1 g'(\xi) + a_2}, \tag{4}$$

where a_0, a_1 and a_2 are constants that are not all zero at the same time. Substituting Equation (4) into Equation (3) and solving the resulting system of equations, we can obtain the following families of equations.

Family 1

$$p_1 = r, q_1 = q, r_1 = p,\ a_0 = 1,\ a_1 = 0,\ a_2 = 0. \tag{5}$$

Family 2

$$p_1 = p, q_1 = q, r_1 = r,\ a_0 = 0,\ a_1 = 0,\ a_2 = 1. \tag{6}$$

Family 3

$$p_1 = q^2 - 4pr, q_1 = -2q, r_1 = 1,\ a_0 = 0,\ a_1 = 1,\ a_2 = 0. \tag{7}$$

Family 4

$$p_1 = 8r \pm 4q\sqrt{\frac{r}{p}}, q_1 = q \pm 6\sqrt{pr}, r_1 = p,\ a_0 = 1,\ a_1 = 0,\ a_2 = \mp\sqrt{\frac{r}{p}}. \tag{8}$$

Family 5

$$p_1 = pr - \frac{3}{4}q^2 \pm 3q\sqrt{pr}, q_1 = \frac{q}{2} \pm 3\sqrt{pr}, r_1 = \frac{1}{4},\ a_0 = \pm\sqrt{p},\ a_1 = 1,\ a_2 = \pm\sqrt{r}. \tag{9}$$

We again assume that $f(\xi)$ and $g(\xi)$ satisfy the following relationship:

Case 2

$$f(\xi) = \sqrt{ag^2(\xi) + bg'(\xi) + c}. \tag{10}$$

By solving this case, we can obtain the following relationship equations,

Family 6

$$a = \pm b\sqrt{p}, c = -b\sqrt{r},\ p_1 = \pm\frac{\sqrt{p}}{2b},\ q_1 = \frac{1}{4}(q \pm 6\sqrt{pr}),\ r_1 = \frac{(q\sqrt{r} \pm 2r\sqrt{p})b}{2},\ b \neq 0. \tag{11}$$

Family 7

$$a = \pm b\sqrt{p}, c = \pm\frac{bq}{2\sqrt{p}},\ p_1 = \pm\frac{\sqrt{p}}{2b},\ q_1 = -\frac{q}{2},\ r_1 = \pm\frac{(q^2 - 4pr)b}{8\sqrt{p}},\ b \neq 0. \tag{12}$$

By using the two transformation relationships of $f(\xi)$ and $g(\xi)$ mentioned above, we obtain the solutions of Equations (4) and (10) for two sets of general expressions of Equation (3). Through Table 1, different forms of complex Jacobian elliptic function solutions can be obtained.

3. Method and Application to the (2+1)-Dimensional Klein–Gordon Equation

We assume Equation (1) has the following nonlinear wave solution,

$$u(x, y, t) = u(\xi),\ \ \xi = \Phi(x, y, t), \tag{13}$$

where $\Phi(x, y, t)$ is an arbitrary wave function about x, y and t. Substituting Equation (13) into Equation (1) yields

$$\left(\Phi_x{}^2 + \Phi_y{}^2 - \Phi_t{}^2\right)u'' + \left(\Phi_{xx} + \Phi_{yy} - \Phi_{tt}\right)u' + \alpha u - \beta u^3 = 0. \tag{14}$$

where u', Φ_x, Φ_y and Φ_t mean $\frac{du(\xi)}{d\xi}$, $\frac{\partial\Phi(x,y,t)}{\partial x}$, $\frac{\partial\Phi(x,y,t)}{\partial y}$ and $\frac{\partial\Phi(x,y,t)}{\partial t}$. Then, according to the method in Ref. [24], we assume Equation (14) has the following formal solutions,

$$u(\xi) = a_0 + a_1 f(\xi) + b_1\frac{1}{f(\xi)} + c_1\frac{f'(\xi)}{f(\xi)}, \tag{15}$$

where a_0, a_1, b_1 and c_1 are constants determined later, and f (ξ) represents the solutions of the Equation (3). Substituting Equation (15) into Equation (14) and setting the coefficients of $f^i(\xi)$ and $f^i(\xi)f'(\xi)$ to zero yields a set of algebraic equations about a_0, a_1, b_1, c_1, Φ_x, Φ_y, Φ_t, Φ_{xx}, Φ_{yy} and Φ_{tt}. Solving the resulting equations, the following sets of coefficients can be obtained:

Set 1

$$a_0 = 0,\ a_1 = \pm\sqrt{-\frac{2p_1\alpha}{q_1\beta}},\ b_1 = 0,\ c_1 = 0,\ \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0,\ {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{q_1}. \tag{16}$$

Set 2

$$a_0 = 0,\ a_1 = 0,\ b_1 = \pm\sqrt{-\frac{2r_1\alpha}{q_1\beta}},\ c_1 = 0,\ \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0,\ {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{q_1}. \tag{17}$$

Set 3

$$a_0 = 0,\ a_1 = 0,\ b_1 = 0,\ c_1 = \pm\sqrt{\frac{\alpha}{q_1\beta}},\ \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0,\ {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{2q_1}. \tag{18}$$

Set 4

$$a_0 = 0, \quad a_1 = \pm\sqrt{\frac{2p_1\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}},\ b_1 = \pm\sqrt{\frac{2r_1\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}},\ c_1 = 0,\ \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0,\ {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{q_1 \pm 6\sqrt{p_1r_1}}. \tag{19}$$

Set 5

$$a_0 = 0,\ a_1 = \pm\sqrt{\frac{p_1\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}},\ b_1 = \pm\sqrt{\frac{r_1\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}},\ c_1 = \pm\sqrt{\frac{\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}},\ \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0,\ {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{2\alpha}{q_1 \pm 6\sqrt{p_1r_1}}. \tag{20}$$

From the solution set obtained above, we can obtain the general expression of the Jacobian elliptic function solutions of the (2+1)-dimensional Klein–Gordon equation in the following forms:

$$u_1(\xi) = \pm\sqrt{-\frac{2p_1\alpha}{q_1\beta}}f(\xi), \tag{21}$$

where $\xi = \Phi(x, y, t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{q_1}$.

$$u_2(\xi) = \pm\sqrt{-\frac{2r_1\alpha}{q_1\beta}}\frac{1}{f(\xi)}, \tag{22}$$

where $\xi = \Phi(x, y, t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{q_1}$.

$$u_3(\xi) = \pm\sqrt{\frac{\alpha}{q_1\beta}}\frac{f'(\xi)}{f(\xi)}, \tag{23}$$

where $\xi = \Phi(x, y, t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{2q_1}$.

$$u_4(\xi) = \pm\sqrt{\frac{2p_1\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}}f(\xi) \pm \sqrt{\frac{2r_1\alpha}{(q_1 \pm 6\sqrt{p_1r_1})\beta}}\frac{1}{f(\xi)}, \tag{24}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{q_1 \pm 6\sqrt{p_1 r_1}}$.

$$u_5(\xi) = \pm\sqrt{\frac{p_1\alpha}{(q_1 \pm 6\sqrt{p_1 r_1})\beta}} f(\xi) \pm \sqrt{\frac{r_1\alpha}{(q_1 \pm 6\sqrt{p_1 r_1})\beta}} \frac{1}{f(\xi)} \pm \sqrt{\frac{\alpha}{(q_1 \pm 6\sqrt{p_1 r_1})\beta}} \frac{f'(\xi)}{f(\xi)}, \tag{25}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{2\alpha}{q_1 \pm 6\sqrt{p_1 r_1}}$.

The solutions of Equations (21)–(25) are the general form of the (2+1)-dimensional Klein–Gordon equation expressed by $f(\xi)$, and the value of $f(\xi)$ is determined by Equation (3). Through Equations (2)–(12), p_1, q_1 and r_1 in Equation (3) can be expressed by p, q and r in Equation (2). Then, according to p, q and r in Table 1, the various Jacobian elliptic function solutions of the (2+1)-dimensional Klein–Gordon equation can be obtained. These five sets of solutions contain a large number of Jacobi elliptic function solutions, trigonometric function solutions when the modulus $m = 0$ and hyperbolic function solutions when the modulus $m = 1$. These solutions are generally new exact solutions to the (2+1)-dimensional Klein– Gordon equation, which have not been found in other literature.

4. Example of a Set of Solutions

In the following, we will provide a set of examples to demonstrate the power and effectiveness of this method. According to Table 1, if we select $p = m^2$, $q = -\left(1 + m^2\right)$ and $r = 1$, $g(\xi\ ) = cn(\xi)/dn(\xi)$, from Equation (5), we can obtain

$$p_1 = 1, q_1 = -\left(1 + m^2\right), r_1 = m^2,\ a_0 = 1,\ a_1 = 0,\ a_2 = 0. \tag{26}$$

Substituting Equation (26) into Equations (21)–(25), we can obtain the following Jacobian elliptic function solutions of the (2+1)-dimensional Klein–Gordon equation:

$$u_{11}(\xi) = \pm\sqrt{\frac{2\alpha}{(1+m^2)\beta}} \frac{dn(\xi)}{cn(\xi)}, \tag{27}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{1+m^2}$.

$$u_{12}(\xi) = \pm\sqrt{\frac{2m^2\alpha}{(1+m^2)\beta}} \frac{cn(\xi)}{dn(\xi)}, \tag{28}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{1+m^2}$.

$$u_{13}(\xi) = \pm\sqrt{\frac{\alpha}{(1+m^2)\beta}} \frac{\left(m^2-1\right)sn(\xi)}{cn(\xi)dn(\xi)}, \tag{29}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{2(1+m^2)}$.

$$u_{14}(\xi) = \pm\sqrt{\frac{2\alpha}{[-(1+m^2) \pm 6m]\beta}} \frac{dn(\xi)}{cn(\xi)} \pm \sqrt{\frac{2m^2\alpha}{[-(1+m^2) \pm 6m]\beta}} \frac{cn(\xi)}{dn(\xi)}, \tag{30}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{-(1+m^2) \pm 6m}$.

$$u_{15}(\xi) = \pm\sqrt{\frac{\alpha}{[-(1+m^2) \pm 6m]\beta}} \frac{dn(\xi)}{cn(\xi)} \pm \sqrt{\frac{2m^2\alpha}{[-(1+m^2) \pm 6m]\beta}} \frac{cn(\xi)}{dn(\xi)} \pm \sqrt{\frac{\alpha}{[-(1+m^2) \pm 6m]\beta}} \frac{\left(m^2-1\right)sn(\xi)}{cn(\xi)dn(\xi)}, \tag{31}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{2\alpha}{-(1+m^2) \pm 6m}$.

Due to the construction form of the solution in Equation (5), the solutions obtained in **Family 2** are the same as those in **Family 1**. From Equation (7) we can obtain the following:

Family 3

$$p_1 = \left(1 - m^2\right)^2, q_1 = 2\left(1 + m^2\right), r_1 = 1,\ a_0 = 0,\ a_1 = 1,\ a_2 = 0. \tag{32}$$

According to Equations (32) and (21)–(25), the solutions of the (2+1)-dimensional Klein–Gordon equation read as follows:

$$u_{31}(\xi) = \pm\sqrt{-\frac{\alpha}{(1+m^2)\beta}}\frac{cn(\xi)dn(\xi)}{sn(\xi)}, \tag{33}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{2(1+m^2)}$.

$$u_{32}(\xi) = \pm\sqrt{-\frac{\alpha}{(1+m^2)\beta}}\frac{\left(m^2-1\right)sn(\xi)}{cn(\xi)dn(\xi)}, \tag{34}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{2(1+m^2)}$, which is the same as $u_{13}(\xi)$.

$$u_{33}(\xi) = \pm\sqrt{-\frac{\alpha}{2(1+m^2)\beta}}\frac{1 - m^2 sn^4(\xi)}{sn(\xi)cn(\xi)dn(\xi)}, \tag{35}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{4(1+m^2)}$.

$$\begin{aligned} u_{34}(\xi) = &\pm\sqrt{\frac{\alpha}{[(1+m^2)\pm3(1-m^2)]\beta}}\frac{cn(\xi)dn(\xi)}{sn(\xi)} \\ &\pm\sqrt{\frac{\alpha}{[(1+m^2)\pm3(1-m^2)]\beta}}\frac{\left(m^2-1\right)sn(\xi)}{cn(\xi)dn(\xi)}, \end{aligned} \tag{36}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{2(1+m^2)\pm6(1-m^2)}$.

$$\begin{aligned} u_{35}(\xi) = &\pm\sqrt{\frac{\alpha}{[2(1+m^2)\pm6(1-m^2)]\beta}}\frac{cn(\xi)dn(\xi)}{sn(\xi)} \\ &\pm\sqrt{\frac{\alpha}{[2(1+m^2)\pm6(1-m^2)]\beta}}\frac{\left(m^2-1\right)sn(\xi)}{cn(\xi)dn(\xi)} \\ &\pm\sqrt{\frac{\alpha}{[2(1+m^2)\pm6(1-m^2)]}}\frac{1-m^2sn^4(\xi)}{sn(\xi)cn(\xi)dn(\xi)}, \end{aligned} \tag{37}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{(1+m^2)\pm3(1-m^2)}$.

From Equation (8) we can obtain the following:

Family 4

$$p_1 = 8 \mp \frac{4\left(1+m^2\right)}{m}, q_1 = -(1+m^2) \pm 6m, r_1 = m^2,\ a_0 = 1,\ a_1 = 0,\ a_2 = \mp\frac{1}{\mathrm{m}}. \tag{38}$$

According to Equations (36) and (21)–(25), the solutions of the (2+1)-dimensional Klein–Gordon equation read as follows:

$$u_{41}(\xi) = \pm m\sqrt{\frac{2\left[8 + \lambda\frac{4(1+m^2)}{m}\right]\alpha}{[(1+m^2)+6\lambda m]\beta}}\frac{cn(\xi)dn(\xi)}{mcn^2(\xi) + \lambda dn^2(\xi)}, \tag{39}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{(1+m^2)-6\lambda m}$, $\lambda^2 = 1$.

$$u_{42}(\xi) = \pm m\sqrt{\frac{2\alpha}{[(1+m^2)+6\lambda m]\beta}}\left[\frac{cn(\xi)}{dn(\xi)} + \frac{\lambda dn(\xi)}{mcn(\xi)}\right], \tag{40}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{(1+m^2)+6\lambda m}$, $\lambda^2 = 1$.

$$\begin{aligned} &u_{43}(\xi) \\ &= \pm\sqrt{\frac{\alpha}{[-(1+m^2)+6\lambda m]\beta}}\frac{m^3cn^4(\xi)sn(\xi) - msn(\xi)cn^2(\xi)dn^2(\xi) - \lambda sn(\xi)dn^4(\xi) + \lambda m^2 sn(\xi)cn^2(\xi)dn^2(\xi)}{m^2cn^3(\xi)dn(\xi) - \lambda mdn^3(\xi)cn(\xi)}, \end{aligned} \tag{41}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{2[-(1+m^2)+6\lambda m]}$, $\lambda^2 = 1$.

$$\begin{aligned} u_{44}(\xi) = \pm &\sqrt{\frac{2\left[8-\lambda\frac{4(1+m^2)}{m}\right]\alpha}{\left[-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}\right]\beta}}\frac{mcn(\xi)dn(\xi)}{mcn^2(\xi) - \lambda dn^2(\xi)} \\ &\pm m\sqrt{\frac{2\alpha}{\left[-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}\right]\beta}}\left[\frac{cn(\xi)}{dn(\xi)}\right. \\ &\left. - \frac{\lambda dn(\xi)}{mcn(\xi)}\right], \end{aligned} \tag{42}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}}$, $\lambda^2 = 1$.

$$\begin{aligned} &u_{45}(\xi) \\ &= \pm\sqrt{\frac{\left[8-\lambda\frac{4(1+m^2)}{m}\right]\alpha}{\left[-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}\right]\beta}}\frac{mcn(\xi)dn(\xi)}{mcn^2(\xi) - \lambda dn^2(\xi)} \\ &\pm m\sqrt{\frac{\alpha}{\left[-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}\right]\beta}}\left[\frac{cn(\xi)}{dn(\xi)} - \frac{\lambda dn(\xi)}{mcn(\xi)}\right] \\ &\pm\sqrt{\frac{\alpha}{\left[-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}\right]\beta}}\frac{m^3cn^4(\xi)sn(\xi) - msn(\xi)cn^2(\xi)dn^2(\xi) - \lambda sn(\xi)dn^4(\xi) + \lambda m^2 sn(\xi)cn^2(\xi)dn^2(\xi)}{m^2cn^3(\xi)dn(\xi) - \lambda mdn^3(\xi)cn(\xi)}, \end{aligned} \tag{43}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{2\alpha}{-(1+m^2)+6\lambda m \pm 6m\sqrt{8-\frac{4\lambda(1+m^2)}{m}}}$, $\lambda^2 = 1$.

From Equation (9) we can obtain the following:

Family 5

$$\begin{aligned} p_1 = m^2 - \tfrac{3}{4}(1+m^2)^2 + 3\lambda\mu m(1+m^2), q_1 = -\tfrac{1+m^2}{2} - 3\lambda\mu m, r_1 = \tfrac{1}{4},\ a_0 \\ = \lambda m,\ a_1 = 1,\ a_2 = \mu,\ \lambda^2 = 1, \mu^2 = 1. \end{aligned} \tag{44}$$

According to Equations (44) and (21)–(25), the solutions of the (2+1)-dimensional Klein–Gordon equation read as follows:

$$\begin{aligned} &u_{51}(\xi) \\ &= \pm\sqrt{\frac{2\left[m^2 - \frac{3}{4}(1+m^2)^2 + 3\lambda\mu m(1+m^2)\right]\alpha}{\left[\frac{1+m^2}{2} + 3\lambda\mu m\right]\beta}}\frac{cn(\xi)dn(\xi)}{\lambda mcn^2(\xi) + (m^2-1)sn(\xi) + \mu dn^2(\xi)}, \end{aligned} \tag{45}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = -\frac{\alpha}{\frac{1+m^2}{2} + 3\lambda\mu m}, \lambda^2 = 1, \mu^2 = 1$.

$$u_{52}(\xi) = \pm\sqrt{\frac{\alpha}{2\left[\frac{1+m^2}{2} + 3\lambda\mu m\right]\beta}}\frac{\lambda mcn^2(\xi) + (m^2-1)sn(\xi) + \mu dn^2(\xi)}{cn(\xi)dn(\xi)}, \tag{46}$$

where $\xi = \Phi(x,y,t), \Phi_{xx}+\Phi_{yy}-\Phi_{tt}=0, {\Phi_x}^2+{\Phi_y}^2-{\Phi_t}^2=-\frac{\alpha}{\frac{1+m^2}{2}+3\lambda\mu m}, \lambda^2=1, \mu^2=1.$

$$u_{53}(\xi)=\pm\sqrt{-\frac{\alpha}{\left[\frac{1+m^2}{2}+3\lambda\mu m\right]\beta}}\frac{-\lambda m\left(m^2-1\right)sn(\xi)cn^2(\xi)-m^2cn^4(\xi)+dn^4(\xi)+\mu sn(\xi)dn^2(\xi)}{\lambda mcn^3(\xi)dn(\xi)+(m^2-1)sn(\xi)cn(\xi)dn(\xi)+\mu cn(\xi)dn^3(\xi)}, \tag{47}$$

where $\xi = \Phi(x,y,t), \Phi_{xx}+\Phi_{yy}-\Phi_{tt}=0, {\Phi_x}^2+{\Phi_y}^2-{\Phi_t}^2=\frac{\alpha}{-\frac{1+m^2}{2}-3\lambda\mu m}, \lambda^2=1, \mu^2=1.$

$$\begin{aligned}&u_{54}(\xi)\\&=\pm\sqrt{\frac{2\left[m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)\right]\alpha}{\left[-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}\right]\beta}}\frac{cn(\xi)dn(\xi)}{\lambda mcn^2(\xi)+(m^2-1)sn(\xi)+\mu dn^2(\xi)}\\&\pm\sqrt{\frac{\alpha}{2\left[-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}\right]\beta}}\frac{\lambda mcn^2(\xi)+\left(m^2-1\right)sn(\xi)+\mu dn^2(\xi)}{cn(\xi)dn(\xi)},\end{aligned} \tag{48}$$

where $\xi = \Phi(x,y,t), \Phi_{xx}+\Phi_{yy}-\Phi_{tt}=0, {\Phi_x}^2+{\Phi_y}^2-{\Phi_t}^2=\frac{\alpha}{-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}}, \lambda^2=1, \mu^2=1.$

$$\begin{aligned}&u_{55}(\xi)\\&=\pm\sqrt{\frac{\left[m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)\right]\alpha}{\left[-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}\right]\beta}}\frac{cn(\xi)dn(\xi)}{\lambda mcn^2(\xi)+(m^2-1)sn(\xi)+\mu dn^2(\xi)}\\&\pm\sqrt{\frac{\alpha}{4\left[-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}\right]\beta}}\frac{\lambda mcn^2(\xi)+\left(m^2-1\right)sn(\xi)+\mu dn^2(\xi)}{cn(\xi)dn(\xi)}\\&\pm\sqrt{\frac{\alpha}{\left[-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}\right]\beta}}\frac{-\lambda m\left(m^2-1\right)sn(\xi)cn^2(\xi)-m^2cn^4(\xi)+dn^4(\xi)+\mu sn(\xi)dn^2(\xi)}{\lambda mcn^3(\xi)dn(\xi)+(m^2-1)sn(\xi)cn(\xi)dn(\xi)+\mu cn(\xi)dn^3(\xi)},\end{aligned} \tag{49}$$

where $\xi = \Phi(x,y,t), \Phi_{xx}+\Phi_{yy}-\Phi_{tt}=0, {\Phi_x}^2+{\Phi_y}^2-{\Phi_t}^2=\frac{2\alpha}{-\frac{1+m^2}{2}-3\lambda\mu m\pm3\sqrt{m^2-\frac{3}{4}(1+m^2)^2+3\lambda\mu m(1+m^2)}}, \lambda^2=1, \mu^2=1.$

Since **Family 6** and **Family 7** represent the same type of solution, we only provide a demonstration represented by **Family 6**. From Equation (11) we can obtain the following:

Family 6

$$a=\pm bm, c=-b,\ p_1=\pm\frac{m}{2b},\ q_1=\frac{1}{4}[-(1+m^2)\pm6m],\ r_1=\frac{\left[-\left(1+m^2\right)\pm2m\right]b}{2},\ b\neq0. \tag{50}$$

Therefore, the corresponding solutions of the (2+1)-dimensional Klein–Gordon equation can be represented as follows:

$$u_{61}(\xi)=\pm\sqrt{\frac{4\lambda m\alpha}{(-1-m^2+6\lambda m)\beta}}\sqrt{-\lambda m\frac{cn^2(\xi)}{dn^2(\xi)}-m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)}+sn(\xi)+1}, \tag{51}$$

where $\xi = \Phi(x,y,t), \Phi_{xx}+\Phi_{yy}-\Phi_{tt}=0, {\Phi_x}^2+{\Phi_y}^2-{\Phi_t}^2=\frac{4\alpha}{1+m^2-6\lambda m},\ \lambda^2=1.$

$$u_{62}(\xi)=\pm\sqrt{\frac{4(1+m^2-2\lambda m)\alpha}{(1+m^2-6\lambda m)\beta}}\frac{1}{\sqrt{-\lambda m\frac{cn^2(\xi)}{dn^2(\xi)}-m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)}+sn(\xi)+1}}, \tag{52}$$

where $\xi = \Phi(x,y,t), \Phi_{xx}+\Phi_{yy}-\Phi_{tt}=0, {\Phi_x}^2+{\Phi_y}^2-{\Phi_t}^2=\frac{4\alpha}{1+m^2-6\lambda m},\ \lambda^2=1.$

$$u_{63}(\xi)=\pm\sqrt{\frac{4(1+m^2-2\lambda m)\alpha}{(-1-m^2+6\lambda m)\beta}}\frac{\lambda m\frac{sn(\xi)cn^3(\xi)}{dn^3(\xi)}-\lambda\frac{sn(\xi)cn(\xi)}{dn(\xi)}+\frac{m^4sn^2(\xi)cn^3(\xi)}{dn^3(\xi)}+\frac{\frac{1}{2}m^2cn^3(\xi)-m^2sn^2(\xi)cn(\xi)}{dn(\xi)}-cn(\xi)dn(\xi)}{\lambda m\frac{cn^2(\xi)}{dn^2(\xi)}+m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)}-sn(\xi)-1}, \tag{53}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{2\alpha}{-1-m^2+6\lambda m}$, $\lambda^2 = 1$.

$$\begin{aligned} & u_{64}(\xi) \\ & = \pm\sqrt{\frac{\lambda m\alpha}{\left[\frac{1}{4}(-1-m^2+\lambda 6m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}\right]\beta}}\sqrt{\lambda m\frac{cn^2(\xi)}{dn^2(\xi)} + m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)} - sn(\xi) - 1} \\ & \pm\sqrt{\frac{[-(1+m^2)+2\lambda m]\alpha}{\left[\frac{1}{4}(-1-m^2+6\lambda m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}\right]\beta}}\frac{1}{\sqrt{\lambda m\frac{cn^2(\xi)}{dn^2(\xi)}+m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)}-sn(\xi)-1}}, \end{aligned} \tag{54}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{\frac{1}{4}(-1-m^2+6\lambda m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}}$, $\lambda^2 = 1$.

$$\begin{aligned} & u_{65}(\xi) \\ & = \pm\sqrt{\frac{\frac{1}{2}\lambda m\alpha}{\left[\frac{1}{4}(-1-m^2+6\lambda m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}\right]\beta}}\sqrt{\lambda m\frac{cn^2(\xi)}{dn^2(\xi)} + m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)} - sn(\xi) - 1} \\ & \pm\sqrt{\frac{\frac{1}{2}[-(1+m^2)+2\lambda m]\alpha}{\left[\frac{1}{4}(-1-m^2+6\lambda m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}\right]\beta}}\frac{1}{\sqrt{\lambda m\frac{cn^2(\xi)}{dn^2(\xi)}+m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)}-sn(\xi)-1}} \\ & \pm\sqrt{\frac{\alpha}{\left[\frac{1}{4}(-1-m^2+6\lambda m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}\right]\beta}}\frac{\lambda m\frac{sn(\xi)cn^3(\xi)}{dn^3(\xi)}-\lambda\frac{sn(\xi)cn(\xi)}{dn(\xi)}+\frac{m^4sn^2(\xi)cn^3(\xi)}{dn^3(\xi)}+\frac{\frac{1}{2}m^2cn^3(\xi)-m^2sn^2(\xi)cn(\xi)}{dn(\xi)}-cn(\xi)dn(\xi)}{\lambda m\frac{cn^2(\xi)}{dn^2(\xi)}+m^2\frac{sn(\xi)cn^2(\xi)}{dn^2(\xi)}-sn(\xi)-1}, \end{aligned} \tag{55}$$

where $\xi = \Phi(x,y,t), \Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, {\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{2\alpha}{\frac{1}{4}(-1-m^2+6\lambda m)\pm 3\sqrt{-\lambda(1+m^2)m+2m^2}}$, $\lambda^2 = 1$.

From this set of example solutions, it can be seen that this method is very effective and powerful for constructing complex exact solutions. In addition, these solutions also include early simple forms of Jacobian elliptic function solutions of the (2+1)-dimensional Klein– Gordon equation. Among these solutions, Equations (39)–(55) are the new exact solutions of the (2+1)-dimensional Klein–Gordon equation that we have discovered for the first time. Of course, there are still many new types of Jacobian elliptic function solutions for the (2+1)-dimensional Klein–Gordon equation, which may also include corresponding trigonometric and hyperbolic function solutions under limit conditions. Due to space limitations, we will not provide examples one-by-one.

5. Local Nonlinear Wave Structures of (2+1)-Dimensional Klein–Gordon Equation

For all the Jacobian elliptic function solutions mentioned above, under the conditions of general traveling wave transformation, we can obtain the periodic wave solutions of the equation. However, since Equation (13) is a general wave transformation, it indicates that the equation can have complex wave solutions. In this paper, we present a novel nonlinear wave construction method to explore complex wave solutions that have not been mentioned in other literatures. Through this method, we can construct various nonlinear complex wave structures for the (2+1)-dimensional Klein–Gordon equation. In the following section, the local nonlinear wave structure of the (2+1)-dimensional Klein–Gordon equation is discussed by taking the solutions of Equation (39) as examples. In this set of solution, the general wave $\xi = \Phi(x, y, t)$ satisfies

$$\Phi_{xx} + \Phi_{yy} - \Phi_{tt} = 0, \tag{56}$$

$${\Phi_x}^2 + {\Phi_y}^2 - {\Phi_t}^2 = \frac{\alpha}{(1+m^2) \mp 6m}, \tag{57}$$

Equation (56) can have the following general solution:

$$\Phi(x,y,t) = \Psi_1(kx + ly + \omega t) + \Psi_2(kx + ky - \omega t) + h(x,y,t), \tag{58}$$

where k, l and ω are any nonzero constants and satisfy $k^2 + l^2 = \omega^2$, Ψ_1, Ψ_2, and h is any second order differentiable function and satisfies $h_{xx} + h_{yy} - h_{tt} = 0$. Taking Equation (58) into Equation (57), we can get the following limiting condition:

$$2(k^2 + l^2 + \omega^2)\Psi_1'\Psi_2' + 2kh_x(\Psi_1' + \Psi_2') + 2lh_y(\Psi_1' + \Psi_2') - 2\omega h_t(\Psi_1' - \Psi_2') + {h_x}^2 + {h_y}^2 - {h_t}^2 = \frac{\alpha}{(1+m^2)\mp 6m}. \quad (59)$$

In Equation (58), if $h(x, y, t)$ is not introduced, Equation (59) can only have traveling wave solutions. The traveling wave solutions commonly used in solving the Klein–Gordon equation in the literature were mentioned above. The introduction of $h(x, y, t)$ makes Equation (59) have complex wave solutions. So, we can construct various nonlinear waves of the (2+1)-dimensional Klein–Gordon equation by selecting different values for Ψ_1, Ψ_2, and h that can meet Equations (56) and (59). In the following section, we will provide several nonlinear wave examples.

Case 1

If we choose $\Psi_1 = 0, \Psi_2 = 0$ and $h(x, y, t) = k_1 x + l_1 y + \omega_1 t$, and $\alpha = \left({k_1}^2 + {l_1}^2 + {\omega_1}^2\right)\left[(1 + m^2) \mp 6m\right]$, where k_1, l_1 and ω_1 are nonzero constants, we can obtain the periodic wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 1. This is a common wave structure that undergoes periodic changes with the phase space (x, y, t). Although Equation (39) is a novel periodic wave solution of the (2+1)-dimensional Klein–Gordon equation, the wave structure is similar to the periodic wave structure in ref. [7].

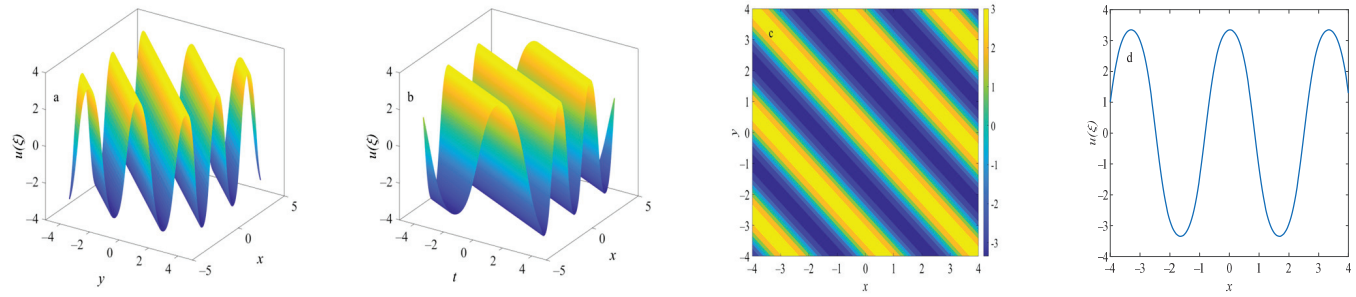


Figure 1. Three-dimensional plots in (x, y) phase space (**a**) and (x, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) for $t = 0$ and a two-dimensional plot (**d**) in $y = 0$ and $t = 0$ represent the periodic wave solution of Equation (39), under conditions of $k = 1$, $l = 1$, $\omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2$, $\lambda = 1$, $\Psi_1 = 0$, $\Psi_2 = 0$, $h(x, y, t) = k_1 x + l_1 y + \omega_1 t$ and $\alpha = \left({k_1}^2 + {l_1}^2 + {\omega_1}^2\right)\left[(1 + m^2) \mp 6m\right]$.

Case 2

If we choose $\Psi_1 = -\frac{kk_1 x + ll_1 y + \omega\omega_1 t}{\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \cosh(kx + ly - \omega t)$, $h(x, y, t) = k_1 x^2 + l_1 y^2 + \omega_1 t^2$, and $\alpha = \frac{(lk_1 x + kl_1 y)^2}{\omega^2}\left[(1 + m^2) \mp 6m\right]$, we can obtain the new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 2. This is a nonlinear wave propagating along the x or y direction in two-dimensional space under certain conditions. The center of the (x, t, u) or (y, t, u) contour is an approximately circular wave structure, and many harmonic waves will appear as time increases or decreases.

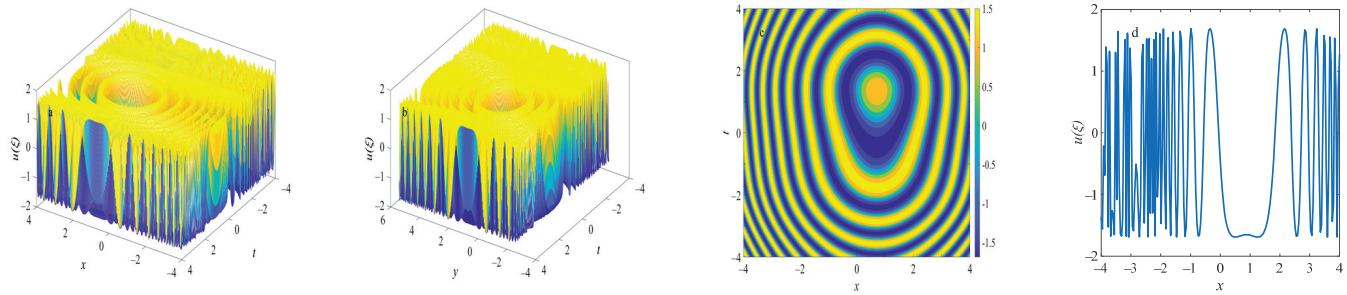


Figure 2. Three-dimensional plots in (x, t) phase space (**a**) and (y, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) and a two-dimensional plot (**d**) represent the complex wave solution of Equation (39), under the conditions of $k = 1, \omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2$, $\lambda = 1$, $\alpha = 4, \Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx+ky+\omega t)$, $\Psi_2 = \cosh(kx+ly-\omega t)$, $h(x,y,t) = k_1x^2 + l_1y^2 + \omega_1t^2$, and $\alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1+m^2) \mp 6m]$.

Case 3

If we choose $\Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx+ky+\omega t)$, $\Psi_2 = \sin(kx+ly-\omega t)$, $h(x,y,t) = k_1x^2 + l_1y^2 + \omega_1t^2$ and $\alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1+m^2) \mp 6m]$, we can obtain another new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 3. This is a nonlinear wave that periodically propagates along the x or y direction in two-dimensional space under certain conditions. The (x, t, u) or (y, t, u) contour is an approximately elliptical nonlinear wave structure, in which the amplitude remains constant except for the center position during propagation.

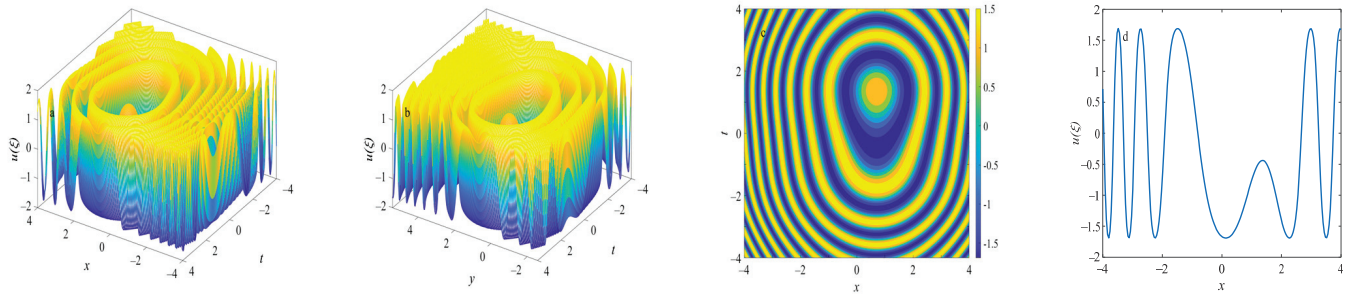


Figure 3. Three-dimensional plots in (x, t) phase space (**a**) and (y, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) and a two-dimensional plot (**d**) represent the complex wave solution of Equation (39), under the conditions of $k = 1, \omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2$, $\lambda = 1$, $\alpha = 4, \Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx+ky+\omega t)$, $\Psi_2 = \sin(kx+ly-\omega t)$, $h(x,y,t) = kk_1x^2 + l_1y^2 + \omega_1t^2$, and$\alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1+m^2) \mp 6m]$.

Case 4

If we choose $\Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx+ky+\omega t)$, $\Psi_2 = \ln(kx+ly-\omega t)^2$,$h(x,y,t) = k_1x^2 + l_1y^2 + \omega_1t^2$ and $\alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1+m^2) \mp 6m]$, we can obtain the new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 4. This is a nonlinear wave that periodically propagates along the x or y direction in two-dimensional space under certain conditions, and the amplitude remains constant except for the center position during propagation.

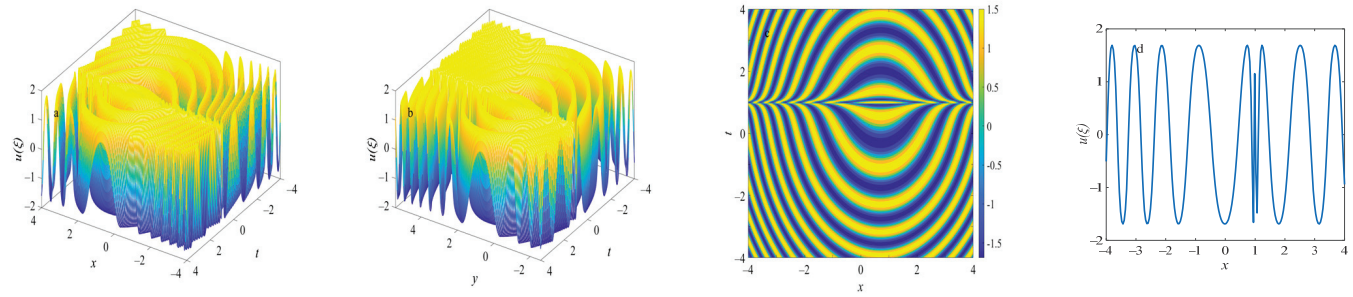


Figure 4. Three-dimensional plots in (x, t) phase space (**a**) and (y, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) and a two-dimensional plot (**d**) represent the complex wave solution of Equation (39), under the conditions of $k = 1$, $\omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2$, $\lambda = 1$, $\alpha = 4, \Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \ln(kx + ly - \omega t)^2$ and $h(x, y, t) = kk_1x^2 + l_1y^2 + \omega_1t^2, \alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1 + m^2) \mp 6m]$.

Case 5

If we choose $\Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \tan(kx + ly - \omega t)$, $h(x, y, t) = k_1x^2 + l_1y^2 + \omega_1t^2$, and $\alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1 + m^2) \mp 6m]$, we can obtain the new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 5. This is a nonlinear wave that propagates along the x or y direction in two-dimensional space under certain conditions. The propagation period and amplitude of this wave will be interrupted by the singularity brought by $\Psi_2 = tan(kx + ly - \omega t)$.

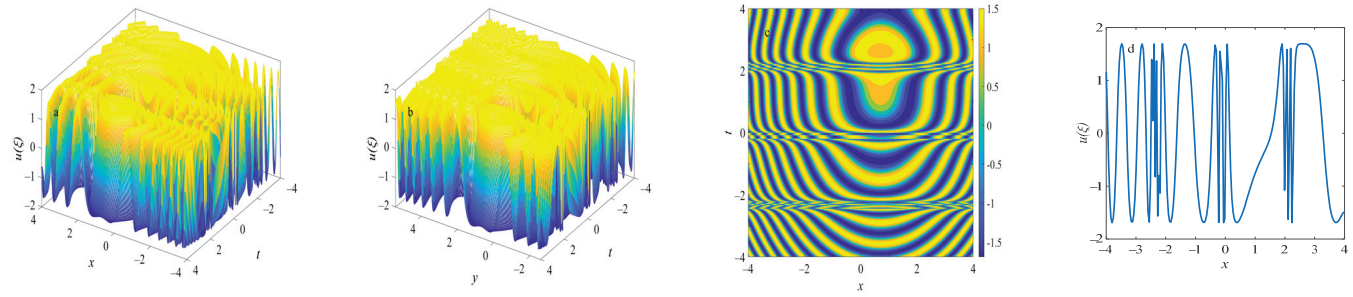


Figure 5. Three-dimensional plots in (x, t) phase space (**a**) and (y, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) and a two-dimensional plot (**d**) represent the complex wave solution of Equation (39), under the conditions of $k = 1, \omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2$, $\lambda = 1$, $\alpha = 4, \Psi_1 = -\frac{kk_1x+ll_1y+\omega\omega_1t}{\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \tan(kx + ly - \omega t)$, $h(x, y, t) = k_1x^2 + l_1y^2 + \omega_1t^2$, and $\alpha = \frac{(lk_1x+kl_1y)^2}{\omega^2}[(1 + m^2) \mp 6m]$.

Case 6

If we choose $\Psi_1 = -\frac{kk_1+ll_1+\omega\omega_1}{2\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \sin(kx + ly - \omega t)$, $h(x, y, t) = k_1x + l_1y + \omega_1t$, and $\alpha = \frac{(lk_1+kl_1)^2}{\omega^2}[(1 + m^2) \mp 6m]$, we can obtain the new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 6. This is a nonlinear wave that propagates along the x and y direction in two-dimensional space. As shown in the figure, they are still quasi-periodic waves in the (x, y) phase space and the (x, t) phase space, but their amplitudes are modulated by a sine wave.

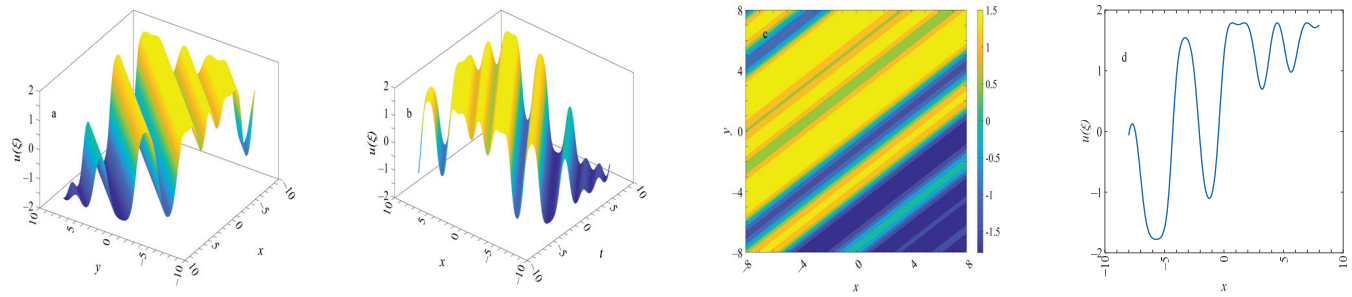


Figure 6. Three-dimensional plots in (x, y) phase space (**a**) and (x, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) in $y = 0$ and a two-dimensional plot (**d**) in $y = 0$ and $t = 0$ represent the complex wave solution of Equation (39), under the conditions of $k = 1, \omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2, \lambda = 1, \Psi_1 = -\frac{kk_1+ll_1+\omega\omega_1}{2\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \sin(kx + ly - \omega t)$, $h(x, y, t) = k_1x + l_1y + \omega_1 t$, and $\alpha = \frac{(lk_1+kl_1)^2}{\omega^2}[(1 + m^2) \mp 6m]$.

Case 7

If we choose $\Psi_1 = -\frac{kk_1+ll_1+\omega\omega_1}{2\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \ln(kx + ly - \omega t)^2$, $h(x, y, t) = k_1x + l_1y + \omega_1 t$, and $\alpha = \frac{(lk_1+kl_1)^2}{\omega^2}[(1 + m^2) \mp 6m]$, we can obtain the new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 7. This is a nonlinear wave that propagates along the x and y direction in two-dimensional space. Figure 7a,b shows the central structure of this nonlinear wave. As a matter of fact, this nonlinear wave propagates periodically in addition to its central position.

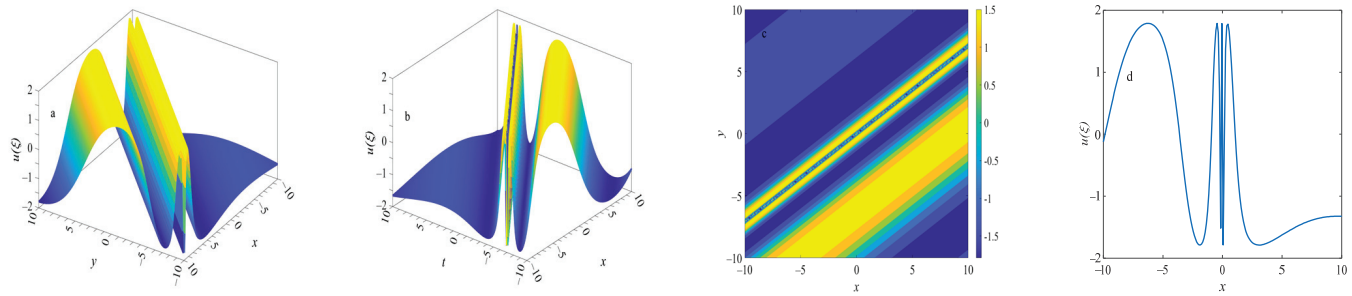


Figure 7. Three-dimensional plots in (x, y) phase space (**a**) and (x, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) in $y = 0$ and a two-dimensional plot (**d**) in $y = 0$ and $t = 0$ represent the complex wave solution of Equation (39), under the conditions of $k = 1, \omega = \sqrt{2}$, $k_1 = 2, l_1 = 2, \omega_1 = 1, m = 0.2, \lambda = 1, \Psi_1 = -\frac{kk_1+ll_1+\omega\omega_1}{2\omega^2}(kx + ky + \omega t)$, $\Psi_2 = \ln(kx + ly - \omega t)^2$, $h(x, y, t) = k_1x + l_1y + \omega_1 t$, and $\alpha = \frac{(lk_1+kl_1)^2}{\omega^2}[(1 + m^2) \mp 6m]$.

Case 8

If we choose $\Psi_1 = -\frac{kk_1+ll_1+\omega\omega_1}{2\omega^2}(kx + ky + \omega t)$, $\Psi_2 = e^{kx+ly-\omega t}$, $h(x, y, t) = k_1x + l_1y + \omega_1 t$, and $\alpha = \frac{(lk_1+kl_1)^2}{\omega^2}[(1 + m^2) \mp 6m]$, we can obtain the new type of complex wave structure of the (2+1)-dimensional Klein–Gordon equation shown in Figure 8. This is a nonlinear wave propagating along the x-direction, which is a quasi-periodic nonlinear wave with gradually changing periods due to the exponential increase in the value of $\Psi_2 = e^{kx+ly-\omega t}$.

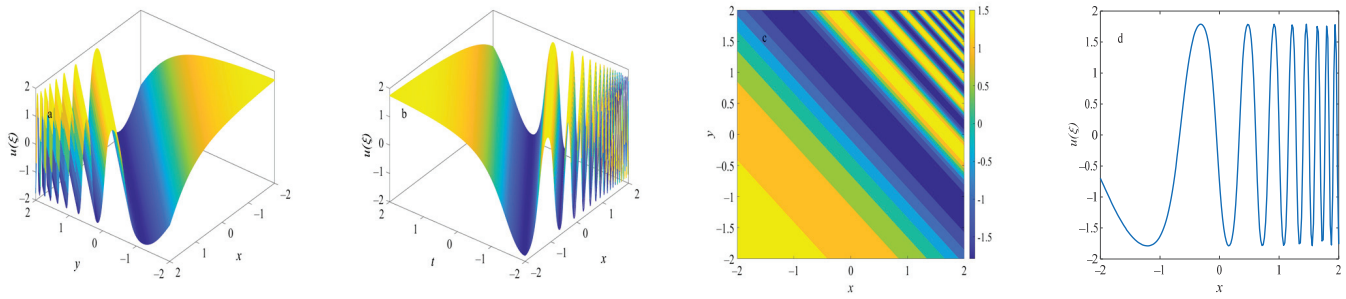


Figure 8. Three-dimensional plots in (x, y) phase space (**a**) and (x, t) phase space (**b**), as well as a two-dimensional contour plot (**c**) in $y = 0$ and a two-dimensional plot (**d**) in $y = 0$ and $t = 2$ represent the complex wave solution of Equation (39), under the conditions of $k = 1, \omega = \sqrt{2}$, $k_1 = 2, l_1 = 2$, $\omega_1 = 1, m = 0.2, \lambda = 1, \Psi_1 = -\frac{kk_1+ll_1+\omega\omega_1}{2\omega^2}(kx + ky + \omega t)$, $\Psi_2 = e^{kx+ly-\omega t}$, $h(x, y, t) = k_1x + l_1y + \omega_1 t$, and $\alpha = \frac{(lk_1+kl_1)^2}{\omega^2}[(1 + m^2) \mp 6m]$.

6. Discussion and Conclusions

This paper utilizes a new extended Jacobian elliptic function expansion method to construct many Jacobian elliptic equation solutions for the (2+1)-dimensional Klein–Gordon equation. This is a new expansion method proposed by us, resulting in a large number of new types of Jacobian elliptic function solutions. The method we used is simple and effective. In this paper, we applied it to handle the (2+1) Klein–Gordon equation as an example. In fact, it can also handle more complex equations, such as the Benjamin equation with $n = 2$ in ref. [15]. We only need to substitute Equations (5)–(12) in this paper into Equations (31)–(35) in ref. [15] to obtain solutions that include the solutions listed in that reference but far exceed those expressed in their structural forms, such as Equations (4) and (10) in this paper. Compared to early Jacobian expansion methods, such as refs. [12–14], their solutions are only a few special cases of our solution. For the F-expansion method in refs. [8,9], their solution is only a special case under the conditions given in Equation (6). The tanh–sech expansion method and its extensions in refs. [10,11] have fewer solutions than the ones we obtained, yet they are included in the limit conditions of the solutions in m→1 in this paper.

The solutions obtained in this article can be used to describe the propagation of nonlinear waves in plasma, condensed matter physics, and so on. Then, we proposed a new method of Equations (56)–(59) for constructing complex nonlinear waves, which has not been reported in other literature. Through this method, various nonlinear wave structures for this equation were constructed, which indicate new types of complex wave phenomena. To provide an intuitive understanding of the structure of complex nonlinear waves and their propagation in time and space, we set the parameters of the (2+1)-dimensional Klein–Gordon equation and complex waves to given values and applied the mathematical software MATLAB 2021b to draw three-dimensional, two-dimensional and two-dimensional contour plots of some examples (see Figures 1–8). These waves exhibit complex composite wave structures propagating in spacetime. Different parameters of the (2+1)-dimensional Klein–Gordon equation, different travelling wave parameters and different composite waves can cause different changes in the types, amplitude and period of these nonlinear waves. The images shown in Figures 1–8 indicate that under specific conditions, different composite waves can be excited, which may experience unstable amplitude growth, attenuation or oscillation, as well as the generation of high-order harmonics.

All the solutions and formulas obtained in this paper have been checked by MATLAB 2021b. By applying the same scheme, the method used in this paper can also be used to handle other types of NLEEs to obtain various complex wave structures. However, the construction method of nonlinear complex waves in Section 5 can only be applied to nonlinear Klein–Gordon-type equations. In the future, we will derive this type of nonlinear complex wave transformation for other NLEEs.

Author Contributions: Methodology, Y.G.; formal analysis, G.W.; investigation, G.W.; data curation, Y.Y.; writing—original draft, G.W.; writing—review and editing, Y.G.; project administration, Y.Y.; funding acquisition, Y.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Specialized Fund for the Doctoral of Kaili University (grant No. BS20240209) and National Natural Science Foundation of China with Contract Nos. 12275307 and 11575238.

Data Availability Statement: No data, models, or code are generated or used during the study.

Acknowledgments: The authors thank the referees for their valuable comments and suggestions, which improved the presentation of this manuscript.

Conflicts of Interest: The authors state no conflicts of interest.

References

1. Wazwaz, A.M. The extended tanh method for new compact and noncompact solutions for the KP-BBM and the ZK-BBM equations. *Chaos Solitons Fractals* **2008**, *38*, 1505. [CrossRef]
2. Yusofoğlu, E. New solitonary solutions for the MBBM equations using Exp-function method. *Phys. Lett. A* **2008**, *372*, 442. [CrossRef]
3. Gardner, C.S.; Greene, J.M.; Kruskal, M.D.; Miura, R.M. Method for solving Korteweg-de Vries equation. *Phys. Rev. Lett.* **1967**, *19*, 1095. [CrossRef]
4. Su, C.H.; Gardner, C.S. Korteweg-de Vries Equation and Generalizations—III—Derivation of the Korteweg-de Vries Equation and Burgers Equation. *J. Math. Phys.* **1969**, *10*, 536. [CrossRef]
5. Li, Z.B.; Wang, M.L. Travelling wave solutions to the two-dimensional KdV-Burgers equation. *J. Phys. A Math. Gen.* **1993**, *26*, 6027. [CrossRef]
6. Ito, M. An Extension of Nonlinear Evolution Equations of the K-dV (mK-dV) Type to Higher Orders. *J. Phys. Soc. Jpn.* **1980**, *49*, 771. [CrossRef]
7. Wang, M.L. Solitary wave solutions for variant Boussinesq equations. *Phys. Lett. A* **1995**, *199*, 169. [CrossRef]
8. Liu, J.B.; Yang, K.Q. The extended F-expansion method and exact solutions of nonlinear PDEs. *Chaos Solitons Fractals* **2004**, *22*, 111. [CrossRef]
9. Zhang, S.; Xia, T.C. A generalized F-expansion method and new exact solutions of Konopelchenko-Dubrovsky equations. *Appl. Math. Comput.* **2006**, *183*, 1190. [CrossRef]
10. Wazwaz, A.M. The extended tanh method for new solitons solutions for many forms of the fifth-order KdV equations. *Appl. Math. Comput.* **2007**, *184*, 1014. [CrossRef]
11. Wazwaz, A.M. The tanh-coth method for solitons and kink solutions for nonlinear parabolic equations. *Appl. Math. Comput.* **2007**, *188*, 1467. [CrossRef]
12. Liu, S.K.; Fu, Z.T.; Liu, S.D.; Zhao, Q. Jacobi elliptic function expansion method and periodic wave solutions of nonlinear wave equations. *Phys. Lett. A* **2001**, *289*, 69. [CrossRef]
13. Fu, Z.T.; Liu, S.K.; Liu, S.D.; Zhao, Q. New Jacobi elliptic function expansion and new periodic solutions of nonlinear wave equations. *Phys. Lett. A* **2001**, *290*, 72. [CrossRef]
14. Elgarayhi, A. New periodic wave solutions for the shallow water equations and the generalized Klein-Gordon equation. *Commun. Nonlinear Sci. Numer. Simul.* **2008**, *13*, 877. [CrossRef]
15. Wu, G.; Han, J.; Zhang, W.; Zhang, M. New periodic wave solutions to nonlinear evolution equations by the extended mapping method. *Phys. D Nonlinear Phenom.* **2007**, *229*, 116. [CrossRef]
16. Jiong, S. Auxiliary equation method for solving nonlinear partial differential equations. *Phys. Lett. A* **2003**, *309*, 387. [CrossRef]
17. Sirendaoreji. New exact travelling wave solutions for the Kawahara and modified Kawahara equations. *Chaos Solitons Fractals* **2004**, *19*, 147–150. [CrossRef]
18. Zhu, X.; Cheng, J.; Chen, Z.; Wu, G. New Solitary-Wave Solutions of the Van der Waals Normal Form for Granular Materials via New Auxiliary Equation Method. *Mathematics* **2022**, *10*, 2560. [CrossRef]
19. Wu, G.; Guo, Y. Construction of New Infinite-Series Exact Solitary Wave Solutions and Its Application to the Korteweg-De Vries Equation. *Fractal Fract.* **2023**, *7*, 75. [CrossRef]
20. Wang, M.; Li, X.; Zhang, J. The (G'/G)-expansion method and travelling wave solutions of nonlinear evolution equations in mathematical physics. *Phys. Lett. A* **2008**, *372*, 417. [CrossRef]
21. Zayed, E.M.E.; Gepreel, K.A. The G'/G-expansion method for finding the traveling wave solutions of nonlinear partial differential equations in mathematical physics. *J. Math. Phys.* **2009**, *50*, 013502. [CrossRef]
22. Guo, S.; Zhou, Y. The extended G'/G-expansion method and its applications to the Whitham-Broer-Kaup-Like equations and coupled Hirota-Satsuma KdV equations. *Appl. Math. Comput.* **2010**, *215*, 3214. [CrossRef]
23. Islam, M.S.; Khan, K.; Akbar, M.A. An analytical method for finding exact solutions of modified Korteweg-de Vries equation. *Results Phys.* **2015**, *5*, 131. [CrossRef]

24. Wu, G.; Guo, Y. New Complex Wave Solutions and Diverse Wave Structures of the (2+1)-Dimensional Asymmetric Nizhnik-Novikov-Veselov Equation. *Fractal Fract.* **2023**, *7*, 170. [CrossRef]
25. Itzykson, C.; Zuber, J.B. *Quantum Field Theory*; McGraw-Hill International Book Co.: New York, NY, USA, 1980.
26. Weinberg, S. *Quantum Theory of Fields*; Cambridge University Press: Cambridge, UK, 1995.
27. Rahman, M.; Dulat, S.; Li, K. Wigner Function for Klein—Gordon Landau Problem. *Commun. Theor. Phys.* **2010**, *54*, 809. [CrossRef]
28. Wang, Z.; Zhang, H.Q. Many new kinds exact solutions to (2+1)-dimensional Burgers equation and Klein-Gordon equation used a new method with symbolic computation. *Appl. Math. Comput.* **2007**, *186*, 693. [CrossRef]
29. Seadawy, A.R.; Lu, D.C.; Arshad, M. Stability Analysis of Solitary Wave Solutions for Coupled and (2+1)-Dimensional Cubic Klein-Gordon Equations and Their Applications. *Commun. Theor. Phys.* **2018**, *69*, 676. [CrossRef]
30. Ebaid, A. Exact solutions for the generalized Klein-Gordon equation via a transformation and Exp-function method and comparison with Adomian's method. *J. Comput. Appl. Math.* **2009**, *223*, 278. [CrossRef]
31. González, J.A.; Bellorín, A.; Guerrero, L.E. Kink-soliton explosions in generalized Klein-Gordon equations. *Chaos Solitons Fractals* **2007**, *33*, 143. [CrossRef]
32. Roshid, M.M.; Karim, M.F.; Azad, A.K.; Rahman, M.M.; Sultana, T. New solitonic and rogue wave solutions of a Klein-Gordon equation with quadratic nonlinearity. *Partial. Differ. Equ. Appl. Math.* **2021**, *3*, 100036. [CrossRef]
33. Joseph, S.P. New traveling wave exact solutions to the coupled Klein-Gordon system of equations. *Partial. Differ. Equ. Appl. Math.* **2022**, *5*, 100208. [CrossRef]
34. Hafez, M.G.; Alam, M.N.; Akbar, M.A. Exact traveling wave solutions to the Klein-Gordon equation using the novel (G'/G)-expansion method. *Results Phys.* **2014**, *4*, 177. [CrossRef]

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

mathematics

MDPI

Article

Impulsive Discrete Runge–Kutta Methods and Impulsive Continuous Runge–Kutta Methods for Nonlinear Differential Equations with Delayed Impulses

Gui-Lai Zhang *, Zhi-Yong Zhu, Yu-Chen Wang and Chao Liu

College of Sciences, Northeastern University, Shenyang 110819, China
* Correspondence: zhangguilai@neuq.edu.cn

Abstract: In this paper, we study the asymptotical stability of the exact solutions of nonlinear impulsive differential equations with the Lipschitz continuous function $f(t, x)$ for the dynamic system and for the impulsive term Lipschitz continuous delayed functions I_k. In order to obtain numerical methods with a high order of convergence and that are capable of preserving the asymptotical stability of the exact solutions of these equations, impulsive discrete Runge–Kutta methods and impulsive continuous Runge–Kutta methods are constructed, respectively. For these different types of numerical methods, different convergence results are obtained and the sufficient conditions for asymptotical stability of these numerical methods are also obtained, respectively. Finally, some numerical examples are provided to confirm the theoretical results.

Keywords: impulsive discrete Runge–Kutta method; impulsive continuous Runge–Kutta method; Lipschitz condition; convergence; asymptotical stability

MSC: 65L06

1. Introduction

Impulsive differential equations (IDEs) are widely applied in numerous fields of science and technology: theoretical physics, mechanics, population dynamics, pharmacokinetics, industrial robotics, chemical technology, biotechnology, economics, etc. (see [1–5] and references therein). Recently, there has been growing interest in the study of impulsive differential equations with delayed impulses (DEDIs) [5–17]. In particular, the stability of the exact solutions of DEDIs has been widely studied [5–13]. However, to the best of our knowledge, our present paper is the first paper to study the asymptotical stability of nonlinear DEDIs under Lipschitz conditions.

In recent years, the theory of numerical methods for IDEs has been developed rapidly. The convergence and stability of numerical methods for scalar linear IDEs [18–20], multidimensional linear IDEs [21], semi-linear IDEs [22], nonlinear IDEs[23–29], impulsive time-delay differential equations [30–35] and stochastic impulsive time-delay differential equations [36] have been studied. But little work has been conducted on numerical methods for DEDIs. In [37], we investigated asymptotical stability and convergence of impulsive discrete Runge–Kutta methods for linear DEDIs. In our present paper, we further investigate the convergence and stability of impulsive discrete Runge–Kutta methods for nonlinear DEDIs.

Continuous numerical methods are widely applied to delay differential equations without impulsive perturbations (see [38–43], etc.). But the exact solutions of impulsive differential equations are not continuous, so the continuous numerical methods are not applicable for impulsive differential equations. In [20], asymptotical stability and convergence of impulsive collocation methods for impulsive ordinary differential equations were studied. In [35], the convergence of the impulsive continuous Runge–Kutta methods was

studied. As far as we know, our present paper is the first to study the convergence and stability of impulsive continuous Runge–Kutta methods (ICRKMs) for nonlinear DEDIs.

The Runge–Kutta method ([44–47]) is applicable to various types of ordinary differential equations and its advantages mainly include high accuracy, generally good numerical stability and convergence. A natural question is, when applying the Runge–Kutta method to solve DEDIs, does the Runge–Kutta method method still have good stability and convergence when we treat the impulse terms in different ways? The continuous Runge–Kutta method ([39,46]) is described above and is suitable for solving ordinary differential equations and delay differential equations. It is also important for many practical questions such as graphical output, and even location or treatment of discontinuities in differential equations. Another natural question is whether the application of the continuous Runge–Kutta method to solve DEDIs also has good stability and convergence. This paper will answer both of these questions.

The remainder of this paper is arranged as follows. In Section 2, sufficient conditions for asymptotical stability of the exact solution of a class of nonlinear DEDIs are provided. In Section 3, the scheme 1 are correct. impulsive Runge–Kutta methods (S1IRKMs) are constructed. It is proved that S1IRKM is convergent of order p if the corresponding Runge–Kutta method is p-th order. S1IRKMs are obtained to preserve asymptotical stability of the exact solutions under the sufficient conditions obtained in Section 2, applying the theory of Padé approximation. Moreover, the scheme 1 impulsive θ method (S1IθM) are obtained to preserve asymptotical stability of the exact solutions under the sufficient conditions. In Section 4, the scheme 2 impulsive Runge–Kutta methods (S2IRKM) are constructed. It is proved that S2IRKM is only convergent of order 1 if the corresponding Runge–Kutta method is p-th order. S2IRKMs are obtained to preserve asymptotical stability of the exact solutions under the sufficient conditions applying the theory of Padé approximation. Moreover, the scheme 2 impulsive θ method (S2IθM) is obtained to preserve asymptotical stability of the exact solutions under sufficient conditions. In Section 5, the convergence and asymptotical stability of ICRKMs are studied. In Section 6, we provide two numerical examples to confirm our theoretical results. Finally, in Section 7, conclusions and future work are provided.

2. Asymptotical Stability of the Exact Solutions of DEDIs

Consider the DEDI [6] of the following form

$$\begin{cases} x'(t) = f(t, x(t)), & t \geq t_0,\ t \neq \tau_k,\ k \in \mathbb{Z}^+, \\ x(\tau_k^+) = I_k(x(r_k^-)), & r_k \in \mathcal{F}_{\tau_k}^{\sigma},\ k \in \mathbb{Z}^+, \\ x(t_0) = x_0, \end{cases} \tag{1}$$

where $\mathbb{Z}^+ = \{1, 2, \cdots\}$, $x(t^+)$ is the right limit of $x(t)$, $t_0 = \tau_0 < \tau_1 < \tau_2 < \cdots$, $\lim_{k\to\infty} \tau_k = \infty$, the function $f : [t_0, +\infty) \times \mathbb{R}^d \to \mathbb{R}^d$ is continuous in t and Lipschitz continuous with respect to the second variable in the following sense: there is a positive real constant α such that

$$\|f(t, x_1) - f(t, x_2)\| \leq \alpha \|x_1 - x_2\| \tag{2}$$

for arbitrary $t \in [t_0, \infty)$, $x_1, x_2 \in \mathbb{R}^d$, where $\|\cdot\|$ is any convenient norm on $\mathbb{R}^d$. Define the functions I_k to be from $\mathbb{R}^d$ to $\mathbb{R}^d$, $k \in \mathbb{Z}^+$. Assume that each function I_k ($k \in \mathbb{Z}^+$) is Lipschitz continuous, i.e., there is a positive constant β_k such that

$$\|I_k(x) - I_k(y)\| \leq \beta_k \|x - y\|, \ \text{ for } \forall x, y \in \mathbb{R}^d. \tag{3}$$

For any given impulse sequence τ_k, $k \in \mathbb{Z}^+$ and any constant $\sigma \in (0, 1]$, the set $\mathcal{F}_{\tau_k}^{\delta}$ is defined as follows

$$\mathcal{F}_{\tau_k}^{\sigma} = \{r_k : r_k = (1 - \sigma)\tau_{k-1} + \sigma\tau_k, k \in \mathbb{Z}^+\}.$$

Definition 1. *A function $x : [t_0, \infty) \to \mathbb{R}^d$ is said to be a solution of* (1) *, if*

(i) $\lim\limits_{t \to t_0^+} x(t) = x_0$,

(ii) *For $t \in (t_0, +\infty), t \neq \tau_k, k = 1, 2, \cdots$, $x(t)$ is differentiable and $x'(t) = f(t, x(t))$,*

(iii) *$x(t)$ is right continuous in $(t_0, +\infty)$ and $x(\tau_k) = I_k(x(r_k^-)), k = 1, 2, \cdots$.*

In order to investigate the asymptotical stability of $x(t)$, consider Equation (1) with another initial datum:

$$\begin{cases} y'(t) = f(t, y(t)), & t > t_0,\ t \neq \tau_k,\ k \in \mathbb{Z}^+, \\ y(\tau_k) = I_k(y(r_k^-)), & r_k \in \mathcal{F}_{\tau_k}^{\delta},\ k \in \mathbb{Z}^+, \\ y(t_0) = y_0. \end{cases} \tag{4}$$

Definition 2. *The exact solution $x(t)$ of* (1) *is said to be*

1. *stable if, for an arbitrary $\epsilon > 0$, there exists a positive number $\delta = \delta(\epsilon)$ such that, for any other solution $y(t)$ of* (4)*, $\|x_0 - y_0\| < \delta$ implies*

$$\|x(t) - y(t)\| < \epsilon,\ \forall\ t > t_0;$$

2. *asymptotically stable, if it is stable and $\lim\limits_{t \to \infty} \|x(t) - y(t)\| = 0$.*

Theorem 1. *Assume that there exists a positive constant γ such that $\tau_k - \tau_{k-1} \leq \gamma$, $k \in \mathbb{Z}^+$. The exact solution of* (1) *is asymptotically stable if there is a positive constant C such that*

$$\beta_k \mathrm{e}^{\alpha\sigma(\tau_k - \tau_{k-1})} \leq C < 1 \tag{5}$$

for arbitrary $k \in \mathbb{Z}^+$.

Proof. For arbitrary $t \in [\tau_k, \tau_{k+1})$, $k = 0, 1, 2, \cdots$, we can obtain that

$$\begin{aligned} \|x(t) - y(t)\| &= \|x(\tau_k) - y(\tau_k) + \int_{\tau_{k-1}}^{t} (f(s, x(s)) - f(s, y(s)))ds\| \\ &\leq \|x(\tau_k) - y(\tau_k)\| + \int_{\tau_k}^{t} \|f(s, x(s)) - f(s, y(s))\|ds \\ &\leq \|x(\tau_k) - y(\tau_k)\| + \alpha \int_{\tau_k}^{t} \|x(s) - y(s)\|ds \end{aligned}$$

By Gronwall's Theorem, for arbitrary $t \in [\tau_k, \tau_{k+1})$, $k = 0, 1, 2, \cdots$, we have

$$\|x(t) - y(t)\| \leq \|x(\tau_k) - y(\tau_k)\|\mathrm{e}^{\alpha(t - \tau_k)},$$

which implies

$$\|x(r_{k+1}^-) - y(r_{k+1}^-)\| \leq \|x(\tau_k) - y(\tau_k)\|\mathrm{e}^{\alpha(r_{k+1} - \tau_k)} = \|x(\tau_k) - y(\tau_k)\|\mathrm{e}^{\alpha\sigma(\tau_{k+1} - \tau_k)}.$$

Consequently, we can obtain that

$$\begin{aligned} &\|x(\tau_{k+1}) - y(\tau_{k+1})\| \\ &= \|I_{k+1}(x(r_{k+1}^-)) - I_{k+1}(y(r_{k+1}^-))\| \\ &\leq \beta_{k+1}\|x(r_{k+1}^-) - y(r_{k+1}^-)\| \\ &\leq \|x(\tau_k) - y(\tau_k)\|\beta_{k+1}\mathrm{e}^{\alpha\sigma(\tau_{k+1} - \tau_k)}. \end{aligned}$$

Therefore, by the method of introduction and the conditions (3) and (5), for arbitrary $t \in [\tau_k, \tau_{k+1})$, $k = 0, 1, 2, \cdots$, we can obtain that

$$\begin{aligned}
&\|x(t) - y(t)\| \\
&\leq \|x_0 - y_0\| \mathrm{e}^{\alpha(t-\tau_k)} \prod_{i=1}^{k} \beta_i \mathrm{e}^{\alpha\sigma(\tau_i - \tau_{i-1})} \\
&\leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha(t-\tau_k)} \\
&\leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha(\tau_{k+1}-\tau_k)} \\
&\leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha\gamma},
\end{aligned}$$

which implies $\|x(\tau_{k+1}^-) - y(\tau_{k+1}^-)\| \leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha\gamma}$ and $\|x(\tau_{k+1}) - y(\tau_{k+1})\| \leq \|x_0 - y_0\| C^{k+1}$. Hence for an arbitrary $\epsilon > 0$, there exists $\delta = \mathrm{e}^{-\alpha\gamma}\epsilon$ such that $\|x_0 - y_0\| < \delta$ implies

$$\|x(t) - y(t)\| \leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha\gamma} \leq \|x_0 - y_0\| \mathrm{e}^{\alpha\gamma} < \epsilon$$

for arbitrary $t \in [\tau_k, \tau_{k+1})$, $k = 0, 1, 2, \cdots$, i.e.,

$$\|x(t) - y(t)\| < \epsilon, \ \forall \ t > t_0.$$

So the exact solution of (1) is stable. Obviously, for arbitrary $t \in [\tau_k, \tau_{k+1})$, $k = 0, 1, 2, \cdots$,

$$\|x(t) - y(t)\| \leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha\gamma} \to 0, \ k \to \infty.$$

Similarly, we can also obtain that

$$\|x(\tau_{k+1}^-) - y(\tau_{k+1}^-)\| \leq C^k \|x_0 - y_0\| \mathrm{e}^{\alpha\gamma} \to 0, \ k \to \infty,$$

and

$$\|x(\tau_{k+1}^+) - y(\tau_{k+1}^+)\| \leq C^{k+1} \|x_0 - y_0\| \to 0, \ k \to \infty.$$

Consequently, the exact solution of (1) is asymptotically stable. □

From the proof of Theorem 1, we can obtain the following result.

Remark 1. *If condition* (5) *of Theorem 1 is changed into the weaker condition*

$$\beta_k \mathrm{e}^{\alpha\sigma(\tau_k - \tau_{k-1})} \leq 1, \ \forall \ k \in \mathbb{Z}^+, \tag{6}$$

then the exact solution of (1) *is stable.*

3. Scheme 1 Impulsive Discrete Runge–Kutta Methods

In the following part of this paper, we will focus on the case of $0 < \sigma < 1$; the special case of $\sigma = 1$ has already been studied in paper [29]. The simplest and most straightforward idea is to take all points in the set $\{\tau_k, r_k : \mathbb{Z}^+\}$ as the numerical mesh. For convenience, we divide the intervals $[\tau_{k-1}, r_k]$ and $[r_k, \tau_k)$ ($k \in \mathbb{Z}^+$) equally by m; m is a positive integer. In this case, for $k \in \mathbb{N}$, the step sizes are as follows

$$h_{k,l} = \begin{cases} \bar{h}_{k,1} := \frac{r_k - \tau_k}{m}, & l = 1, 2, \cdots, m \\ \bar{h}_{k,2} := \frac{\tau_{k+1} - r_k}{m}, & l = m+1, m+2, \cdots, 2m, \end{cases} \tag{7}$$

which implies that the mesh point $t_{k,0} = \tau_k$, $t_{k,2m} = \tau_{k+1}^-$, $t_{k,l} = \tau_k + \sum_{j=0}^{l} h_{k,j}$, $\forall k \in \mathbb{N}$, $l = 1, 2, \cdots, 2m-1$.

The S1IRKM for DEDI (1) can be constructed as follows:

$$\begin{cases} X^i_{k,l+1} = x_{k,l} + h_{k,l+1} \sum\limits_{j=1}^{v} a_{ij} f(t^j_{k,l+1}, X^j_{k,l+1}), & k \in \mathbb{N}, i = 1, 2, \cdots, v, \\ x_{k,l+1} = x_{k,l} + h_{k,l+1} \sum\limits_{i=1}^{v} b_i f(t^i_{k,l+1}, X^i_{k,l+1}), & l = 0, 1, \cdots, 2m-1, \\ x_{k+1,0} = I_{k+1}(x_{k,m}), \\ x_{0,0} = x_0, \end{cases} \tag{8}$$

where v is referred to as the number of stages, $t^i_{k,l} = t_{k,l} + c_i h_{k,l+1}$, $x_{k,l}$ is an approximation to the exact solution $x(t_{k,l})$ and $X^i_{k,l+1}$ is an approximation to the exact solution $x(t^i_{k,l+1})$, $k \in \mathbb{N}, l = 0, 1, \cdots, 2m-1, i = 1, 2, \cdots, v$. The weights b_i, the abscissae $c_i = \sum\limits_{j=1}^{v} a_{ij}$ and the matrix $A = [a_{ij}]^v_{i,j=1}$ will be denoted by (A, b, c).

3.1. Convergence of S1IRKMs

In order to study the convergence of S1IRKMs, the DEDI (1) is restricted to the interval $[0, T]$ in this subsection. For convenience, assume that there exists a positive integer N such that $T = \tau_N$.

To analyze the local truncation errors of S1IRKM (8) for DEDI (1), consider the following local problem

$$\begin{cases} Z^i_{k,l+1} = z_{k,l} + h_{k,l+1} \sum\limits_{j=1}^{v} a_{ij} f(t^j_{k,l+1}, Z^j_{k,l+1}), & k \in \mathbb{N}, i = 1, 2, \cdots, v, \\ z_{k,l+1} = z_{k,l} + h_{k,l+1} \sum\limits_{i=1}^{v} b_i f(t^i_{k,l+1}, Z^i_{k,l+1}), & l = 0, 1, \cdots, 2m-1, \end{cases} \tag{9}$$

where $z_{k,l} = x(t_{k,l})$, $k = 0, 1, 2, \cdots, N-1$, $l = 0, 1, 2, \cdots, 2m-1$.

Because it can be seen as a problem of ordinary differential equation (see [44–46])when we consider the local problem, we can directly obtain the following result.

Theorem 2. *Consider the DEDI* (1) *where $f(t, x)$ is C^p-continuous in $[t_0, T] \times \mathbb{R}^d$. If the corresponding Runge–Kutta method is convergent of order p, then local errors between the numerical solutions obtained from* (9) *and the exact solutions obtained from DEDI* (1) *satisfy that there exists a constant C such that, for arbitrary $k = 0, 1, 2, \cdots, N-1$, $l = 0, 1, 2, \cdots, 2m-2$,*

$$R_{k,l+1} := \|z_{k,l+1} - x(t_{k,l+1})\| \leq C h^{p+1}_{k,l+1}$$

and

$$R_{k,2m} := \|z_{k,2m} - x(\tau^-_{k+1})\| \leq C h^{p+1}_{k,2m}.$$

Theorem 3. *Assume that $f(t, x)$ of DEDI* (1) *is C^p-continuous in $[t_0, T] \times \mathbb{R}^d$, the functions I_k are bounded, and Lipschitz conditions* (2) *and* (3) *hold. If the corresponding Runge–Kutta method is convergent of order p, then the global errors $e_{k,l}$ between the numerical solutions $x_{k,l}$ obtained from* (8) *and the exact solutions $x(t_{k,l})$ obtained from DEDI* (1) *satisfy that there exists a constant C_1 such that, when h is small enough, for arbitrary $k = 0, 1, 2, \cdots, N-1$, $l = 0, 1, 2, \cdots, 2m-1$,*

$$e_{k,l} := \|x_{k,l} - x(t_{k,l})\| \leq C_1 h^p \tag{10}$$

and

$$e_{k,2m} := \|x_{k,2m} - x(\tau^-_{k+1})\| \leq C_1 h^p, \tag{11}$$

where $h = \max\limits_{k,l}\{h_{k,l}\} = \max\limits_{k}\{\bar{h}_{k,1}, \bar{h}_{k,2}\}$.

Proof. From (8) and (9), we can obtain that

$$\begin{aligned}
&\|X^i_{k,l+1} - Z^i_{k,l+1}\| \\
&= \|x_{k,l} - z_{k,l} + h_{k,l+1}\sum_{j=1}^{v} a_{ij}(f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Z^j_{k,l+1}))\| \\
&\le \|x_{k,l} - z_{k,l}\| + h_{k,l+1}\sum_{j=1}^{v} |a_{ij}|\|f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Z^j_{k,l+1})\| \\
&\le \|x_{k,l} - z_{k,l}\| + \alpha h_{k,l+1}\sum_{j=1}^{v} |a_{ij}|\|X^j_{k,l+1} - Z^j_{k,l+1}\| \\
&\le \|x_{k,l} - z_{k,l}\| + \alpha h\left(\max_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)\max_{1\le j\le v}\{\|X^j_{k,l+1} - Z^j_{k,l+1}\|\}
\end{aligned}$$

which implies that, for $h < \alpha^{-1}\left(\max_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)^{-1}$,

$$\max_{1\le i\le v}\{\|X^i_{k,l+1} - Z^i_{k,l+1}\|\} \le \Lambda_1\|x_{k,l} - z_{k,l}\|.$$

where $\Lambda = \left(1 - \alpha h\left(\max\limits_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)\right)^{-1}$. Hence,

$$\begin{aligned}
&\|x_{k,l+1} - z_{k,l+1}\| \\
&= \|x_{k,l} - z_{k,l} + h_{k,l+1}\sum_{j=1}^{v} b_j(f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l}, Z^j_{k,l}))\| \\
&\le \|x_{k,l} - z_{k,l}\| + h_{k,l+1}\sum_{j=1}^{v} |b_j|\|f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Z^j_{k,l+1})\| \\
&\le \|x_{k,l} - z_{k,l}\| + \alpha h_{k,l+1}\left(\sum_{j=1}^{v}|b_j|\right)\max_{1\le i\le v}\{\|X^i_{k,l+1} - Z^i_{k,l+1}\|\} \\
&\le (1 + \alpha B\Lambda h_{k,l+1})\|x_{k,l} - z_{k,l}\|
\end{aligned}$$

where $B = \sum\limits_{j=1}^{v}|b_j|$. From Theorem 2, we have

$$\begin{aligned}
\bar{R}_{k,1} &:= \max_{1\le l\le m} R_{k,l} \le C\bar{h}_{k,1}h^p, \\
/\bar{R}_{k,2} &:= \max_{m+1\le l\le 2m} R_{k,l} \le C\bar{h}_{k,2}h^p.
\end{aligned}$$

If $0 \le l \le m-1$,

$$\begin{aligned}
e_{k,l+1} &:= \|x(t_{k,l+1}) - x_{k,l+1}\| \\
&\le \|x(t_{k,l+1}) - z_{k,l+1}\| + \|z_{k,l+1} - x_{k,l+1}\| \\
&\le \left(1 + \alpha B\Lambda\bar{h}_{k,1}\right)e_{k,l} + \bar{R}_{k,1} \\
&\le \left(1 + \alpha B\Lambda\bar{h}_{k,1}\right)^{l+1}e_{k,0} + \left(\left(1 + \alpha B\Lambda\bar{h}_{k,1}\right)^{l+1} - 1\right)\frac{\bar{R}_{k,1}}{\alpha\beta\Lambda\bar{h}_{k,1}} \\
&\le \mathrm{e}^{(l+1)\alpha B\Lambda\bar{h}_{k,1}}e_{k,0} + \left(\mathrm{e}^{(l+1)\alpha B\Lambda\bar{h}_{k,1}} - 1\right)\frac{\bar{R}_{k,1}}{\alpha B\Lambda\bar{h}_{k,1}} \\
&\le \mathrm{e}^{\alpha B\Lambda\sigma(\tau_{k+1}-\tau_k)}e_{k,0} + \left(\mathrm{e}^{\alpha B\Lambda\sigma(\tau_{k+1}-\tau_k)} - 1\right)\frac{Ch^p}{\alpha B\Lambda} \\
&\le \mathrm{e}^{\alpha B\Lambda\sigma T}e_{k,0} + \left(\mathrm{e}^{\alpha B\Lambda_1\sigma T} - 1\right)\frac{Ch^p}{\alpha B\Lambda}
\end{aligned} \tag{12}$$

Otherwise, if $m \le l \le 2m-1$,

$$\begin{aligned} e_{k,l+1} &:= \|x(t_{k,l+1}) - x_{k,l+1}\| \\ &\le \|x(t_{k,l+1}) - z_{k,l+1}\| + \|z_{k,l+1} - x_{k,l+1}\| \\ &\le \left(1 + \alpha B\Lambda \bar{h}_{k,2}\right) e_{k,l} + \bar{R}_{k,2} \\ &\le \left(1 + \alpha B\Lambda \bar{h}_{k,2}\right)^{l-m+1} e_{k,m} + \left(\left(1 + \alpha B\Lambda \bar{h}_{k,2}\right)^{l-m+1} - 1\right) \frac{\bar{R}_{k,2}}{\alpha B\Lambda \bar{h}_{k,2}} \\ &\le \mathrm{e}^{(l-m+1)\alpha B\Lambda \bar{h}_{k,2}} e_{k,m} + \left(\mathrm{e}^{(l-m+1)\alpha B\Lambda \bar{h}_{k,2}} - 1\right) \frac{\bar{R}_{k,2}}{\alpha\beta\Lambda \bar{h}_{k,2}} \\ &\le \mathrm{e}^{\alpha B\Lambda(1-\sigma)(\tau_{k+1}-\tau_k)} e_{k,m} + \left(\mathrm{e}^{\alpha B\Lambda(1-\sigma)(\tau_{k+1}-\tau_k)} - 1\right) \frac{Ch^p}{\alpha\beta\Lambda} \\ &\le \mathrm{e}^{\alpha B\Lambda(1-\sigma)T} e_{k,m} + \left(\mathrm{e}^{\alpha B\Lambda(1-\sigma)T} - 1\right) \frac{Ch^p}{\alpha B\Lambda} \\ &\le \mathrm{e}^{\alpha B\Lambda T} e_{k,0} + \left(\mathrm{e}^{\alpha B\Lambda T} - 1\right) \frac{Ch^p}{\alpha B\Lambda} \end{aligned} \tag{13}$$

Otherwise,

$$\begin{aligned} e_{k+1,0} &:= \|x_{k+1,0} - x(t_{k+1,0})\| = \|x_{k+1,0} - x(\tau_{k+1})\| \\ &= \|I_{k+1}(x_{k,m}) - I_{k+1}(x(\tau_{k+1}^-))\| \\ &\le \beta_{k+1}\|x_{k,m} - x(t_{k,m}^-)\| \\ &\le \beta_{k+1}\mathrm{e}^{\alpha B\Lambda\sigma T} e_{k,0} + \beta_{k+1}\left(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\right) \frac{Ch^p}{\alpha B\Lambda} \\ &\le \bar{B}\mathrm{e}^{\alpha B\Lambda\sigma T} e_{k,0} + \bar{B}\left(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\right) \frac{Ch^p}{\alpha B\Lambda} \\ &\le \left(\bar{B}\mathrm{e}^{\alpha B\Lambda\sigma T}\right)^{k+1} e_{0,0} + \left(\left(\bar{B}\mathrm{e}^{\alpha B\Lambda\sigma T}\right)^{k+1} - 1\right) \frac{Ch^p}{\alpha B\Lambda}, \end{aligned} \tag{14}$$

where $\bar{B} = \max\{\beta_1, \beta_2, \cdots, \beta_N\}$. In fact, we can choose $\|x_{0,0} - x_0\| = O(h^p))$; that is, $\|e_{0,0}\| = O(h^p)$. For convenience, we only choose $x_{0,0} = x_0$, which impiles that $e_{0,0} = 0$. So from (14), for arbitrary $k = 0, 1, 2, \ldots, N$, we have

$$e_{k+1,0} \le C_2 h^p \tag{15}$$

where $C_2 = \left(\left(\bar{B}\mathrm{e}^{\alpha B\Lambda\sigma T}\right)^N - 1\right) \frac{C}{\alpha B\Lambda}$. Combining (12) and (15), for $0 \le l \le m-1$, we can obtain that

$$e_{k,l+1} \le C_3 h^p \tag{16}$$

where $C_3 = \mathrm{e}^{\alpha B\Lambda\sigma T} C_2 + \left(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\right) \frac{C}{\alpha B\Lambda}$. Similarly, combining (13) and (16), for $m \le l \le 2m-1$, we obtain

$$e_{k,l+1} \le C_4 h^p \tag{17}$$

where $C_4 = \mathrm{e}^{\alpha B\Lambda T} C_2 + \left(\mathrm{e}^{\alpha B\Lambda T} - 1\right) \frac{Ch^p}{\alpha B\Lambda}$. Consequently, from (15), (16) and (17), we know that (10) and (11) hold for $C_1 = \max\{C_2, C_3, C_4\}$ and $h < \alpha^{-1}\left(\max_{1\le i\le v} \sum_{j=1}^{v} |a_{ij}|\right)^{-1}$. □

3.2. Asymptotical Stability of S1IRKMs

In order to study asymptotical stability of S1IRKMs, we also consider S1IRKM for DEDI (4) as follows:

$$\begin{cases} Y_{k,l+1}^i = y_{k,l} + h_{k,l+1} \sum\limits_{j=1}^{v} a_{ij} f(t_{k,l+1}^j, Y_{k,l+1}^j), \ \ k \in \mathbb{N}, i = 1, 2, \cdots, v, \\ y_{k,l+1} = y_{k,l} + h_{k,l+1} \sum\limits_{i=1}^{v} b_i f(t_{k,l+1}^i, Y_{k,l+1}^i), \ \ l = 0, 1, \cdots, 2m-1, \\ y_{k+1,0} = I_{k+1}(y_{k,m}), \\ y_{0,0} = y_0. \end{cases} \tag{18}$$

Definition 3. *The S1IRKM* (8) *for DEDI* (1) *is said to be*

1. *stable, if* $\exists \bar{h} > 0$,
   (i) $I - z_{k,l}A$ *is invertible for all* $z_{k,l} = \alpha h_{k,l}$, $h_{k,l} \le \bar{h}$, $\forall \in \mathbb{N}$, $l = 1, 2, \cdots, 2m$,

(ii) for an arbitrary $\epsilon > 0$, there exists such a positive number $\delta = \delta(\epsilon)$ that, for any other numerical solutions of (29)*, $\|x_0 - y_0\| < \delta$ implies*

$$\|X_k - Y_k\| < \epsilon, \ \forall\, k \in \mathbb{N},$$

where $X_k = (x_{k,0}, x_{k,1}, \cdots, x_{k,m})^T$ and $Y_k = (y_{k,0}, y_{k,1}, \cdots, y_{k,m})^T$.

2. *asymptotically stable, if it is stable and if $\exists \bar{h}_1 > 0$, for $h_{k,l} \leq \bar{h}_1$, $k \in \mathbb{N}$, $l = 1, 2, \cdots, 2m$; the following holds:*

$$\lim_{k\to\infty} \|X_k - Y_k\| = 0.$$

Lemma 1. *([44–46,48]). The $(\boldsymbol{j}, \boldsymbol{k})$-Padé approximation to e^z is given by*

$$\boldsymbol{R}(z) = \frac{P_j(z)}{Q_k(z)}, \tag{19}$$

where

$$P_j(z) = 1 + \frac{\boldsymbol{j}}{\boldsymbol{j}+\boldsymbol{k}} \cdot z + \frac{\boldsymbol{j}(\boldsymbol{j}-1)}{(\boldsymbol{j}+\boldsymbol{k})(\boldsymbol{j}+\boldsymbol{k}-1)} \cdot \frac{z^2}{2!} + \cdots + \frac{\boldsymbol{j}!\boldsymbol{k}!}{(\boldsymbol{j}+\boldsymbol{k})!} \cdot \frac{z^{\boldsymbol{j}}}{\boldsymbol{j}!},$$

$$Q_k(z) = 1 - \frac{\boldsymbol{k}}{\boldsymbol{j}+\boldsymbol{k}} \cdot z + \frac{\boldsymbol{k}(\boldsymbol{k}-1)}{(\boldsymbol{j}+\boldsymbol{k})(\boldsymbol{j}+\boldsymbol{k}-1)} \cdot \frac{z^2}{2!} + \cdots + (-1)^{\boldsymbol{k}} \cdot \frac{\boldsymbol{k}!\boldsymbol{j}!}{(\boldsymbol{j}+\boldsymbol{k})!} \cdot \frac{z^{\boldsymbol{k}}}{\boldsymbol{k}!},$$

with error

$$\mathrm{e}^z - \boldsymbol{R}(z) = (-1)^{\boldsymbol{k}} \cdot \frac{\boldsymbol{j}!\boldsymbol{k}!}{(\boldsymbol{j}+\boldsymbol{k})!(\boldsymbol{j}+\boldsymbol{k}+1)!} \cdot z^{\boldsymbol{j}+\boldsymbol{k}+1} + O(z^{\boldsymbol{j}+\boldsymbol{k}+2}).$$

It is the unique rational approximation to e^z of order $\boldsymbol{j}+\boldsymbol{k}$, such that the degrees of numerator and denominator are $\boldsymbol{j}$ and $\boldsymbol{k}$, respectively.

Lemma 2. *([49–51]). Assume that $\boldsymbol{R}(z)$ is the $(\boldsymbol{j}, \boldsymbol{k})$-Padé approximation to e^z. Then $\boldsymbol{R}(z) < \mathrm{e}^z$ for all $z > 0$ if and only if $\boldsymbol{k}$ is even.*

Theorem 4. *Assume that $\boldsymbol{R}(z)$ is the stability function of S1IRKM* (8)*; that is,*

$$\boldsymbol{R}(z) = 1 + zb^T(I - zA)^{-1}e = \frac{P_j(z)}{Q_k(z)},$$

where $e = (1, 1, \cdots, 1)^T$ is a v-dimensional vector. Let the coefficients of the corresponding Runge–Kutta method of S1IRKM (8) *be nonnegative, that is, $a_{ij} \geq 0$ and $b_i \geq 0$, $1 \leq i \leq v$, $1 \leq j \leq v$. Under the conditions of Theorem 1, S1IRKM* (8) *for* (1) *is asymptotically stable when the step sizes satisfy* (7) *and $m > M$, if $\boldsymbol{k}$ is even, where $M = \inf\{m : I - zA$ is invertible and $(I - zA)^{-1}e \geq 0, z = \alpha h, h = \max_{k,l}\{h_{k,l}\}\}$. (The last inequality should be interpreted entrywise.)*

Proof. Because $a_{ij} \geq 0$ and $b_i \geq 0$,$1 \leq i \leq v$, $1 \leq j \leq v$, we can obtain that

$$\begin{aligned}
&\|X^i_{k,l+1} - Y^i_{k,l+1}\| \\
&= \|x_{k,l} - y_{k,l} + h_{k,l+1} \sum_{j=1}^{v} a_{ij}(f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1}))\| \\
&\leq \|x_{k,l} - y_{k,l}\| + h_{k,l+1} \sum_{j=1}^{v} a_{ij}\|f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1})\| \\
&\leq \|x_{k,l} - y_{k,l}\| + \alpha h_{k,l+1} \sum_{j=1}^{v} a_{ij}\|X^j_{k,l+1} - Y^j_{k,l+1}\|.
\end{aligned}$$

Since $m > M$, $(I - \alpha h_{k,l+1}A)^{-1}e \geq 0$. Hence

$$[\|X^i_{k,l+1} - Y^i_{k,l+1}\|] \leq (I - \alpha h_k A)^{-1}e\|x_{k,l} - y_{k,l}\|$$

where $[\|X^i_{k,l+1} - Y^i_{k,l+1}\|] = (\|X^1_{k,l+1} - Y^1_{k,l+1}\|, \|X^2_{k,l+1} - Y^2_{k,l+1}\|, \cdots, \|X^v_{k,l+1} - Y^v_{k,l+1}\|)^T$. By Lemmas 1 and 2, we can obtain

$$\begin{aligned}
&\|x_{k,l+1} - y_{k,l+1}\| \\
&= \|x_{k,l} - y_{k,l} + h_{k,l+1}\sum_{j=1}^{v} b_j(f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l}, Y^j_{k,l}))\| \\
&\leq \|x_{k,l} - y_{k,l}\| + h_{k,l+1}\sum_{j=1}^{v} b_j\|f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1})\| \\
&\leq \|x_{k,l} - y_{k,l}\| + \alpha h_{k,l+1}\sum_{j=1}^{v} b_j\|X^j_{k,l+1} - Y^j_{k,l+1}\| \\
&= \|x_{k,l} - y_{k,l}\| + \alpha h_{k,l+1}b^T[\|X^i_{k,l+1} - Y^i_{k,l+1}\|] \\
&\leq (1 + \alpha h_{k,l+1}b^T(I - \alpha h_{k,l+1}A)^{-1}e)\|x_{k,l} - y_{k,l}\| \\
&= \mathbf{R}(\alpha h_{k,l+1})\|x_{k,l} - y_{k,l}\| \\
&\leq \mathrm{e}^{\alpha h_{k,l+1}}\|x_{k,l} - y_{k,l}\|.
\end{aligned}$$

Hence for arbitrary $k = 0, 1, 2, \cdots$ and $l = 0, 1, \cdots, 2m$, we have

$$\|x_{k,l} - y_{k,l}\| \leq \|x_{k,0} - y_{k,0}\|\mathrm{e}^{\alpha(t_{k,l} - \tau_k)}.$$

which implies

$$\begin{aligned}
&\|x_{k+1,0} - y_{k+1,0}\| = \|I_{k+1}(x_{k,m}) - I_{k+1}(y_{k,m})\| \\
&\leq \beta_{k+1}\|x_{k,m} - y_{k,m}\| \\
&\leq \beta_{k+1}\|x_{k,0} - y_{k,0}\|\mathrm{e}^{\alpha(t_{k,m} - \tau_k)} \\
&= \beta_{k+1}\|x_{k,0} - y_{k,0}\|\mathrm{e}^{\alpha\sigma(\tau_{k+1} - \tau_k)}.
\end{aligned}$$

Therefore, by the method of introduction and condition (5), we can obtain that

$$\begin{aligned}
&\|x_{k,l} - y_{k,l}\| \\
&\leq \|x_0 - y_0\|\left(\beta_1\mathrm{e}^{\alpha\sigma(\tau_1 - \tau_0)}\right)\left(\beta_2\mathrm{e}^{\alpha\sigma(\tau_2 - \tau_1)}\right)\left(\beta_k\mathrm{e}^{\alpha\sigma(\tau_k - \tau_{k-1})}\right)\mathrm{e}^{\alpha(t_{k,l} - \tau_k)} \\
&\leq \|x_0 - y_0\|C^k\mathrm{e}^{\alpha\gamma}
\end{aligned}$$

which implies that S1IRKM for DEDI (1) is asymptotically stable. □

Remark 2. *For z sufficiently close to zero, the matrix $I - zA$ is invertible and $(I - zA)^{-1}e \geq 0$. Therefore, taking step sizes according to (7) and $m \geq M$ and $M = \inf\{m : I - zA$ is invertible and $(I - zA)^{-1}e \geq 0, z = \alpha h, h = \max_{k,l}\{h_{k,l}\}\}$ in Theorem 4 is reasonable.*

Remark 3. *When the corresponding Runge–Kutta method chooses these formats as follows, which is also the special case $\boldsymbol{k} = 0$, the S1IRKM satisfies Theorem 4.*
(1) Explicit Euler method
(2) Two-stage second-order explicit Runge–Kutta methods

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & 0 & 1 \end{array}$$

Modified Euler method

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Heun′s method, order 2

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \frac{3}{4} & \frac{3}{4} & 0 \\ \hline & \frac{1}{3} & \frac{2}{3} \end{array}$$

Ralston′s method

(3) Three-stage third-order explicit Runge–Kutta methods

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{3} & 0 & 0 \\ \frac{2}{3} & 0 & \frac{2}{3} & 0 \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array}$$

Heun′s method, order 3

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \frac{2}{3} & \frac{2}{3} & 0 & 0 \\ \frac{2}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array}$$

Runge − −Kutta method, order 3

(4) The classical four-stage fourth-order explicit Runge–Kutta method

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

Unfortunately, we cannot obtain the p-stage explicit Runge–Kutta methods of order p for $p \geq 5$ because of the Butcher Barriers (See [44] (Theorem 370B, pp.259) or [46] (Theorem 5.1 pp.173)).

3.3. Asymptotical Stability of S1IθMs

Similar to (8), the scheme 1 impulsive θ method (S1IθM) for (1) is constructed as follows:

$$\begin{cases} x_{k,l+1} = x_{k,l} + h_{k,l+1}((1-\theta)f(t_{k,l}, x_{k,l}) + \theta f(t_{k,l+1}, x_{k,l+1})), & l = 0, 1, \ldots, 2m-1 \\ x_{k+1,0} = I_{k+1}(x_{k,m}), & k \in \mathbb{N} \\ x_{0,0} = x_0. \end{cases} \tag{20}$$

From [49] (Lemma 2 and Lemma 3) or [18] (Theorem 2.2 and Lemma 2.3), we can obtain the following result.

Lemma 3. *When z is small enough,*

$$1 + \frac{z}{1 - z\theta} \leq \mathrm{e}^z$$

if and only if $0 \leq \theta \leq \varphi(1)$, *where* $\varphi(x) = \frac{1}{x} - \frac{1}{\mathrm{e}^x - 1}$.

Theorem 5. *Under the conditions of Theorem 1, if* $0 \leq \theta \leq \varphi(1)$, *S1I$\theta$M* (20) *for DEDI* (1) *is asymptotically stable when the step sizes satisfy* (7) *and are small enough.*

Proof. Obviously, we can obtain

$$\begin{aligned} & \|x_{k,l+1} - y_{k,l+1}\| \\ \leq & \|x_{k,l} - y_{k,l} + (1-\theta)h_{k,l+1}(f(t_{k,l}, x_{k,l}) - f(t_{k,l}, y_{k,l}))\| \\ & + \theta h_{k,l+1}\|f(t_{k,l+1}, x_{k,l+1}) - f(t_{k,l+1}, y_{k,l+1})\| \\ \leq & (1 + (1-\theta)\alpha h_{k,l+1})\|x_{k,l} - y_{k,l}\| + \theta\alpha h_{k,l+1}\|x_{k,l+1} - y_{k,l+1}\| \end{aligned}$$

which implies

$$\|x_{k,l+1} - y_{k,l+1}\| \leq \frac{1 + (1-\theta)\alpha h_{k,l+1}}{1 - \theta\alpha h_{k,l+1}} \cdot \|x_{k,l} - y_{k,l}\|$$

Therefore, by Lemma 3 and the method of introduction, we can obtain that

$$\|x_{k,l+1} - y_{k,l+1}\| \leq \mathrm{e}^{\alpha h_{k,l+1}} \|x_{k,l} - y_{k,l}\|.$$

Similar to the proof of Theorem 4, we can obtain that S1IθM (20) for DEDI (1) is asymptotically stable. □

4. Scheme 2 Impulsive Discrete Runge–Kutta Methods

In this section, S2IRKM for DEDI (1) can be constructed as follows:

$$\begin{cases} X^i_{k,l+1} = x_{k,l} + h_k \sum\limits_{j=1}^{v} a_{ij} f(t^j_{k,l+1}, X^j_{k,l+1}), & k \in \mathbb{N}, i = 1, 2, \cdots, v, \\ x_{k,l+1} = x_{k,l} + h_k \sum\limits_{i=1}^{v} b_i f(t^i_{k,l+1}, X^i_{k,l+1}), & l = 1, 2, \cdots, m-1, \\ x_{k+1,0} = I_{k+1}(x_{k,\lfloor \sigma m \rfloor}), \\ x_{0,0} = x_0, \end{cases} \tag{21}$$

where $h_k = \frac{\tau_{k+1} - \tau_k}{m}$, $t_{k,l} = \tau_k + l h_k$, $t^i_{k,l} = t_{k,l} + c_i h_k$, $x_{k,l}$ is an approximation to the exact solution $x(t_{k,l})$ and $X^i_{k,l+1}$ is an approximation to the exact solution $x(t^i_{k,l+1})$, $k \in \mathbb{N} = \{0, 1, 2, \cdots\}$, $l = 0, 1, \cdots, m-1$, $i = 1, 2, \cdots, v$; v is referred to as the number of stages.

4.1. Convergence of S2IRKMs

In order to study the convergence of S2IRKMs, DEDI (1) is restricted to the interval $[0, T]$ in this subsection. For convenience, assume that there exists a positive integer N such that $T = \tau_N$.

To analyze the local truncation errors of S2RKM (21) for DEDI (1), consider the following local problem

$$\begin{cases} Z^i_{k,l+1} = z_{k,l} + h_k \sum\limits_{j=1}^{v} a_{ij} f(t^j_{k,l+1}, Z^j_{k,l+1}), & k \in \mathbb{N}, i = 1, 2, \cdots, v, \\ z_{k,l+1} = z_{k,l} + h_k \sum\limits_{i=1}^{v} b_i f(t^i_{k,l+1}, Z^i_{k,l+1}), & l = 0, 1, \cdots, m-1, \end{cases} \tag{22}$$

where $z_{k,l} = x(t_{k,l})$, $k = 0, 1, 2, \cdots, N-1$, $l = 0, 1, 2, \cdots, m-1$.

Because it can be seen as a problem of ordinary differential equation (see [44–46]) when we consider the local problem, we can directly obtain the following result.

Theorem 6. *Consider DEDI* (1) *where $f(t, x)$ is C^p-continuous in $[t_0, T] \times \mathbb{R}^d$. If the corresponding Runge–Kutta method is convergent of order p, then local errors between the numerical solutions obtained from* (22) *and the exact solutions obtained from DEDI* (1) *satisfy that there exists a constant C_5 such that, for arbitrary $k = 0, 1, 2, \cdots, N-1$, $l = 0, 1, 2, \cdots, m-2$,*

$$R_{k,l+1} := \|z_{k,l+1} - x(t_{k,l+1})\| \leq C_5 h_k^{p+1}$$

and

$$R_{k,m} := \|z_{k,m} - x(\tau^-_{k+1})\| \leq C_5 h_k^{p+1}.$$

Theorem 7. *Assume that $f(t,x)$ of DEDI* (1) *is C^p-continuous in $[t_0,T]\times\mathbb{R}^d$, the functions I_k are bounded, and Lipschitz conditions* (2) *and* (3) *hold. If the corresponding Runge–Kutta method is convergent of order p, then the global errors $e_{k,l}$ between the numerical solutions $x_{k,l}$ obtained from* (8) *and the exact solutions $x(t_{k,l})$ obtained from DEDI* (1) *satisfy that there exists a constant C_6 such that, for arbitrary $k=0,1,2,\cdots,N-1$, $l=0,1,2,\cdots,m-1$,*

$$e_{k,l} := \|x_{k,l}-x(t_{k,l})\| \le C_6 h \tag{23}$$

and

$$e_{k,m} := \|x_{k,m}-x(\tau_{k+1}^-)\| \le C_6 h, \tag{24}$$

where $h=\max\limits_{0\le k<N}\{h_k\}$.

Proof. From (8) and (9), we can obtain that

$$\begin{aligned}
&\|X_{k,l+1}^i - Z_{k,l+1}^i\| \\
&= \|x_{k,l}-z_{k,l}+h_k\sum_{j=1}^{v}a_{ij}(f(t_{k,l+1}^j,X_{k,l+1}^j)-f(t_{k,l+1}^j,Z_{k,l+1}^j))\| \\
&\le \|x_{k,l}-z_{k,l}\|+h_k\sum_{j=1}^{v}|a_{ij}|\|f(t_{k,l+1}^j,X_{k,l+1}^j)-f(t_{k,l+1}^j,Z_{k,l+1}^j)\| \\
&\le \|x_{k,l}-z_{k,l}\|+\alpha h_k\sum_{j=1}^{v}|a_{ij}|\|X_{k,l+1}^j-Z_{k,l+1}^j\| \\
&\le \|x_{k,l}-z_{k,l}\|+\alpha h\left(\max_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)\max_{1\le j\le v}\{\|X_{k,l+1}^j-Z_{k,l+1}^j\|\}
\end{aligned}$$

which implies that

$$\max_{1\le i\le v}\{\|X_{k,l+1}^i-Z_{k,l+1}^i\|\}\le\Lambda\|x_{k,l}-z_{k,l}\|,$$

where $\Lambda=\left(1-\alpha h\left(\max\limits_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)\right)^{-1}$. Hence

$$\begin{aligned}
&\|x_{k,l+1}-z_{k,l+1}\| \\
&= \|x_{k,l}-z_{k,l}+h_k\sum_{j=1}^{v}b_j(f(t_{k,l+1}^j,X_{k,l+1}^j)-f(t_{k,l}^j,Z_{k,l}^j))\| \\
&\le \|x_{k,l}-z_{k,l}\|+h_k\sum_{j=1}^{v}|b_j|\|f(t_{k,l+1}^j,X_{k,l+1}^j)-f(t_{k,l+1}^j,Z_{k,l+1}^j)\| \\
&\le \|x_{k,l}-z_{k,l}\|+\alpha h_k\left(\sum_{j=1}^{v}|b_j|\right)\max_{1\le i\le v}\{\|X_{k,l+1}^i-Z_{k,l+1}^i\|\} \\
&\le (1+\alpha B\Lambda h_k)\|x_{k,l}-z_{k,l}\|.
\end{aligned}$$

From Theorem 2, we have

$$\bar{R}_k := \max_{1\le l\le m}R_{k,l}\le C_5 h_k h^p.$$

For $0 \le l \le m-1$,

$$\begin{aligned} e_{k,l+1} &:= \|x(t_{k,l+1}) - x_{k,l+1}\| \\ &\le \|x(t_{k,l+1}) - z_{k,l+1}\| + \|z_{k,l+1} - x_{k,l+1}\| \\ &\le (1 + \alpha B\Lambda h_k) e_{k,l} + \bar{R}_k \\ &\le (1 + \alpha B\Lambda h_k)^{l+1} e_{k,0} + \Big((1 + \alpha B\Lambda h_k)^{l+1} - 1\Big) \frac{\bar{R}_k}{\alpha B\Lambda h_k} \\ &\le \mathrm{e}^{(l+1)\alpha B\Lambda h_k} e_{k,0} + \Big(\mathrm{e}^{(l+1)\alpha B\Lambda h_k} - 1\Big) \frac{\bar{R}_k}{\alpha B\Lambda h_k} \\ &\le \mathrm{e}^{\alpha B\Lambda\sigma(\tau_{k+1}-\tau_k)} e_{k,0} + \Big(\mathrm{e}^{\alpha B\Lambda\sigma(\tau_{k+1}-\tau_k)} - 1\Big) \frac{C_5 h^p}{\alpha B\Lambda} \\ &\le \mathrm{e}^{\alpha B\Lambda\sigma T} e_{k,0} + \big(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\big) \frac{C_5 h^p}{\alpha B\Lambda}. \end{aligned} \tag{25}$$

Applying Taylor's formula, for any $k = 1, 2, \cdots, N$,

$$x(r_k) - x(t_{k,\lfloor \sigma m \rfloor}) = x'(t_{k,\lfloor \sigma m \rfloor})(r_k - t_{k,\lfloor \sigma m \rfloor}) + \frac{1}{2!} x''(\xi)(r_k - t_{k,\lfloor \sigma m \rfloor})^2$$

which implies that

$$\|x(r_k) - x(t_{k,\lfloor \sigma m \rfloor})\| = C_7 h,$$

where $\xi \in (t_{k,\lfloor \sigma m \rfloor}, r_k)$. Consequently, we can obtain that

$$\begin{aligned} e_{k+1,0} &:= \|x_{k+1,0} - x(t_{k+1,0})\| \\ &= \|I_{k+1}(x_{k+1,\lfloor \sigma m \rfloor}) - I_{k+1}(x(r_{k+1}^-))\| \\ &\le \beta_{k+1} \|x_{k+1,\lfloor \sigma m \rfloor} - x(t_{k,\lfloor \sigma m \rfloor})\| + \beta_{k+1} \|x(t_{k,\lfloor \sigma m \rfloor}) - x(r_{k+1}^-)\| \\ &\le \beta_{k+1} \|x_{k+1,\lfloor \sigma m \rfloor} - x(r_{k+1}^-)\| + \beta_{k+1} C_7 h \\ &\le \beta_{k+1} \mathrm{e}^{\alpha B\Lambda\sigma T} e_{k,0} + \beta_{k+1} \big(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\big) \frac{C_5 h^p}{\alpha B\Lambda} + \beta_{k+1} C_7 h \\ &\le \bar{B} \mathrm{e}^{\alpha B\Lambda\sigma T} e_{k,0} + \bar{B} \big(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\big) \frac{C_5 h^p}{\alpha B\Lambda} + (k+1) \bar{B} C_7 h \\ &\le \big(\bar{B} \mathrm{e}^{\alpha B\Lambda\sigma T}\big)^{k+1} e_{0,0} + \Big(\Big(\big(\bar{B} \mathrm{e}^{\alpha B\Lambda\sigma T}\big)^{k+1} - 1\Big) \frac{C_5 T^{p-1}}{\alpha B\Lambda} + (k+1) \bar{B} C_7\Big) h. \end{aligned} \tag{26}$$

For convenience, we choose $x_{0,0} = x_0$, which impiles that $e_{0,0} = 0$. So from (26), for arbitrary $k = 0, 1, 2, \ldots, N$, we have

$$e_{k+1,0} \le C_8 h \tag{27}$$

where $C_8 = \Big(\big(\bar{B} \mathrm{e}^{\alpha B\Lambda\sigma T}\big)^N - 1\Big) \frac{C_5 T^{p-1}}{\alpha B\Lambda} + N \bar{B} C_7$. Combining (25) and (27), for $0 \le l \le m-1$, we can obtain that

$$e_{k,l+1} \le C_9 h \tag{28}$$

where $C_9 = \mathrm{e}^{\alpha B\Lambda\sigma T} C_8 + \big(\mathrm{e}^{\alpha B\Lambda\sigma T} - 1\big) \frac{C_5 T^{p-1}}{\alpha B\Lambda}$. Consequently, from (27) and (28), we know that (23) and (24) hold for $C_6 = \max\{C_8, C_9\}$ and $h < \alpha^{-1} \Big(\max_{1 \le i \le v} \sum_{j=1}^{v} |a_{ij}|\Big)^{-1}$. □

4.2. Asymptotical Stability of S2IRKMs

In order to study the asymptotical stability of S2IRKMs, we consider S2IRKM for (4) as follows:

$$\begin{cases} Y_{k,l+1}^i = y_{k,l} + h_k \sum\limits_{j=1}^{v} a_{ij} f(t_{k,l+1}^j, Y_{k,l+1}^j), \ \ k \in \mathbb{N}, i = 1, 2, \cdots, v, \\ y_{k,l+1} = y_{k,l} + h_k \sum\limits_{i=1}^{v} b_i f(t_{k,l+1}^i, Y_{k,l+1}^i), \ \ l = 1, 2, \cdots, m-1, \\ y_{k+1,0} = I_{k+1}(y_{k,\lfloor \sigma m \rfloor}), \\ y_{0,0} = y_0. \end{cases} \tag{29}$$

Theorem 8. *Assume that* $\boldsymbol{R}(z)$ *is the stability function of S2IRKM* (21)*, that is*

$$\boldsymbol{R}(z) = 1 + zb^T(I - zA)^{-1}e = \frac{P_j(z)}{Q_k(z)},$$

where $e = (1, 1, \cdots, 1)^T$ *is a v-dimensional vector. Let the coefficients of the corresponding Runge–Kutta method of S2IRKM* (21) *be nonnegative; that is,* $a_{ij} \geq 0$ *and* $b_i \geq 0$, $1 \leq i \leq v$, $1 \leq j \leq v$. *Under the conditions of Theorem 1, S2IRKM* (21) *for* (1) *is asymptotically stable for* $h_k = \frac{\tau_{k+1} - \tau_k}{m}$, $k \in \mathbb{N}$, $m \in \mathbb{Z}^+$ *and* $m \geq M$, *if* $\boldsymbol{k}$ *is even, where* $M = \inf\{m : I - zA$ *is invertible and* $(I - zA)^{-1}e \geq 0, z = \alpha h, h = \max_k\{h_k\}, m \in \mathbb{Z}^+\}$.

Proof. Because $a_{ij} \geq 0$ and $b_i \geq 0$, $1 \leq i \leq v$, $1 \leq j \leq v$, we can obtain that

$$\begin{aligned}
&\|X^i_{k,l+1} - Y^i_{k,l+1}\| \\
&= \|x_{k,l} - y_{k,l} + h_k \sum_{j=1}^{v} a_{ij}(f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1}))\| \\
&\leq \|x_{k,l} - y_{k,l}\| + h_k \sum_{j=1}^{v} a_{ij}\|f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1})\| \\
&\leq \|x_{k,l} - y_{k,l}\| + \alpha h_k \sum_{j=1}^{s} a_{ij}\|X^j_{k,l+1} - Y^j_{k,l+1}\|.
\end{aligned}$$

When $m \geq M$, $(I - zA)^{-1}e \geq 0, z = \alpha h_k, k \in \mathbb{Z}^+$, so

$$[\|X^i_{k,l+1} - Y^i_{k,l+1}\|] \leq (I - \alpha h_k A)^{-1}e\|x_{k,l} - y_{k,l}\|$$

where $[\|X^i_{k,l+1} - Y^i_{k,l+1}\|] = (\|X^1_{k,l+1} - Y^1_{k,l+1}\|, \|X^2_{k,l+1} - Y^2_{k,l+1}\|, \cdots, \|X^v_{k,l+1} - Y^v_{k,l+1}\|)^T$. By Lemmas 1 and 2, we can obtain

$$\begin{aligned}
&\|x_{k,l+1} - y_{k,l+1}\| \\
&= \|x_{k,l} - y_{k,l} + h_k \sum_{j=1}^{s} b_j(f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1}))\| \\
&\leq \|x_{k,l} - y_{k,l}\| + h_k \sum_{j=1}^{s} b_j\|f(t^j_{k,l+1}, X^j_{k,l+1}) - f(t^j_{k,l+1}, Y^j_{k,l+1})\| \\
&\leq \|x_{k,l} - y_{k,l}\| + \alpha h_k \sum_{j=1}^{s} b_j\|X^j_{k,l+1} - Y^j_{k,l+1}\| \\
&= \|x_{k,l} - y_{k,l}\| + \alpha h_k b^T[\|X^i_{k,l+1} - Y^i_{k,l+1}\|] \\
&\leq (1 + \alpha h_k b^T(I - \alpha h_k A)^{-1}e)\|x_{k,l} - y_{k,l}\| \\
&= \mathbf{R}(\alpha h_k)\|x_{k,l} - y_{k,l}\| \\
&\leq \mathrm{e}^{\alpha h_k}\|x_{k,l} - y_{k,l}\|.
\end{aligned}$$

Hence, for arbitrary $k = 0, 1, 2, \cdots$ and $l = 0, 1, \cdots, m$, we have

$$\|x_{k,l} - y_{k,l}\| \leq \|x_{k,0} - y_{k,0}\|\mathrm{e}^{\alpha l h_k}.$$

which implies

$$\begin{aligned}&\|x_{k+1,0}-y_{k+1,0}\|=\|I_{k+1}(x_{k,\lfloor\sigma m\rfloor})-I_{k+1}(y_{k,\lfloor\sigma m\rfloor})\|\\&\leq\beta_{k+1}\|x_{k,\lfloor\sigma m\rfloor}-y_{k,\lfloor\sigma m\rfloor}\|\\&\leq\beta_{k+1}\|x_{k,0}-y_{k,0}\|\mathrm{e}^{\alpha\lfloor\sigma m\rfloor h_k}\\&\leq\beta_{k+1}\|x_{k,0}-y_{k,0}\|\mathrm{e}^{\alpha\sigma(\tau_{k+1}-\tau_k)}.\end{aligned}$$

Therefore, by the method of introduction and condition (5), we can obtain that

$$\begin{aligned}&\|x_{k,l}-y_{k,l}\|\\&\leq\|x_0-y_0\|\left(\beta_1\mathrm{e}^{\alpha\sigma(\tau_1-\tau_0)}\right)\left(\beta_2\mathrm{e}^{\alpha\sigma(\tau_2-\tau_1)}\right)\left(\beta_k\mathrm{e}^{\alpha\sigma(\tau_k-\tau_{k-1})}\right)\mathrm{e}^{\alpha lh_k}\\&\leq\|x_0-y_0\|C^k\mathrm{e}^{\alpha\gamma}\end{aligned}$$

which implies that the Runge–Kutta method for (1) is asymptotically stable for $h_k=\frac{\tau_{k+1}-\tau_k}{m}$, $k\in\mathbb{N}$, $m\in\mathbb{Z}^+$ and $m\geq M$. □

4.3. Asymptotical Stability of S2Iθ M

S2IθM for (1) can be constructed as follows:

$$\begin{cases}x_{k,l+1}=x_{k,l}+h_k(1-\theta)f(t_{k,l},x_{k,l})+h_k\theta f(t_{k,l+1},x_{k,l+1})\\x_{k+1,0}=I_{k+1}(x_{k,\lfloor\sigma m\rfloor}),\\x_{0,0}=x_0,\end{cases}\tag{30}$$

where $h_k=\frac{\tau_{k+1}-\tau_k}{m}$, $m\geq1$, $m\in\mathbb{Z}^+$, $k\in\mathbb{N}$.

Theorem 9. *Under the conditions of Theorem 1, if* $0\leq\theta\leq\varphi(1)$*, there is a positive M such that S2Iθ M* (30) *for* (1) *is asymptotically stable for* $h_k=\frac{\tau_{k+1}-\tau_k}{m}$, $k\in\mathbb{N}$, $m\in\mathbb{Z}^+$ *and* $m\geq M$ *.*

Proof. Obviously, we can obtain

$$\begin{aligned}&\|x_{k,l+1}-y_{k,l+1}\|\\&\leq\|x_{k,l}-y_{k,l}+(1-\theta)h_k(f(t_{k,l},x_{k,l})-f(t_{k,l},y_{k,l}))\|\\&\quad+\theta h_k\|f(t_{k,l+1},x_{k,l+1})-f(t_{k,l+1},y_{k,l+1})\|\\&\leq(1+(1-\theta)\alpha h_k)\|x_{k,l}-y_{k,l}\|+\theta\alpha h_k\|x_{k,l+1}-y_{k,l+1}\|\end{aligned}$$

which implies

$$\|x_{k,l+1}-y_{k,l+1}\|\leq\frac{1+(1-\theta)\alpha h_k}{1-\theta\alpha h_k}\cdot\|x_{k,l}-y_{k,l}\|$$

Therefore, by Lemma 2 and the method of introduction, we can obtain that

$$\|x_{k,l+1}-y_{k,l+1}\|\leq\mathrm{e}^{\alpha h_k}\|x_{k,l}-y_{k,l}\|.$$

Similarly to the proof of Theorem 8, we can prove that S2IθM (30) for (1) is asymptotically stable for $h_k=\frac{\tau_{k+1}-\tau_k}{m}$, $k\in\mathbb{N}$, $m\in\mathbb{Z}^+$ and $m>\alpha(\tau_{k+1}-\tau_k)$, $k\in\mathbb{Z}^+$ if $0\leq\theta\leq\varphi(1)$. □

5. Impulsive Continuous Runge–Kutta Methods

The purpose of this section is to construct impulsive continuous Runge–Kutta methods (ICRKMs) for DEDI (1) and study the convergence and stability of the constructed numerical methods, respectively.

To ensure the high-order convergence of the numerical methods, the mesh

$$\bar{\mathbf{S}} = \{t_0, t_1, \cdots, t_n, \cdots\}$$

includes all discontinuous points (the points at the moments of impulsive effect), i.e., $\mathbf{S} \subset \bar{\mathbf{S}}$, where $\mathbf{S} = \{\tau_k : k \in \mathbb{Z}^+\}$.

Remark 4.

(1) *The same as S1IRKMs in Section 3, all points in the set* $\{\tau_k, r_k : k \in \mathbb{Z}^+\}$ *are chosen as the numerical mesh. We can divide the intervals* $[\tau_{k-1}, r_k]$ *and* $[r_k, \tau_k)$ $(k \in \mathbb{Z}^+)$ *equally by m; m is a positive integer. In this case, ICRKM* (31) *in this section and S1IRKM* (8) *have the same values at the discrete points, if they have the same corresponding Runge–Kutta method. Because they have similar properties, we ignore this case for the sake of brevity.*

(2) *For convenience, in the next part of this section, we divide the intervals* $[\tau_{k-1}, \tau_k]$ $(k \in \mathbb{Z}^+)$ *equally by m; m is a positive integer. Unlike the S2IRKMs, when we compute the numerical solutions at the moments of impulsive effect, the numerical solutions of ICRKMs at points* $\{r_k : k \in \mathbb{Z}^+\}$ *can be obtained directly without substituting nearby values.*

When interpolants (constructed using no extra stages) of the corresponding continuous Runge–Kutta method are interpolants of the first class, ICRKM for DEDI (1) is constructed as follows.

$$\begin{cases} X^i_{n+1} = x_n + h_{n+1} \sum\limits_{j=1}^{v} a_{ij} f(t^j_{n+1}, X^j_{n+1}), & i = 1, 2, \cdots, v, \\ \eta(t_n + \vartheta h_{n+1}) = x_n + h_{n+1} \sum\limits_{i=1}^{v} b_i(\vartheta) f(t^i_{n+1}, X^i_{n+1}), & 0 \le \vartheta < 1, \\ \eta(t^-_{n+1}) = x_n + h_{n+1} \sum\limits_{i=1}^{v} b_i f(t^i_{n+1}, X^i_{n+1}), & \\ x_{n+1} = \eta(t_{n+1}) = \begin{cases} I_k(\eta(r^-_k)), & \text{if } \exists k \text{ such that } t_{n+1} = \tau_k, \\ \eta(t^-_{n+1}), & \text{otherwise,} \end{cases} & \\ x_0 = x(t_0) \end{cases} \tag{31}$$

where $y_n = \eta(t_n)$, $t_{n+1} = t_n + h_{n+1}$, $t^i_{n+1} = t_n + c_i h_{n+1}$, $b_i(0) = 0$, $b_i = b_i(1)$, $c_i = \sum\limits_{j=1}^{v} a_{ij}$, $i = 1, 2, \cdots, v$, for $j = 1, 2, \cdots, v$,

$$X^j_{n+1} = \begin{cases} \eta(t^-_{n+1}), & \text{if } \exists k \in \mathbb{Z}^+, \text{ such that } t^j_{n+1} = \tau_k \text{ and } c_j = 1, \\ \eta(t^j_{n+1}), & \text{otherwise.} \end{cases}$$

According to Remark 4 (2), the step sizes are chosen as follows, for $km < n \le (k+1)m$, $k \in \mathbb{N}$,

$$h_n = \tilde{h}_k = \frac{\tau_{k+1} - \tau_k}{m},$$

where m is a positive integer.

When interpolants (constructed by means of additional stages) of the corresponding continuous Runge–Kutta method are interpolants of the second class, ICRKM for DEDI (1) is constructed as follows.

$$\begin{cases} X^i_{n+1} = x_n + h_{n+1}\sum\limits_{j=1}^{v} a_{ij} f(t^j_{n+1}, X^j_{n+1}), & i = 1, 2, \cdots, v, \\ X^i_{n+1} = x_n + h_{n+1}\sum\limits_{j=1}^{s} a_{ij} f(t^j_{n+1}, X^j_{n+1}), & i = v+1, \cdots, s, \\ \eta(t_n + \vartheta h_{n+1}) = x_n + h_{n+1}\sum\limits_{i=1}^{s} b_i(\vartheta) f(t^i_{n+1}, X^i_{n+1}), & 0 \le \vartheta < 1, \\ \eta(t^-_{n+1}) = y_n + h_{n+1}\sum\limits_{i=1}^{s} b_i f(t^i_{n+1}, X^i_{n+1}), & \\ x_{n+1} = \eta(t_{n+1}) = \begin{cases} I_k(\eta(r^-_k)), & \text{if } \exists k \text{ such that } t_{n+1} = \tau_k, \\ \eta(t^-_{n+1}), & \text{otherwise,} \end{cases} & \end{cases} \tag{32}$$

where

$$\begin{aligned} b_i(0) &= 0, && i = 1, 2, \cdots, s; \\ b_i(1) &= b_i, && i = 1, 2, \cdots, v; \\ b_i(1) &= 0, && i = v+1, v+2, \cdots, s. \end{aligned}$$

5.1. Convergence of ICRKMs

To analyze the local truncation errors of ICRKM for DEDI (1), consider the following local problem of (31) on $[t_n, t_{n+1}]$, $n = 0, 1, 2, \cdots, M-1$,

$$\begin{cases} Z^i_{n+1} = z_n + h_{n+1}\sum\limits_{j=1}^{v} a_{ij} f(t^j_{n+1}, Z^j_{n+1}), & i = 1, 2, \cdots, v, \\ \zeta(t_n + \vartheta h_{n+1}) = z_n + h_{n+1}\sum\limits_{i=1}^{v} b_i(\vartheta) f(t^i_{n+1}, Z^i_{n+1}), & 0 \le \vartheta < 1, \\ \zeta(t^-_{n+1}) = z_n + h_{n+1}\sum\limits_{i=1}^{v} b_i f(t^i_{n+1}, Z^i_{n+1}), & \\ z_{n+1} = \zeta(t_{n+1}) = \begin{cases} I_k(\zeta(r^-_k)), & \text{if } \exists k \text{ such that } t_{n+1} = \tau_k, \\ \zeta(t^-_{n+1}), & \text{otherwise,} \end{cases} & \end{cases} \tag{33}$$

and the following local problem of (32) on $[t_n, t_{n+1}]$,

$$\begin{cases} Z^i_{n+1} = z_n + h_{n+1}\sum\limits_{j=1}^{v} a_{ij} f(t^j_{n+1}, Z^j_{n+1}), & i = 1, 2, \cdots, v, \\ Z^i_{n+1} = z_n + h_{n+1}\sum\limits_{j=1}^{s} a_{ij} f(t^j_{n+1}, Z^j_{n+1}), & i = v+1, \cdots, s, \\ \zeta(t_n + \vartheta h_{n+1}) = z_n + h_{n+1}\sum\limits_{i=1}^{s} b_i(\vartheta) f(t^i_{n+1}, Z^i_{n+1}), & 0 \le \vartheta < 1, \\ \zeta(t^-_{n+1}) = z_n + h_{n+1}\sum\limits_{i=1}^{s} b_i f(t^i_{n+1}, Z^i_{n+1}), & \\ z_{n+1} = \zeta(t_{n+1}) = \begin{cases} I_k(\zeta(r^-_k)), & \text{if } \exists k \text{ such that } t_{n+1} = \tau_k, \\ \zeta(t^-_{n+1}), & \text{otherwise,} \end{cases} & \end{cases} \tag{34}$$

where $z_n = x(t_n)$ and for $j = 1, 2, \cdots, v$,

$$Z^j_{n+1} = \begin{cases} \zeta(t^-_{n+1}), & \text{if } \exists k \in \mathbb{Z}^+, \text{ such that } t^j_{n+1} = \tau_k \text{ and } c_j = 1, \\ \zeta(t^j_{n+1}), & \text{otherwise.} \end{cases}$$

Because it can be seen as a problem of ordinary differential equations when we consider the local problem, from [39] (page 114, Definition 5.1.3), we can directly obtain the following result.

Theorem 10. *Consider DEDI* (1) *where $f(t,x)$ is C^p-continuous in $[t_0,T]\times\mathbb{R}^d$. If the corresponding continuous Runge–Kutta method is consistent of order p, then local errors between the numerical solutions obtained from* (33) *(or* (34)*) and the exact solutions obtained from DEDI* (1) *satisfy that there exists a constant C such that, for arbitrary $n=0,1,2,\cdots,N-1$, if $t_{n+1}\neq\tau_k$, for $\forall k$,*

$$R_n := \|z_{n+1}-x(t_{n+1})\| \le Ch_{n+1}^{p+1}; \tag{35}$$

otherwise, there exists an integer k such that $t_{n+1}=\tau_k$,

$$\|\zeta(t_{n+1}^-)-x(\tau_k^-)\| \le Ch_{n+1}^{p+1}, \tag{36}$$

If the corresponding continuous Runge–Kutta method is consistent of uniform order q, then local errors between the numerical solutions obtained from (33) *(or* (34)*) and the exact solutions obtained from DEDI* (1) *satisfy that there exists a constant C such that, for arbitrary $n=0,1,2,\cdots,N-1$, when $t_{n+1}\neq\tau_k$, for $\forall k$,*

$$\|\zeta(t)-x(t)\| \le Ch_{n+1}^{q+1},\ \forall t\in[t_n,t_{n+1}]; \tag{37}$$

otherwise, there exists an integer k such that $t_{n+1}=\tau_k$,

$$\|\zeta(t)-x(t)\| \le Ch_{n+1}^{q+1},\ \forall t\in[t_n,\tau_k). \tag{38}$$

Theorem 11. *Assume that $f(t,x)$ of DEDI* (1) *is C^p-continuous in $[t_0,T]\times\mathbb{R}^d$, the functions I_k are bounded, and Lipschitz conditions* (2) *and* (3) *hold. If the corresponding continuous Runge–Kutta method is consistent of order p and is consistent of uniform order q, then the global errors $e(t)=\|x(t)-\eta(t)\|$ between the numerical solutions $\eta(t)$ obtained from* (31) *(or* (32)*) and the exact solutions $x(t)$ obtained from DEDI* (1) *satisfy that there exists a constant C_1 such that, for arbitrary $n=0,1,2,\cdots,N-1$, when $t_{n+1}\neq\tau_k, k\in\mathbb{Z}^+$*

$$e(t)=\|x(t)-\eta(t)\| \le C_1h^{q'}, t\in[t_n,t_{n+1}] \tag{39}$$

when $\exists k\in\mathbb{Z}^+, t_{n+1}=\tau_k$,

$$e(t)=\|x(t)-\eta(t)\| \le C_1h^{q'}, t\in[t_n,\tau_k) \tag{40}$$

and

$$e(\tau_k)=\|x(\tau_k)-\eta(\tau_k)\| \le C_1h^{q'}, \tag{41}$$

where $q'=\min\{p,q+1\}$.

Proof. From (8) and (9), we can obtain that

$$\begin{aligned}
&\|X_{n+1}^i - Z_{n+1}^i\| \\
&= \|x_n - z_n + h_{n+1}\sum_{j=1}^{v} a_{ij}(f(t_{n+1}^j, X_{k,l+1}^j) - f(t_{n+1}^j, Z_{n+1}^j))\| \\
&\le \|x_n - z_n\| + h_{n+1}\sum_{j=1}^{v} |a_{ij}|\|f(t_{n+1}^j, X_{n+1}^j) - f(t_{n+1}^j, Z_{n+1}^j)\| \\
&\le \|x_n - z_n\| + \alpha h_{n+1}\sum_{j=1}^{v} |a_{ij}|\|X_{n+1}^j - Z_{n+1}^j\| \\
&\le \|x_n - z_n\| + \alpha h\left(\max_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)\max_{1\le j\le v}\{\|X_{n+1}^j - Z_{n+1}^j\|\}
\end{aligned}$$

which implies that

$$\max_{1\le i\le v}\{\|X_{n+1}^i - Z_{n+1}^i\|\} \le \Lambda\|x_n - z_n\|.$$

where $\Lambda = \left(1 - \alpha h\left(\max\limits_{1\le i\le v}\sum_{j=1}^{v}|a_{ij}|\right)\right)^{-1}$. Hence

$$\begin{aligned}
&\|x_{n+1} - z_{n+1}\| \\
&= \|x_n - z_n + h_{n+1}\sum_{j=1}^{v} b_j(f(t_{n+1}^j, X_{n+1}^j) - f(t_{n+1}^j, Z_{n+1}^j))\| \\
&\le \|x_n - z_n\| + h_{n+1}\sum_{j=1}^{v}|b_j|\|f(t_{n+1}^j, X_{n+1}^j) - f(t_{n+1}^j, Z_{n+1}^j)\| \\
&\le \|x_n - z_n\| + \alpha h_{n+1}\left(\sum_{j=1}^{v}|b_j|\right)\max_{1\le i\le v}\{\|X_{n+1}^i - Z_{n+1}^i\|\} \\
&\le (1 + \alpha B\Lambda h_{n+1})\|x_n - z_n\|.
\end{aligned}$$

From Theorem 2, we have

$$\bar{R}_k := \max_{km\le n<(k+1)m} R_n = \max_{km\le n<(k+1)m}\|z_n - x(t_n)\| \le C\tilde{h}_k h^p,$$

where $h = \max_{1\le n\le N} h_n$.

For $km \le n < (k+1)m - 1$, $(t_{n+1} \ne \tau_{k+1})$, $k \in \mathbb{N}$,

$$\begin{aligned}
e_{n+1} &:= \|x(t_{n+1}) - \eta(t_{n+1})\| \\
&\le \|x(t_{n+1}) - z_{n+1}\| + \|z_{n+1} - x_{n+1}\| \\
&\le \left(1 + \alpha B\Lambda\tilde{h}_k\right)e_n + \bar{R}_{n+1} \\
&\le \left(1 + \alpha B\Lambda\tilde{h}_k\right)^{n+1-km} e_{km} + \left(\left(1 + \alpha B\Lambda h_{n+1}\right)^{n+1-km} - 1\right)\frac{\bar{R}_k}{\alpha\beta\Lambda\tilde{h}_k} \\
&\le \mathrm{e}^{\alpha B\Lambda(\tau_{k+1}-\tau_k)}e_{km} + \left(\mathrm{e}^{\alpha B\Lambda(\tau_{k+1}-\tau_k)} - 1\right)\frac{Ch^p}{\alpha B\Lambda} \\
&\le \mathrm{e}^{\alpha B\Lambda T}e_{km} + \left(\mathrm{e}^{\alpha B\Lambda T} - 1\right)\frac{Ch^p}{\alpha B\Lambda}
\end{aligned} \tag{42}$$

and on the interval $[t_n, t_{n+1}]$,

$$\begin{aligned}
&\|x(t_n + \vartheta h_{n+1}) - \eta(t_n + \vartheta h_{n+1})\| \\
&\le \|x(t_n + \vartheta h_{n+1}) - \zeta(t_n + \vartheta h_{n+1})\| + \|\zeta(t_n + \vartheta h_{n+1}) - \eta(t_n + \vartheta h_{n+1})\| + Ch^{q+1} \\
&\le \|x_n - z_n\| + h_{n+1}\sum_{i=1}^{v} b_i(\vartheta)\|f(t_{n+1}^i, X_{n+1}^i) - f(t_{n+1}^i, Z_{n+1}^i)\| + Ch^{q+1} \\
&\le \|x_n - z_n\| + h_{n+1}\tilde{B}\max_{1\le i\le v}\|f(t_{n+1}^i, X_{n+1}^i) - f(t_{n+1}^i, Z_{n+1}^i)\| + Ch^{q+1} \\
&\le (1 + h_{n+1}\alpha\Lambda\tilde{B})\|x_n - z_n\| + Ch^{q+1} \\
&\le \mathrm{e}^{\alpha\Lambda\tilde{B}\tilde{h}_k}e_n + Ch^{q+1} \\
&\le \mathrm{e}^{\alpha\Lambda\tilde{B}\tilde{h}_k}\mathrm{e}^{\alpha B\Lambda T}e_{km} + \mathrm{e}^{\alpha\Lambda\tilde{B}\tilde{h}_k}\left(\mathrm{e}^{\alpha B\Lambda T} - 1\right)\frac{Ch^p}{\alpha B\Lambda} + Ch^{q+1} \\
&\le \mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}e_{km} + \mathrm{e}^{\alpha\Lambda\tilde{B}T}\left(\mathrm{e}^{\alpha B\Lambda T} - 1\right)\frac{Ch^p}{\alpha B\Lambda} + Ch^{q+1}.
\end{aligned} \tag{43}$$

where $\tilde{B} = \max_{0\le\vartheta\le 1}\sum_{i=1}^{v} b_i(\vartheta)$. For $n = (k+1)m-1$, $(t_{n+1} = \tau_{k+1})$, $k \in \mathbb{N}$, from (43), we can obtain

$$\begin{aligned} e_{n+1} &:= \|x(t_{n+1}) - \eta(t_{n+1})\| = \|x(\tau_{k+1}) - \eta(\tau_{k+1})\| \\ &= \|I_{k+1}(x(r_{k+1}^-)) - I_{k+1}(\eta(x(r_{k+1}^-))\| \\ &\le \beta_{k+1}\|x(r_{k+1}^-) - \eta(x(r_{k+1}^-)\| \\ &\le \bar{B}\mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}e_{km} + \bar{B}\mathrm{e}^{\alpha\Lambda\tilde{B}T}\Big(\mathrm{e}^{\alpha B\Lambda T}-1\Big)\frac{Ch^p}{\alpha B\Lambda} + \bar{B}Ch^{q+1} \\ &\le \bar{B}\mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}e_{km} + Dh^{q'}. \end{aligned} \tag{44}$$

where $\bar{B} = \max\{\beta_1, \beta_2, \cdots, \beta_N\}$ and $D = \bar{B}\mathrm{e}^{\alpha\Lambda\tilde{B}T}\big(\mathrm{e}^{\alpha B\Lambda T}-1\big)\frac{C}{\alpha B\Lambda} + \bar{B}C$. By (44) and mathematical induction, we know that

$$e(\tau_k) = e_{km} \le \bar{B}^N\mathrm{e}^{N\alpha(B+\tilde{B})\Lambda T}e_0 + \left(\frac{\bar{B}^N\mathrm{e}^{N\alpha(B+\tilde{B})\Lambda T}-1}{\bar{B}\mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}-1}\right)Dh^{q'}. \tag{45}$$

If the inital data $\eta(t_0) = x(t_0)$, $e_0 = 0$. Combining (43) and (45), the theorem holds for $C_1 = \max\{\Big(\frac{\bar{B}^N\mathrm{e}^{N\alpha(B+\tilde{B})\Lambda T}-1}{\bar{B}\mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}-1}\Big)D, \mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}\Big(\frac{\bar{B}^N\mathrm{e}^{N\alpha(B+\tilde{B})\Lambda T}-1}{\bar{B}\mathrm{e}^{\alpha(B+\tilde{B})\Lambda T}-1}\Big)D + \mathrm{e}^{\alpha\Lambda\tilde{B}T}\big(\mathrm{e}^{\alpha B\Lambda T}-1\big)\frac{C}{\alpha B\Lambda} + C\}$. □

5.2. Asymptotical Stability of ICRKMs

In order to study the asymptotical stability of ICRKMs, we first consider that ICRKM for DEDI (4) is constructed as follows.

$$\begin{cases} \bar{Y}_{n+1}^i = \bar{y}_n + h_{n+1}\sum\limits_{j=1}^{v} a_{ij}f(t_{n+1}^j, \bar{Y}_{n+1}^j), & i = 1, 2, \cdots, v, \\ \bar{\eta}(t_n + \vartheta h_{n+1}) = \bar{y}_n + h_{n+1}\sum\limits_{i=1}^{v} b_i(\vartheta)f(t_{n+1}^i, \bar{Y}_{n+1}^i), & 0 \le \vartheta < 1, \\ \bar{\eta}(t_{n+1}^-) = \bar{y}_n + h_{n+1}\sum\limits_{i=1}^{v} b_i f(t_{n+1}^i, \bar{Y}_{n+1}^i), & \\ \bar{y}_{n+1} = \bar{\eta}(t_{n+1}) = \begin{cases} I_k(\bar{\eta}(r_k^-)), & \text{if } \exists k \text{ such that } t_{n+1} = \tau_k, \\ \bar{\eta}(t_{n+1}^-), & \text{otherwise,} \end{cases} & \\ \bar{y}_0 = y(t_0) & \end{cases} \tag{46}$$

where

$$\bar{Y}_{n+1}^j = \begin{cases} \bar{\eta}(t_{n+1}^-), & \text{if } \exists k \in \mathbb{Z}^+, \text{ such that } t_{n+1}^j = \tau_k \text{ and } c_j = 1, \\ \bar{\eta}(t_{n+1}^j), & \text{otherwise.} \end{cases}$$

Theorem 12. *Assume that $f(t,x)$ is C^p-continuous in $[t_0, T]\times\mathbb{R}^d$ and satisfies the Lipschitz conditions* (2) *and* (3)*; $\eta(t)$ and $\bar{\eta}(t)$ are the numerical solutions obtained from ICRKMs* (31) *and* (46) *for DEDI* (1) *and* (4)*, respectively. If there are positive constants $\bar{h}_0$ and C such that $\beta_k\mathrm{e}^{\alpha\Lambda\tilde{B}\left[\sigma(\tau_k-\tau_{k-1})+\bar{h}_0\right]} < C < 1$ for all $k \in \mathbb{Z}^+$, then ICRKMs* (31) *and* (46) *for DEDI* (1) *and* (4) *are asymptotically stable.*

Proof. Because of the Lipschitz condition of f, we can obtain that

$$\begin{aligned}
&\|f(t_{n+1}^i, X_{n+1}^i) - f(t_{n+1}^i, \bar{Y}_{n+1}^i)\| \\
&\le \alpha \|X_{n+1}^i - \bar{Y}_{n+1}^i\| \\
&\le \alpha \|x_n - \bar{y}_n\| + h_{n+1}\alpha \sum_{j=1}^{v} |a_{ij}| \|f(t_{n+1}^j, X_{n+1}^j) - f(t_{n+1}^j, \bar{Y}_{n+1}^j)\| \\
&\le h_{n+1}\alpha (\max_{1\le i\le s} \sum_{j=1}^{s} |a_{ij}|) \max_{1\le i\le s} \|f(t_{n+1}^j, X_{n+1}^j) - f(t_{n+1}^j, \bar{Y}_{n+1}^j)\| \\
&\quad + \alpha \|x_n - \bar{y}_n\|
\end{aligned}$$

Therefore, if $h_{n+1} \le \bar{h}_1$ for some $\bar{h}_1 < (\alpha \max\limits_{1\le i\le v} \sum\limits_{j=1}^{v} |a_{ij}|)^{-1}$, then

$$\max_{1\le i\le v} \|f(t_{n+1}^i, X_{n+1}^i) - f(t_{n+1}^i, \bar{Y}_{n+1}^i)\| \le \alpha\Lambda \|x_n - \bar{y}_n\|$$

where

$$\Lambda = (1 - h\alpha \max_{1\le i\le v} \sum_{j=1}^{v} |a_{ij}|)^{-1}.$$

Hence

$$\begin{aligned}
&\|\eta(t_n + \vartheta h_{n+1}) - \bar{\eta}(t_n + \vartheta h_{n+1})\| \\
&\le \|x_n - \bar{y}_n\| + h_{n+1} \sum_{i=1}^{v} b_i(\vartheta) \|f(t_{n+1}^i, X_{n+1}^i) - f(t_{n+1}^i, \bar{Y}_{n+1}^i)\| \\
&\le \|x_n - \bar{y}_n\| + h_{n+1}\tilde{B} \max_{1\le i\le v} \|f(t_{n+1}^i, X_{n+1}^i) - f(t_{n+1}^i, \bar{Y}_{n+1}^i)\| \\
&\le (1 + h_{n+1}\alpha\Lambda\tilde{B}) \|x_n - \bar{y}_n\| \\
&\le \mathrm{e}^{\alpha\Lambda\tilde{B}h_{n+1}} \|x_n - \bar{y}_n\|
\end{aligned} \tag{47}$$

which implies

$$\|x_{n+1} - \bar{y}_{n+1}\| \le \mathrm{e}^{\alpha\Lambda\tilde{B}h_{n+1}} \|x_n - \bar{y}_n\|.$$

So, for $km \le n < (k+1)m$,

$$\|x_n - \bar{y}_n\| \le \mathrm{e}^{\alpha\Lambda\tilde{B}(t_n - t_{km})} \|x_{km} - \bar{y}_{km}\|. \tag{48}$$

From (47) and (48), we obtain

$$\begin{aligned}
&\|x_{(k+1)m} - \bar{y}_{(k+1)m}\| = \|\eta(\tau_{k+1}) - \bar{\eta}(\tau_{k+1})\| \\
&\le \beta_{k+1} \|\eta(r_{k+1}^-) - \bar{\eta}(r_{k+1}^-)\| \\
&\le \beta_{k+1} \mathrm{e}^{\alpha\Lambda\tilde{B}\bar{h}_0} \|x_{km+\lfloor\sigma m\rfloor} - \bar{y}_{km+\lfloor\sigma m\rfloor}\| \\
&\le \beta_{k+1} \mathrm{e}^{\alpha\Lambda\tilde{B}\bar{h}_0} \mathrm{e}^{\alpha\Lambda\tilde{B}(t_{km+\lfloor\sigma m\rfloor} - t_{km})} \|x_{km} - \bar{y}_{km}\| \\
&\le \beta_{k+1} \mathrm{e}^{\alpha\Lambda\tilde{B}[\sigma(\tau_{k+1} - \tau_k) + \bar{h}_0]} \|x_{km} - \bar{y}_{km}\|.
\end{aligned} \tag{49}$$

Combining (47), (48) and (49) and applying mathematical induction, we can obtain that ICRKMs (31) and (46) for DEDI (1) and (4) are asymptotically stable when the step sizes satisfy $h_n \le \min\{\bar{h}_0, \bar{h}_1\}$, $\forall n \in \mathbb{Z}^+$. □

6. Numerical Experiments

In this section, two simple numerical examples in real space are given.

Example 1. *Consider the following scalar DEDI:*

$$\begin{cases} x'(t) = \sin(x(t)), & t > 0,\ t \neq k, k \in \mathbb{Z}^+, \\ x(k) = (\frac{1}{2})x((k - \frac{\pi}{4})^-), & k \in \mathbb{Z}^+, \\ x(0) = x_0. \end{cases} \tag{50}$$

Obviously, $\sigma = 1 - \frac{\pi}{4}$, $\beta_k = \frac{1}{2}$, $\tau_k = k$, $k \in \mathbb{Z}^+$. *For arbitrary* $x, y \in \mathbb{R}$, *we can obtain that*

$$|\sin(x) - \sin(y)| = |2\cos(\frac{x+y}{2})\sin(\frac{x-y}{2})| \leq 2|\frac{x-y}{2}| = |x-y|,$$

which implies the Lipschitz coefficient $\alpha = 1$. *Hence*

$$\beta_k e^{\alpha\sigma(\tau_k - \tau_{k-1})} = \frac{e^{1-\frac{\pi}{4}}}{2} < 1.$$

Therefore, by Theorem 1, the exact solution of (50) *is asymptotically stable.*

This statement is correct. By Theorems 4 and 8, if the stability function $\boldsymbol{R}(z) = \frac{P_j(z)}{Q_k(z)}$ *with nonnegative coefficients of S1IRKM* (8) *(or S2IRKM* (21)*), then S1IRKM* (8) *(or S2IRKM* (21)*) for* (50) *is asymptotically stable if* $\boldsymbol{k}$ *is even and the step sizes are small enough. For example, the scheme 1 impulsive Heun's method (S1IHM) (see Figure 1) and scheme 1 impulsive four-stage four-order classical Runge–Kutta method (S1IRKM) (see Figure 2) for* (50) *is asymptotically stable.*

When $x_0 = 1$, *solving* (50), *we can obtain*

$$\begin{aligned}
&x(t) = \arccos\left(\frac{C_0 e^{-2t} - 1}{C_0 e^{-2t} + 1}\right),\ t \in (0,1),\\
&x(1) = (\frac{1}{2})x((1 - \frac{\pi}{4})^-) = \left(\frac{1}{2}\right)\arccos\left(\frac{C_0 e^{-2\left(1-\frac{\pi}{4}\right)} - 1}{C_0 e^{-2\left(1-\frac{\pi}{4}\right)} + 1}\right),\\
&x(t) = \arccos\left(\frac{C_1 e^{-2t} - 1}{C_1 e^{-2t} + 1}\right),\ t \in (1,2),\\
&x(2) = (\frac{1}{2})x((2 - \frac{\pi}{4})^-) = \left(\frac{1}{2}\right)\arccos\left(\frac{C_1 e^{-2\left(2-\frac{\pi}{4}\right)} - 1}{C_1 e^{-2\left(2-\frac{\pi}{4}\right)} + 1}\right),\\
&x(t) = \arccos\left(\frac{C_2 e^{-2t} - 1}{C_2 e^{-2t} + 1}\right),\ t \in (2,3),\\
&x(3) = (\frac{1}{2})x((3 - \frac{\pi}{4})^-) = \left(\frac{1}{2}\right)\arccos\left(\frac{C_2 e^{-2\left(3-\frac{\pi}{4}\right)} - 1}{C_2 e^{-2\left(3-\frac{\pi}{4}\right)} + 1}\right),\\
&\vdots
\end{aligned}$$

where

$$C_0 = \frac{1+\cos 1}{1-\cos 1},$$

$$C_1 = \mathrm{e}^2\left(\frac{1+\cos\left[\left(\frac{1}{2}\right)\arccos\left(\frac{C_0\mathrm{e}^{-2\left(1-\frac{\pi}{4}\right)}-1}{C_0\mathrm{e}^{-2\left(1-\frac{\pi}{4}\right)}+1}\right)\right]}{1-\cos\left[\left(\frac{1}{2}\right)\arccos\left(\frac{C_0\mathrm{e}^{-2\left(1-\frac{\pi}{4}\right)}-1}{C_0\mathrm{e}^{-2\left(1-\frac{\pi}{4}\right)}+1}\right)\right]}\right),$$

$$C_2 = \mathrm{e}^4\left(\frac{1+\cos\left[\left(\frac{1}{2}\right)\arccos\left(\frac{C_1\mathrm{e}^{-2\left(2-\frac{\pi}{4}\right)}-1}{C_1\mathrm{e}^{-2\left(2-\frac{\pi}{4}\right)}+1}\right)\right]}{1-\cos\left[\left(\frac{1}{2}\right)\arccos\left(\frac{C_1\mathrm{e}^{-2\left(2-\frac{\pi}{4}\right)}-1}{C_1\mathrm{e}^{-2\left(2-\frac{\pi}{4}\right)}+1}\right)\right]}\right).$$

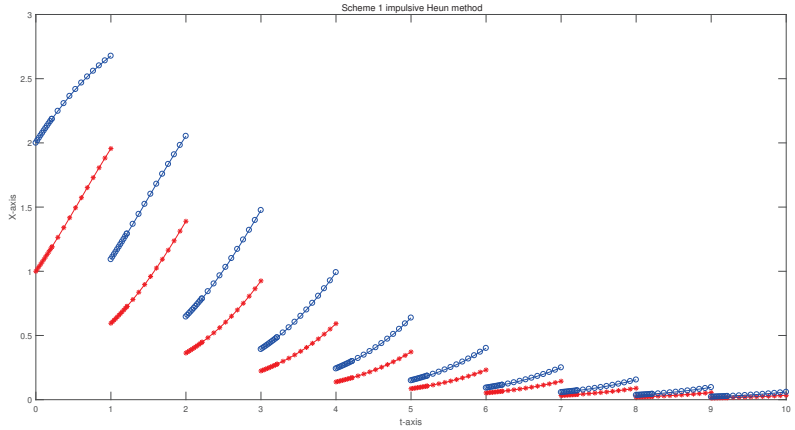


Figure 1. The S1IHM for DEDI (50) with initial values $x_0 = 1$ and $y_0 = 2$, respectively.

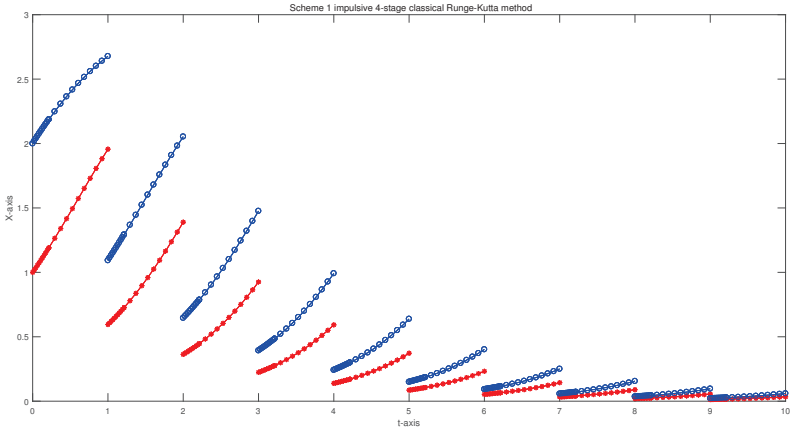


Figure 2. The S1IRKM with corresponding classical 4-stage 4-order Runge–Kutta method for DEDI (50) with intitial values $x_0 = 1$ and $y_0 = 2$, respectively.

From the theory in the previous part of this paper, we can see that the numerical solutions obtained by scheme 1 impulsive discrete Runge–Kutta methods (See Section 3) converge best. From Tables 1–4, we can also see that the scheme 1 impulsive discrete Runge–Kutta methods have the best convergence when we use computational simulation, even when the step sizes are not precise enough to have truncation errors. From Figure 3, we can see that the curves of the exact solution of DEDI (50) *seem to overlap with those obtained by the S1IRKM and the difference between the curves of the exact solution and those obtained by the S1IHM is not very large. Even if we take the maximum step sizes* $\bar{h}_{k,1} = 1 - \frac{\pi}{4}$ *and* $\bar{h}_{k,2} = \frac{\pi}{4}$, $k \in \mathbb{N}$ $(m = 1)$, *the scheme 1 impulsive discrete Runge–Kutta methods are simulated very well.*

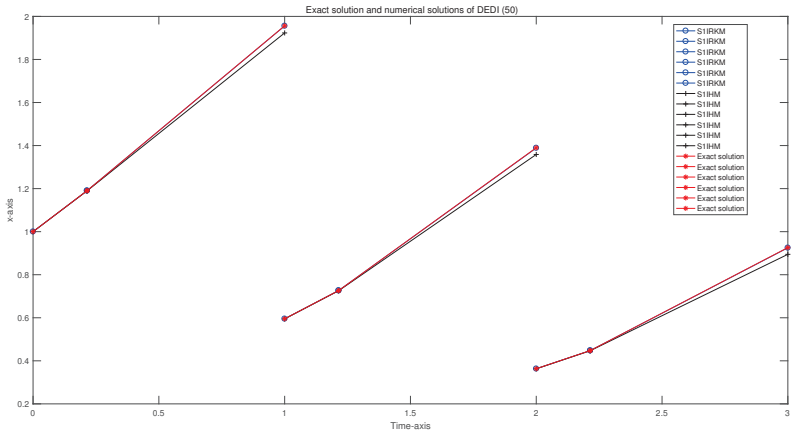


Figure 3. The exact solution of DEDI (50) and the numerical solutions obtained from S1IRKM and S1IHM for DEDI (50), respectively, when the initial values $x_0 = 1$.

Table 1. The errors between the numerical solutions obtained from different impulsive schemes with corresponding Heun's method for (50) and the exact solution of (50) at $t = 3$.

	S1IHM		**S2IHM**		**ICHM**	
m	**AE**	**RE**	**AE**	**RE**	**AE**	**RE**
10	7.553082×10^{-6}	3.375092×10^{-5}	0.008562	0.038261	9.937190×10^{-5}	4.440430×10^{-4}
20	1.902566×10^{-6}	8.501609×10^{-6}	0.008458	0.037796	6.538494×10^{-5}	2.921723×10^{-4}
40	4.774370×10^{-7}	2.133426×10^{-6}	0.008431	0.037674	2.077670×10^{-5}	9.284060×10^{-5}
80	1.195843×10^{-7}	5.343618×10^{-7}	0.001232	0.005504	$1.88584928 \times 10^{-6}$	8.426910×10^{-6}
Ratio	3.982459	3.982459	2.953555	2.953555	5.227996	5.227996

Table 2. The errors between the numerical solutions obtained from different impulsive schemes with corresponding classical 4-stage 4-order Runge–Kutta method for (50) and the exact solution of (50) at $t = 3$.

	S1IRKM		**S2IRKM**		**ICCRKM**	
m	**AE**	**RE**	**AE**	**RE**	**AE**	**RE**
10	1.626758×10^{-10}	7.269161×10^{-10}	0.008422	0.037633	3.287970×10^{-7}	1.469228×10^{-6}
20	1.022230×10^{-11}	4.567829×10^{-11}	0.008422	0.037633	4.455852×10^{-8}	1.991096×10^{-7}
40	6.374623×10^{-13}	2.848498×10^{-12}	0.008422	0.037633	1.003505×10^{-9}	4.484159×10^{-9}
80	3.674838×10^{-14}	1.642100×10^{-13}	0.001229	0.005492	5.924100×10^{-10}	2.647182×10^{-9}
Ratio	16.432140	16.432140	2.950583	2.950583	17.825269	17.825269

Table 3. The errors between the numerical solutions obtained from different impulsive schemes with corresponding Heun's method for (51) and the exact solution of (51) at $t = 3$.

	S1IHM		**S2IHM**		**ICHM**	
m	**AE**	**RE**	**AE**	**RE**	**AE**	**RE**
10	1.818238×10^{-5}	1.806009×10^{-4}	0.009707	0.096421	1.826220×10^{-4}	0.001814
20	4.602998×10^{-6}	4.572040×10^{-5}	0.009614	0.095489	4.296101×10^{-5}	4.267208×10^{-4}
40	1.157979×10^{-6}	1.150191×10^{-5}	0.002496	0.024787	1.073603×10^{-5}	1.066382×10^{-4}
80	2.904017×10^{-7}	2.884486×10^{-6}	0.002488	0.024715	2.638715×10^{-6}	2.620969×10^{-5}
Ratio	3.970884	3.970884	1.955017	1.955017	4.107037	4.107037

Table 4. The errors between the numerical solutions obtained from different impulsive schemes with corresponding classical 4-stage 4-order Runge–Kutta method for (51) and the exact solution of (51) at $t = 3$.

	S1IRKM		S2IRKM		ICCRKM	
m	**AE**	**RE**	**AE**	**RE**	**AE**	**RE**
10	1.007330×10^{-9}	1.000556×10^{-8}	0.009581	0.095163	2.293592×10^{-6}	2.278166×10^{-5}
20	6.376583×10^{-11}	6.333697×10^{-10}	0.009581	0.095163	2.057541×10^{-7}	2.043703×10^{-6}
40	3.940667×10^{-12}	3.914164×10^{-11}	0.002486	0.0246901	3.088753×10^{-8}	3.067981×10^{-7}
80	1.746658×10^{-13}	1.734911×10^{-12}	0.002486	0.0246901	3.531197×10^{-9}	3.507448×10^{-8}
Ratio	18.180002	18.180002	1.9514301	1.9514301	8.851896	8.851896

Example 2. *The above theory also holds for the following linear DEDI:*

$$\begin{cases} x'(t) = x(t), & t \geq 0,\ t \neq k,\ k = 1, 2, \cdots, \\ x(k) = (\frac{1}{3})x((k - \frac{2}{3})^-), & k \in \mathbb{Z}^+, \\ x(0) = x_0. \end{cases} \tag{51}$$

Applying mathematical induction, the exact solution to DEDI (51) *can be obtained by direct calculation as follows, for* $k \in \mathbb{N}$,

$$x(t) = x_0 \left(\frac{1}{3}e^{\frac{1}{3}}\right)^k e^{t-k},\ t \in [k, k+1).$$

Obviously, $\alpha = 1$, $\sigma = \frac{1}{3}$, $\tau_k = k$, $\beta_k = \frac{1}{3}$, $k \in \mathbb{Z}^+$. *So*

$$\beta_k e^{\alpha\sigma(\tau_k - \tau_{k-1})} = (\frac{1}{3})e^{(\frac{1}{3})(k-(k-1))} = \frac{e^{\frac{1}{3}}}{3} < 1.$$

Therefore, by Theorem 1, the exact solution of (51) *is asymptotically stable.*

By Theorems 4 and 8, if the stability function $\boldsymbol{R}(z) = \frac{P_j(z)}{Q_{\boldsymbol{k}}(z)}$ *with nonnegative coefficients of S1IRKM* (8) *(or S2IRKM* (21)*), then S1IRKM* (8) *(or S2IRKM* (21)*) for* (51) *is asymptotically stable if* $\boldsymbol{k}$ *is even and the step sizes are small enough. For example, the S1IHM (see Figure 4) and S1IRKM (see Figure 5) for* (51) *is asymptotically stable.*

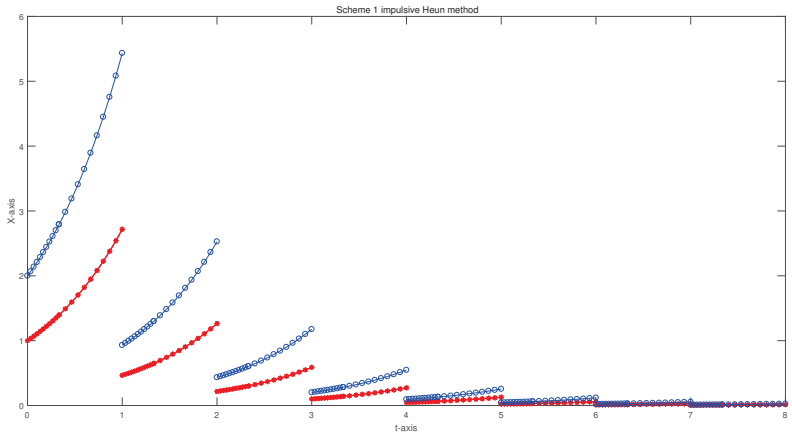


Figure 4. The S1IHM for DEDI (51) with initial values $x_0 = 1$ and $y_0 = 2$, respectively.

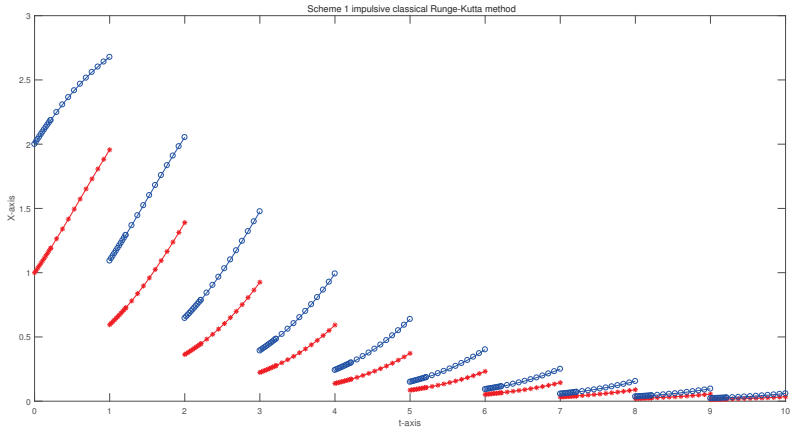


Figure 5. The S1IRKM for DEDI (51) with initial values $x_0 = 1$ and $y_0 = 2$, respectively.

Even if we take the maximum step sizes $\bar{h}_{k,1} = \frac{2}{3}$ *and* $\bar{h}_{k,2} = \frac{1}{3}$, $k \in \mathbb{N}$, $(m = 1)$, *the scheme 1 impulsive discrete Runge–Kutta methods (See Section 3) are simulated very well. From Figure 6, we can see that the curves of the exact solution of DEDI* (51) *seem to overlap with those obtained by the S1IRKM and the difference between the curves of the exact solution and those obtained by the S1IHM is not very large.*

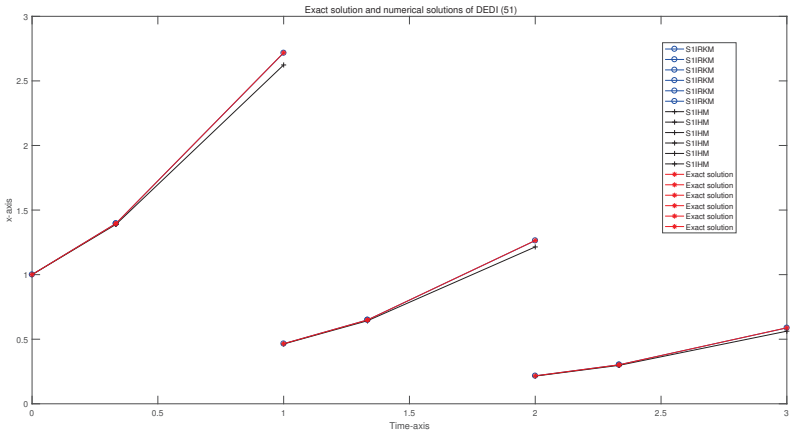


Figure 6. The exact solution of DEDI (51) and the numerical solutions obtained from S1IRKM and S1IHM for DEDI (51), respectively, when the initial values $x_0 = 1$.

Consider the following impulsive continuous Heun's method (ICHM) with corresponding two-stage Heun's method of order $p = 2$, interpolated by its unique natural continuous extension of order $q = 1$.

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \qquad b_1(\vartheta) = \frac{\vartheta}{2},$$

Heun's method, order 2 $\qquad b_2(\vartheta) = \frac{\vartheta}{2}.$

From Theorem 3, S1IHM for DEDIs (1), (50) and (51) is convergent of order 2. From Theorem 7, S2IHM for DEDIs (1), (50) and (51) is convergent at least of order 1. Applying Theorem 11, we know that the above ICHM for DEDIs (1), (50) and (51) is convergent of order $q' = 2$. These results are in general agreement with those obtained from the numerical experiments in Tables 1 and 3.

Similarly, consider the following impulsive continuous classical Runge–Kutta method (ICCRKM) with corresponding four-stage classical Runge–Kutta method of order $p = 4$, interpolated by its unique natural continuous extension of order $q = 2$.

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \qquad \begin{array}{c} b_1(\vartheta) = \left(-\frac{\vartheta}{2} + \frac{2}{3}\right)\vartheta, \\ b_2(\vartheta) = \frac{\vartheta}{3}, \\ b_3(\vartheta) = \frac{\vartheta}{3}, \\ b_4(\vartheta) = \left(\frac{\vartheta}{2} - \frac{1}{3}\right)\vartheta. \end{array}$$

Classical $4-$ stage Runge $-$ $-$Kutta method

From Theorem 3, S1IRKM with corresponding classical four-stage four-order Runge–Kutta method for DEDIs (1), (50) and (51) is convergent of order 4. From Theorem 7, S2IRKM for DEDIs (1), (50) and (51) is convergent at least of order 1. Applying Theorem 11, we know that the above ICCRKM for DEDIs (1), (50) and (51) is convergent of order $q' = 3$. These results are in general agreement with those obtained from the numerical experiments in Tables 2 and 4.

AE denotes the absolute errors between the numerical solutions and the exact solutions of DEDIs in Tables 1–4. Similarly, RE denotes the relative errors between the numerical solutions and the exact solutions of DEDIs.

As can be seen from Tables 1 and 3, when the step size is halved, both AE and RE of the scheme 1 impulsive Heun's method (S1IHM) and impulsive continuous Heun's method ((ICHM)) for DEDIs (50) and (51) become one-quarter of the original ones, respectively, which roughly indicates that both the S1IHM and ICHM for DEDIs (50) and (51) are convergent of order 2. On the other hand, when the step size is halved, both AE and RE of the scheme 2 impulsive Heun's method (S2IHM) for DEDIs (50) and (51) become one half of the original ones, respectively, which roughly indicates that both the S2IHM for DEDIs (50) and (51) are convergent of order 1.

As can be seen from Tables 2 and 4, when the step size is halved, both AE and RE of the scheme 1 impulsive classical four-stage four-order Runge–Kutta method (S1IRKM) for DEDIs (50) and (51) become one-sixteenth of the original ones, which roughly indicates that the S1ICRKM for DEDIs (50) and (51) is convergent of order 4. On the other hand, when the step size is halved, both AE and RE of the scheme 2 impulsive classical four-stage four-order Runge–Kutta method (S2IRKM)for DEDIs (50) and (51) become half of the original ones, which roughly indicates that the S2IRKM for DEDIs (50) and (51) is convergent of order 1.

As can be seen from Table 4, when the step size is halved, both AE and RE of ICCRKM for DEDI (51) become one-eighth of the original ones, which roughly indicates that ICRKM for DEDI (51) is convergent of order 3. However, in Table 2, the magnitude of the ratios of AE and RE of ICCRKM for DEDI (50) vary a little bit, but the overall look of convergence is faster than one-eighth.

7. Conclusions and Future Works

The first innovation of this paper is to consider nonlinear Lipschitz continuous function $f(t, x)$ for the dynamic system and for the impulsive term a Lipschitz continuous function I_k, which also implies new sufficient conditions for asymptotical stability of the exact solutions, and numerical solutions of DEDIs are obtained. Another innovation of this paper is that different numerical methods are constructed in order to obtain efficient numerical formats for higher-order convergence, as follows. (1) The simplest and most straightforward idea is to select the times at discontinuous points τ_k, (the moments of impulsive effects) $k \in \mathbb{Z}^+$ and the past times r_k involved in calculating the exact (or numerical) solution of the discontinuities as step nodes of the numerical method, which is also the numerical method (S1RKM) with the best convergence. The S1IRKMs are convergent of order p if the corresponding Runge–Kutta method is p-order. (2) The second idea is to select only the discontinuities τ_k as step nodes and instead of the past times r_k being selected as step nodes for the numerical method, the times $t_{k,\lfloor \sigma m \rfloor}$ near the past times r_k are taken and selected as step nodes, which is the main idea behind the construction of S2RKM. The S2IRKMs for DEDI (1) in the general case are only convergent of order 1, but are more efficient and

may be suitable for more complex DEDIs. Thus in this case, we only need to use the S2θM, which is also convergent of order 1 and simpler. (3) When the past times r_k are not chosen as step nodes, in order to overcome the convergence order problem that occurs, in the second idea, we can use the ICRKM. In this article, we prove that ICRKM for DEDI (1) is convergent of order $q' = \min\{p, q\}$, if the corresponding continuous Runge–Kutta method is consistent of order p and is consistent of uniform order q.

When the past times r_k involved in DEDIs at the moments of impulsive effects are state-dependent or stochastic, it is difficult or impossible for the past moments to be taken as step nodes, which is a problem we will address in the future. In other words, applying S2IθMs or ICRKMs to solve time-delay differential equations with state-dependent delayed impulses or differential equations with stochastic delayed impulses will be the future work. What happens if the function I_k in an impulsive term is not a continuous Lipschitz function? This is also a question we will study in the future.

Author Contributions: Conceptualization, G.-L.Z.; Software, Z.-Y.Z., Y.-C.W. and G.-L.Z.; Writing—original draft, G.-L.Z.; Writing—review and editing, G.-L.Z. and C.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (No. 11701074) and Hebei Natural Science Foundation (No. A2020501005).

Data Availability Statement: The datasets generated during the current study are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare no competing interests.

References

1. Bainov, D.D.; Simeonov, P.S. *Systems with Impulsive Effect: Stability, Theory and Applications*; Ellis Horwood: Chichester, UK, 1989.
2. Bainov, D.D.; Simeonov, P.S. *Impulsive Differential Equations: Asymptotic Properties of the Solutions*; World Scientific: Singapore, 1995.
3. Lakshmikantham, V.; Bainov, D.D.; Simeonov, P.S. *Theory of Impulsive Differential Equations*; World Scientific: Singapore, 1989.
4. Samoilenko, A.M.; Perestyuk, N.A.; Chapovsky, Y. *Impulsive Differential Equations*; World Scientific: Singapore, 1995.
5. Li, X.D.; Song, S.J. *Impulsive Systems with Delays: Stability and Control*; Science Press: Beijing, China, 2022.
6. Li, X.D.; Song, S.J.; Wu, J.H. Exponential stability of nonlinear systems with delayed impulses and applications. *IEEE Trans. Automat. Control* **2019**, *64*, 4024–4034. [CrossRef]
7. Yu, Z.Q.; Ling, S.; Liu, P.X. Exponential stability of time-delay systems with flexible delayed impulse. *Asian J. Control.* **2024**, *26*, 265–279. [CrossRef]
8. Jiang, B.; Lu, J.; Liu, Y. Exponential stability of delayed systems with average-delay impulses. *SIAM J. Control Optim.* **2020**, *58*, 3763–3784. [CrossRef]
9. He, Z.L.; Li, C.D.; Cao, Z.R.; Li, H.F. Stability of nonlinear variable-time impulsive differential systems with delayed impulses. *Nonlinear Anal. Hybrid Syst.* **2021**, *39*, 100970. [CrossRef]
10. Lu, Y.; Zhu, Q.X. Exponential stability of impulsive random delayed nonlinear systems with average-delay impulses. *J. Frankl. Inst.* **2024**, *361*, 106813. [CrossRef]
11. Chen, X.Y.; Liu, Y.; Ruan, Q.H.; Cao, J.D. Stabilization of nonlinear time-delay systems: Flexible delayed impulsive control. *Appl. Math. Model.* **2023**, *114*, 488–501. [CrossRef]
12. Chen, W.H.; Zheng, W.X. Exponential stability of nonlinear time-delay systems with delayed impulse effects. *Automatica* **2011**, *47*, 1075–1083. [CrossRef]
13. Cui, Q.; Li, L.L.; Cao, J.D. Stability of inertial delayed neural networks with stochastic delayed impulses via matrix measure method. *Neurocomputing* **2022**, *471*, 70–78. [CrossRef]
14. Li, X.D.; Zhang, X.L.; Song, S.J. Effect of delayed impulses on input-to-state stability of nonlinear systems. *Automatica* **2017**, *76*, 378–382. [CrossRef]
15. Liu, W.L.; Li, P.; Li, X.D. Impulsive systems with hybrid delayed impulses: Input-to-state stability. *Nonlinear Anal. Hybrid Syst.* **2022**, *46*, 101248. [CrossRef]
16. Niu, S.N.; Chen, W.H.; Lu, X.M.; Xu, W.X. Integral sliding mode control design for uncertain impulsive systems with delayed impulses. *J. Frankl. Inst.* **2023**, *360*, 13537–13573. [CrossRef]
17. Kuang, D.P.; Li, J.L.; Gao, D.D. Input-to-state stability of stochastic differential systems with hybrid delay-dependent impulses. *Commun. Nonlinear Sci. Numer. Simul.* **2024**, *128*, 107661. [CrossRef]
18. Ran, X.J.; Liu, M.Z.; Zhu, Q.Y. Numerical methods for impulsive differential equation. *Math. Comput. Model.* **2008**, *48*, 46–55. [CrossRef]

19. Liu, X.; Song, M.H.; Liu, M.Z. Linear multistep methods for impulsive differential equations. *Discrete Dyn. Nat. Soc.* **2012**, *2012*, 652928. [CrossRef]
20. Zhang, Z.H.; Liang, H. Collocation methods for impulsive differential equations. *Appl. Math. Comput.* **2014**, *228*, 336–348. [CrossRef]
21. Liu, M.Z.; Liang, H.; Yang, Z.W. Stability of Runge–Kutta methods in the numerical solution of linear impulsive differential equations. *Appl. Math. Comput.* **2007**, *192*, 346–357. [CrossRef]
22. Zhang, G.L. Asymptotical stability of numerical methods for semi-linear impulsive differential equations. *Comput. Appl. Math.* **2020**, *39*, 17. [CrossRef]
23. Liang, H.; Song, M.H.; Liu, M.Z. Stability of the analytic and numerical solutions for impulsive differential equations. *Appl. Numer. Math.* **2011**, *61*, 1103–1113. [CrossRef]
24. Liang, H.; Liu, M.Z.; Song, M.H. Extinction and permanence of the numerical solution of a two-preyone-predator system with impulsive effect. *Int. J. Comput. Math.* **2011**, *88*, 1305–1325. [CrossRef]
25. Liang, H. hp-Legendre-Gauss collocation method for impulsive differential equations. *Int. J. Comput. Math.* **2015**, *94*, 151–172. [CrossRef]
26. Wen, L.P.; Yu, Y.X. The analytic and numerical stability of stiff impulsive differential equations in Banach space. *Appl. Math. Lett.* **2011**, *24*, 1751–1757. [CrossRef]
27. Zhang, G.L. Convergence, consistency and zero stability of impulsive one-step numerical methods. *Appl. Math. Comput.* **2022**, *423*, 127017. [CrossRef]
28. Liu, X.; Zhang, G.L.; Liu, M.Z. Analytic and numerical exponential asymptotic stability of nonlinear impulsive differential equations. *Appl. Numer. Math.* **2014**, *81*, 40–49. [CrossRef]
29. Zhang, G.L. Asymptotical stability of Runge–Kutta methods for nonlinear impulsive differential equations. *Adv. Differ. Equ.* **2020**, *2020*, 42. . [CrossRef]
30. Ding, X.; Wu, K.N.; Liu, M.Z. The Euler scheme and its convergence for impulsive delay differential equations. *Appl. Math. Comput.* **2010**, *216*, 1566–1570. [CrossRef]
31. Zhang, G.L.; Song, M.H.; Liu, M.Z. Asymptotical stability of the exact solutions and the numerical solutions for a class of impulsive differential equations. *Appl. Math. Comput.* **2015**, *258*, 12–21. [CrossRef]
32. Zhang, G.L.; Song, M.H. Asymptotical stability of Runge–Kutta methods for advanced linear impulsive differential equations with piecewise constant arguments. *Appl. Math. Comput.* **2015**, *259*, 831–837. [CrossRef]
33. Zhang, G.L.; Song, M.H.; Liu, M.Z. Exponential stability of the exact solutions and the numerical solutions for a class of linear impulsive delay differential equations. *J. Comput. Appl. Math.* **2015**, *285*, 32–44. [CrossRef]
34. Zhang, G.L. High order Runge–Kutta methods for impulsive delay differential equations. *Appl. Math. Comput.* **2017**, *313*, 12–23. [CrossRef]
35. Zhang, G.L.; Song, M.H. Impulsive continuous Runge–Kutta methods for impulsive delay differential equations. *Appl. Math. Comput.* **2019**, *341*, 160–173. [CrossRef]
36. Wu, K.N.; Ding, X. Convergence and stability of Euler method for impulsive stochastic delay differential equations. *Appl. Math. Comput.* **2014**, *229*, 151–158. [CrossRef]
37. Zhang, G.L.; Liu, C. Two schemes of impulsive Runge–Kutta methods for linear differential equations with delayed impulses. *Mathematics* **2024**, *12*, 2075. [CrossRef]
38. Bellen, A. One-step collocation for delay differential equations. *J. Comput. Appl. Math.* **1984**, *183*, 275–283. [CrossRef]
39. Bellen, A.; Zennaro, M. *Numerical Methods for Delay Differential Equations*; Clarendon Press: Oxford, UK, 2003.
40. Brunner, H. *Collocation Methods for Volterra Integral and Related Functional Differential Equations*; Cambridge University Press: Cambridge, UK, 2004.
41. Brunner, H.; Liang, H. Stability of collocation methods for delay differential equations with vanishing delays. *BIT Numer. Math.* **2010**, *50*, 693–711. [CrossRef]
42. Liang, H.; Brunner, H. Collocation methods for differential equations with piecewise linear delays. *Commun. Pure Appl. Anal.* **2012**, *11*, 1839–1857. [CrossRef]
43. Engelborghs, K.; Luzyanina, T.; Houty, K.J.I.; Roose, D. Collocation methods for the computation of periodic solutions of delay differential equations. *SIAM J. Sci. Comput.* **2001**, *5*, 1593–1609. [CrossRef]
44. Butcher, J.C. *Numerical Methods for Ordinary Differential Equations*; Wiley: Hoboken, NJ, USA, 2003.
45. Dekker, K.; Verwer, J.G. *Stability of Runge–Kutta Methods for Stiff Nonlinear Differential Equations*; North-Holland: Amsterdam, The Netherlands, 1984.
46. Hairer, E.; Nørsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations I*; Nonstiff Problems; Springer: New York, NY, USA, 1993.
47. Hairer, E.; Nørsett, S.P.; Wanner, G. *Solving Ordinary Differential Equations II*; Stiff Problems; Springer: New York, NY, USA, 1993.
48. Wanner, G.; Hairer, E.; Nϕrsett, S.P. Order stars and stability theorems. *BIT* **1978**, *18*, 475–489. [CrossRef]
49. Song, M.H.; Yang, Z.W.; Liu, M.Z. Stability of θ-methods for advanced differential equations with piecewise continuous arguments. *Comput. Math. Appl.* **2005**, *49*, 1295–1301. [CrossRef]

50. Wang, Q.; Qiu, S. Oscillation of numerical solution in the Runge–Kutta methods for equation $x'(t) = ax(t) + a_0x([t])$. *Acta Math. Appl. Sin. Engl. Ser.* **2014**, *30*, 943–950. [CrossRef]
51. Yang, Z.W.; Liu, M.Z.; Song, M.H. Stability of Runge–Kutta methods in the numerical solution of equation $u'(t) = au(t) + a_0u([t]) + a_1u([t-1])$. *Appl. Math. Comput.* **2005**, *162*, 37–50. [CrossRef]

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

mathematics

MDPI

Article

Solving Nonlinear Equation Systems via a Steffensen-Type Higher-Order Method with Memory

Shuai Wang [1], Haomiao Xian [2,3], Tao Liu [4] and Stanford Shateyi [5,*]

[1] Foundation Department, Changchun Guanghua University, Changchun 130033, China; math_wangshuai@126.com
[2] School of Statistics, Beijing Normal University, Beijing 100875, China; xian_haomiao@163.com
[3] IT Department, Sichuan Rural Commercial United Bank, Chengdu 610000, China
[4] School of Mathematics and Statistics, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China; liutao@neuq.edu.cn
[5] Department of Mathematics and Applied Mathematics, School of Mathematical and Natural Sciences, University of Venda, P. Bag X5050, Thohoyandou 0950, South Africa
* Correspondence: stanford.shateyi@univen.ac.za

Abstract: This article introduces a multi-step solver for sets of nonlinear equations. To achieve this, we consider and develop a multi-step Steffensen-type method without memory, which does not require evaluations of the Fréchet derivatives, and subsequently extend it to a method with memory. The resulting order is $\sqrt{5}+2$, utilizing the identical number of functional evaluations as the solver without memory, thereby demonstrating a higher computational index of efficiency. Finally, we illustrate the advantages of the proposed scheme with memory through various test problems.

Keywords: with memory; Steffensen-type; higher-order methods; fractal attraction basins; efficiency index

MSC: 65H10; 41A15

1. Introductory Notes

Consider a set of square algebraic nonlinear problems as [1,2]:

$$H(\eta) = 0, \tag{1}$$

wherein $H(\eta) = (\nu_1(\eta), \nu_2(\eta), \ldots, \nu_\omega(\eta))^T$ and $\nu_i(\eta), 1 \leq i \leq \omega$ are coordinate functions. Assume that $H(\eta)$ is an enough differentiable function of η within a convex open set denoted by $D \subseteq \mathbb{R}^\omega$. Now we first revisit some pioneering iterative methods to address (1). Newton's method (NM), widely used for such problems, is formulated as follows [3]:

$$\begin{cases} H'(g^{(\chi)})\delta^{(\chi)} = -H(g^{(\chi)}), \quad \chi = 0, 1, 2, \ldots, \\ g^{(\chi+1)} = \delta^{(\chi)} + g^{(\chi)}. \end{cases} \tag{2}$$

This approach achieves second-order convergence, provided that the starting vector $g^{(0)}$ is adequately near to the true root θ. To overcome certain constraints associated with the NM, alternative solvers [4,5] have been developed, including Steffensen's method (SM), which operates without the need for derivative calculations [6]. The formulation of the SM for addressing nonlinear systems is outlined in [7]:

$$\begin{cases} \lambda^{(\chi)} = H(g^{(\chi)}) + g^{(\chi)}, \\ g^{(\chi+1)} = -[g^{(\chi)}, \lambda^{(\chi)}; H]^{-1} H(g^{(\chi)}) + g^{(\chi)}, \quad \chi = 0, 1, 2, \ldots, \end{cases} \tag{3}$$

which utilizes the divided difference operator (DDO). The first-order DDO of H for high-dimensional knots ς and ζ is defined via ($1 \leq i, j \leq \omega$):

$$[\varsigma, \zeta; H]_{i,j} = \frac{-H_i(\varsigma_1, \ldots, \varsigma_{j-1}, \zeta_j, \ldots, \zeta_\omega) + H_i(\varsigma_1, \ldots, \varsigma_j, \zeta_{j+1}, \ldots, \zeta_\omega)}{-\zeta_j + \varsigma_j}. \tag{4}$$

More generally, the DDO for H on $\mathbb{R}^\omega$ can be defined as [7]:

$$[\cdot, \cdot; H] : D \subset \mathbb{R}^\omega \times \mathbb{R}^\omega \to \mathcal{L}(\mathbb{R}^\omega), \tag{5}$$

that satisfies $[\zeta, \varsigma; H](-\varsigma + \zeta) = H(\zeta) - H(\varsigma), \quad \forall \varsigma, \zeta \in D$. Using $h = \zeta - \varsigma$, the first-order DDO can also be given by [8]:

$$[\varsigma + h, \varsigma; H] = \int_0^1 H'(\varsigma + th)\, dt. \tag{6}$$

However the definitions (4)–(6) mainly yield in dense matrices for the representation of the DDO, which restrict the applicability of the SM for tackling (1) to some extent.

The author in [9] provided another procedure to compute the DDO in a similar way as follows:

$$\begin{cases} [g^{(x)} + F^{(x)}, g^{(x)}; H] = \\ (H(g^{(x)} + F^{(x)}e^1) - H(g^{(x)}), \ldots, H(g^{(x)} + F^{(x)}e^\omega) - H(g^{(x)}))F^{(x)^{-1}}, \end{cases} \tag{7}$$

with $F^{(x)} = \text{diag}(\nu_1(g^{(x)}), \nu_2(g^{(x)}), \ldots, \nu_\omega(g^{(x)}))$. Traub (TM) introduced an improvement to NM with local cubic convergence [9]:

$$\begin{cases} \gamma^{(x)} = g^{(x)} - H'(g^{(x)})^{-1}H(g^{(x)}), \\ g^{(x+1)} = \gamma^{(x)} - H'(g^{(x)})^{-1}H(\gamma^{(x)}). \end{cases} \tag{8}$$

Another well-known and effective method for resolving (1) is the fourth-order Jarratt technique (JM) [10,11], which is described as:

$$\begin{cases} y^{(x)} = g^{(x)} - \dfrac{2}{3}H'(g^{(x)})^{-1}H(g^{(x)}), \\ g^{(x+1)} = g^{(x)} - \dfrac{1}{2}(3H'(y^{(x)}) - H'(g^{(x)}))^{-1} \\ \qquad\qquad \cdot(3H'(y^{(x)}) + H'(g^{(x)}))H'(g^{(x)})^{-1}H(g^{(x)}). \end{cases} \tag{9}$$

An improvement of (3) was furnished in [12,13] as:

$$\begin{cases} z^{(x)} = g^{(x)} - [g^{(x)}, w^{(x)}; H]^{-1}H(g^{(x)}), \\ g^{(x+1)} = z^{(x)} - [g^{(x)}, w^{(x)}; H]^{-1}H(z^{(x)}), \end{cases} \tag{10}$$

where

$$w^{(x)} = g^{(x)} + \vartheta H(g^{(x)}),\ \vartheta \in \mathbb{R}. \tag{11}$$

The distinction in (10), as opposed to (3), lies in its utilization of two iterations and consequently two ω-D function assessments, enabling it to achieve a rate superior to quadratic. The core concept is to stabilize the DDO during each cycle and subsequently augment the substeps to maximize order enhancement and contribute to the computational efficiency index (CEI) of the solver.

This manuscript is motivated by the objective of creating a multi-step fast iterative solver that improves both accuracy and efficiency in addressing nonlinear equation sets via (10). By eliminating the reliance on Fréchet derivatives, our proposed method based on an extension over TM seeks to alleviate the complexities and computational demands associated with derivative-involved approaches, thereby advancing the domain of this field.

The primary objective is to furnish a higher-order derivative-free Steffensen-type solver capable of addressing nonlinear systems, encompassing both complex and real solutions. Our intention is to enhance computational efficiency by reducing the frequency of matrix inversions and functional evaluations, in accordance with the principles of numerical analysis. In this work, functional evaluations refer to both function and derivative evaluations, which differ from the concept typically used in the Calculus of Variations.

This article is constructed as comes next. Section 2 explores the memorization technique utilized in the Steffensen-type scheme for addressing nonlinear sets of equations. Section 3 formulates a multistep approach comprising several substeps to achieve rapid convergence while minimizing the number of LU decompositions using a with memory structure to accelerate the convergence as much as possible. Section 4 presents an error analysis and assesses the rate of convergence. Subsequently, Section 5 examines the CEI of various methods, focusing on the number of functional assessments and the flops-type index. Furthermore, Section 6 demonstrates the applicability and advantages of the proposed method with memory through its application to several problems. Finally, Section 7 offers concluding remarks.

2. With Memorization of the Iterative Methods

In this context, our objective is to enhance the CEI of (10) without adding additional steps or further DDOs for every iterate; see [14]. To achieve this, we utilize the concept of memory-based methods, which suggest that the speed of convergence and overall efficiency of iterative techniques can be enhanced by retaining and utilizing previously calculated function values.

Noting that the error equation for (10) (the notations here will be pointed out further in Section 3):

$$\varepsilon^{(\chi+1)} = (I + \vartheta A'(\theta))(2I + \vartheta A'(\theta))C_2\varepsilon^{(\chi)^3} + \mathcal{O}(\varepsilon^{(\chi)^4}), \tag{12}$$

contains a term expressed as follows:

$$I + \vartheta H'(\theta) = 0. \tag{13}$$

Here the non-zero scalar ϑ in (13) significantly influences both the convergence domain and the enhancement of the convergence speed. When addressing a nonlinear set of problems, and since θ is unknown, one could approximate $H'(\theta)$ to bring the entire relation in (13) close to zero. Thus, we can express this as

$$\vartheta \simeq -H'(\bar{\theta})^{-1}, \tag{14}$$

where $\bar{\theta}$ represents an estimation of the root (for each iteration).

It is crucial to elaborate on how one estimates the matrix $\vartheta := A^{(\chi)}$, $(\chi \geq 1)$ by utilizing certain approximations of $-H'(\theta)$ derived from the available data [15].

To enhance the performance of (10) through the principle of memory-based methods [13], we take into account the following iterative expression without memory shown as PM1:

$$\begin{cases} w^{(\chi)} &= g^{(\chi)} + \vartheta H(g^{(\chi)}) \\ q^{(\chi)} &= g^{(\chi)} - [g^{(\chi)}, w^{(\chi)}; H]^{-1} H(g^{(\chi)}), \\ z^{(\chi)} &= q^{(\chi)} - [g^{(\chi)}, w^{(\chi)}; H]^{-1} H(q^{(\chi)}), \\ g^{(\chi+1)} &= z^{(\chi)} - [g^{(\chi)}, w^{(\chi)}; H]^{-1} H(z^{(\chi)}). \end{cases} \tag{15}$$

This solver reads the following error equation:

$$\varepsilon^{(\chi+1)} = (I + \vartheta H'(\theta))(2I + \vartheta H'(\theta))C_2^2\varepsilon^{(\chi)^4} + \mathcal{O}(\varepsilon^{(\chi)^5}). \tag{16}$$

Without loss of generality, we focus on *the scalar case* to analyze the dynamical behavior of the iterative methods in the complex plane, rather than extending to the multi-dimensional case for (19). Visualizing the fractal attraction basins of iterative methods for polynomial

equations of various degrees in the complex plane is critical for several reasons [16,17], particularly when shading the plot based on the number of iterations required for convergence. In this context, different polynomial roots correspond to distinct regions of attraction. By mapping these basins, one can illustrate where initial guesses converge to specific roots. This step is essential in our work, demonstrating how the use of memory and small free parameter values can expand the convergence radii, thereby enlarging the region for selecting initial approximations.

Shading the plot based on the number of iterations needed for convergence offers insights into the solver's effectiveness. Regions where the method converges rapidly can be identified as more stable or efficient, whereas areas requiring more iterations (or failing to converge) suggest potential inefficiencies or instability. Such analyses are illustrated in Figures 1–4 over the domain $[-2,2] \times [-2,2]$, with a maximum iteration count of 150 and a tolerance of 10^{-2} for the residual as the stopping criterion. They reveal that the higher the order is for Steffensen-type methods, the larger the attraction basin is. Note that PM1 and PM2 both are Steffensen-type methods. In these new methods, the number of iterations to get the root is lower than SM; due to this, they have lighter and fewer shaded areas in their attraction basins. Moreover, the convergence radii could be enlarged by selecting small values for the free non-zero scalar ϑ. Thus, memorization will not only contribute to a higher-efficiency index but also to larger attraction basins, which means higher stability for such a solver in contrast to the Steffensen-type method without memory.

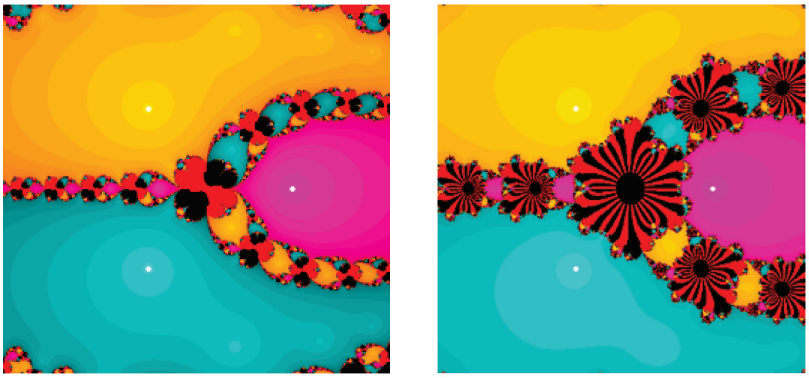


Figure 1. Fractal attraction basins for $z^3 - 1 = 0$, SM on the left and PM1 on the right using $\vartheta = 0.2$.

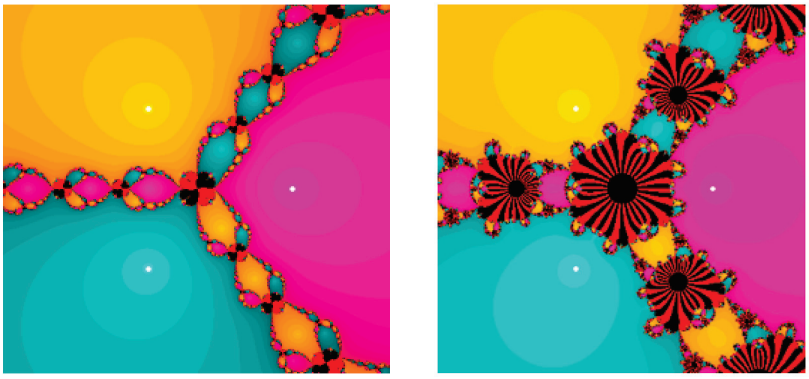


Figure 2. Fractal attraction basins for $z^3 - 1 = 0$, SM on the left and PM1 on the right using $\vartheta = 0.02$.

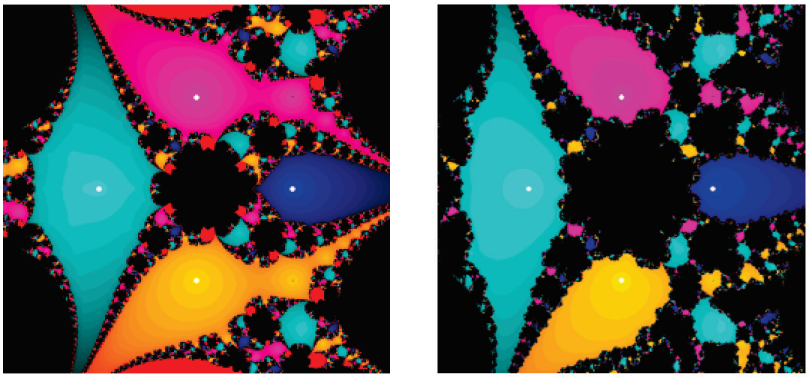


Figure 3. Fractal attraction basins for $z^4 - 1 = 0$, SM on the left and PM1 on the right using $\vartheta = 0.2$.

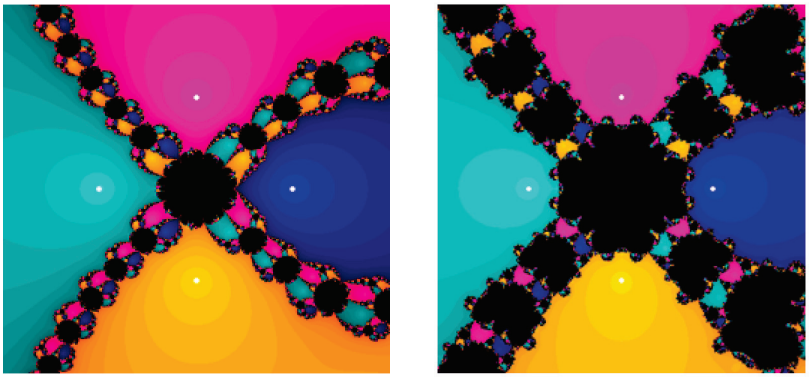


Figure 4. Fractal attraction basins for $z^4 - 1 = 0$, SM on the left and PM1 on the right using $\vartheta = 0.02$.

Figures 1–4 also reveal that by observing how the number of iterations varies across the complex plane, one can assess the convergence properties of any methods. The fractal boundaries highlight regions where small changes in the initial guess can yield drastically different outcomes (i.e., converging to different roots or diverging). This sensitivity is crucial to understand, especially when implementing these methods in practical applications where precision of the initial guess might be limited. They are used later in the paper by implying to select a small value for the free nonzero parameter (carried forward the relation (37)) and how memorization can enhance the convergence domain.

3. Derivation of the Scheme

To facilitate the implementation of the memory-based scheme (15), we will first examine

$$\vartheta := A^{(\chi)} = -[w^{(\chi-1)}, g^{(\chi-1)}; H]^{-1} \approx -H'(\theta)^{-1}, \tag{17}$$

and

$$\begin{cases} N^{(\chi-1)}\delta^{(\chi)} = -H(g^{(\chi)}), \\ N^{(\chi-1)}\gamma^{(\chi)} = -H(q^{(\chi)}), \\ N^{(\chi-1)}\psi^{(\chi)} = -H(z^{(\chi)}). \end{cases} \tag{18}$$

Consequently, we now present the subsequent scheme (PM2) as our main contribution, ($A^{(\chi)} = -[w^{(\chi-1)}, g^{(\chi-1)}; H]^{-1}, \chi \geq 1$):

$$\begin{cases} w^{(\chi)} = g^{(\chi)} + A^{(\chi)}H(g^{(\chi)}), & \chi \geq 1, \\ q^{(\chi)} = g^{(\chi)} + \delta^{(\chi)}, & \chi \geq 0, \\ z^{(\chi+1)} = q^{(\chi)} + \gamma^{(\chi)}, \\ g^{(\chi+1)} = z^{(\chi)} + \psi^{(\chi)}. \end{cases} \tag{19}$$

The error equation of this solver with memory will be given in the next section.

It is well known [15] when $D \subset \mathbb{R}^{\omega}$ represents a convex nonempty domain. Then, assume that H is three-times Fréchet smooth over D, and $[u, v; H] \in \mathcal{L}(D, D)$ for any distinct points $u, v \in D$ (where $v \neq u$). Additionally, let the starting vector $g^{(0)}$ and the zero θ be in close proximity to each other. By defining $A^{(\chi)} = -[w^{(\chi-1)}, g^{(\chi-1)}; H]^{-1}$ and setting $d^{(\chi)} := I + A^{(\chi)} H'(\theta)$, finally we can derive the equation below

$$d^{(\chi)} \sim e^{(\chi-1)}. \tag{20}$$

The relation (20) will be used later in Section 4 of this work.

To implement (19), it is essential to resolve a series of linear algebraic sets of equations. This entails performing a new LU factorization at each iteration, without leveraging any information from preceding steps. Nevertheless, a substantial body of the literature exists regarding the recycling of such information to derive updated preconditioners for iterative solvers [18]. The advantage of (19) lies in the fact that all linear systems share the identical coefficient matrix. Consequently, a single LU factorization suffices; by retaining this decomposition, it can be applied to several distinct right-hand side parts to obtain the resolution vectors for each sub-cycle of (19).

The solution of the nonlinear equation systems that we consider here are in $D \subseteq \mathbb{R}^{\omega}$ as stated in Section 1. The roots that we are seeking for are assumed to be simple zeros. Both real and complex roots can be obtained by the discussed methods (if existed). In fact, by choosing a suitable complex initial guess, a complex root (if existed), can be obtained.

4. Convergence Order

Here, we furnish a theoretical analysis of the convergence speed of the iterative scheme presented in (19). Before introducing the main contribution, we represent the ω-dimensional Taylor expansion.

The rate at which the iteration without memory PM1 converges is determined via ω-dimensional Taylor expansions. Let $\varepsilon^{(\chi)} = g^{(\chi)} - \theta$ denote the error at the χ-th iterate. As noted in [19]:

$$\varepsilon^{(\chi+1)} = G\varepsilon^{(\chi)^p} + \mathcal{O}(\varepsilon^{(\chi)^{p+1}}), \tag{21}$$

this error equation implies G is a p-linear functional, where $G \in \mathcal{L}(\mathbb{R}^{\omega}, \mathbb{R}^{\omega}, \ldots, \mathbb{R}^{\omega})$ and p is the speed. Additionally, we have:

$$\varepsilon^{(\chi)^p} = (\underbrace{\varepsilon^{(\chi)}, \varepsilon^{(\chi)}, \ldots, \varepsilon^{(\chi)}}_{p \text{ terms}}). \tag{22}$$

Assume that $H : D \subseteq \mathbb{R}^{\omega} \to \mathbb{R}^{\omega}$ is sufficiently differentiable in the Fréchet sense in D. Following [10], the ω-th derivative of H at $u \in \mathbb{R}^{\omega}$, $\omega \geq 1$, is the ω-linear functional, i.e.,

$$H^{(\omega)}(u) : \mathbb{R}^{\omega} \times \cdots \times \mathbb{R}^{\omega} \to \mathbb{R}^{\omega}, \tag{23}$$

so that $H^{(\omega)}(u)(v_1, \ldots, v_{\omega}) \in \mathbb{R}^{\omega}$. For $\theta + h \in \mathbb{R}^{\omega}$ located in a vicinity of the solution θ of (1), the Taylor expansion can be formulated as [10]:

$$H(\theta + h) = H'(\theta)\left[h + \sum_{\omega=2}^{p-1} M_{\omega} h^{\omega}\right] + \mathcal{O}(h^p), \tag{24}$$

where $M_{\omega} = \dfrac{1}{\omega!}[H'(\theta)]^{-1} H^{(\omega)}(\theta)$, $\omega \geq 2$. It follows that $M_{\omega} h^{\omega} \in \mathbb{R}^{\omega}$, as $H^{(\omega)}(\theta) \in \mathcal{L}(\mathbb{R}^{\omega} \times \cdots \times \mathbb{R}^{\omega}, \mathbb{R}^{\omega})$ and $[H'(\theta)]^{-1} \in \mathcal{L}(\mathbb{R}^{\omega})$. Additionally, for H', we have:

$$H'(\theta + h) = H'(\theta)\left[I + \sum_{\omega=2}^{p-1} \omega M_{\omega} h^{\omega-1}\right] + \mathcal{O}(h^p), \tag{25}$$

wherein I shows the unit matrix. Additionally, $\omega M_{\omega} h^{\omega-1} \in \mathcal{L}(\mathbb{R}^{\omega})$.

Theorem 1. *Let in (1) $H : D \subseteq \mathbb{R}^{\omega} \to \mathbb{R}^{\omega}$ be adequately Fréchet differentiable at every point in D and that $H(\theta) = 0$ at $\theta \in \mathbb{R}^{\omega}$. Additionally, let $H'(\eta)$ be continuous and nonsingular at θ. Next, $\{g^{(\chi)}\}_{\chi \geq 0}$ produced by (19) with memory with a selection of an appropriate starting value has 4.23607 R-convergence order.*

Proof. For proving the convergence speed, we consider (24) and (25) to write

$$H(g^{(\chi)}) = H'(\theta)\left[\varepsilon^{(\chi)} + M_2 {\varepsilon^{(\chi)}}^2 + M_3 {\varepsilon^{(\chi)}}^3\right] + \mathcal{O}({\varepsilon^{(\chi)}}^4). \tag{26}$$

In the context of the scheme (15), when operating without memory (i.e., PM1) and applying (24)–(26), we ultimately derive

$$\varepsilon^{(\chi+1)} = (I + \vartheta H'(\theta))(2I + \vartheta H'(\theta)) C_2^2 {\varepsilon^{(\chi)}}^4 + \mathcal{O}({\varepsilon^{(\chi)}}^5). \tag{27}$$

Now by considering the with memorization in (19), we shall express (27) in its asymptotic form as follows:

$$\varepsilon^{(\chi+1)} \sim d_1^{(\chi)} {\varepsilon^{(\chi)}}^4, \tag{28}$$

where $\sim$ shows for the error equation without the asymptotical term. A variety of symbolic computations conducted by considering that the coefficients of the error terms in our ω-D scenario are all matrices, and applying (20), along with the understanding that multiplication does not allow for commutativity, we can deduce

$$d_1^{(\chi)} \sim \varepsilon^{(\chi-1)}, \quad \forall \chi \geq 1. \tag{29}$$

Consequently, one arrives at

$$ {d_1^{(\chi)}}^2 \sim {\varepsilon^{(\chi-1)}}^2, \qquad \forall \chi \geq 1. \tag{30}$$

By imposing (29) and (30) into (28), we obtain:

$$\varepsilon^{(\chi+1)} \sim {\varepsilon^{(\chi-1)}}^1 {\varepsilon^{(\chi)}}^4. \tag{31}$$

This demonstrates that $\frac{1}{p} + 4 = p$, where the convergence R-order is expressed as

$$p = \left(\sqrt{5} + 2\right) \simeq 4.23607. \tag{32}$$

The proof is concluded. □

Before ending this section, it is pointed out that with a simple change the structure of PM2, it is possible to provide another iteration scheme with memory of a similar kind as follows (PM3), (this time: $A^{(\chi)} = [w^{(\chi-1)}, g^{(\chi-1)}; H]^{-1}, \chi \geq 1$):

$$\begin{cases} w^{(\chi)} = g^{(\chi)} - A^{(\chi)} H(g^{(\chi)}), & \chi \geq 1, \\ q^{(\chi)} = g^{(\chi)} + \delta^{(\chi)}, & \chi \geq 0, \\ z^{(\chi+1)} = q^{(\chi)} + \gamma^{(\chi)}, & \\ g^{(\chi+1)} = z^{(\chi)} + \psi^{(\chi)}. & \end{cases} \tag{33}$$

If we aim to further improve the convergence order, two possible approaches can be considered. First, we could construct a solver by incorporating additional subsets and then apply the memorization procedure. Second, one might explore faster methods to accelerate

the free parameter using alternative types of interpolation, which, however, falls beyond the scope of this paper.

5. Computational Efficiency Comparisons

For the proposed solver with memory, PM2 (or equivalently PM3), only a single LU factorization is required, followed by matrix-vector multiplications, which enhances computational efficiency by eliminating the need to compute multiple matrix inverses in each iteration. The CEI for iterative solvers is defined as follows [7]:

$$CEI = p^{1/\mathcal{C}}, \tag{34}$$

where $\mathcal{C}$ represents the total computational cost and p signifies the convergence speed, considering the quantity of functional evaluations. The cost of computing each scalar function is considered a unit, while the costs associated with other calculations are expressed as multiples of this unit cost. To evaluate the CEI for PM2, we first outline the number of functional evaluations (cost) required for ω-dimensional functions, as detailed below (excluding the indicator χ):

- In $H(g), H(w), H(q), H(z)$: 4ω evaluations.
- In the DDO: ω^2 evaluations.

Furthermore, we consider the costs to solve two triangular systems, all quantified in floating-point operations (flops). The flops necessary for executing the LU procedure amount to $(2\omega^3)/3$, while resolving the two related triangular systems requires approximately $2\omega^2$ flops. Noting that, here, we assume that the cost for one functional evaluation is roughly equal to the cost for one flop. The findings displayed in Table 1 demonstrate that for varying ω, the CEI of our method surpasses that of competing approaches. Comparisons of different derivative-free Steffensen-type solvers with and without memory based on various choices of ω are illustrated in Figures 5 and 6, confirming the superiority and improvement in both classic and flops-type efficiency indices of PM relative to its main competitors.

Table 1. Efficiency indices for several Steffensen-type methods.

Compared Methods	**SM**	**PM1**	**PM2**
Order	2	4	4.23607
Function assessments	$\omega+\omega^2$	$4\omega+\omega^2$	$4\omega+\omega^2$
The classical CEI	$2^{\frac{1}{\omega+\omega^2}}$	$4^{\frac{1}{4\omega+\omega^2}}$	$4.23607^{\frac{1}{4\omega+\omega^2}}$
No. of LU decomposition	1	1	1
Assessments for LU decompositions (flops)	$\frac{2\omega^3}{3}$	$\frac{2\omega^3}{3}$	$\frac{2\omega^3}{3}$
Assessments for linear systems (flops)	$\frac{2\omega^3}{3}+2\omega^2$	$\frac{2\omega^3}{3}+6\omega^2$	$\frac{2\omega^3}{3}+6\omega^2$
Flops-type CEI	$2^{\frac{1}{\frac{2\omega^3}{3}+3\omega^2+\omega}}$	$4^{\frac{1}{\frac{2\omega^3}{3}+7\omega^2+4\omega}}$	$4.23607^{\frac{1}{\frac{2\omega^3}{3}+7\omega^2+4\omega}}$

In real-world applications, there is a trade-off between eliminating Fréchet-derivatives and increasing the method's overall computational complexity. Generally speaking, it relies on the specific problem, which leads to a nonlinear system of equations. However, by eliminating the Fréchet derivative, we make the solver derivative-free, which is suitable for problems, in which the derivative is not available. Moreover, the concept of memorization can be accompanied in methods without Fréchet derivatives to increase the computational complexity.

It is also remarked that in the absence of Fréchet derivatives, one might ask that what specific mechanisms ensure that the proposed Steffensen-type technique remains robust over a variety of problem sets? To tackle this, it is stated that in the absence of Fréchet derivatives, the convergence radius mainly reduced tremendously, which is one weak point of Steffensen-type techniques at first sight. However, this can simply be improved by

choosing very small values for the free non-zero parameter (as seen in the attraction basins of Section 2), as well as imposing the with memorization concept.

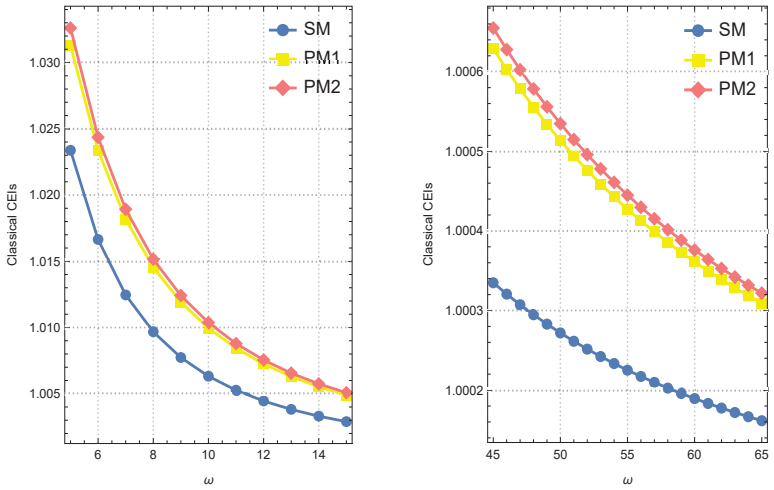


Figure 5. Comparison of classic CEIs for various values of ω.

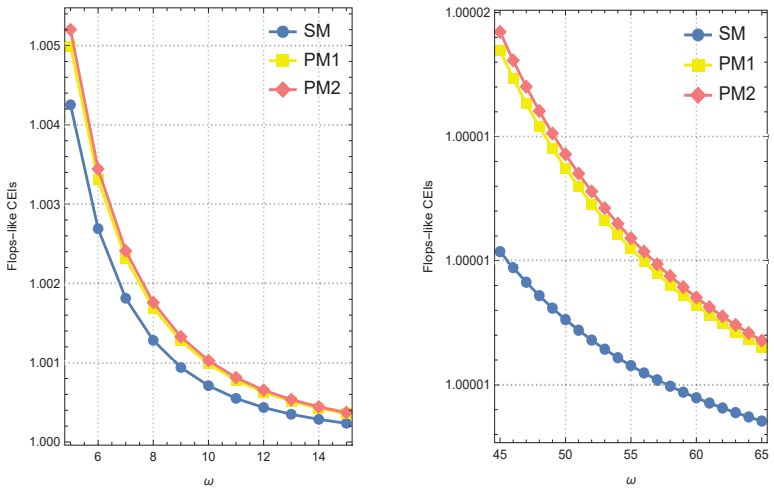


Figure 6. Comparison of flops-type CEIs for various values of ω.

6. Numerical Aspects

The objective of this section is to facilitate the implementation of our proposed scheme, PM2. The computations were executed using Mathematica 13.3 [20] in standard machine precision to handle the round-off errors. The linear systems were resolved employing LU decomposition via `LinearSolver[]`. All computational examples were conducted in a uniform environment. We adopt the stopping criterion as follows:

$$||H(g^{(\chi+1)})||_\infty \leq 10^{-6}. \tag{35}$$

A significant challenge in implementing (19) involves the integration of $A^{(\chi)}$, which is no longer constant and must be characterized as a matrix. Here, $A^{(0)}$ is delineated follows:

$$A^{(0)} = \text{diag}(-0.01). \tag{36}$$

In fact due to Figures 1–4 and the results discussed previously in [13], other small values for the free parameter can also be used which can yield in different choices such as

$$A^{(0)} = \text{diag}(-0.001), \tag{37}$$

or

$$A^{(0)} = \text{diag}(-0.0001), \tag{38}$$

or

$$A^{(0)} = \text{diag}(-0.00001). \tag{39}$$

In fact, for implementation of method with memory, here, for the first iterate, we start with the without memory version of the method and after that, from the second iterate, the information of the previous step can be used to update the parameters and thus with memorization of the scheme can be done. The selection of $A^{(0)}$ has a direct impact on the entire process, influencing the speed at which convergence is achieved. Here, (37) aligns with the dynamic investigations of Steffensen-type solvers with memory, where larger basins of attraction occur when the free parameter is near zero.

To validate the analytical convergence rate in the computational experiments, we determine the numeric speed of convergence using [7]

$$\rho \approx \frac{\ln(||H(g^{(\chi+1)})||_2/||H(g^{(\chi)})||_2)}{\ln(||H(g^{(\chi)})||_2/||H(g^{(\chi-1)})||_2)}. \tag{40}$$

Example 1 ([14]). *We investigate a nonlinear system $H(\eta) = 0$, which possesses a complex root, as detailed below*

$$H(\eta) = \begin{cases} \eta_1 \sin(\eta_2) - 2\eta_{10}{}^{\eta_8} + \eta_{10} - 5\eta_6 - 10\eta_9, \\ 10\eta_1 + \eta_3{}^2 - 5\eta_5{}^2 + 10\eta_6{}^{\eta_8} - \sin(\eta_7) + 2\eta_9, \\ \cos^{-1}(-10\eta_{10} + \eta_8 + \eta_9) + \eta_4 \sin(\eta_2) + \eta_3 - 15\eta_5{}^2 + \eta_7, \\ \eta_1\eta_2{}^{\eta_7} - \eta_8{}^{\eta_{10}} + \eta_3{}^5 - 5\eta_5{}^3 + \eta_7, \\ 10\eta_1{}^2 - \eta_{10} + \cos(\eta_2) + \eta_3{}^2 - 5\eta_6{}^3 - 2\eta_8 - 4^{\eta_9}, \\ \cos^{-1}(\eta_1{}^2)\sin(\eta_2) - 2\eta_{10}\eta_5{}^4\eta_6\eta_9 + \eta_3{}^2, \\ 2\tan(\eta_1{}^2) + 2^{\eta_2} + \eta_3{}^2 - 5\eta_5{}^3 - \eta_6 + \eta_8{}^{\cos(\eta_9)}, \\ \eta_1{}^2 - \eta_{10}\eta_5\eta_6\eta_7\eta_8\eta_9 + \tan(\eta_2) + 2\eta_3{}^{\eta_4} - 5\eta_6{}^3, \\ 5\tan(\eta_1 + 2) + \cos(\eta_9{}^{\eta_{10}}) + \eta_2{}^3 + 7\eta_3{}^4 - 2\sin^3(\eta_6), \\ 5\exp(\eta_1 - 2)\eta_2 + 2\eta_7{}^{\eta_{10}} + 8\eta_3{}^{\eta_4} - 5\eta_6{}^3 - \eta_9, \end{cases} \tag{41}$$

in which the precise solution is presented up to eight floating points as

$$\begin{aligned} \theta \simeq\ & (1.32734904 + 0.35029249i, 1.0585993 - 1.7487246i, \\ & 1.02761867 - 0.01413080i, 3.2739500 + 0.1278283i, \\ & 0.83182439 + 0.00175519i, -0.4853245912 + 0.68487764i, \\ & 0.16936676 + 0.1840917i, 1.5344199 - 0.3212147i, \\ & 2.0863796 + 0.42634275i, -1.9895923 + 1.4783953i)^*. \end{aligned} \tag{42}$$

The computational evidence and the numerical speed, denoted as ρ, are detailed in Table 2. We utilized 1000 fixed floating points, with the initial value set as $g^{(0)} = (1.20 + 0.30i, 1.10 - 1.90i, 1.00 - 0.10i, 2.50 + 0.50i, 0.80 - 0.10i, -0.40 + 1.00i, 0.10 + 0.10i, 1.30 - 0.70i, 2.00 + 0.50i, -1.90 + 1.40i)^*$. The choice of this is based on $g^{(0)}$ [14]. Rather than $g^{(0)}$ and (37), no other parameters should be chosen and the method works based on satisfying (35). Additionally, the residual norm is expressed using the $\|\cdot\|_2$ notation. The numerical pieces of evidence seen in Table 2 support the observations in Figures 1–4, discussing that the smaller the choice of the free parameter for PM2 would results in arriving at the convergence phase faster than consider larger values for this parameter.

Table 2. Comparisons of different methods with and without memory in Example 1.

Methods	$\|H(g^{(3)})\|$	$\|H(g^{(4)})\|$	$\|H(g^{(5)})\|$	$\|H(g^{(6)})\|$	$\|H(g^{(7)})\|$	$\|H(g^{(8)})\|$	$\|H(g^{(9)})\|$	ρ
NM	8.19×10^{-1}	2.73×10^{-2}	1.79×10^{-5}	1.28×10^{-11}	2.52×10^{-23}	8.28×10^{-47}	2.50×10^{-94}	2.02
SM	7.68×10^{-1}	1.83×10^{-2}	7.33×10^{-6}	5.17×10^{-12}	1.51×10^{-24}	2.03×10^{-49}	3.91×10^{-99}	1.99
PM1	1.01×10^{-4}	6.95×10^{-20}	1.19×10^{-80}	2.12×10^{-323}				3.99
PM2 with (36)	2.19×10^{-3}	6.12×10^{-16}	3.29×10^{-69}	7.45×10^{-295}				4.23
PM2 with (37)	1.44×10^{-4}	2.99×10^{-21}	7.96×10^{-92}	1.09×10^{-390}				4.23
PM2 with (38)	1.60×10^{-4}	2.88×10^{-21}	1.05×10^{-91}	2.88×10^{-390}				4.23
PM2 with (39)	1.58×10^{-4}	2.61×10^{-21}	8.16×10^{-92}	9.18×10^{-391}				4.24

Example 2 ([21])**.** *In this test, we investigate the nonlinear systems extracted through the computational resolution of the following Partial Differential Equation (PDE)*

$$\begin{cases} \mathfrak{u}_\tau + \mathfrak{u}\mathfrak{u}_z = \bar{\rho}\mathfrak{u}_{zz}, \\ \mathfrak{u}(1,\tau) = 0,\ \tau \geq 0, \\ \mathfrak{u}(0,\tau) = 0,\ \tau \geq 0, \\ \mathfrak{u}(z,0) = \dfrac{2\bar{\rho}\bar{\vartheta}\pi \sin(\pi z)}{\bar{\xi} + \bar{\vartheta}\cos(\pi z)},\ 0 \leq z \leq 2, \end{cases} \tag{43}$$

where the coefficient of diffusion is $\bar{\rho}$*. If we take into account* $\mathfrak{u} = \mathfrak{u}(z,\tau)$*, then the computational resolution is represented by:*

$$\varrho_{i,j} \simeq \mathfrak{u}(z_i, \tau_j), \tag{44}$$

at the grid points (i,j) *on a equidistant mesh. Let* ϖ_1 *and* ϖ_2 *show the number of steps in spatial and temporal variables, respectively. The parameters are set as follows [21]:* $\bar{\xi} = 5$*,* $T = 1$*,* $\bar{\rho} = 0.5$*, and* $\bar{\vartheta} = 4$*. The parameters have been chosen so as to obtain a unique and well-defined numerical solution and stay away from stiffness or irregularity for the solution of PDEs.*

To address this, we may utilize the first-order backward FD method for the first differentiation in temporal τ:

$$\mathfrak{u}_\tau(z_i, \tau_j) \simeq \frac{\varrho_{i,j} - \varrho_{i,j-1}}{k}, \tag{45}$$

wherein k denotes the time step size. For the spatial terms of the PDE, we apply the second-order central difference FD method as follows:

$$\mathfrak{u}_z(z_i, \tau_j) \simeq \frac{\varrho_{i+1,j} - \varrho_{i-1,j}}{2h}, \tag{46}$$

and

$$\mathfrak{u}_{zz}(z_i, \tau_j) \simeq \frac{\varrho_{i+1,j} - 2\varrho_{i,j} + \varrho_{i-1,j}}{h^2}, \tag{47}$$

where h represents the spatial step size along z. Following this discretization and imposing the boundary conditions will lead a to a set of nonlinear equations that must be tackled iteratively.

The numerical solutions using PM2 are illustrated in Figure 7, and Table 3 provides the comparative evidence for this case in double precision when the residual of the numerical obtained solution is less than 10^{-5}. We set $\varpi_1 = \varpi_2 = 21$, i.e., 21 equally spaced nodes in both space and time direction and after imposing the initial and boundary conditions, we derive to a set of nonlinear system of the dimension 400×400, where the initial vector $g^{(0)} = \mathbf{1}$. Along space, we have used central three-point second-order FD approximations, and along time, we have employed first order forward FD approximations to discretize the problem.

For an alternative set of parameters, specifically $\varpi_1 = \varpi_2 = 31$, representing 31 equally spaced points in both the spatial and temporal directions, we obtained a nonlinear system of dimension 900×900 after applying the initial and boundary conditions. The initial vector is again defined as $g^{(0)} = \mathbf{1}$. Figure 8 presents the numerical simulations for this setup, further demonstrating the effectiveness of PM2 and the concept of memorization.

Table 3. Computational outcomes obtained in Example 2.

Different Solvers	SM	PM1	PM2 with (37)
No. of iterates	7	2	1
Time (in seconds)	6.12	5.11	3.27

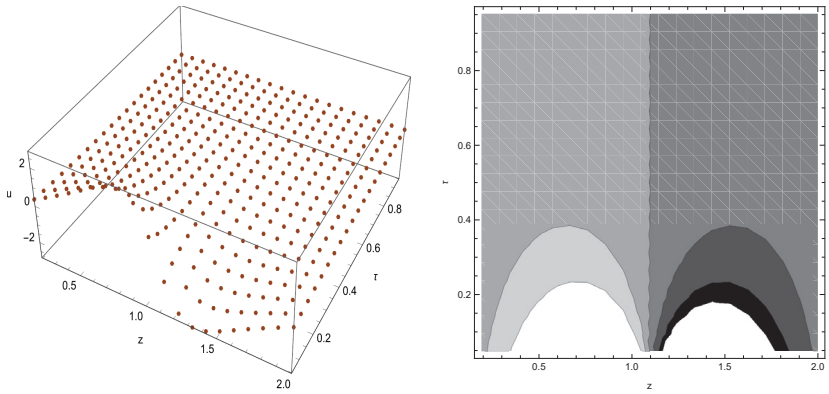


Figure 7. Numerical solutions using PM2 with (37) based on the given grid and $\varpi_1 = \varpi_2 = 21$ in left and its contour plot in right for Example 2.

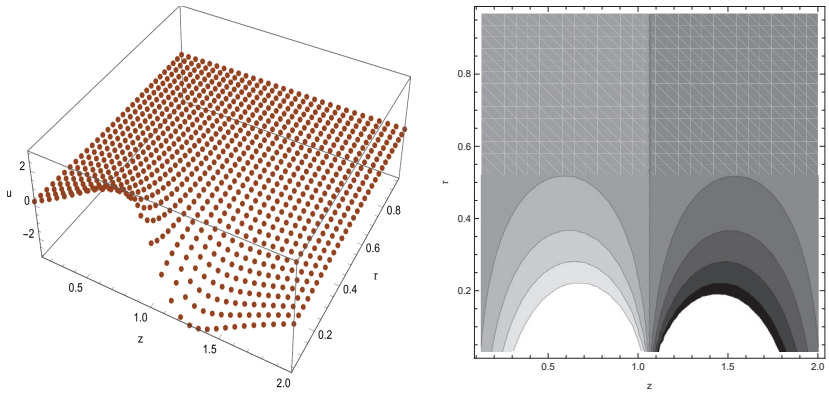


Figure 8. Numerical solutions using PM2 with (37) based on the given grid and $\varpi_1 = \varpi_2 = 31$ in left and its contour plot in right for Example 2.

The numerical performance of the proposed solver is demonstrated through root-finding for various nonlinear equations. Results from the numerical tests clearly indicate that the solver achieves convergence in fewer iterations and with higher accuracy per iteration.

We finish this section by pointing out the following matter. Another concern may arise regarding how the method handles convergence issues in nonlinear systems with multiple solutions or when the initial guess significantly influences convergence behavior. To address this, the focus of this article is on simple roots, not multiple roots; while the methods can be applied to find multiple zeros, their orders of convergence drop significantly in such cases. In fact, if a nonlinear system has multiple roots (whether the multiplicity is known or unknown), specialized solvers designed for multiple roots, along with appropriate initial guesses, must be developed to maximize efficiency.

7. Concluding Summaries

In this paper, we have presented an advanced Steffensen-type iteration expression aimed at solving nonlinear systems of equations, specifically tailored to eliminate the need for computing Fréchet derivatives. This approach has exhibited significant computational efficiency by reducing the number of matrix inversions and functional evaluations, as outlined in Sections 3–5. Our examination has validated the efficacy of the proposed method. Future endeavors will focus on further improving the efficiency of the scheme and expanding its applicability to a wider variety of nonlinear challenges, including nonlinear stochastic differential equations as highlighted in [22].

Author Contributions: Conceptualization, S.W., H.X., T.L. and S.S.; formal analysis, S.W., H.X., T.L. and S.S.; funding acquisition, T.L. and S.S.; investigation, S.W., H.X. and T.L.; methodology, S.W., H.X., T.L. and S.S.; supervision, T.L.; validation, T.L. and S.S.; writing—original draft, S.W., H.X. and T.L.; writing—review & editing, S.W., H.X. and T.L. All authors have read and agreed to the published version of the manuscript.

Funding: The research was funded by the Research Project on Graduate Education and Teaching Reform of Hebei Province of China (YJG2024133), the Open Fund Project of Marine Ecological Restoration and Smart Ocean Engineering Research Center of Hebei Province (HBMESO2321), the Technical Service Project of Eighth Geological Brigade of Hebei Bureau of Geology and Mineral Resources Exploration (KJ2022-021), the Technical Service Project of Hebei Baodi Construction Engineering Co., Ltd. (KJ2024-012), the Natural Science Foundation of Hebei Province of China (A2020501007), and the Fundamental Research Funds for the Central Universities (N2123015).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Acknowledgments: We would like to express our sincere gratitude to the four referees whose insightful comments significantly enhanced both the clarity and robustness of this manuscript.

Conflicts of Interest: We have no financial conflicts of interest or personal relationships that might have affected the research presented in this manuscript.

References

1. McNamee, J.M.; Pan, V.Y. *Numerical Methods for Roots of Polynomials–Part I*; Elsevier: Amsterdam, The Netherlands, 2007.
2. Bini, D.A. Numerical computation of the roots of Mandelbrot polynomials: An experimental analysis. *Electron. Trans. Numer. Anal.* **2024**, *61*, 1–27. [CrossRef]
3. Ortega, J.M.; Rheinboldt, W.C. *Iterative Solution of Nonlinear Equations in Several Variables*; Academic Press: New York, NY, USA, 1970.
4. Khdhr, F.W.; Soleymani, F.; Saeed, R.K.; Akgül, A. An optimized Steffensen–type iterative method with memory associated with annuity calculation. *Eur. Phys. J. Plus* **2019**, *134*, 146. [CrossRef]
5. Torkashvand, V.; Kazemi, M.; Azimi, M. Efficient family of three-step with-memory methods and their dynamics. *Comput. Methods Differ. Equ.* **2024**, *12*, 599–609.
6. Noda, T. The Steffensen iteration method for systems of nonlinear equations. *Proc. Japan Acad.* **1987**, *63*, 186–189. [CrossRef]
7. Grau-Sánchez, M.; Grau, À.; Noguera, M. On the computational efficiency index and some iterative methods for solving systems of nonlinear equations. *J. Comput. Appl. Math.* **2011**, *236*, 1259–1266. [CrossRef]
8. Rostami, M.; Lotfi, T.; Brahmand, A. A fast derivative-free iteration scheme for nonlinear systems and integral equations. *Mathematics* **2019**, *7*, 637. [CrossRef]
9. Traub, J.F. *Iterative Methods for the Solution of Equations*; Prentice Hall: New York, NY, USA, 1964.
10. Cordero, A.; Hueso, J.L.; Martínez, E.; Torregrosa, J.R. A modified Newton–Jarratt's composition. *Numer. Algorithms* **2010**, *55*, 87–99. [CrossRef]
11. Kansal, M.; Cordero, A.; Bhalla, S.; Torregrosa, J.R. New fourth-and sixth-order classes of iterative methods for solving systems of nonlinear equations and their stability analysis. *Numer. Algorithms* **2021**, *87*, 1017–1060. [CrossRef]
12. Amat, S.; Busquier, S. Convergence and numerical analysis of a family of two-step Steffensen's methods. *Comput. Math. Appl.* **2005**, *49*, 13–22. [CrossRef]
13. Soleymani, F.; Sharifi, M.; Shateyi, S.; Khaksar Haghani, F. A class of Steffensen-type iterative methods for nonlinear systems. *J. Appl. Math.* **2014**, *2014*, 705375. [CrossRef]
14. Lotfi, T.; Momenzadeh, M. Constructing an efficient multi-step iterative scheme for nonlinear system of equations. *Comput. Methods Differ. Equ.* **2021**, *9*, 710–721.

15. Ahmad, F.; Soleymani, F.; Khaksar Haghani, F.; Serra-Capizzano, S. Higher order derivative-free iterative methods with and without memory for systems of nonlinear equations. *Appl. Math. Comput.* **2017**, *314*, 199–211. [CrossRef]
16. Shi, L.; Ullah, M.Z.; Nashine, H.K.; Alansari, M.; Shateyi, S. An enhanced numerical iterative method for expanding the attraction basins when computing matrix signs of invertible matrices. *Fractal Fract.* **2023**, *7*, 684. [CrossRef]
17. Wang, X.; Li, W. A class of sixth-order iterative methods for solving nonlinear systems: The convergence and fractals of attractive basins. *Fractal Fract.* **2024**, *8*, 133. [CrossRef]
18. Bertaccini, D.; Durastante, F. Interpolating preconditioners for the solution of sequence of linear systems. *Comput. Math. Appl.* **2016**, *72*, 1118–1130. [CrossRef]
19. Sharma, J.R.; Guha, R.K.; Sharma, R. An efficient fourth order weighted-Newton method for systems of nonlinear equations. *Numer. Algorithms* **2013**, *62*, 307–323. [CrossRef]
20. Sánchez León, J.G. *Mathematica Beyond Mathematics: The Wolfram Language in the Real World;* Taylor & Francis Group: Boca Raton, FL, USA, 2017.
21. Sauer, T. *Numerical Analysi*s, 2nd ed.; Pearson: New York, NY, USA, 2012.
22. Soheili, A.R.; Soleymani, F. Iterative methods for nonlinear systems associated with finite difference approach in stochastic differential equations. *Numer. Algorithms* **2016**, *71*, 89–102. [CrossRef]

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

mathematics

MDPI

Article

Numerical Simulation for the Wave of the VariableCoefficient Nonlinear Schrödinger Equation Based on the Lattice Boltzmann Method

Huimin Wang, Hengjia Chen * and Ting Li

College of Applied Mathematics, Jilin University of Finance and Economics, Changchun 130117, China; whm780921@sina.com or 109011@jlufe.edu.cn (H.W.); li15568099662@163.com (T.L.)
* Correspondence: chennachen51@163.com

Abstract: The variable coefficient nonlinear Schrödinger equation has a wide range of applications in various research fields. This work focuses on the wave propagation based on the variable coefficient nonlinear Schrödinger equation and the variable coefficient fractional order nonlinear Schrödinger equation. Due to the great challenge of accurately solving such problems, this work considers numerical simulation research on this type of problem. We innovatively consider using a mesoscopic numerical method, the lattice Boltzmann method, to study this type of problem, constructing lattice Boltzmann models for these two types of equations, and conducting numerical simulations of wave propagation. Error analysis was conducted on the model, and the convergence of the model was numerical validated. By comparing it with other classic schemes, the effectiveness of the model has been verified. The results indicate that lattice Boltzmann method has demonstrated advantages in both computational accuracy and time consumption. This study has positive significance for the fields of applied mathematics, nonlinear optics, and computational fluid dynamics.

Keywords: lattice Boltzmann method; numerical simulation; variable coefficient nonlinear Schrödinger equation; variable coefficient fractional order nonlinear Schrödinger equation

MSC: 37M05

1. Introduction

The Schrödinger equation is a class of partial differential equations, and it is a very important fundamental equation in the field of quantum mechanics. Every microscopic system has a Schrödinger equation corresponding to it, which can be used to describe the motion of microscopic particles and is widely used in several fields. In recent years, in many fields, such as fluid mechanics, nonlinear optics, and biology, models that can be described using the variable coefficient nonlinear Schrödinger equation have emerged, and the corresponding soliton solutions can provide further scientific explanations for many phenomena. Therefore, the variable coefficient nonlinear Schrödinger equation is widely used in disciplines, such as nonlinear optics and quantum mechanics [1,2]. The study of the variable coefficient nonlinear Schrödinger equation has received much attention. And how to solve the variable coefficient nonlinear Schrödinger equation has become an important branch of nonlinear science and a research focus for scholars due to its significant value.

For most nonlinear partial differential equations, to solute their exact solutions is challenging, and the presence of variable coefficients in partial differential equations further enhances the nonlinearity of the equation, making soluting its solution increasingly difficult. Moreover, due to practical needs, people have introduced the variable coefficient fractional order Schrödinger equation to more accurately describe real-world problems. The addition of fractional derivatives further increases the difficulty of solving variable coefficient fractional partial differential equations. So far, some methods have been used

to analytically study the variable coefficient nonlinear Schrödinger equation, including the bilinear technique and symbolic computations, the generalized Darboux transformation, the general algebraic method, the extended (G'/G)-expansion method, the inverse scattering transformation method [3–7], etc. Due to the difficulty of solving these equations, in addition to further developing research methods for exact solutions of these equations, it is also necessary to develop numerical research methods to numerically solve these equations. However, there are currently not many numerical studies on these equations, such as fourth-order split-step Runge–Kutta, split-step Fourier and Runge–Kutta methods, the Crank-Nicolson (CN) implicit finite-difference method, the deep learning method [8–10], etc., and each method has its advantages and disadvantages. However, for rich nonlinear systems, these studies are still far from sufficient, and we need to develop more numerical methods for research. Therefore, this work innovatively considers developing a new numerical method, namely the lattice Boltzmann method, to numerically simulate the propagation of waves in optical fibers described by the variable coefficient nonlinear Schrödinger equation and the variable coefficient fractional order nonlinear Schrödinger equation. This study has positive implications for many research fields, such as applied mathematics, nonlinear optics, and fluid mechanics.

The lattice Boltzmann method (LBM) is a new modeling and numerical simulation method developed in recent years [11–19], which is a mesoscopic scale method based on the fundamental theory of nonequilibrium statistical physics and molecular dynamics theory, which is not limited to the macroscopic equations, but connects the macroscopic and mesoscopic levels, and the particles transfer the energy through motion and collision. Compared with traditional numerical methods, the lattice Boltzmann method has unique advantages, such as simple programming, stable algorithms, and easy boundary handling. It also shows advantages in computational accuracy and time consumption. In recent years, the lattice Boltzmann method has become a powerful tool in the field of computational fluid dynamics and has made many achievements in the field of nonlinear partial differential equations.

Next, we will construct lattice Boltzmann models for two types of variable coefficient nonlinear Schrödinger equations and variable coefficient fractional order nonlinear Schrödinger equations and numerically simulate the wave propagation described by them.

2. Basic Theory of Lattice Boltzmann Model

In the lattice Boltzmann method, regular lattices are usually used to discretize the space, such as the D1Q3 model, D1Q5 model, D2Q5 model, D2Q7 model, etc., and particles can only move along the lines on the grid. At each time step, particles move to adjacent neighboring grid points or stay at their original grid points. In this work, we choose the D1Q3 model to discretize the one dimensional space; see Figure 1. In the D1Q3 model, the particle velocity is $\mathbf{e}_\alpha = [e_0, e_1, e_2] = [0, c, -c]$, and $\alpha = 0, 1, 2$ represent the three directions of particle motion, respectively, where $\alpha = 0$ represents stationary particles.

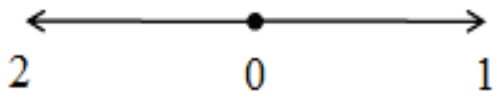


Figure 1. D1Q3 model.

Let $f_\alpha^\sigma(\mathbf{x}, t)$ be the single-particle distribution function with velocity $\mathbf{e}_\alpha$. at position $\mathbf{x}$, time t, and $f_\alpha^{\sigma,eq}(\mathbf{x}, t)$ be the corresponding equilibrium distribution function, where σ represents the component number. Assuming that the distribution function satisfies the conservation condition,

$$\sum_\alpha f_\alpha^\sigma(\mathbf{x}, t) = \sum_\alpha f_\alpha^{\sigma,eq}(\mathbf{x}, t). \tag{1}$$

The evolution of the distribution function satisfies the lattice Boltzmann equation:

$$f_\alpha^\sigma(\mathbf{x}+\mathbf{e}_\alpha, t+1) - f_\alpha^\sigma(\mathbf{x}, t) = -\frac{1}{\tau}[f_\alpha^\sigma(\mathbf{x}, t) - f_\alpha^{\sigma,eq}(\mathbf{x}, t)] + \Omega_\alpha^\sigma(\mathbf{x}, t), \tag{2}$$

where τ is the single relaxation time and Ω_α^σ is an additional term. By applying Taylor expansion, the multiscale expansion technique, and Chapman–Enskog expansion to the lattice Boltzmann equation, a series of partial differential equations on different time scales can be obtained [20]. Please see Appendix A for the detailed derivation.

3. Lattice Boltzmann Model and Numerical Simulation of Variable Coefficient Nonlinear Schrödinger Equation

3.1. Variable Coefficient Nonlinear Schrödinger Equation with Perturbation Term

We consider constructing a lattice Boltzmann model for a class of nonlinear Schrödinger equations with perturbation terms:

$$i\phi_t = \phi_{xx} + \lambda(x,t)\phi + \beta\phi_x^* + \gamma(x,t)|\phi|^2\phi. \tag{3}$$

where $\lambda(x,t)$, β, and $\gamma(x,t)$ represent the loss factor of the optical fiber, disturbance coefficient, and nonlinear coefficient, respectively. Due to the presence of perturbation terms in the equation, for the convenience of solving, we separate the imaginary and real parts of the equation. Let $\phi = u + iv$, then the Equation (3) is rewritten as a coupled equation system in the following form:

$$u_t = v_{xx} + \lambda(x,t)v - \beta v_x + \gamma(x,t)\left(u^2+v^2\right)v. \tag{4}$$

$$v_t = -u_{xx} - \lambda(x,t)u - \beta u_x - \gamma(x,t)\left(u^2+v^2\right)u. \tag{5}$$

Next, we will use the series of equations at different time scales to recover the coupled equation system.

3.1.1. Recovery of Macroscopic Equations

Define the macro quantity u and v as

$$u = \sum_\alpha f_\alpha^1(x,t), \tag{6}$$

$$v = \sum_\alpha f_\alpha^2(x,t). \tag{7}$$

According to the conservation conditions (1), there yields

$$u = \sum_\alpha f_\alpha^{1,(0)}(x,t), \tag{8}$$

$$v = \sum_\alpha f_\alpha^{2,(0)}(x,t), \tag{9}$$

Let the moments of the equilibrium distribution function be

$$m^{1,0} = \sum_\alpha f_\alpha^{1,(0)} e_\alpha = \beta v, \tag{10}$$

$$m^{2,0} = \sum_\alpha f_\alpha^{2,(0)} e_\alpha = \beta u, \tag{11}$$

$$\pi^{1,0} = \sum_\alpha f_\alpha^{1,(0)} e_\alpha^2 = \beta^2 u - \frac{v}{\varepsilon c_2}. \tag{12}$$

$$\pi^{2,0} = \sum_{\alpha} f_{\alpha}^{2,(0)} e_{\alpha}^{2} = \beta^2 v + \frac{u}{\varepsilon c_2}. \tag{13}$$

We assume that $\Omega_{\alpha}^{\sigma} = \varepsilon^2 \Omega_{\alpha}^{\sigma,(2)}$, i.e., $\Omega_{\alpha}^{\sigma,(n)} = 0$, $n \neq 2$, $\sigma = 1, 2$. Summing up the parameter α for $(\mathrm{A7}) + \varepsilon \times (\mathrm{A8})$, which yields

$$u_t = v_{xx} - \beta v_x + \varepsilon \sum_{\alpha} \Omega_{\alpha}^{1,(2)} + O(\varepsilon^2). \tag{14}$$

$$v_t = -u_{xx} - \beta u_x + \varepsilon \sum_{\alpha=1}^{3} \Omega_{\alpha}^{2,(2)} + O(\varepsilon^2). \tag{15}$$

Equations (14) and (15) comprise an approximate formula for the recovered macroscopic Equations (4) and (5).

We choose to make the additional source term meet

$$\sum_{\alpha} \Omega^{1,(2)} = \lambda(x,t) v + \gamma(x,t) \left(u^2 + v^2\right) v, \tag{16}$$

$$\sum_{\alpha} \Omega^{2,(2)} = -\lambda(x,t) u - \gamma(x,t) \left(u^2 + v^2\right) u. \tag{17}$$

$\Omega_{\alpha}^{\sigma,(2)}$ is also assumed to be independent of α, then

$$\Omega^{1,(2)} = \frac{\lambda(x,t) v + \gamma(x,t) \left(u^2 + v^2\right) v}{3\varepsilon}, \tag{18}$$

$$\Omega^{2,(2)} = \frac{-\lambda(x,t) u - \gamma(x,t) \left(u^2 + v^2\right) u}{3\varepsilon}. \tag{19}$$

Combining Equations (8)–(13) and the D1Q3 model, the expressions of equilibrium distribution function can be obtained as

$$f_{\alpha}^{1,(0)} = \begin{cases} \frac{\beta v}{2c} + \frac{\beta^2 u}{2c^2} - \frac{v}{2c^2 \varepsilon c_2}, & \alpha = 1, \\ \frac{\beta^2 u}{2c^2} - \frac{v}{2c^2 \varepsilon c_2} - \frac{\beta v}{2c}, & \alpha = 2, \\ u - \frac{\beta^2 u}{c^2} + \frac{v}{c^2 \varepsilon c_2}, & \alpha = 3. \end{cases} \tag{20}$$

$$f_{\alpha}^{2,(0)} = \begin{cases} \frac{\beta u}{2c} + \frac{u}{2c^2 \varepsilon c_2} + \frac{\beta^2 v}{2c^2}, & \alpha = 1, \\ \frac{u}{2c^2 \varepsilon c_2} + \frac{\beta^2 v}{2c^2} - \frac{\beta u}{2c}, & \alpha = 2, \\ v - \frac{u}{c^2 \varepsilon c_2} - \frac{\beta^2 v}{c^2}, & \alpha = 3. \end{cases} \tag{21}$$

Summing $(\mathrm{A7}) + \varepsilon \times (\mathrm{A8}) + \varepsilon^2 \times (\mathrm{A9})$ over α, which yield

$$u_t = v_{xx} + \lambda(x,t) v - \beta v_x + \gamma(x,t) \left(u^2 + v^2\right) v + E_2^1 + O(\varepsilon^3), \tag{22}$$

$$v_t = -u_{xx} - \lambda(x,t) u - \beta u_x - \gamma(x,t) \left(u^2 + v^2\right) u + E_2^2 + O(\varepsilon^3). \tag{23}$$

where E_2^{σ} is the second-order error term. Through error analysis, the error terms are obtained as

$$E_2^1 = -\varepsilon^2 \left\{ \frac{3\beta C_3}{\varepsilon C_2} \frac{\partial^3 u}{\partial x^3} + C_3 (c^2 \beta - \beta^3) \frac{\partial^3 v}{\partial x^3} - \frac{\beta \tau}{\varepsilon} [\lambda(x,t) + \gamma(x,t)(u^2 + v^2) + 2\gamma(x,t) v^2] \frac{\partial u}{\partial x} - \frac{2\tau\beta\gamma(x,t)}{\varepsilon} uv \frac{\partial v}{\partial x} \right\}, \tag{24}$$

$$E_2^2 = -\varepsilon^2 \left\{ -\frac{3\beta C_3}{\varepsilon C_2} \frac{\partial^3 v}{\partial x^3} + C_3 (c^2 \beta - \beta^3) \frac{\partial^3 u}{\partial x^3} + \frac{\beta \tau}{\varepsilon} [\lambda(x,t) + \gamma(x,t)(u^2 + v^2) + 2\gamma(x,t) u^2] \frac{\partial v}{\partial x} + \frac{2\tau\beta\gamma(x,t)}{\varepsilon} uv \frac{\partial u}{\partial x} \right\}. \tag{25}$$

3.1.2. Numerical Simulation of Wave Propagation

In this part, we will provide numerical examples of the variable coefficient nonlinear Schrödinger Equations (4) and (5).

Case I in this example, $\lambda(x,t) = 1$, $\beta = 2$, and $\gamma(x,t) = -1$, refer to [2], and the exact solution is

$$u(x,t) = \frac{u_0}{1 + \exp[\sqrt{\lambda}x + (\beta\sqrt{\lambda} - \lambda)t + x_0]}, \tag{26}$$

$$v(x,t) = \frac{u_0 \exp[\sqrt{\lambda}x + (\beta\sqrt{\lambda} - \lambda)t + x_0]}{1 + \exp[\sqrt{\lambda}x + (\beta\sqrt{\lambda} - \lambda)t + x_0]}, \tag{27}$$

The initial condition and the boundary condition are given according to the exact solution

$$u(x,0) = \frac{u_0}{1 + \exp[\sqrt{\lambda}x + x_0]}, -10 \le x \le 10, \tag{28}$$

$$v(x,0) = \frac{u_0 \exp[\sqrt{\lambda}x + x_0]}{1 + \exp[\sqrt{\lambda}x + x_0]}, -10 \le x \le 10. \tag{29}$$

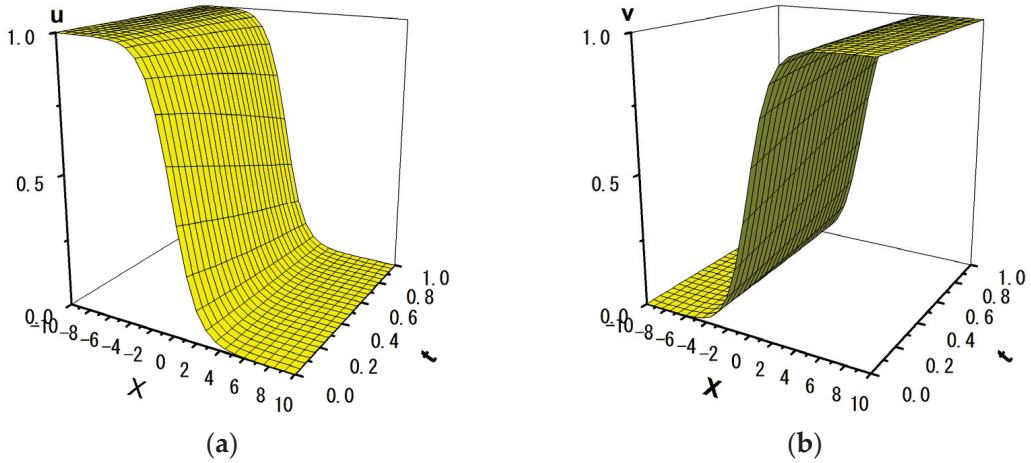
The boundary conditions are

$$u(x_B,t) = \frac{u_0}{1 + \exp[\sqrt{\lambda}x_B + (\beta\sqrt{\lambda} - \lambda)t + x_0]}, t > 0. \tag{30}$$

$$v(x_B,t) = \frac{u_0 \exp[\sqrt{\lambda}x_B + (\beta\sqrt{\lambda} - \lambda)t + x_0]}{1 + \exp[\sqrt{\lambda}x_B + (\beta\sqrt{\lambda} - \lambda)t + x_0]}, t > 0. \tag{31}$$

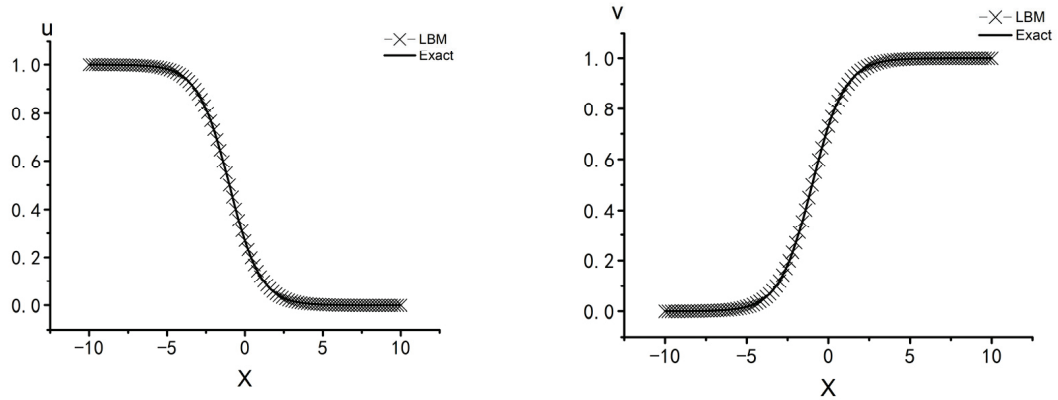
where x_B represents the boundary point, $u_0^2 = -\frac{\lambda}{\gamma}$, $x_0 = 0$. The calculation interval is $[-10, 10]$. The computational parameters are the number of lattices $M = 101$, $\Delta t = 0.001$, $\Delta x = 0.02$, and $\tau = 1.021$. We use the software Fortran Powerstation 4.0 to write code for numerical operations, and the numerical results are shown in Figures 2–6. Figure 2 shows the wave propagation simulated by the lattice Boltzmann method. Figure 3 shows the comparison between the lattice Boltzmann solution and the exact solution. Figure 4 shows the error curves, $Er = \left|u^N - u^E\right|$ and $\left|v^N - v^E\right|$, where u^N and v^N represent the lattice Boltzmann solution, and u^E and v^E represent the exact solution. The results show that the lattice Boltzmann solution agrees with the exact solution. We use the infinite norm of error of u, $\|Er\|_\infty = \max\{Er\} = \max\{\left|u^N - u^E\right|\}$, to evaluate the performance of the constructed lattice Boltzmann model. Figure 5 shows the relationship between the infinite norm of error $\|Er\|_\infty$ and the Knudsen coefficient ε. Through linear fitting, the fitted line is obtained, $\log_{10}(\left|\left|Er\right|\right|_\infty) = 1.5174 \times \log_{10}\varepsilon + 0.58836$. The slope of the straight line represents the order of convergence of the model, and the results show that the constructed model is convergent. In the lattice Boltzmann model, ε is equal to the time step Δt and the spatial step $\Delta x = c\Delta t = c\varepsilon$. For a fixed parameter c, the order of convergence of the model in both time and space is 1.5174. Figure 4 also shows the relationship between the truncation error and the Knudsen number.

To verify the effectiveness of our lattice Boltzmann model, we compared our model with several classical schemes. The comparison results are shown in Figure 6 and Table 1. Figure 6a shows the comparison of solitary waves simulated by these different schemes, and Figure 6b shows the error comparison of these schemes. It can be seen from the results that our lattice Boltzmann model error is lower than that of other schemes. We also compared the errors and time consumption of different schemes, and the results are listed in Table 1. From the data in Table 1, it can be seen that compared with other classic schemes, our lattice Boltzmann method exhibits advantages in both accuracy and time consumption.



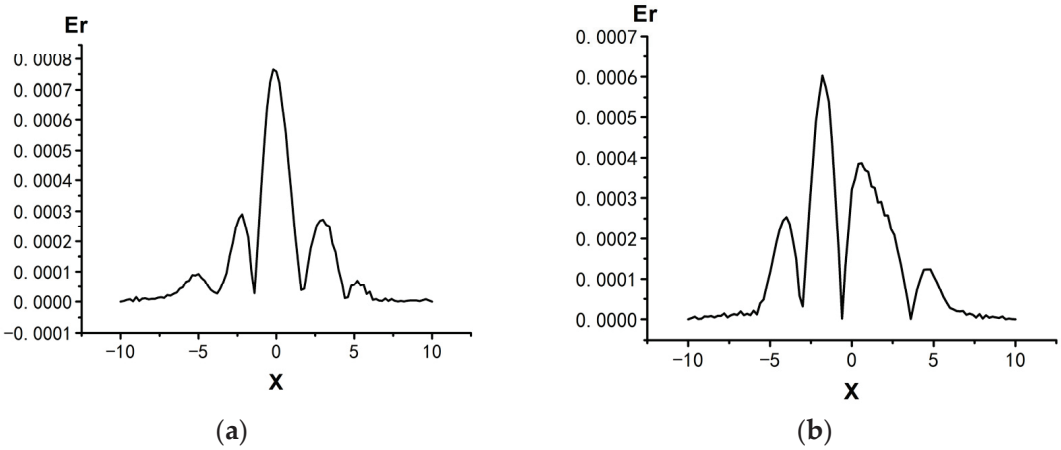
(**a**) (**b**)

Figure 2. Wave propagation simulated by the lattice Boltzmann method. (**a**) Wave of u; (**b**) wave of v.



(**a**) (**b**)

Figure 3. Comparison of the lattice Boltzmann solution and exact solution, $t = 1.0$. (**a**) Wave solution of u; (**b**) wave solution of v.



(**a**) (**b**)

Figure 4. Error curve, $t = 1.0$. (**a**) Error curve of u; (**b**) error curve of v.

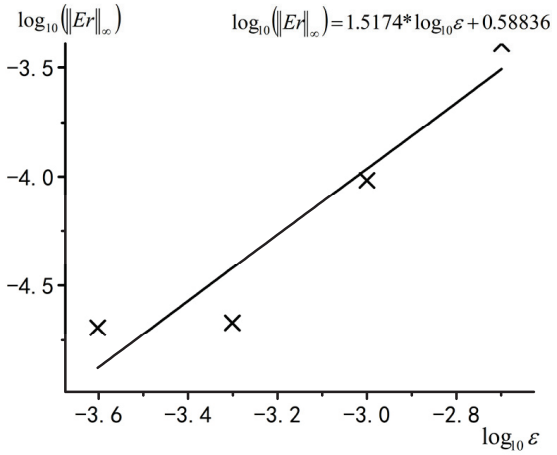
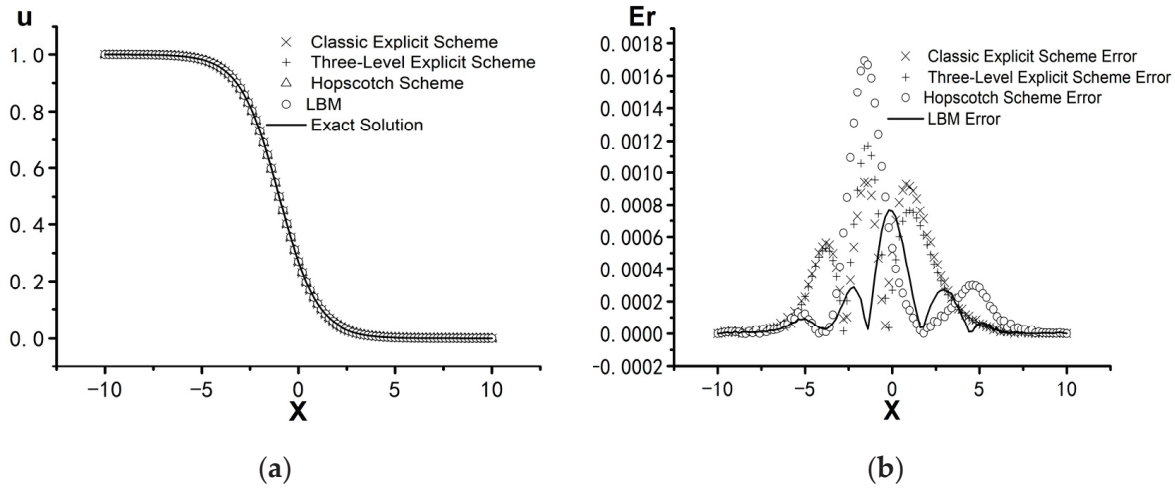


Figure 5. Infinite method of error $\|Er\|_\infty$ versus Knudsen number curve.



(**a**) (**b**)

Figure 6. Comparison of different schemes, $t = 1.0$. (**a**) Wave solution of u; (**b**) error curve of u.

Table 1. Comparison table of different schemes.

Scheme		Time t = 0.2	Time t = 0.4	Time t = 0.6	Time t = 0.8	Time t = 1.0
Classic Explicit Scheme	$\|Er\|_\infty$	2.236962×10^{-4}	3.646016×10^{-4}	6.057620×10^{-4}	8.040071×10^{-4}	9.397864×10^{-4}
	CPU Time (s)	0.094	0.213	0.259	0.263	0.310
Three-Level Explicit Scheme	$\|Er\|_\infty$	3.236234×10^{-4}	6.066561×10^{-4}	8.515120×10^{-4}	1.037896×10^{-3}	1.171649×10^{-3}
	CPU Time (s)	0.133	0.232	0.268	0.284	0.311
Hopscotch Scheme	$\|Er\|_\infty$	3.938675×10^{-4}	7.615089×10^{-4}	1.185596×10^{-3}	1.506567×10^{-3}	1.693845×10^{-3}
	CPU Time (s)	0.132	0.241	0.297	0.325	0.388
LBM	$\|Er\|_\infty$	1.530647×10^{-4}	2.858341×10^{-4}	5.033910×10^{-4}	6.718934×10^{-4}	7.652342×10^{-4}
	CPU Time (s)	0.078	0.140	0.171	0.188	0.203

Case II In this example, $\lambda(x,t) = 1$, $\beta = 1$, $\gamma(x,t) = -1$. The initial conditions are

$$u(x,0) = \frac{u_0 \exp[\sqrt{\lambda}x + x_0]}{1 + \exp[\sqrt{\lambda}x + x_0]}, -10 \le x \le 10, \tag{32}$$

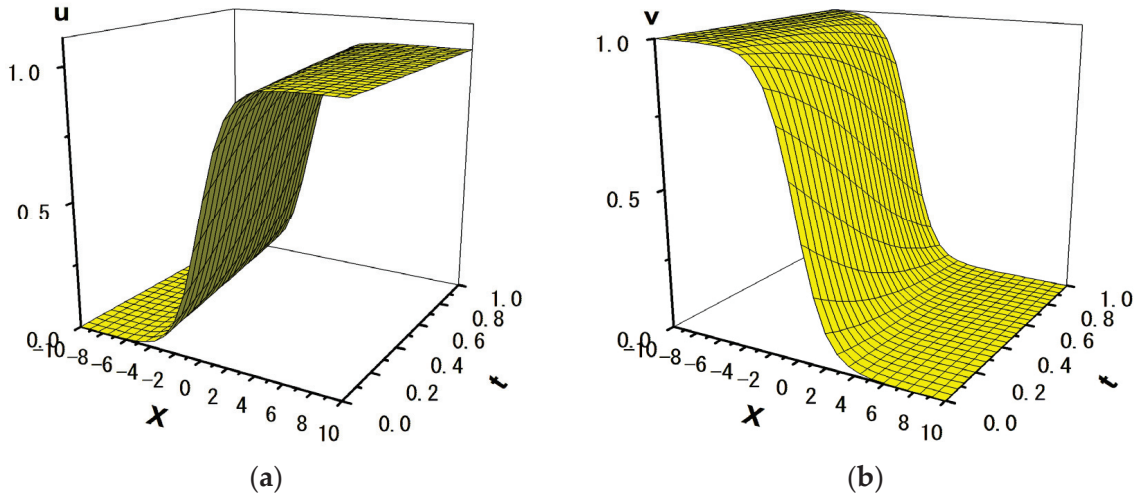
$$v(x,0) = \frac{u_0}{1 + \exp[\sqrt{\lambda}x + x_0]}, -10 \le x \le 10. \tag{33}$$

The boundary conditions are

$$u(x_B,t)=\frac{u_0\exp[\sqrt{\lambda}x_B+(\beta\sqrt{\lambda}+\lambda)t+x_0]}{1+\exp[\sqrt{\lambda}x_B+(\beta\sqrt{\lambda}+\lambda)t+x_0]},t>0. \tag{34}$$

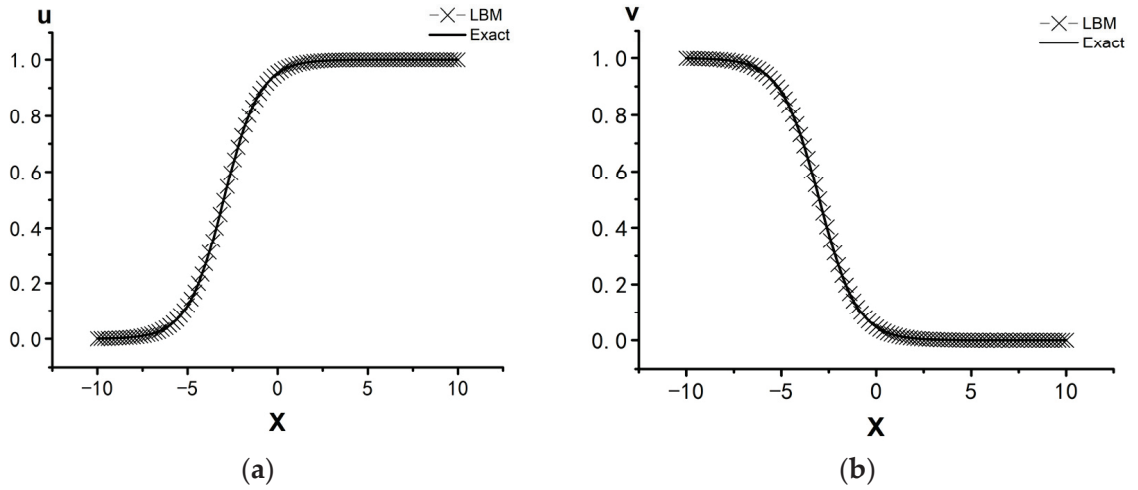
$$v(x_B,t)=\frac{u_0}{1+\exp[\sqrt{\lambda}x_B+(\beta\sqrt{\lambda}+\lambda)t+x_0]},t>0. \tag{35}$$

where x_B represents the boundary point, $u_0^2=-\frac{\lambda}{\gamma}$, $x_0=0$. The calculation interval is $[-10,10]$. The computational parameters are the number of lattices $M=101$, $\Delta t=0.001$, $\Delta x=0.02$, $\tau=0.944$, and the numerical results are shown in Figures 7–9.



(**a**) (**b**)

Figure 7. Wave propagation simulated by the lattice Boltzmann method. (**a**) Wave of u; (**b**) wave of v.



(**a**) (**b**)

Figure 8. Comparison between the lattice Boltzmann solution and exact solution, $t=1.0$. (**a**) Wave solution of u; (**b**) wave solution of v.

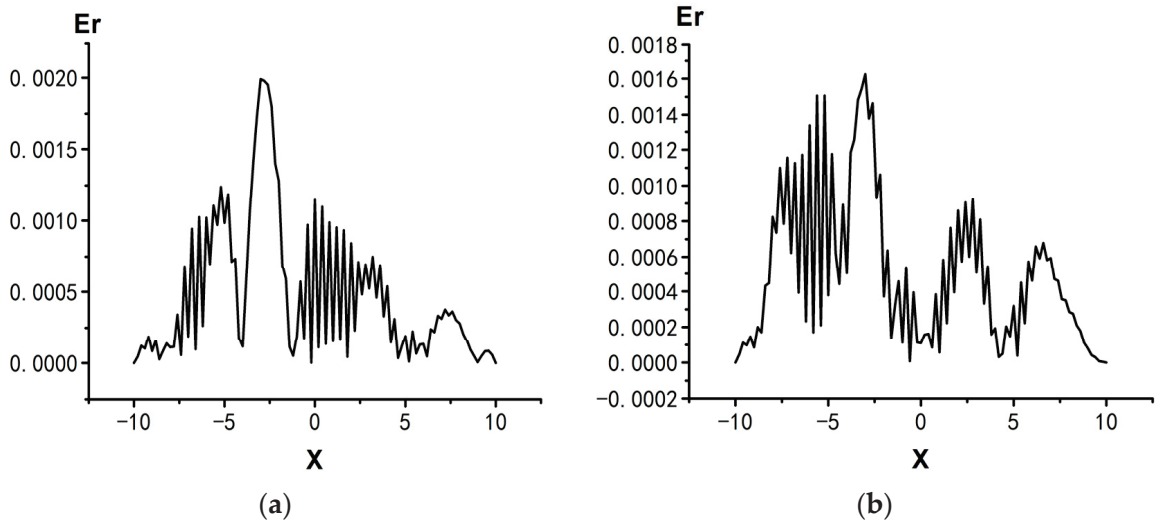


Figure 9. Error curve, $t = 1.0$. (**a**) Error curve of u; (**b**) error curve of v.

Case III In this example, $\lambda(x,t) = 1$, $\beta = 2$, $\gamma(x,t) = 1$. The initial conditions are

$$u(x,0) = u_1 \mathrm{sech}(a_1\sqrt{-\lambda}x + x_0), -10 \le x \le 10, \tag{36}$$

$$v(x,0) = a_2 u_1 \mathrm{sech}(a_1\sqrt{-\lambda}(x + x_0), -10 \le x \le 10, \tag{37}$$

The boundary conditions are

$$u(x_B,t) = u_1 \mathrm{sech}(a_1\sqrt{-\lambda}x_B - a_1 a_2 \beta\sqrt{-\lambda}t + x_0), t > 0. \tag{38}$$

$$v(x_B,t) = a_2 u_1 \mathrm{sech}(a_1\sqrt{-\lambda}(x_B - \beta a_2 t + x_0)), t > 0. \tag{39}$$

where x_B represents the boundary point, $a_1 = 1$, $a_2 = 1$, $u_1 = 1$. The calculation interval is $[-10, 10]$. The computational parameters are the number of lattices $M = 101$, $\Delta t = 0.001$, $\Delta x = 0.02$, $\tau = 0.96$, and the numerical results are shown in Figures 10–12.

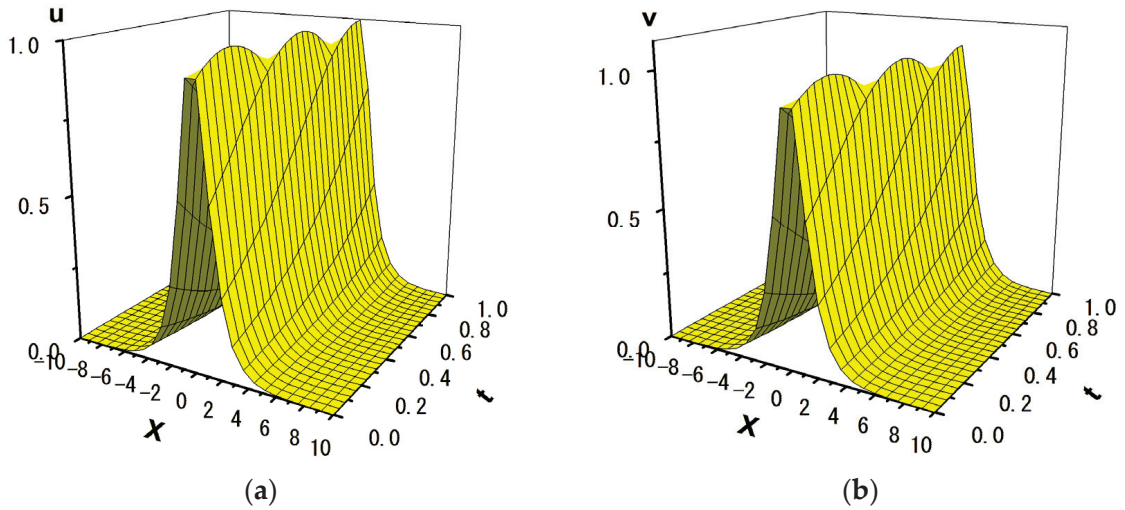


Figure 10. Wave propagation simulated by the lattice Boltzmann method. (**a**) Wave of u; (**b**) wave of v.

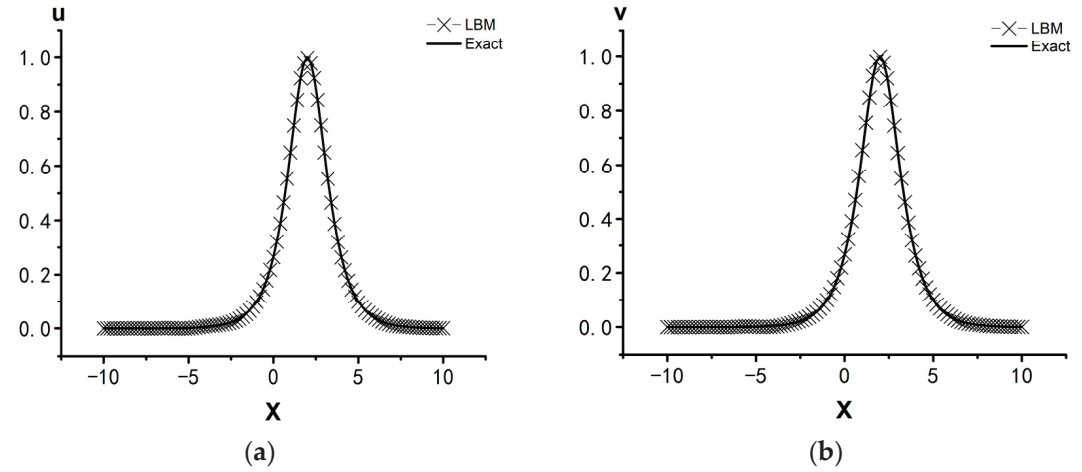


Figure 11. Comparison of the lattice Boltzmann solution and exact solution, $t = 1.0$. (**a**) Wave solution of u; (**b**) wave solution of v.

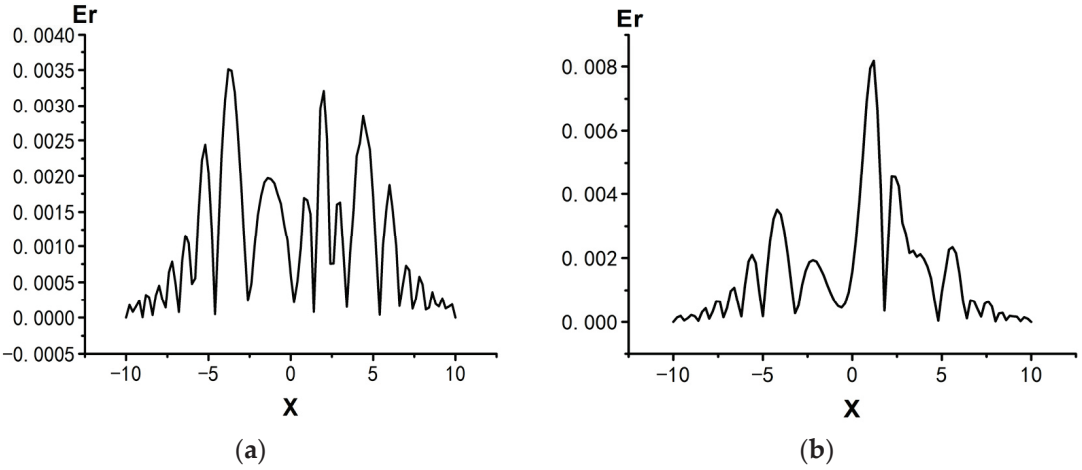


Figure 12. Error curve, $t = 1.0$. (**a**) Error curve of u; (**b**) error curve of v.

3.2. Variable Coefficient Fractional Order Nonlinear Schrödinger Equation

In this part, we consider constructing a lattice Boltzmann model for a class of variable coefficient fractional order nonlinear Schrödinger equation:

$$iu_t + \lambda(t)\frac{\partial^\beta u}{\partial|x|^\beta} + v(x)u + \gamma(t)\,|u|^2 u = 0,\ \mathrm{a} \le \mathrm{x} \le b, 0 \le t \le T. \tag{40}$$

where $u(x,t)$ is a complex valued wave function; $\lambda(t)$, $v(x)$, and $\gamma(t)$ are bounded real functions; $1 < \beta \le 2$, $\frac{\partial^\beta u}{\partial|x|^\beta}$ is the Riesz fractional derivative of order β, defined through Riemann Liouville integration as

$$\frac{\partial^\beta u}{\partial|x|^\beta} = -\theta(I_+^{-\beta} + I_-^{-\beta})u. \tag{41}$$

where $\theta = \frac{1}{2\cos(\beta\pi/2)}$, $I_\pm^{-\beta} = \frac{\partial^2}{\partial x^2} I_\pm^{2-\beta} u(x,t)$. According to Riemann Liouville's integral definition, it can be inferred that

$$\left(I_+^{2-\beta}u\right)(x,t) = \frac{1}{\Gamma(2-\beta)}\int_a^x (x-\xi)^{1-\beta} u(\xi,t)d\xi,\ x > a, \tag{42}$$

$$\left(I_{-}^{2-\beta}u\right)(x,t)=\frac{1}{\Gamma(2-\beta)}\int_{x}^{b}(\xi-x)^{\beta-1}u(\xi,t)d\xi,\ x<b. \tag{43}$$

Therefore, Equation (40) can be expressed in the following form:

$$iu_t-\lambda(t)\theta\frac{\partial^2}{\partial x^2}(I_{+}^{2-\beta}+I_{-}^{2-\beta})u+v(x)u+\gamma(t)\mid u\mid^2 u=0,\ \mathrm{a}\le \mathrm{x}\le b, 0\le t\le T. \tag{44}$$

Next, We will use the series of equations at different time scales to recover Equation (44).

3.2.1. Recovery of Macroscopic Equation

We define the macroscopic quantity u as

$$iu=\sum_{\alpha}f_{\alpha}(\mathbf{x},t). \tag{45}$$

Here, component number $\sigma=1$ is omitted and not written. According to conservation condition $\sum\limits_{\alpha}f(\mathbf{x},t)=\sum\limits_{\alpha}f_{\alpha}^{eq}(\mathbf{x},t)$, it can be concluded that

$$iu=\sum_{\alpha}f_{\alpha}^{(0)}(\mathbf{x},t), \tag{46}$$

let the moments of the equilibrium distribution function be

$$m^0=\sum_{\alpha}f_{\alpha}^{(0)}e_{\alpha}=0, \tag{47}$$

$$\pi^0=\sum_{\alpha}f_{\alpha}^{(0)}e_{\alpha}^2=-\frac{\lambda(t)\theta}{\varepsilon C_2}I_{\pm}^{2-\beta}u. \tag{48}$$

where the Riemann–Liouville integral $I_{\pm}^{2-\beta}u$ can be approximately calculated based on the Grünwald–Letnikov fractional derivative definitions on the left and right sides,

$$I_{+}^{2-\beta}u(x,t)\approx h^{2-\beta}\sum_{r=0}^{[\frac{x-a}{h}]}\begin{bmatrix}2-\beta\\ r\end{bmatrix}u(x-rh,t),\ a<x<b, \tag{49}$$

$$I_{-}^{2-\beta}u(x,t)\approx h^{2-\beta}\sum_{r=0}^{[\frac{b-x}{h}]}\begin{bmatrix}2-\beta\\ r\end{bmatrix}u(x+rh,t),\ a<x<b. \tag{50}$$

where $\begin{bmatrix}2-\beta\\ r\end{bmatrix}=\frac{(2-\beta)(2-\beta+1)\cdots(2-\beta+r-1)}{r!}$, for $r>0$, and $\begin{bmatrix}2-\beta\\ r\end{bmatrix}=1$, for $r=0$.

Assuming that $\Omega_{\alpha}=\varepsilon^2\Omega_{\alpha}^{(2)}$, i.e., $\Omega_{\alpha}^{(n)}=0$, $n\neq 2$. Then, from $\sum\limits_{\alpha}[(\mathrm{A7})+\varepsilon\times(\mathrm{A8})]$, it can be concluded that

$$iu_t-\lambda(t)\theta\frac{\partial^2}{\partial x^2}(I_{+}^{2-\beta}+I_{-}^{2-\beta})u=\varepsilon\sum_{\alpha}\Omega_{\alpha}^{(2)}+O(\varepsilon^2). \tag{51}$$

Equation (51) is an approximate formula for the recovered macroscopic Equation (44). We set

$$\sum_{\alpha}\Omega_{\alpha}^{(2)}=-V(x)u-\gamma(t)|u|^2u, \tag{52}$$

If it is assumed that $\Omega_{\alpha}^{(2)}$ is independent of α, combining the D1Q3 model we can obtain

$$\Omega_{\alpha}^{(2)} = \Omega^{(2)} = \frac{-V(x)u - \gamma(t)|u|^2 u}{3\varepsilon}. \tag{53}$$

Solving Equations (46)–(48), the equilibrium distribution function is obtained

$$f_{\alpha}^{(0)} = \begin{cases} \frac{-\lambda(t)\theta}{2c^2\varepsilon C_2}(I_{+}^{2-\beta} + I_{-}^{2-\beta})u, & \alpha = 1, 2\,, \\ iu - 2f_1^{(0)}, & \alpha = 0. \end{cases} \tag{54}$$

Summing $(A7) + \varepsilon \times (A8) + \varepsilon^2 \times (A9)$ over α yields

$$iu_t - \lambda(t)\theta \frac{\partial^2}{\partial x^2}(I_{+}^{2-\beta} + I_{-}^{2-\beta})u = \varepsilon \sum_{\alpha} \Omega_{\alpha}^{(2)} + E_2 + O(\varepsilon^3). \tag{55}$$

E_2 is the second-order error term. Through error analysis, it can be obtained that

$$\begin{aligned} E_2 &= -\varepsilon^2 \left(C_3 \sum_{\alpha} \Delta^3 f_{\alpha}^{(0)} + 2C_2 \sum_{\alpha} \Delta \frac{\partial}{\partial t_1} f_{\alpha}^{(0)} + \tau \sum_{\alpha} \Delta \Omega_{\alpha}^{(2)} \right) \\ &= -\frac{3\varepsilon C_3 \gamma}{C_2} \frac{\partial^3}{\partial t_0 \partial x^2} \left[I_{+}^{2-\beta} u(x,t) + I_{-}^{2-\beta} u(x,t) \right] \end{aligned}. \tag{56}$$

Thus, the macroscopic Equation (44) is recovered as

$$iu_t - \lambda(t)\theta \frac{\partial^2}{\partial x^2}(I_{+}^{2-\beta} + I_{-}^{2-\beta})u + v(x)u + \gamma(t)|u|^2 u = O(\varepsilon). \tag{57}$$

3.2.2. Numerical Example

We will numerically simulate the wave propagation of the equation in this section. A numerical example is given, $\lambda(t) = t/30$, $v(x) = \sin x$, $\gamma(t) = t/8$.

The initial conditions and the boundary conditions in this example are [21] as follows:

$$u(x, 0) = \operatorname{sech}(x) \cdot \exp(2ix), -20 \le x \le 20, \tag{58}$$

$$u(-20, t) = u(20, t) = 0.\ 0 \le t \le 1. \tag{59}$$

The computational parameters are the number of lattices, $M = 101$, $\Delta t = 0.001$, $\Delta x = 0.02$, $c = \Delta x/\Delta t$, $\tau = 1.0755$. The propagation of the solitary wave solution using the lattice Boltzmann method for $\alpha = 1.2,\ 1.4,\ 1.6,\ 1.8$ from $t = 0$ to $t = 0.5$ is shown in Figures 13–16.

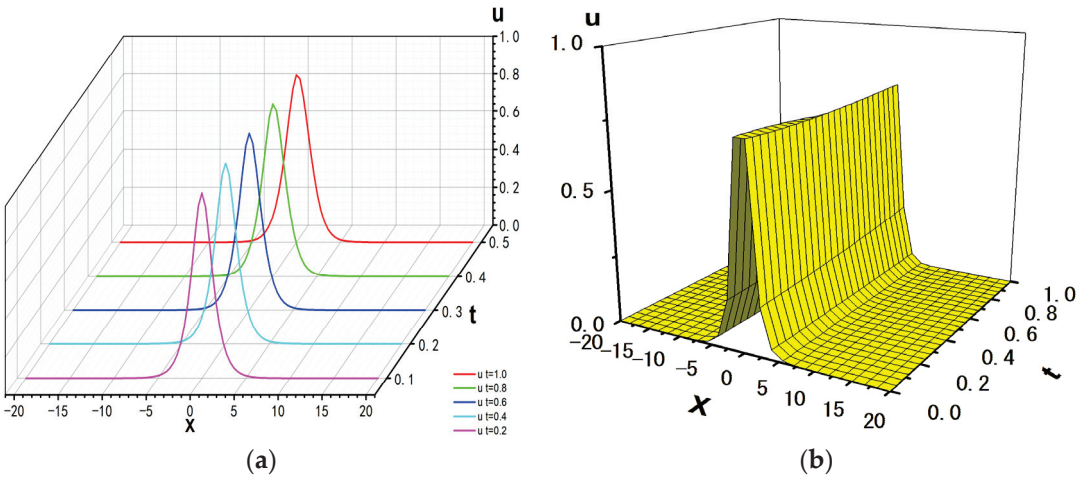
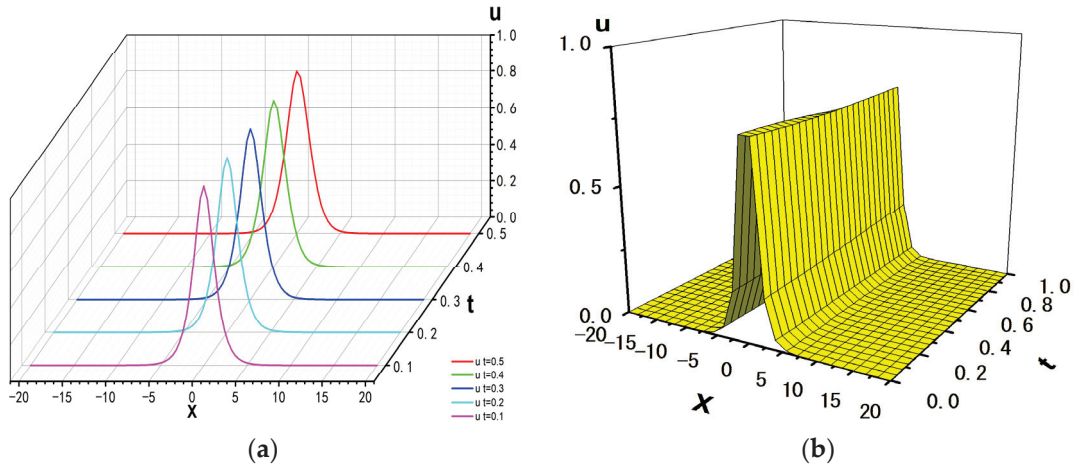
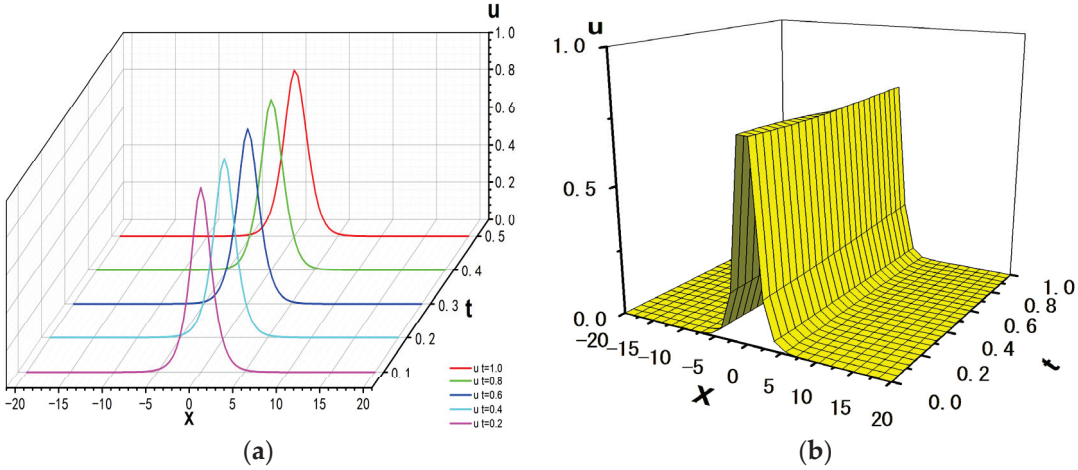


Figure 13. Solitary wave propagation using the lattice Boltzmann method, $\alpha = 1.2$. (**a**) Waterfall plot; (**b**) solitary wave propagation.



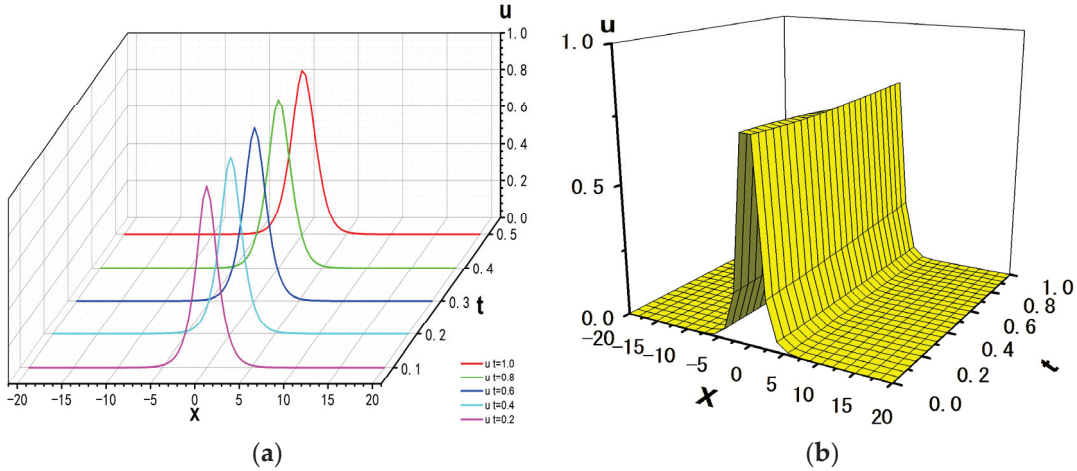
(**a**) (**b**)

Figure 14. Solitary wave propagation using the lattice Boltzmann method, $\alpha = 1.4$. (**a**) Waterfall plot; (**b**) solitary wave propagation.



(**a**) (**b**)

Figure 15. Solitary wave propagation using the lattice Boltzmann method, $\alpha = 1.6$. (**a**) Waterfall plot; (**b**) solitary wave propagation.



(**a**) (**b**)

Figure 16. Solitary wave propagation using the lattice Boltzmann method, $\alpha = 1.8$. (**a**) Waterfall plot; (**b**) solitary wave propagation.

4. Conclusions

In this paper, we use the lattice Boltzmann method to numerically simulate wave propagation based on the variable coefficient nonlinear Schrödinger equation and the variable coefficient fractional order Schrödinger equation.

Lattice Boltzmann models are constructed for the two types of equations, a series of partial differential equations on different time scales are obtained by using the Taylor expansion, the Chapman–Enskog expansion, and the time multiscale expansion based on the basic Lattice Boltzmann equation. The macroscopic equations are recovered by choosing appropriate expressions for the moments of the equilibrium distribution function.

The solutions of the equations are numerically simulated together with numerical examples. By comparing the lattice Boltzmann solution with the exact solution and combining it with the error analysis, it is found that the lattice Boltzmann solution agrees with the exact solution. Furthermore, the effectiveness of our method was verified by comparing the lattice Boltzmann model with other classical schemes. The comparison results indicate that our method has shown advantages in both computational accuracy and time consumption. The convergence of the model has also been numerically verified.

The research results indicate that the lattice Boltzmann method is effective in studying wave propagation based on the variable coefficient nonlinear Schrödinger equation and the variable coefficient fractional order Schrödinger equation. These two types of equations are of great research value in various fields, so this work investigates the strong research significance of the lattice Boltzmann method for solving the solitary wave solutions of these two types of equations. In our future work, we will further research a high-precision and high-efficiency numerical method.

Author Contributions: Conceptualization, H.W.; methodology, H.W.; software, H.W., H.C., and T.L.; validation, H.W. and H.C.; writing—original draft preparation, H.W., H.C., and T.L.; project administration, H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by "the Jilin Provincial Natural Science Foundation of China, grant number YDZJ202201ZYTS535" and "the Project of Education Department of Jilin Province of China, grant number JJKH20220151KJ".

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

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

Appendix A

Let us define the Knudsen number ε as the ratio between the mean free path l and the characteristic length L. Taking ε to be equal to the time step Δt, the lattice Boltzmann Equation (2) in physical units is expressed as Equation (A1).

$$f_\alpha^\sigma(\mathbf{x}+\varepsilon\mathbf{e}_{\boldsymbol{\alpha}}, t+\varepsilon) - f_\alpha^\sigma(\mathbf{x}, t) = -\frac{1}{\tau}[f_\alpha^\sigma(\mathbf{x}, t) - f_\alpha^{\sigma,eq}(\mathbf{x}, t)] + \Omega_\alpha^\sigma(\mathbf{x}, t). \tag{A1}$$

In Equation (A1), it is assumed that the Knudsen number ε is small. Performing a Taylor expansion on the left-hand side of Equation (A1), keeping terms up to $O(\varepsilon^4)$, yields

$$f_\alpha^\sigma(\mathbf{x}+\varepsilon e_\alpha, t+\varepsilon) - f_\alpha^\sigma(\mathbf{x}, t) = \sum_{n=1}^{3} \frac{\varepsilon^n}{n!}\left(\frac{\partial}{\partial t} + e_\alpha \frac{\partial}{\partial \mathbf{x}}\right)^n f_\alpha^\sigma(\mathbf{x}, t) + O\left(\varepsilon^4\right). \tag{A2}$$

Under the assumption of a small Knudsen number, the Chapman–Enskog expansion is performed on f_α^σ,

$$f_\alpha^\sigma = f_\alpha^{\sigma,(0)} + \sum_{n=1}^{\infty} \varepsilon^n f_\alpha^{\sigma,(n)}, \tag{A3}$$

where $f_\alpha^{\sigma,(0)} \equiv f_\alpha^{\sigma,eq}$. Introducing t_0, t_1, t_2, t_3 as different scale times, defined as

$$t_i = \varepsilon^i t, i = 0, 1, 2, 3. \tag{A4}$$

and

$$\frac{\partial}{\partial t} = \sum_{n=0}^{3} \varepsilon^n \frac{\partial}{\partial t_n} + O(\varepsilon^4). \tag{A5}$$

The Chapman–Enskog expansion is also performed on Ω_α^σ,

$$\Omega_\alpha^\sigma = \sum_{n=1}^{\infty} \varepsilon^n \Omega_\alpha^{\sigma,(n)}. \tag{A6}$$

From Equations (A3) to (A6), the equations for each order of ε are given as follows:

$$C_1 \Delta f_\alpha^{\sigma,(0)} = -\frac{1}{\tau} f_\alpha^{\sigma,(1)} + \Omega_\alpha^{\sigma,(1)}, \tag{A7}$$

$$\frac{\partial}{\partial t_1} f_\alpha^{\sigma,(0)} + C_2 \Delta^2 f_\alpha^{\sigma,(0)} + \Delta\tau\Omega_\alpha^{\sigma,(1)} = -\frac{1}{\tau} f_\alpha^{\sigma,(2)} + \Omega_\alpha^{\sigma,(2)}, \tag{A8}$$

$$\begin{aligned} &C_3 \Delta^3 f_\alpha^{\sigma,(0)} + 2C_2 \Delta \tfrac{\partial}{\partial t_1} f_\alpha^{\sigma,(0)} + \tfrac{\partial}{\partial t_2} f_\alpha^{\sigma,(0)} + \tau \tfrac{\partial}{\partial t_1} \Omega_\alpha^{\sigma,(1)} + C_2 \tau \Delta^2 \Omega_\alpha^{\sigma,(1)} \\ &+\tau\Delta\Omega_\alpha^{\sigma,(2)} = -\tfrac{1}{\tau} f_\alpha^{\sigma,(3)} + \Omega_\alpha^{\sigma,(3)}. \end{aligned} \tag{A9}$$

where the partial differential operator $\Delta \equiv \frac{\partial}{\partial t_0} + \mathbf{e}_\alpha \frac{\partial}{\partial \mathbf{x}}$, $f_\alpha^{\sigma,(0)} = f_\alpha^{\sigma,eq}$.

Equations (A7)–(A9) represent a series of partial differential equations across various time scales. In these equations, C_i is the polynomial of the relaxation time factor τ.

$$C_1 = 1, \tag{A10}$$

$$C_2 = \frac{1}{2} - \tau, \tag{A11}$$

$$C_3 = \tau^2 - \tau + \frac{1}{6}, \tag{A12}$$

Based on Equations (1) and (A3), it follows that

$$\sum_\alpha f_\alpha^{\sigma,(n)}(\mathbf{x},t) = 0, \text{ for } n \geq 1. \tag{A13}$$

Equation (A13) indicates that the moment at zero vanishes for each order $n \geq 1$ of ε.

The equilibrium distribution function is characterized by certain moments, which are denoted in the following manner:

$$\sum_\alpha f_\alpha^{\sigma,(0)}(\mathbf{x},t) e_\alpha \equiv m^{\sigma,0}(\mathbf{x},t), \tag{A14}$$

$$\sum_\alpha f_\alpha^{\sigma,(0)}(\mathbf{x},t) e_\alpha^2 \equiv \pi^{\sigma,0}(\mathbf{x},t), \tag{A15}$$

$$\sum_\alpha f_\alpha^{\sigma,(0)}(\mathbf{x},t) e_\alpha^3 \equiv P^{\sigma,0}(\mathbf{x},t). \tag{A16}$$

References

1. Huang, F.T. *Algorithm and Characterization of Optical Soliton Solutions for NLS Equations in Fiber Optic Communications*; University of Science and Technology Beijing: Beijing, China, 2023.
2. Chen, X.W. *A Study of the Soliton Solution of the Schrödinger Equation with Variable Coefficients*; Nanjing University of Information Engineering: Nanjing, China, 2018.

3. Wen, S.T.; Manafian, J.; Sedighi, S.; Atmaca, S.P.; Gallegos, C.; Mahmoud, K.H.; Alsubaie, A.S.A. Interactions among lump optical solitons for coupled nonlinear Schrödinger equation with variable coefficient via bilinear method. *Sci. Rep.* **2024**, *14*, 19568. [CrossRef] [PubMed]
4. Song, N.; Liu, R.; Guo, M.M.; Ma, W.X. Nth order generalized Darboux transformation and solitons, breathers and rogue waves in a variable-coefficient coupled nonlinear Schrödinger equation. *Nonlinear Dynam.* **2023**, *111*, 19347–19357. [CrossRef]
5. Gu, Y.; Chen, B.; Ye, F.; Aminakbari, N. Soliton solutions of nonlinear Schrödinger equation with the variable coefficients under the influence of Woods–Saxon potential. *Results Phys.* **2022**, *42*, 105979. [CrossRef]
6. Hong, B. Exact solutions for the conformable fractional coupled nonlinear Schrödinger equations with variable coefficients. *J. Low Freq. Noise Vib. Act. Control* **2023**, *42*, 628–641. [CrossRef]
7. Yu, F.; Li, L. Soliton robustness, interaction and stability for a variable coefficients Schrödinger(VCNLS) equation with inverse scattering transformation. *Chaos Soliton. Fract.* **2024**, *185*, 115185. [CrossRef]
8. Yin, H.M.; Tian, B.; Chai, J.; Liu, L.; Sun, Y. Numerical solutions of a variable-coefficient nonlinear Schrödinger equation for an inhomogeneous optical fiber. *Comput. Math. Appl.* **2018**, *76*, 1827–1836. [CrossRef]
9. Tay, K.G.; Choy, Y.Y.; Tionng, W.K.; Ong, C.T. Numerical solutions of the dissipative nonlinear Schrödinger equation with variable coefficient arises in elastic tube. *Dyn. Contin. Discret. Impuls. Syst. Ser. B. Appl. Algorithms* **2018**, *25*, 53–61.
10. Sun, J.; Dong, H.; Liu, M.; Fang, Y. Data-driven rogue waves solutions for the focusing and variable coefficient nonlinear Schrödinger equations via deep learning. *Chaos* **2024**, *34*, 073134. [CrossRef] [PubMed]
11. Qian, Y.H.; d'Humieres, D.; Lallemand, P. Lattice BGK Models for Navier- Stokes Equations. *Europhys. Lett.* **1992**, *17*, 479–484. [CrossRef]
12. Chen, S.Y.; Doolen, G.D. Lattice Boltzmann method for fluid flows. *Annu. Rev. Fluid Mech.* **1998**, *30*, 329–364. [CrossRef]
13. Succi, S.; Benzi, R. Lattice Boltzmann equation for quantum mechanics. *Phys. D Nonlinear Phenom.* **1993**, *69*, 327–332. [CrossRef]
14. He, Y.B.; Lin, X.Y. Numerical analysis and simulations for coupled nonlinear Schrödinger equations based on lattice Boltzmann method. *Appl. Math. Lett.* **2020**, *106*, 106391. [CrossRef]
15. Shi, B.C.; Guo, Z.L. Lattice Boltzmann model for nonlinear convection-diffusion equations. *Phys. Rev. E* **2009**, *79*, 016701.
16. Nie, X.B.; Doolen, G.D.; Chen, S.Y. Lattice-Boltzmann Simulations of Fluid Flows in MEMS. *J. Stat. Phys.* **2002**, *107*, 279–289. [CrossRef]
17. Lallemand, P.; Luo, L.S. Lattice Boltzmann equation with Overset method for moving objects in two-dimensional flows. *J. Comput. Phys.* **2020**, *407*, 109223. [CrossRef]
18. Dubois, F.; Lallemand, P.; Tekitek, M.M. On anti bounce back boundary condition for lattice Boltzmann schemes. *Comput. Math. Appl.* **2019**, *79*, 555–575. [CrossRef]
19. Boghosian, B.M.; Dubois, F.; Lallemand, P. Numerical approximations of a lattice Boltzmann scheme with a family of partial differential equations. *Comput. Fluids* **2024**, *284*, 106410. [CrossRef]
20. Wang, H.M. A lattice Boltzmann model for the ion- and electron-acoustic solitary waves in beam-plasma system. *Appl. Math. Comput.* **2016**, *279*, 62–75. [CrossRef]
21. Shen, H. Finite difference study of Schrödinger equation with variable coefficients of fractional order. *J. Ningxia Norm. Coll.* **2023**, *44*, 27–34.

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

mathematics

MDPI

Article

Efficient Scalar Multiplication of ECC Using Lookup Table and Fast Repeating Point Doubling

Fu-Jung Kan [1], Yan-Haw Chen [2], Jeng-Jung Wang [2,*] and Chong-Dao Lee [2]

[1] Department of Electronic Engineering, I-Shou University, Kaohsiung 84001, Taiwan; isu11102001d@isu.edu.tw
[2] Department of Information Engineering, I-Shou University, Kaohsiung 84001, Taiwan; yanchen@isu.edu.tw (Y.-H.C.); chongdao@isu.edu.tw (C.-D.L.)
* Correspondence: jjwang@isu.edu.tw

Abstract: Reducing the computation time of scalar multiplication for elliptic curve cryptography is a significant challenge. This study proposes an efficient scalar multiplication method for elliptic curves over finite fields $GF(2^m)$. The proposed method first converts the scalar into a binary number. Then, using Horner's rule, the binary number is divided into fixed-length bit-words. Each bit-word undergoes repeating point doubling, which can be precomputed. However, repeating point doubling typically involves numerous inverse operations. To address this, significant effort has been made to develop formulas that minimize the number of inverse operations. With the proposed formula, regardless of how many times the operation is repeated, only a single inverse operation is required. Over $GF(2^m)$, the proposed method for scalar multiplication outperforms the sliding window method, which is currently regarded as the fastest available. However, the introduced formulas require more multiplications, squares, and additions. To reduce these operations, we further optimize the square operations; however, this introduces a trade-off between computation time and memory size. These challenges are key areas for future improvement.

Keywords: elliptic curve; scalar multiplication; inverse operation; finite field

MSC: 68P25

1. Introduction

Elliptic curve cryptography (abbreviated as ECC) was introduced by Miller [1] in 1986 and Koblitz [2] in 1987. ECC is typically defined over prime finite fields $GF(p)$ or binary finite fields $GF(2^m)$. Public key cryptographic primitives can be implemented using abelian groups generated by elliptic curves over $GF(p)$ or $GF(2^m)$. ECC provides the same level of security as traditional public key cryptography, but with a smaller number of parameters. In practical applications, ECC over $GF(p)$ and $GF(2^m)$ each possess distinct advantages, and the choice between them depends on the specific requirements of the application. For example, $GF(p)$ is often preferred in scenarios demanding high security and versatility, such as financial transactions, digital signatures, and SSL/TLS protocols. ECC over $GF(p)$ generally provides stronger security guarantees and is well supported in both hardware and software implementations. On the other hand, $GF(2^m)$ is particularly suitable for resource-constrained environments, such as embedded systems and Internet of Things (IoT) devices, due to its computational efficiency. Operations over $GF(2^m)$ can be significantly accelerated through hardware optimization, making them more advantageous in scenarios where high computational efficiency is critical.

ECC designs over prime fields generally offer stronger resistance to side-channel attacks, while designs over binary fields benefit from a carry-free feature, making arithmetic operations more suitable for hardware implementation. ECC employs an encryption technique based on the discrete logarithm problem. The discrete logarithm problem is defined as follows: Given an elliptic curve E over a finite field and two points P and Q on E, the task is to find the value of k such that $Q = kP$. However, scalar multiplications and point inversions both are computationally intensive and represent key challenges. Regarding ECC defined over finite fields, numerous methods have been developed to optimize scalar multiplication and point inversion, including algebraic theorem-based designs [3], bit-slicing techniques [4], lookup tables [5], non-adjacent forms (NAFs) [6], and so on. For instance, the method in [7] minimizes the number of non-zero bits using the direct recoding method [8] to enhance scalar multiplication. Implementing ECC arithmetic operations on various coordinates can lead to faster computations. In [9], Jacobian coordinates are used to achieve high-efficiency point addition and doubling without requiring point inversions. In [10], the authors derive formulas for $3P$ in λ-projective coordinates and for $5P$ in both affine and λ-projective coordinates, marking the first study in λ-projective coordinates.

The methods presented in [11–13] transform scalar multiplication processes from affine to projective coordinate systems, with implementations verified on FPGA boards. For a more in-depth analysis on using various coordinates, we refer the reader to [14]. In terms of hardware implementations, the lookup table approach in [15] optimizes double point-doubling operations, while the triple-based chain method [16] reduces time consumption in elliptic curve cryptosystems. A low-latency window algorithm [17] enhances security, as does an enhanced comb method for point addition and doubling. In [18], a configurable ECC crypto-processor defined with the Weierstrass equation over prime fields was implemented and verified on a Xilinx FPGA board. Modular multipliers over $GF(2^m)$ are discussed in [19], and algorithmic improvements for computational complexity in low-power devices are presented in [20].

Reducing the number of inverse operations in scalar multiplication is crucial, as inversion over finite fields is the most time-consuming of all basic operations. In this work, a modified Horner's rule based on binary scalar representation and a grouping technique are employed to accelerate scalar multiplication. Using the grouping technique, the scalar is partitioned into bit-words, where each represents a sum of repeating point doublings that can be precomputed and stored. Instead of traditional point doubling, this study derives formulas for performing repeating point doubling. These formulas require only one inversion operation regardless of the number of repetitions. Unlike projective coordinate systems, the derived formulas are based on the affine coordinate system. To the best of our knowledge, these formulas are the first to compute scalar multiplication in this manner. Additionally, the proposed method is suitable for both software and hardware implementations, as the arithmetic operations are simple and consistent in execution. From a software perspective, the proposed method achieves faster scalar multiplication computation compared with the sliding window algorithm [21]. While the sliding window method [21] is a highly efficient general-purpose technique and is widely regarded as the fastest available, it may not be optimal for all scenarios. Integrating the proposed method with the sliding window algorithm can further enhance its performance.

The contributions of this study are as follows:

- We propose an efficient repeating point-doubling algorithm that relies solely on standard inversion operations.
- A generalizable accelerated squaring method is introduced, which can be applied to inverse element computation.

- The proposed repeating point-doubling algorithm can enhance the performance of the sliding window method or any other technique requiring repeating point-doubling operations.
- The calculation of repeated point doubling is a critical component in algorithms for computing scalar multiplication. By replacing these operations with our proposed method, we can achieve further improvements in efficiency. For instance, our approach demonstrates significant performance gains when applied to techniques such as the sliding window algorithm, as evidenced by the experimental data presented in Section 4.

The rest of this paper is organized as follows: Section 2 introduces finite field arithmetic on elliptic curves. In Section 3, formulas for repeating point doubling are derived, which significantly reduce the computation time compared with traditional point doubling. Additionally, a modified square operation is introduced to further improve efficiency. Section 4 presents the results of the simulation implemented in Python 3.9 and executed on an Intel Core i9-14900K processor, showcasing the performance of the proposed methods. Finally, conclusions are provided in Section 5.

2. Preliminaries

2.1. Basic Operations on $GF(2^m)$

In the following, the binary operator "+" will denote an addition operation, which may vary depending on the context, such as addition of real numbers, bits, polynomials, or points on an elliptic curve. The exact meaning of "+" will be clear from the context in which it is used. When $a, b \in \{0,1\}$, the operation $a+b$ refers to the addition modulo 2 (i.e., binary addition).

Let $A = a_{m-1}a_{m-2}\ldots a_1a_0$ be an element in $GF(2^m)$, where $a_i \in \{0,1\}$ for $0 \le i \le m-1$. Then, A can be represented as a polynomial $A(x) = \sum_{i=0}^{m-1} a_i x^i$. For simplicity, $A(x)$ is referred to as being defined over $GF(2^m)$. Let $B(x) = \sum_{i=0}^{m-1} b_i x^i$ be defined over $GF(2^m)$. Then, the addition of $A(x)$ and $B(x)$, denoted as $A(x)+B(x)$, is defined by $\sum_{i=0}^{m-1}(a_i+b_i)x^i$. For example, suppose that $A = 10101110$ and $B = 11011011$ are elements in $GF(2^8)$. We can express these binary elements as polynomials $A(x) = x^7+x^5+x^3+x^2+x$ and $B(x) = x^7+x^6+x^4+x^3+x+1$. Now, performing the addition $A(x)+B(x)$, which is equivalent to bitwise addition modulo 2, we obtain the following:

$$\begin{aligned} A(x)+B(x) &= (1+1)x^7+(0+1)x^6+(1+0)x^5+(0+1)x^4+(1+1)x^3 \\ &\quad +(1+0)x^2+(1+1)x+(0+1) \\ &= x^6+x^5+x^4+x^2+1. \end{aligned} \tag{1}$$

or equivalently, in binary form: $A+B = 01110101$. In programming, (a_i+b_i) is the XOR operation of a_i and b_i. The multiplication of $A(x)$ and $B(x)$ is the remainder $R(x)$ of the product of $A(x)$ and $B(x)$ by dividing an irreducible polynomial $f(x)$ of degree m defined over $GF(2^m)$. Symbolically, the multiplication of $A(x)$ and $B(x)$ in $GF(2^m)$ is denoted as $R(x) \equiv A(x)B(x) \bmod f(x)$.

Over $GF(2^m)$, the Extended Euclidean Algorithm is employed to compute the remainder of the product of $A(x)$ and $B(x)$ by dividing $f(x)$. However, there are many time-consuming divisions in the algorithm. In order to avoid the divisions, Fermat's Little Theorem is usually employed to compute the remainder or inverse. A polynomial $B(x)$ is said to be the inverse of a polynomial $A(x)$ if

$$A(x)B(x) \equiv \bmod f(x).$$

We denote $B(x)$ as $A^{-1}(x)$. In Fermat's Little Theorem, suppose that A is an element in $GF(2^m)$. Then, the inverse A^{-1} of A is equal to A^{2^m-2}; moreover, $A^{2^m-2} = \prod_{i=1}^{m-1} A^{2^i}$.

2.2. Point Addition and Point Doubling on Elliptic Curve

The elliptic curve $E(x, y)$ defined in $GF(2^m)$ is given by $y^2 + xy = x^3 + Ax^2 + B$, where A and B are elements in $GF(2^m)$. Let $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ be two points in $E(x, y)$. The point addition of P and Q, denoted by $P + Q$, is the point (x_3, y_3) in $E(x, y)$ obtained as follows:

1. If $P \neq Q$, $P + Q$ is defined by the negative of the point that is the intersection of $E(x, y)$ and the line passing through P and Q. Let λ denote the slope of the line. Then,

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2} = (y_1 + y_2)(x_1 + x_2)^{-1}, x_3 = \lambda^2 + \lambda + x_1 + x_2 + A, \text{ and } y_3 = y_1 + (x_1 + x_3)\lambda + x_3.$$

2. If $P = Q$, $P + P$ is the negative of the point that is the intersection of $E(x, y)$ and the tangent line passing through the point P. $P + P$ is written as $2P = (x_2, y_2)$ and is called the point doubling of P, and we have

$$\lambda = x_1 + \frac{y_1}{x_1} = x_1 + y_1 x_1^{-1}, x_2 = \lambda^2 + \lambda + A, \text{ and } y_2 = x_1^2 + (\lambda + 1)x_2. \quad (2)$$

An illustrative example is presented based on the definitions of point addition and point doubling as follows. Over $GF(2^5)$, let $P = (x_1, y_1) = (00110, 10000), Q = (x_2, y_2) = (01010, 10010)$ be points on $y^2 + xy = x^3 + Ax^2 + B$, where $A = 0001, B = 0001$, and an irreducible polynomial $f(x) = x^5 + x + 1$. Let $R = (x_3, y_3) = P + Q$. Then,

$$\begin{aligned}
\lambda &= \tfrac{y_1+y_2}{x_1+x_2} \\
&= \tfrac{10000+10010}{00110+01010} \\
&= \tfrac{00010}{01100} = (00010)(00111) = 01110, \\
x_3 &= \lambda^2 + \lambda + x_1 + x_2 + A \\
&= (01110)^2 + 01110 + 00110 + 01010 + 00001 = 11101, \text{ and} \\
y_3 &= y_1 + (x_1 + x_3)\lambda + x_3 \\
&= 10000 + (00110 + 11101)(01110) + 11101 = 11011.
\end{aligned}$$

Let $P + P = (x_2, y_2)$. Then,

$$\begin{aligned}
\lambda &= x_1 + \tfrac{y_1}{x_1} \\
&= 00110 + \tfrac{10000}{00110} \\
&= 00110 + (10000)(01110) = 00110 + 11011 = 11101 \\
x_2 &= \lambda^2 + \lambda + A \\
&= (11101)^2 + 11101 + 00001 \\
&= 10110 + 11101 + 00001 = 01010, \text{ and} \\
y_2 &= x_1^2 + (\lambda + 1)x_2 \\
&= (00110)^2 + (11101 + 00001)(01010) \\
&= 10100 + 00110 = 10010.
\end{aligned}$$

3. Optimizing Scalar Multiplications

3.1. Scalar Multiplication

Let $k \geq 2$ be an integer and P be a point in $E(x,y) : y^2 + xy = x^3 + Ax^2 + B$. The scalar multiplication kP of P is defined by $kP = \underbrace{P + P + \cdots + P}_{k}$. The computation of kP is lengthy. To reduce the duration, first, k is converted to a binary representation, as follows:

$$k_{m-1}2^{m-1} + k_{m-2}2^{m-2} + \cdots + k_1 2^1 + k_0 2^0, \tag{3}$$

where $k_i \in \{0,1\}$ for $0 \leq i \leq m-1$. Let d, w, r be non-negative integers such that $m = w \cdot d + r$ with $0 \leq r \leq w-1$. Then, using Horner's rule, Equation (3) can be represented as

$$\begin{aligned} k &= \underbrace{((\cdots(}_{d} k_{m-1}2^{w-1} + k_{m-2}2^{w-2} + \cdots + k_{m-(w-1)}2 + k_{m-w})2^w \\ &+ k_{m-w-1}2^{w-1} + k_{m-w-2}2^{w-2} + \cdots + k_{m-w-(w-1)}2 + k_{m-w-w})2^w + \cdots \\ &+ k_{m-(d-1)w-1}2^{w-1} + k_{m-(d-1)w-2}2^{w-2} + \cdots + k_{m-(d-1)w-(w-1)}2 + k_{m-(d-1)w-w})2^r \\ &+ k_{r-1}2^{r-1} + k_{r-2}2^{r-2} + \cdots k_1 2 + k_0. \end{aligned}$$

For example, suppose that $k = 2^{14} + 2^{12} + 2^{11} + 2^9 + 2^8 + 2^6 + 2^5 + 2^4 + 2 + 1$ and $w = 3$. Then,

$$\begin{aligned} k &= ((((2^2 + 0 \cdot 2^1 + 2^0)2^3 + 2^2 + 0 \cdot 2^1 + 2^0)2^3 + 2^2 + 0 \cdot 2^1 + 2^0)2^3 \\ &\quad + 2^2 + 2^1 + 0 \cdot 2^0)2^3 + 0 \cdot 2^2 + 2^1 + 2^0. \end{aligned} \tag{4}$$

As the idea behind the proposed method comes from the sliding window [21], let us briefly introduce the basic concept of the sliding window by the following example. For k in Equation (4) with window size $w = 3$, k can be written as

$$k = 101\ 101\ 101\ 110\ 011. \tag{5}$$

In accordance with the sliding window method, the precomputations are $(\lceil 15/3 \rceil - 1)$ point additions; a point doubling number of 9; and $2P$, $3P(2P + P)$, $5P(3P + 2P)$, and $7P(5P + 2P)$. The number of point doubling is the number of times a window with length w is successively shifted one place from left to right, skipping the zeros if they are not in the window. More details on the sliding window method can be found in [21]. With the proposed method, kP is written as

$$\begin{aligned} kP &= \underbrace{((\cdots(}_{d} k_{m-1}2^{w-1}P + k_{m-2}2^{w-2}P + \cdots + k_{m-(w-1)}2P + k_{m-w}P)2^w \\ &+ k_{m-w-1}2^{w-1}P + k_{m-w-2}2^{w-2}P + \cdots + k_{m-w-(w-1)}2P + k_{m-w-w}P)2^w \\ &+ \cdots \\ &+ k_{m-(d-1)w-1}2^{w-1}P + k_{m-(d-1)w-2}2^{w-2}P + \cdots \\ &\qquad + k_{m-(d-1)w-(w-1)}2P + k_{m-(d-1)w-w}P)2^r \\ &+ k_{r-1}2^{r-1}P + k_{r-2}2^{r-2}P + \cdots k_1 2P + k_0 P. \end{aligned} \tag{6}$$

In Equation (6), for $0 \leq i \leq d-1$, each

$$k_{m-iw-1}2^{w-1}P + k_{m-iw-2}2^{w-2}P + \cdots + k_{m-iw-(w-1)}2P + k_{m-iw-w}P \tag{7}$$

is referred to as a w-bit word, denoted as $(k^i_{w-1}, k^i_{w-2}, \cdots, k^i_1, k^i_0)$. For the last r terms,

$$k_{r-1}2^{r-1}P + k_{r-2}2^{r-2}P + \cdots + k_1 2P + k_0 P \tag{8}$$

in Equation (6) is also represented as a w-bit word $(k^d_{w-1}, k^d_{w-2}, \cdots, k^d_r, k^d_{r-1}, k^d_{r-2}, \cdots, k^d_1, k^d_0)$ with $k^d_j = 0$ for $r \leq j \leq w-1$.

For Equations (7) and (8), it is evident that any scalar multiplication operation can be equivalently expressed as the computation of $2P, 2^2P, \cdots, 2^{w-1}P$ for each i. For a small value of w, the points $2P, 2^2P, \cdots, (2^w - 1)P$ can be precomputed and stored in advance, as illustrated in Table 1. In this table, given the point P, the scalar k, and the word length w, the result of Equation (7) or (8) can be directly retrieved from the entry $L(k_{m-iw+w-1}, k_{m-iw+w-2}, \cdots, k_{m-iw+w-(w-1)}, k_{m-iw+w-w})$, provided that $k^i_j = k_{m-iw+j}$ for $0 \leq i \leq d$ and $0 \leq j \leq w-1$.

Table 1. Precomputations for Equations (7) and (8).

$L(k_{m-iw+w-1}, k_{m-iw+w-2}, \cdots, k_{m-iw+w-(w-1)}, k_{m-iw+w-w})$	$\sum_{j=0}^{w-1} k_{m-iw-(w-j)} 2^j P$
$L(0,0,0,\cdots,0,0,0)$	0
$L(0,0,0,\cdots,0,0,1)$	P
$L(0,0,0,\cdots,0,1,0)$	$2P$
$L(0,0,0,\cdots,0,1,1)$	$L(0,0,0,\cdots,0,0,1) + L(0,0,0,\cdots,0,1,0) = 3P$
$L(0,0,0,\cdots,1,0,0)$	2^2P
$\vdots$	$\vdots$
$L(1,0,0,\cdots,0,0,0)$	$2^{w-1}P$
$L(1,0,0,\cdots,0,0,1)$	$L(1,0,0,\cdots,0,0,0) + L(0,0,0,\cdots,0,0,1) = 2^{w-1}P + P$
$\vdots$	$\vdots$
$L(1,1,1,\cdots,1,1,1)$	$L(1,0,0,\cdots,0,0,0) + L(0,1,1,\cdots,1,1,1) = (2^w - 1)P$

We will use the following example to demonstrate how to look up values in Table 1. Suppose that $w = 3$; we precompute all eight possible combinations, as shown in the following table. For the value of k according to Equation (5), for the combination 110, we obtain the result from the table entry $L(1,1,0)$, which is $6P$.

$L(k_2, k_1, k_0)$	$k_2P^2 + k_1P + k_0P$
$L(0,0,0)$	0
$L(0,0,1)$	P
$L(0,1,0)$	$2P$
$L(0,1,1)$	$2P + P$
$L(1,0,0)$	$4P$
$L(1,0,1)$	$4P + P$
$L(1,1,0)$	$4P + 2P$
$L(1,1,1)$	$4P + 3P$

Therefore, given point P, scalar k, and word length w, kP can be computed with the following Algorithm 1, `ScalarMUL`.

Algorithm 1 `ScalarMUL`(k, P, w)

1. `Set` $Q = 0$
2. `Using` P `to create table` L, `as shown in Table` 1
3. `Set` $d = \lfloor m/w \rfloor$ `and` $r = m - w \cdot d$
4. `For` $i \leftarrow d - 1$ `downto` 0
5. `do`
6. $\quad Q \leftarrow Q + L(k_{m-iw+w-1}, k_{m-iw+w-2}, \cdots, k_{m-iw+w-(w-1)}, k_{m-iw+w-w})$
7. $\quad Q \leftarrow 2^w Q$
8. `Enddo`
9. $Q \leftarrow 2^r Q$
10. $Q \leftarrow Q + L(0, 0, \cdots, 0, k_{r-1}, k_{r-2}, \cdots, k_1, k_0)$
11. `return` Q

3.2. Reducing Inverse in the Repeating Point Doubling

The sliding window method [21] shifts a window of length $w > 0$ and skips over runs of zeros between them while disregarding the fixed digit boundaries. However, in the `ScalarMUL` algorithm, the binary representation of k is partitioned into fixed-length bit-words of size w, where each word is processed sequentially. This approach can also be extended to the sliding window method, as will be demonstrated in Section 4 with the experimental results. Within the `ScalarMUL` algorithm, it is necessary to compute $2^w Q$ in step 7 and $2^r Q$ in step 9.

According to the definition of scalar multiplication kQ of Q and the associative property of point addition on the elliptic curve $E(x, y)$, for any positive integer n, $2^n Q$ can be expressed as the point doubling of $2^{n-1}Q$. Specifically, $2^n Q = 2^{n-1}Q + 2^{n-1}Q = 2(2^{n-1}Q)$. Traditionally, as described in Equation (2), $2^n Q$ can be computed using the following Algorithm 2, referred to as `Tradition`. In the `Tradition` algorithm, line 4 employs Equation (2) to compute $2Q$. Each iteration performs a point-doubling operation on Q requiring five XORs (additions), two multiplications, and one inverse operation. The addition, multiplication, and square operations mentioned here are all operations defined within $GF(2^m)$.

Algorithm 2 `Tradition`(n, P)

1. `Set` $Q = P$
2. `For` $i \leftarrow n - 1$ `downto` 0
3. `do`
4. $\quad Q \leftarrow 2Q$
5. `Enddo`
6. `return` Q

To obtain $2^n Q$, we have to compute $2Q(Q + Q), 4Q(2Q + 2Q), \cdots, 2^n(2^{n-1}Q + 2^{n-1}Q)$. Therefore, in the computation, there are n inverse operations, $5n$ XORs, $2n$ multiplications, and $2n$ squares. Since the inverse operation is computationally expensive, we have developed optimized formulas to replace the point-doubling computation in the `Tradition` algorithm. The derived formulas are designed to ensure that only a single inverse operation is required when computing $2^n Q$ of a given point Q, significantly improving computational efficiency. Let $Q_0 = (x_0, y_0)$ be a point in $E(x, y)$. For $n \geq 1$, let $Q_n = (x_n, y_n)$ be the point doubling of Q_{n-1}. Then, Q_n is the scalar multiplication $2^n Q_0$ of Q_0. Let λ_n be the slope of the tangent line passing through the point Q_{n-1}. Then, to derive formulas for Q_n obtained from Q_0 via the iteration of point doubling, first, consider $Q_1 = (x_1, y_1)$. We have

$$\begin{aligned}
\lambda_1 &= x_0 + \tfrac{y_0}{x_0} = \tfrac{x_0^2+y_0}{x_0} = \tfrac{v_1}{x_0},\\
x_1 &= \lambda_1^2 + \lambda_1 + A = \left(\tfrac{v_1}{x_0}\right)^2 + \tfrac{v_1}{x_0} + A = \tfrac{Ax_0^2+v_1x_0+v_1^2}{x_0^2} = \tfrac{u_1}{x_0^2}, \text{ and}\\
y_1 &= x_0^2 + (\lambda_1+1)x_1,
\end{aligned} \tag{9}$$

where $v_1 = x_0^2 + y_0$ and $u_1 = Ax_0^2 + v_1x_0 + v_1^2$.

In what follows, the formula for y_n will be omitted until λ_n and x_n are obtained.

For $Q_2 = 2Q_1 = (x_2, y_2)$,

$$\begin{aligned}
\lambda_2 &= x_1 + \tfrac{y_1}{x_1} = \lambda_1^2 + (A+1) + \tfrac{x_0^2}{x_1} = \tfrac{v_1^2}{x_0^2} + (A+1) + \tfrac{x_0^2x_0^2}{u_1},\\
&= \tfrac{(A+1)u_1x_0^2+u_1v_1^2+x_0^2(x_0^2)^2}{u_1x_0^2} = \tfrac{v_2}{u_1x_0^2} \text{ and}\\
x_2 &= \lambda_2^2 + \lambda_2 + A = \tfrac{v_2^2}{(u_1x_0^2)^2} + \tfrac{v_2}{u_1x_0^2} + A = \tfrac{A(u_1x_0^2)^2+v_2(u_1x_0^2)+v_2^2}{(u_1x_0^2)^2} = \tfrac{u_2}{(u_1x_0^2)^2},
\end{aligned} \tag{10}$$

where $v_2 = (A+1)u_1x_0^2 + u_1v_1^2 + x_0^2(x_0^2)^2$ and $u_2 = A(u_1x_0^2)^2 + v_2(u_1x_0^2) + v_2^2$.

For $Q_3 = 2Q_2 = (x_3, y_3)$,

$$\begin{aligned}
\lambda_3 &= x_2 + \tfrac{y_2}{x_2} = \lambda_2^2 + (A+1) + \tfrac{x_1^2}{x_2}\\
&= \tfrac{v_2^2}{(u_1x_0^2)^2} + (A+1) + \tfrac{u_1^2/x_0^4}{(u_2/u_1x_0^2)^2}\\
&= \tfrac{(A+1)u_2(u_1x_0^2)^2+u_2v_2^2+(u_1^2)^2(u_1x_0^2)^2}{u_2(u_1x_0^2)^2} = \tfrac{v_3}{u_2(u_1x_0^2)^2} \text{ and}\\
x_3 &= \lambda_3^2 + \lambda_3 + A = \tfrac{v_3^2}{(u_2(u_1x_0^2)^2)^2} + \tfrac{v_3}{u_2(u_1x_0^2)^2} + A\\
&= \tfrac{A(u_2(u_1x_0^2)^2)^2+v_3(u_2(u_1x_0^2)^2)+v_3^2}{(u_2(u_1x_0^2)^2)^2} = \tfrac{u_3}{(u_2(u_1x_0^2)^2)^2},
\end{aligned} \tag{11}$$

where $v_3 = (A+1)u_2(u_1x_0^2)^2 + u_2v_2^2 + (u_1^2)^2(u_1x_0^2)^2$ and $u_3 = A(u_2(u_1x_0^2)^2)^2 + v_3(u_2(u_1x_0^2)^2) + v_3^2$.

For $Q_4 = 2Q_3 = (x_4, y_4)$,

$$\begin{aligned}
\lambda_4 &= x_3 + \tfrac{y_3}{x_3} = \lambda_3^2 + (A+1) + \tfrac{x_2^2}{x_3}\\
&= \tfrac{v_3^2}{(u_2(u_1x_0^2)^2)^2} + (A+1) + \tfrac{u_2^2/((u_1x_0^2)^2)^2}{(u_3/(u_2(u_1x_0^2)^2)^2}\\
&= \tfrac{(A+1)u_3(u_2(u_1x_0^2)^2)^2+u_3v_3^2+(u_2^2)^2(u_2(u_1x_0^2)^2)^2}{u_3(u_2(u_1x_0^2)^2)^2} = \tfrac{v_4}{u_3(u_2(u_1x_0^2)^2)^2} \text{ and}\\
x_4 &= \lambda_4^2 + \lambda_4 + A = \tfrac{v_4^2}{(u_3(u_2(u_1x_0^2)^2)^2)^2} + \tfrac{v_4}{u_3(u_2(u_1x_0^2)^2)^2} + A\\
&= \tfrac{A(u_3(u_2(u_1x_0^2)^2)^2)^2+v_4(u_3(u_2(u_1x_0^2)^2)^2)+v_4^2}{(u_3(u_2(u_1x_0^2)^2)^2)^2} = \tfrac{u_4}{(u_3(u_2(u_1x_0^2)^2)^2)^2},
\end{aligned} \tag{12}$$

where

$$v_4 = (A+1)u_3(u_2(u_1x_0^2)^2)^2 + u_3v_3^2 + (u_2^2)^2(u_2(u_1x_0^2)^2)^2$$

and

$$u_4 = A(u_3(u_2(u_1x_0^2)^2)^2)^2 + v_4(u_3(u_2(u_1x_0^2)^2)^2) + v_4^2.$$

The formulas for x_n and λ_n can be extended iteratively for arbitrarily large values of n, allowing us to compute 2^nQ for any desired n. However, the derivation process becomes increasingly laborious and cumbersome as n grows larger, making it impractical for manual computation. Before establishing that there is only one inverse operation involved in the computation of scalar multiplication, it will be helpful to introduce the following recurrence relations. By following Equations (9)–(12), let $t_1 = x_0, t_2 = u_1x_0^2$, and $t_3 = u_2(u_1x_0^2)^2$. Then,

$$v_3 = (A+1)t_3 + u_2v_2^2 + (u_1^2)^2t_2^2 \text{ and } u_3 = At_3^2 + v_3t_3 + v_3^2 \tag{13}$$

For $n \geq 3$, the following relationships can be easily derived:

$$\begin{aligned} t_n &= u_{n-1}t_{n-1}^2, \\ v_n &= (A+1)t_n + u_{n-1}v_{n-1}^2 + (u_{n-2}^2)^2t_{n-1}^2, \text{ and} \\ u_n &= At_n^2 + v_nt_n + v_n^2. \end{aligned}$$

Table 2 is an illustration of Equations (9)–(12) to compute λ_4 and x_4. In the example, the curve is defined over $GF(2^{163})$. For $m = 233, 283, 409, 571$, the computations of λ_4 and x_4 are shown in Appendix A.

Table 2. Computations of λ_4 and x_4 for four times point doubling of Q_0 for $m = 163$.

$A =$ 0x1, primitive polynomial $f(x) = x^{163} + x^7 + x^6 + x^3 + 1$, $Q_0 = (x_0, y_0)$ $x_0 =$ 0x02FE13C0537BBC11ACAA07D793DE4E6D5E5C94EEE8 $y_0 =$ 0x0289070FB05D38FF58321F2E800536D538CCDAA3D9	
$t_1 = x_0$ $v_1 = t_1^2 + y_0$ $u_1 = At_1^2 + v_1t_1 + v_1^2$	0x2FE13C0537BBC11ACAA07D793DE4E6D5E5C94EEE8 0x4F80CD7EF766D64506FDDADAE0D599F74B2227367 0x32FBC2266652998D2D2F03AFD6241F309DDE4AE1B
$t_2 = u_1t_1^2$ $v_2 = (A+1)t_2^2 + u_1v_1^2 + (x_0^2)^2t_1^2$ $u_2 = At_2^2 + v_2t_2 + v_2^2$	0x5A40058EDE40D0A67D4BD8CB03557EEC05F034063 0x2792546E8CB0EE4CC70AB686063CA9C9EABCE3A12 0x52D96F1338F1AA0962CBCED0BF145D810E7F7E174
$t_3 = u_2t^2$ $v_3 = (A+1)t_3^2 + u_2v_2^2 + (u_1^2)^2t_2^2$ $u_3 = At_3^2 + v_3t_3 + v_3^2$	0x17EA26AC56E2A438890799BFBDF518C44B1769326 0x1A26B8B369A2A6FE9FE00452B82B49FFE32453AC 0x1415E6A7EE1563767A757312679BA44FCFA9C42DF
$t_4 = u_3t_3^2$ $v_4 = (A+1)t_4^2 + u_3v_3^2(u_2^2)^2t_3^2$ $u_4 = At_4^2 + v_4t_4 + v_4^2$	0x2B4B1B80260F6CA1D0BE900A98486B175408B673D 0x79C9EAD3F5B5A625DF7C1D6E3F3C572181F7128E7 0x5DB8FC493F7495E19777B26FAF97457756AD27E2E
$t = t_4^{-1}$ $\lambda_4 = v_4t$ $x_4 = u_4t^2$	0x145E47AFC3228B6070CCC6D1F3B9D178EA838006E 0x7FDE58E3AE8F043ECDE437CA1581911B725743721 0x2E8D15536960EB926E78D9E15CE721DFAE4FE3134

Lemma 1. *For $n \geq 3$, $\lambda_n = \frac{v_n}{t_n}$ and $x_n = \frac{u_n}{t_n^2}$.*

Proof of Lemma 1. We will proceed with induction on n. Equations (9)–(12) show the basis step for λ_n and x_n. For the inductive step,

$$\begin{aligned} \lambda_n &= \lambda_{n-1}^2 + (A+1) + \tfrac{x_{n-2}^2}{x_{n-1}} \\ &= \left(\tfrac{v_{n-1}}{t_{n-1}}\right)^2 + (A+1) + \frac{\left(u_{n-2}/t_{n-2}^2\right)^2}{u_{n-1}/t_{n-1}^2} \\ &= \tfrac{v_{n-1}^2}{t_{n-1}^2} + (A+1) + \tfrac{u_{n-2}^2t_{n-1}^2}{u_{n-1}t_{n-2}^4} \\ &= \tfrac{v_{n-1}^2}{t_{n-1}^2} + (A+1) + \tfrac{(u_{n-2}^2)^2}{u_{n-1}} \\ &= \tfrac{(A+1)u_{n-1}t_{n-1}^2 + u_{n-1}v_{n-1}^2 + (u_{n-2}^2)^2t_{n-1}^2}{u_{n-1}t_{n-1}^2} = \tfrac{v_n}{t_n} \text{ and} \\ x_n &= \lambda_n^2 + \lambda_n + A = \left(\tfrac{v_n}{t_n}\right)^2 + \tfrac{v_n}{t_n} + A = \tfrac{At_n^2 + v_nt_n + v_n^2}{t_n^2} = \tfrac{u_n}{t_n^2}. \end{aligned} \tag{14}$$

This lemma holds. □

Corollary 1. *For* $n \geq 3$, $y_n = \frac{u_{n-1}^2 u_{n-1}^2 + (\lambda_n + 1)u_n}{t_n^2}$.

Proof of Corollary 1. According to Lemma 1, $\lambda_n = \frac{v_n}{t_n^2} \cdot t_n$,

$$\begin{aligned} y_n &= x_{n-1}^2 + (\lambda_n + 1)x_n = \left(\frac{u_{n-1}}{t_{n-1}^2}\right)^2 + (\lambda_n + 1)\frac{u_n}{t_n^2} \\ &= \frac{u_{n-1}^2}{t_{n-1}^4} + (\lambda_n + 1)\frac{u_n}{(u_{n-1}t_{n-1}^2)^2} = \frac{u_{n-1}^2 u_{n-1}^2 + (\lambda_n + 1)u_n}{(u_{n-1}t_{n-1}^2)^2} \\ &= \frac{u_{n-1}^2 u_{n-1}^2 + (\lambda_n + 1)u_n}{t_n^2}. \end{aligned}$$

□

Given a point $Q = (x, y)$ and a positive integer n, the n-times point doubling $2^n Q$ of Q can be efficiently computed using the following Algorithm 3, referred to as `PDNTimes`.

Algorithm 3 `PDNTimes`$(Q = (x, y), n)$

1. `Set` $Q_1 = (0, 0)$
2. `If` $x \neq 0$, `then`
3. $x_0 \leftarrow x$; $y_0 \leftarrow y$
4. $u_0 \leftarrow x_0$; $t_1 \leftarrow x_0$
5. $v_1 \leftarrow t_1^2 + y_0$
6. $u_1 \leftarrow A \cdot t_1^2 + v_1 \cdot t_1 + v_1^2$
7. `For` $i \leftarrow 2$ `upto` n
8. `do`
9. $t_i \leftarrow u_{i-1} \cdot t_{i-1}^2$
10. $v_i \leftarrow (A + 1) \cdot t_i + u_{i-1} \cdot v_{i-1}^2 + (u_{i-2}^2)^2 \cdot t_{i-1}^2$
11. $u_i \leftarrow A \cdot t_i^2 + v_i \cdot t_i + v_i^2$
12. `Enddo`
13. $t \leftarrow t_n^{-1}$
14. $\lambda_n \leftarrow v_n \cdot t$
15. $t' \leftarrow t^2$
16. $x_n \leftarrow u_n \cdot t'$
17. $y_n \leftarrow ((u_{n-1}^2)^2 + (\lambda_n + 1) \cdot u_n) \cdot t'$
18. $Q_1 \leftarrow (x_n, y_n)$
19. `EndIf`
20. `return` Q_1

In the `PDNTimes` algorithm, the computational complexity can be broken down as follows:

- Lines 5 and 6: These lines involve 3 XOR operations, 2 multiplications, and 2 square operations.
- Lines 9–11: Each iteration of the loop in these lines requires 5 XOR operations, 6 multiplications, and 6 square operations.
- Lines 13–17: These lines consist of 2 XOR operations, 4 multiplications, 3 square operations, and 1 inverse operation.

Therefore, a total of $5n$ XORs, $6n$ multiplications, and $6n - 1$ squares are required. However, in the case of hardware devices, the time complexity of adding any two n-bit numbers is currently $O(1)$, while the time complexity of their multiplication is $O(n)$.

Lemma 2. *Over $GF(2^m)$, let $n \geq 2$ be an integer and Q be a point in $E(x,y) : y^2 + xy = x^3 + Ax^2 + B$. The computation of n times point doubling 2^nQ of Q requires $O(n)$ multiplications, $O(n)$ squares, and one inverse operation.*

For the repeating point doubling on $GF(2^m)$, Table 3 demonstrates the execution times of the `Tradition` algorithm and the `PDNTimes` algorithm involved in the `ScalarMUL` algorithm. In other words, in line 7 of the `ScalarMUL` algorithm, the computation of 2^wQ is compared using `PDNTimes` and `Tradition`. Let t_{prev} and t_{prop} denote the execution time of the previous method and the proposed method, respectively. Then, in the table, the decreasing ratio is given by

$$\frac{t_{prev} - t_{prop}}{t_{prev}} \times 100 \tag{15}$$

When comparing the performance of `Tradition` with that of `PDNTimes` for different values of m, it is observed that while the reduction in inverse operations has led to a decrease in computation time, the increased number of multiplication and square operations in the formula results in a slowdown of the computation time reduction as n approaches 8. This trend is illustrated in Figure 1. This trend is attributed to the increase in word length, which leads to longer table construction times and a corresponding rise in memory consumption. Furthermore, as depicted in the figure, this behavior remains consistent across different values of m, indicating that the trade-off between reduced inversions and increased multiplication and square operations persists regardless of the specific parameters.

Table 3. The execution times (10^{-3} s) for `Tradition` and `PDNTimes` over $GF(2^m)$ and the decreasing ratio, where $m = 163, 233, 283, 409, 571$.

m	Method / Ratio	n						
		2	**3**	**4**	**5**	**6**	**7**	**8**
163	`Tradition`	11.024	17.369	25.359	31.436	36.060	41.873	48.851
	`PDNTimes`	6.217	6.462	6.641	6.762	7.034	7.205	7.319
	ratio	43.60	62.80	73.81	78.49	80.49	82.79	85.02
233	`Tradition`	24.701	37.534	50.078	61.188	73.936	89.568	99.046
	`PDNTimes`	12.859	13.195	13.374	13.557	14.197	14.222	14.541
	ratio	47.94	64.85	73.29	77.84	80.80	84.12	85.32
283	`Tradition`	39.671	62.621	79.533	103.221	123.985	144.111	163.431
	`PDNTimes`	20.762	21.181	21.731	21.733	22.341	22.653	23.138
	ratio	47.66	66.18	72.68	78.95	81.98	84.28	85.84
409	`Tradition`	88.431	132.481	176.825	218.813	262.292	306.032	349.894
	`PDNTimes`	44.307	45.134	45.792	46.015	46.691	47.969	48.554
	ratio	49.90	65.93	74.10	78.97	82.20	84.33	86.12
571	`Tradition`	176.102	264.810	348.533	433.089	526.804	606.048	693.713
	`PDNTimes`	87.794	88.223	88.982	89.849	91.223	91.481	92.432
	ratio	50.15	66.68	74.47	79.25	82.68	84.91	86.68

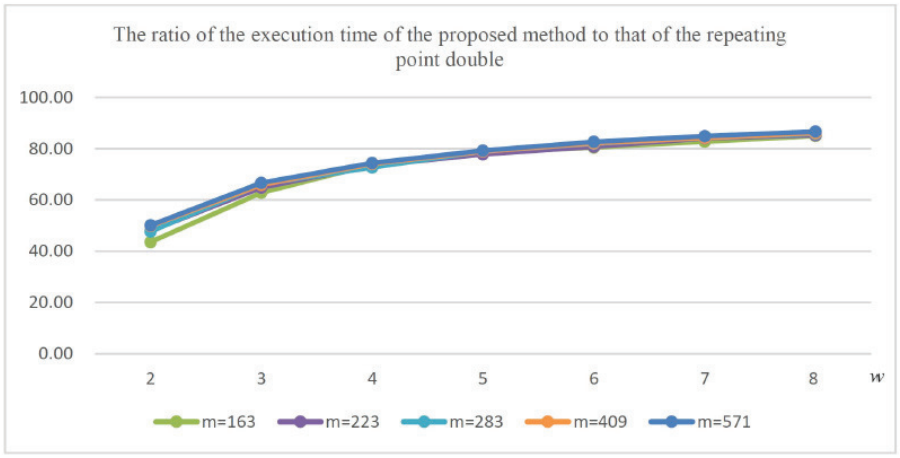


Figure 1. The decreasing behavior shown by the data shown in Table 3.

3.3. Reducing Square Operation Time

In the `PDNTimes` algorithm, there are many square operations in t_i, v_i, and u_i. To further reduce the computation time for scalar multiplication or repeating point doubling, precomputations for square operations are employed again. The method we propose below will enable the square operation to utilize three main operations: XOR, bit shifting, and table lookup. Recall that $A(x) = \sum_{i=0}^{m-1} a_i x^i$ is a polynomial defined over $GF(2^m)$. Then, given an integer $w \geq 2$, let d and r be integers such that $m = w \cdot d + r$ and $0 \leq r \leq w - 1$. Using Horner's rule again (note that the m we are considering is odd),

$$\begin{aligned} A^2(x) \equiv\ & ((\cdots((((\sum_{j=0}^{w-1} a_{m-w+j}x^{2j})x^{2w} + \sum_{j=0}^{w-1} a_{m-2w+j}x^{2j})x^{2w} \bmod f(x) \\ &+ \sum_{j=0}^{w-1} a_{m-3w+j}x^{2j})x^{2w} + \sum_{j=0}^{w-1} a_{m-4w+j}x^{2j})x^{2w} \bmod f(x) \\ &+ \cdots \\ &+ \sum_{j=0}^{w-1} a_{m-(d-1)w+j}x^{2j})x^{2w} + \sum_{j=0}^{w-1} a_{m-dw+j}x^{2j})x^{2r} \bmod f(x) \\ &+ \sum_{j=0}^{r-1} a_j x^{2j}. \end{aligned} \tag{16}$$

In Equation (16), the computation of $A^2(x)$ involves sequentially evaluating the expression

$$((\sum_{j=0}^{w-1} a_{m-iw+j}x^{2j})x^{2w} + \sum_{j=0}^{w-1} a_{m-(i+1)w+j}x^{2j})x^{2w} \bmod f(x) \tag{17}$$

for increasing values of i. Similar to Equation (7) (respectively, Equation (8)), the expression $\sum_{j=0}^{w-1} a_{m-iw+j}x^{2j}$ (respectively, $\sum_{j=0}^{r-1} a_j x^{2j}$) represents a w-bit word, denoted as $(a^i_{w-1}, a^i_{w-2}, \cdots, a^i_1, a^i_0)$ (respectively, $(a^{d+1}_{w-1}, a^{d+1}_{w-2}, \cdots, a^{d+1}_1, a^i_0)$). The result of computing $\sum_{j=0}^{w-1} a_{m-iw+j}x^{2j}$, for $1 \leq i \leq d$, and $\sum_{j=0}^{r-1} a_j x^{2j}$ can be found in the entry $TE(a_{w-1}, a_{w-2}, \cdots, a_0)$ in Table 4 provided that $a_{m-iw+j} = a_j$ for $0 \leq j \leq w - 1$. In the subsequent discussion, the notation "$\bullet << n$" will be used to denote shifting $\bullet$ to the left by n positions, with all the least significant bits set to zero, where n is a positive integer.

In Equation (17), since the maximum degree before applying the modulo operation with respect to $f(x)$ is less than m, the remainder obtained through traditional long division depends on the polynomial $f(x) + x^m$. The result of this modulo operation, denoted as $RD(r_{2w-1}, r_{2w-2}, \cdots, r_0)$, is provided in Table 5, which represents the remainder of (17). Table 5 comprehensively lists all possible outcomes for $r_{2w-1}, r_{2w-2}, \cdots, r_0$.

Therefore, the square operation $A^2(x)$ **mod** $f(x)$ can be computed with the following Algorithm 4, `SquareMod`(A, f, m, w).

Algorithm 4 `SquareMod`(A, f, m, w)

1. `Set` $C = (c_{2w-1}, c_{2w-2}, \cdots, c_0) = (0, 0, \cdots, 0), d = \lfloor \frac{m}{w} \rfloor, r = m - w \cdot d$
2. `Make table` TE `and` RD `such as Table` 4 `and Table` 5`, respectively`
3. `For` $i = 1$ `to` $d - 1$
4. `do`
5. $\quad C \leftarrow C + TE(a_{m-iw+w-1}, a_{m-iw+w-2}, \cdots, a_{m-iw+1}, a_{m-iw})$
6. $\quad C \leftarrow C(x^{2w}) + RD(c_{2w-1}, c_{2w-2}, \cdots, c_0)$
7. `Enddo`
8. $C \leftarrow C + TE(a_{r+w-1}, a_{r+w-2}, \cdots, a_{r+1}, a_r)$
9. $C \leftarrow C(x^{2r}) + RD(c_{2w-1}, c_{2w-2}, \cdots, c_0)$
10. $C \leftarrow C + TE(0, 0, \cdots, 0, a_{r-1}, a_{r-2}, \cdots, a_1, a_0)$
11. `return` C

Table 4. The precomputations for $\sum_{j=0}^{w-1} a_{m-iw+j} x^{2j}$.

$TE(a_{m-iw+w-1}, a_{m-iw+w-2}, \cdots, a_{m-iw+1}, a_{m-iw})$	$\sum_{j=0}^{w-1} a_{m-iw+j} x^{2j}$
$TE(0, 0, \cdots, 0, 0, 0)$	0
$TE(0, 0, \cdots, 0, 0, 1)$	1
$TE(0, 0, \cdots, 0, 1, 0)$	$TE(0, 0, \cdots, 0, 1) << 2$
$TE(0, 0, \cdots, 0, 1, 1)$	$TE(0, 0, \cdots, 0, 1) + TE(0, 0, \cdots, 1, 0)$
$TE(0, 0, \cdots, 1, 0, 0)$	$TE(0, 0, \cdots, 0, 1, 0) << 2$
$TE(0, 0, \cdots, 1, 0, 1)$	$TE(0, 0, \cdots, 0, 1) + TE(0, 0, \cdots, 1, 0, 0)$
$\vdots$	$\vdots$
$TE(1, 1, \cdots, 1, 1, 1)$	$TE(1, 0, \cdots, 0, 0, 0) + TE(0, 1, \cdots, 1, 1, 1)$

Table 5. The precomputations for (17). $f' = f(x) + x^m$.

$RD(r_{2w-1}, r_{2w-2}, \cdots, r_0)$	**Result**
$RD(0, 0, \cdots, 0, 0, 0)$	0
$RD(0, 0, \cdots, 0, 0, 1)$	f'
$RD(0, 0, \cdots, 0, 1, 0)$	$f' << 1$
$RD(0, 0, \cdots, 0, 1, 1)$	$RD(0, 0, \cdots, 0, 0, 1) + RD(0, 0, \cdots, 0, 1, 0)$
$RD(0, 0, \cdots, 1, 0, 0)$	$f' << 2$
$RD(0, 0, \cdots, 1, 0, 1)$	$RD(0, 0, \cdots, 1, 0, 0) + RD(0, 0, \cdots, 0, 0, 1)$
$RD(0, 0, \cdots, 1, 1, 0)$	$RD(0, 0, \cdots, 1, 0, 0) + RD(0, 0, \cdots, 0, 1, 0)$
$RD(0, 0, \cdots, 1, 1, 1)$	$RD(0, 0, \cdots, 1, 0, 0) + RD(0, 0, \cdots, 0, 1, 1)$
$\vdots$	$\vdots$
$RD(1, 1, \cdots, 1, 1, 1)$	$RD(1, 0, \cdots, 0, 0, 0) + RD(0, 1, \cdots, 1, 1, 1)$

In the `SquareMod` algorithm, for each iteration i, the result of the equation of Equation (17) is represented as $C = (c_{2w-1}, c_{2w-2}, \cdots, c_1, c_0)$. In practical implementation, the term x^{2w} in Equation (17) implies that each c_j in C is shifted to the left by $2w$ positions, with all lower-order bits set to zero, where $0 \leq j \leq 2w-1$. Let m' denote the maximum degree of the polynomial in Equation (17) before applying the modulo operation with $f(x)$, and let $f' = f(x) + x^m$. As $A^2(x)$ is stored in an m-bit array in the code, there is a constraint on the shifting of C. Specifically, m' must be greater than the sum of $2w+1$ and the maximum degree of f'. This ensures that the shifting operation does not exceed the bounds of the array and that the modulo operation can be correctly applied.

In the `SquareMod` algorithm, the computational time can be broken down as follows:

- Lines 5 and 6: Each iteration of the loop in these lines requires 2 XOR operations, 1 shift, and 2 table lookups.
- Lines 8–10: These lines consist of 3 XOR operations, 1 multiplication, and 3 table lookups.

Therefore, a total of $(2d+1)$ XORs, d shifts, and $(2d+1)$ table lookups are required. From the perspective of time complexity, this time is negligible compared with the time required for multiplication.

In the `ScalarMUL` and `SquareMod` algorithms, scalar multiplication corresponds to retrieving precomputed values stored in Tables 1, 4, and 5. As a result, this approach significantly enhances computational efficiency by reducing the need for repeated calculations.

Lemma 3. *Given an integer w, the scalar multiplication of a point on $y^2 + xy = x^3 + Ax^2 + B$ over $GF(2^m)$ can be computed in $\lceil \frac{m}{w} \rceil$ iterations in the algorithms* `ScalarMUL` *and* `SquareMod`.

In Lemma 3, the $\lceil \frac{m}{w} \rceil$ iterations imply that a scalar multiplication of the form $2^{\lceil \frac{m}{w} \rceil}Q$ of a given point Q is performed on a given point Q. To evaluate the execution time of the `SquareMod` algorithm, a test code was implemented to execute the algorithm 100,000 times for each word length w with $2 \leq w \leq 8$. Additionally, the memory size required for the lookup table in `SquareMod` was measured for each word length. For instance, in the case of $GF(2^{163})$, Table 6 summarizes the execution time and the corresponding memory size needed for the lookup table in `SquareMod`. Figure 2 provides a graphical representation of the data presented in Table 6. As evident from the table or figure, there is a trade-off between execution time and memory usage. While increasing the word length w can enhance computational efficiency, it also results in a significant increase in the memory size required and construction times for the lookup table. This highlights the need to carefully balance performance optimization with memory constraints when implementing the `SquareMod` algorithm. Finding the optimal word length will also determine the performance of scalar multiplication, meaning the efficiency of scalar multiplication is adjustable. Taking $GF(2^{163})$ as an example, in our program execution environment, the memory size required for each word length w is shown in Table 7. The execution time can be optimized by selecting an appropriate value of w based on the hardware and software specifications of the specific execution environment.

Table 6. The execution time (seconds) and memory size ($2^w \times 163$ bits/8 bits) of the implementation of the `SquareMod` algorithm for $w = 2, 3, \ldots, 8$ in computing $A(x)^2$ over $GF(2^{163})$.

w	Time	Memory Size		
		$RD(\bullet)$	$TE(\bullet)$	Total
2	0.180	82	82	0.16
3	0.122	163	163	0.32
4	0.094	326	326	0.64
5	0.07	652	652	1.27
6	0.69	304	1304	2.55
7	0.060	2608	2608	5.09
8	0.054	5216	5216	10.19

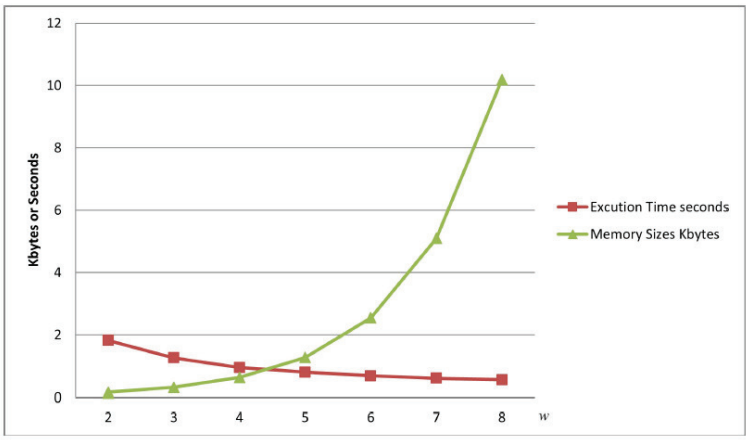


Figure 2. The execution time and memory size used for the algorithm `SquareMod` over $GF(2^{163})$.

Table 7. The execution time and memory size ($2^w \times m$ bits/8 bits) of the `ScalarMUL` algorithm for $w = 2, 3, \ldots, 8$ over $GF(2^m)$, where $m = 163, 233, 283, 409, 571$. Note that the memory size does not include the $RD(\bullet)$ and $TE(\bullet)$ values listed in Table 6.

m	Time	w						
	Size	2	3	4	5	6	7	8
163	seconds	1.18	0.83	0.71	0.70	0.83	1.22	2.07
	bytes	82	163	326	652	1304	2608	5216
233	seconds	3.36	2.32	1.89	1.81	2.08	2.79	4.46
	bytes	117	233	466	932	1864	3728	7456
283	seconds	6.44	4.46	3.63	3.40	3.62	4.85	7.47
	bytes	142	283	566	1132	2264	4528	9056
409	seconds	19.98	13.64	10.71	9.62	9.96	12.01	17.33
	bytes	205	409	818	1636	3272	6544	13,088
571	seconds	50.36	33.67	26.29	22.74	21.91	25.40	34.24
	bytes	286	571	1142	2284	4568	9136	18,272

4. Inverse Algorithm Use `ScalarMUL`

ECC parameters over $GF(2^m)$ used in the `ScalarMUL` algorithm and the sliding window method [21] are provided in Table A5 of Appendix A. The execution times for each word length, both with and without the formulas utilized in the `ScalarMUL` algorithm, as

well as for each window size in the sliding window method [21], are presented in Table 8. Note that the scalar k used in the algorithm `ScalarMUL` and the sliding window are the extension degree m of $GF(2^m)$. Additionally, Table 9 illustrates the decreasing ratio, which compares the execution time of the proposed method with that of the sliding window method [21], highlighting the efficiency improvements achieved by the proposed approach. The decreasing trend in execution time is illustrated in Figure 3. The proposed formulas are specifically tailored for scenarios that involve repeating point-doubling operations, enabling a significant reduction in the number of inverse operations required. The application of our proposed method to the sliding window technique simply requires replacing the formulas we derived for repeating point doubling in Algorithm 2 in [21] with our proposed formulas. Furthermore, these formulas can be seamlessly integrated into the sliding window method to further improve its computational efficiency, as demonstrated by the results presented in Table 9. From Table 9, we observe that the sliding window method with formulas exhibits better efficiency. This is because the sliding window method utilizes a window based on the positions of the bit 1s in the binary representation of k for repeated point doubling. In contrast, our method uses a fixed word length, which requires more precomputation. However, this also demonstrates the value of our derived formulas. This integration highlights the versatility and effectiveness of the proposed approach in optimizing elliptic curve operations.

Table 8. The execution times (seconds) for the `ScalarMUL` algorithm (both the `PDNTimes` and `SquareMod` algorithms are utilized) and sliding window method [21] over $GF(2^m)$.

m	Methods	Window Size or Word Length, w						
		2	**3**	**4**	**5**	**6**	**7**	**8**
163	Sliding window	1.72	1.51	1.42	1.43	1.48	1.67	2.08
	`ScalarMUL` without formulas	1.66	1.50	1.46	1.50	1.73	2.13	2.98
	`ScalarMUL` with formulas	1.18	0.83	0.71	0.70	0.83	1.22	2.07
233	Sliding window	5.08	4.51	4.21	4.14	4.25	4.58	5.44
	`ScalarMUL` without formulas	4.96	4.44	4.31	4.32	4.63	5.46	7.18
	`ScalarMUL` with formulas	3.36	2.32	1.89	1.81	2.08	2.79	4.46
283	Sliding window	9.60	8.56	8.08	7.98	8.09	8.59	9.89
	`ScalarMUL` without formulas	9.48	8.49	8.10	8.24	8.71	9.88	12.6
	`ScalarMUL` with formulas	6.44	4.46	3.63	3.40	3.62	4.85	7.47
409	Sliding window	29.97	26.57	24.93	24.30	24.14	25.24	27.96
	`ScalarMUL` without formulas	29.54	26.31	24.98	24.80	25.71	28.44	34.29
	`ScalarMUL` with formulas	19.98	13.64	10.71	9.62	9.96	12.01	17.33
571	Sliding window	74.58	66.62	63.20	61.38	60.52	61.29	66.18
	`ScalarMUL` without formulas	74.17	67.25	62.87	61.56	61.61	65.59	76.50
	`ScalarMUL` with formulas	50.36	33.67	26.29	22.74	21.91	25.40	34.24

Over $GF(2^m)$, given a point Q, word length w, and setting $n = m$, Figure 4 illustrates the advantages of the `PDNTimes` algorithm in reducing the number of inverse operations required. In line 7 of the `ScalarMUL` algorithm, the operation $Q \leftarrow 2^w Q$ requires computing $2^w Q$, which involves performing w consecutive point doublings on the point Q. We compare the performances based on the number of multiplication operations required. In finite fields, the performance is largely determined via the inverse operations, as they require multiple multiplication operations to compute. The exact number of multiplications depends on the algorithm used. For example, if the Extended Euclidean Algorithm is used, an inverse operation generally takes about $2m$ to $3m$ multiplications, depending on the

implementation's optimization. The exact number of multiplications required for an inverse operation using Fermat's Little Theorem is $m-2$. In line 13 of the `PDNTimes` algorithm, we utilize Fermat's Little Theorem to compute the inverse t'_n. For the `PDNTimes` algorithm, $(m-2)+6w$ multiplication operations are required. If we replace the computation of $Q \leftarrow 2^w Q$ in line 7 of `PDNTimes` with Equation (2) (in line 4 of `Tradition`) to compute $2^w Q$, we will require $w(m-2)+2w$ multiplication operations.

Table 9. For $m = 163, 233, 283, 409, 571$, on $w = 2, 3, \ldots, 8$ over $GF(2^m)$, the decreasing ratio for the `ScalarMUL` algorithm with formulas for the sliding window method [21] and the sliding window method with formulas to the sliding window method.

m	Algorithm	w						
		2	**3**	**4**	**5**	**6**	**7**	**8**
163	`ScalarMUL`	31	45	50	51	44	27	0.5
	sliding window	33	47	55	58	57	52	43
233	`ScalarMUL`	34	49	55	56	51	39	18
	sliding window	34	49	56	60	61	57	49
283	`ScalarMUL`	33	48	55	57	55	44	24
	sliding window	34	48	58	62	64	61	54
409	`ScalarMUL`	33	49	57	60	59	52	38
	sliding window	33	49	58	64	65	65	59
571	`ScalarMUL`	32	49	58	63	64	59	48
	sliding window	32	50	58	64	67	68	64

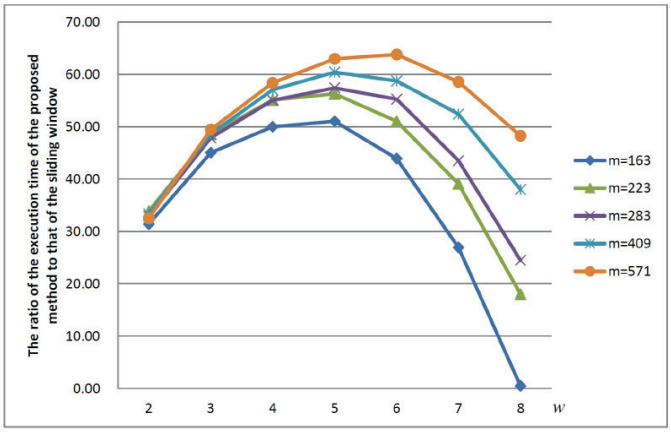


Figure 3. The decreasing behavior based on the data shown in Table 9.

In the affine coordinate system, both point addition and point doubling require one inverse to compute the slope λ. Additionally, each operation involves five multiplications, as follows:

- Two multiplications for calculating λ;
- Two multiplications for determining the new x-coordinate;
- One multiplication for determining the new y-coordinate.

Although our algorithm demonstrates reduced time complexity compared with the sliding window method, as shown in Table 10, its practical execution requires the construction of a larger lookup table. As a result, while our approach still outperforms the sliding window method in terms of efficiency, the performance gap is not as significant as indicated in Table 10.

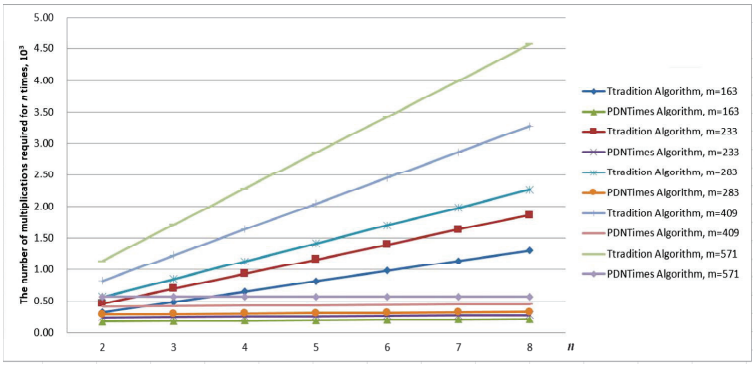


Figure 4. Comparison of the number of multiplications required for n iterations in `Tradition` and `PDNTimes`.

Table 10. Summary of the number of operations required for scalar multiplication over $GF(2^m)$ in the affine coordinate system.

Algorithm	Multiplications	Inverse Operations
Double and Add [22]	$7.5m$	$\frac{3m}{2}$
Sliding Window	$5m + \frac{5m}{w+1}$	$m + \frac{m}{w+1}$
Montgomery Ladder [23]	$10m$	$2m$
The proposed methods	$m - 2 + 6w$	1

5. Conclusions

In this work, we focused on significantly reducing the computation time of scalar multiplication, which can be easily implemented in software, by further expanding the application of Horner's rule and optimizing the square operations, specifically, through the introduction of several formulas for the inverse operations involved in repeating point doubling.

In elliptic curve cryptography and other cryptographic protocols, scalar multiplication is a critical operation that can be computationally expensive, primarily due to the repeated use of inverses and point doubling, which are key to optimizing efficiency.

The introduced formulas can help to minimize the number of inverse operations needed, thereby streamlining the computational process. Computation using the introduced formulas for λ_n and x_n requires more multiplication, square, and addition operations. We also developed the `ScalarMod` algorithm to reduce the computation time for square operations.

Figure 1 demonstrates that if the `ScalarMul` algorithm does not optimize for square operations, the overall reduction in computation time begins to plateau when the word length w reaches 8. This highlights the importance of optimizing square operations to achieve consistent performance improvements. On the other hand, Figure 2 illustrates that while the `Square` algorithm optimizes square operations, a trade-off must be made between execution time and the required memory size. These two phenomena represent key challenges that we aim to overcome and improve upon in future work.

From a theoretical perspective, analyzing the trade-off between execution time and memory usage is an intriguing research topic and a promising direction for future exploration. Understanding this balance could lead to more efficient algorithms that are both fast and resource efficient, making them suitable for a wider range of applications, including resource-constrained environments such as embedded systems and IoT devices.

On the other hand, an important consideration lies in the potential trade-offs between security and implementation complexity. While the primary focus of our work was to

reduce the number of inverse operations in scalar multiplication—a critical bottleneck in ECC—any optimization technique must be carefully assessed for its impact on both security and practical implementation.

- Security Considerations
  Our proposed method is based on well-established mathematical principles and does not introduce new assumptions or structures that could weaken the cryptographic security of the system. The repeating point-doubling formulas and grouping technique are derived directly from the affine coordinate system, ensuring that the underlying security properties of the elliptic curve are preserved. However, we recognize that side-channel attacks (e.g., timing or power analysis) could still pose a risk, as with any cryptographic implementation. While our current work does not explicitly address side-channel resistance, we plan to investigate this aspect in future research, potentially integrating countermeasures such as constant-time execution or masking techniques.
- Implementation Complexity
  The proposed method is designed to be simple and consistent in execution, making it suitable for both software and hardware implementations. The grouping technique and modified Horner's rule introduce minimal overhead in terms of precomputation and memory usage, as the bit-words and repeated point-doubling results can be efficiently stored and reused. Our approach achieves faster scalar multiplication with a comparable level of implementation complexity in comparison with traditional methods like the sliding window algorithm. That said, we acknowledge that further evaluation is needed to assess its performance in highly resource-constrained environments, such as IoT devices or embedded systems.
- Future Work
  While our initial results demonstrate significant improvements in computational efficiency, we agree that a more comprehensive evaluation of security and implementation complexity is essential. Future work will involve the following:
  1. A thorough security analysis, including resistance to side-channel attacks;
  2. Evaluation of the performance of methods in a wider range of hardware and software environments; particularly in resource-constrained settings.
  3. Comparison of the proposed method with other state-of-the-art techniques to identify potential trade-offs and optimize its practical applicability.

Finally, the formulas we derived are completely independent of B in the elliptic curve equation $E(x, y) : y^2 + xy = x^3 + Ax^2 + B$. This independence simplifies the application of our formulas across different elliptic curves. However, we are also curious whether it is possible to derive formulas that are independent of the parameter A in the equation. Exploring this possibility could lead to even more generalized and versatile results, potentially opening new avenues for optimization in elliptic curve cryptography. Such advancements could further enhance the efficiency and applicability of cryptographic protocols in real-world scenarios.

Author Contributions: Conceptualization, F.-J.K., Y.-H.C. and J.-J.W.; methodology, J.-J.W.; software, F.-J.K. and Y.-H.C.; validation, C.-D.L. and J.-J.W.; formal analysis, Y.-H.C.; investigation, C.-D.L.; resources, C.-D.L.; data curation, F.-J.K. and Y.-H.C.; writing—original draft preparation, J.-J.W.; writing—review and editing, J.-J.W.; visualization, F.-J.K. and Y.-H.C.; supervision, J.-J.W.; project administration, F.-J.K., Y.-H.C., C.-D.L. and J.-J.W.; funding acquisition, F.-J.K. and Y.-H.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSTC grant numbers 113-2221-E-214-021 and 113-2221-E-214-017, which may include administrative and technical support.

Data Availability Statement: Data are contained within the article.

Acknowledgments: We sincerely thank the National Science and Technology Council (NSTC) projects 113-2221-E-214-021 and 113-2221-E-214-017 for their funding and support. Heartfelt thanks to the reviewers for their suggestions and comments.

Conflicts of Interest: The authors declare that this study and its results were conducted independently, without any influence from financial, academic, or personal interests that could affect the outcomes. There are no direct or indirect conflicts of interest to disclose.

Appendix A

Table A1. Computations of λ_4 and x_4 for four times point doubling of Q_0 for $m = 233$.

$A =$ 0x1, primitive polynomial $f(x) = x^{233} + x^{74} + 1$, $Q_0 = (x_0, y_0)$	
$x_0 =$ 0x017232BA853A7E731AF129F22FF4149563A419C26BF50A4C9D6EEFAD6126	
$y_0 =$ 0x01DB537DECE819B7F70F555A67C427A8CD9BF18AEB9B56E0C11056FAE6A3	
t_1	0x17232BA853A7E731AF129F22FF4149563A419C26BF50A4C9D6EEFAD6126
v_1	0xC8EFD200D0B85E058BEB9366C7B9D7C0D0323D4A7084B2ABAE8EBB7B92
u_1	0x46B65EFCDD714E1FB3D046F17BFA928F300C397396A6D72A7BCEE4623E
t_2	0x1282331550168DE9B9630E825A5E58AB9A1D2D81F63051833AD8662D99C
v_2	0x277012D40FD385CB18ABFB705129D6A9709385C81D184AF636C800F25D
u_2	0x3E6B5DE2E8141083DAC00140C5936CD62A17ACE5620EEF8BD6763661FF
t_3	0xE7DC44EEB5FB14897829274A375A200B9D227AB7277745638B12045E3C
v_3	0x1D42DB6C4FC88A78613881210C5DCD474641567C546AFD60F1F3C70A52
u_3	0xC90CB3BA4AFEF4FC089394671D12533FA38AC99B369E16AE91D06E541C
t_4	0xD5E4BAC01DB2D0DDF4B5818595D13B649FE1C22CAC6AE6DFF91A267AB6
v_4	0x1F255D6098337C7B333913B56B4208769192550D64956C18F2DC35ABDF4
u_4	0x1DFAC578A9A7E1742AA21F99C4BD2233A785F011584EF8BD203D7899E0
t_4^{-1}	0x1FFF77FF7E8D784B371FB9F83CFD834653F1BA3F507898E3536808A7651
λ_4	0x66B6351AF207F92AA3F52AE7BDF78E1E6F8CDF51E918A8CB63AD38741A
x_4	0x30BD692E27C7A151D6CC09E18FEA36E6EB710B197A6A96D0840183BBAA

Table A2. Computations of λ_4 and x_4 for four times point doubling of Q_0 for $m = 283$.

$A =$ 0x1, primitive polynomial $f(x) = x^{283} + x^{12} + x^7 + x^5 + 1$, $Q_0 = (x_0, y_0)$	
$x_0 =$ 0x00FAC9DFCBAC8313BB2139F1BB755FEF65BC391F8B36F8F8EB7371FD558B	
$y_0 =$ 0x01006A08A41903350678E58528BEBF8A0BEFF867A7CA36716F7E01F8105	
t_1	0xFAC9DFCBAC8313BB2139F1BB755FEF65BC391F8B36F8F8EB7371FD558B
v_1	0x54515111155544512B1CF113BF10D7F74CDDDFA7FE4BAFCED7 D841A5294DD45E322F068
u_1	0xCE238EFFA4284AF0160D29B4F683E93BAC0F38BDC8B297AE78 B59107EA9443D30936D9
t_2	0x1CB0EDF1BB4103B60965BF4190FB920B757DDACEF61CC7603C0 E001ECE6278B3C48085E
v_2	0x17C10F70CF0C20F2ABECDA0639B1E878BD05271DF7A8FB00A42 3673F8426B106AF66A61
u_2	0x2A2FD15E5F8B390EAE362C83D0337A73D290A9FEBC241E5244D 8B6F32BDE4828821B7F9
t_3	0x1CF2B34F4C12286EB5EE15DDD1A49369CF15CDF44DD8B69C421 DCC278F977F68CF7B750
v_3	0x6BF096B1D4E3B3CCA95469A1794EF15B1AD97DADA461C6F350EA B6C32507AE181D9339C
u_3	0x1F1DA1BD7AAFBB0B823DBA1653B4A5D866F57845BE0099617B7 D2EDD3405B74A9F0B23B
t_4	0x245783E6BD266915C279EEAB5E9E657FFA5749E367E8E996655 72D234B01C95C5BC9E89
v_4	0x26FA03DF5515517EF2501202F38AF8C04B7F1F8773941DCCFD2 C38C54F35E0CA419A372
u_4	0x2195B07257881D6D374F60B424FC79F4229C30C8207EC6B07EDF E3C86232B41CED9F992
t_4^{-1}	0x1C161B5BF6C81EF2CC6E80280A98CC5CFD395F0EA246525B10A 930DCCC2734A208D49D9
λ_4	0x6553144C06D11F234DE640CF8CF399B2B85634FEFDE4089350B C151CBD12EC5306113EE
x_4	0x3A55D77017BA6EC0D6AF87B67F8C33B3F7661FD7D3FF5033DBB9523 F5625EBB78BF623F

Table A3. Computations of λ_4 and x_4 for four times point doubling of Q_0 for $m = 409$.

$A =$ 0x1, primitive polynomial $f(x) = x^{409} + x^{87} + 1$, $Q_0 = (x_0, y_0)$	
$x_0 =$	0x0060F05F658F49C1AD3AB1890F7184210EFD0987E307C84C27ACC FB8F9F67CC2C460189EB5AAAA62EE222EB1B35540CFE9023746
$y_0 =$	0x01E369050B7C4E42ACBA1DACBF04299C3460782F918EA427E632516 5E9EA10E3DA5F6C42E9C55215AA9CA27A5863EC48D8E0286B
t_1	0x60F05F658F49C1AD3AB1890F7184210EFD0987E307C84C27ACCFB8 F9F67CC2C460189EB5AAAA62EE222EB1B35540CFE9023746
v_1	0x1AC786F8F21F08DA00A37308FA9787E4DA69A59142AD5B7C8EC95C4E 0BD5522A561845C2DC240CBFFD1E788D8C28EEEC557F1AF
u_1	0x1026BFE44829CBA15B8C26B8E906F2241E47775A7C5D996AAA9AD28 88EC57CEDB82F6BB23EFD18F5F269C4D34984B7BA0B1F3CB
t_2	0xFC0EF81DA9679D4FC66DB292971CABBB552D78D6A48C67650940273 F4CB7957F8FAAE7F94D5DA38A03A74C6CA3111BE362DEAF
v_2	0x3B899DCBB8BE70261408872B757C476E5F89DE93E596B68B36ECD6EB 75649E82B6723804E3B2959FE7CE14F0E2DF1F8E9242AB
u_2	0x117EDD4AF7D8EF95B46D0DE4548C89B872F3D1A00198675B0490AFBEA BE3413E19237E92A1FC940F2289E9E3F2AE2BC69502DDB
t_3	0x7588FD6A097CF42FA6B4A8F8C315A33012989C406217A7FC23E034632B 43C0C8C8797EB2A2FD0643611B1E499858B2C9F8E9D8
v_3	0xA150FFB3B4919CD47A10A2AABA0486114E2BDB2C63FEDC14CD2B71695 BF91868E9533CBC63E811EC16BEEB8DF8A3941D2C551F
u_3	0x93682A817E4F27E418633D540CAC43A6952FABC521CBD6DC88A6EA6B1B4 B2CEC6C603276E6E3B267468E4A034134FBDF25EB4A
t_4	0xC0B0B103E854D8505C71151018952F6E4955B646F001C28ECA78B73E0 53E6E2B7BC849150054F0040D0C9F3204869ED4106EE5
v_4	0x18F980F54D4A5327DAB97A7DB060A75D44BBA63B6AE60E1E1DF3B8495 BD0D06304CA90EB77B145E5885ED59EFDD49BB25426A1E
u_4	0x1AF76EEA319ED639010848C7FC6F027FD701D8F2063348E0920BAF9AC EEBCC07033951B6FBF140957DB90DA12292F80B0819528
t_4^{-1}	0x1DF450CBAC4D70BBF94C8E5219AB0C775EE4F37CF033275682BEA7 8F7C25E2DB292A52B95F2B92FB1588AD285A7570570175A69
λ_4	0x1E8F34968A9C9C65B1D056D71ABCF13D93C2211550AB0F59FBE9 01756646108E70960C750069112300120BA1A1DE6A31D5FADBA
x_4	0x528673FF64BD082F3A60914056944B3BA99AC518D0D93F5F1CB3FB3DA0B 6F4579BC9C1125345DAE9BFCE973BC477747BA4CAF5

Table A4. Computations of λ_4 and x_4 for four times point doubling of Q_0 for $m = 571$.

$A =$ 0x1, primitive polynomial $f(x) = x^{571} + x^{10} + x^5 + x^2 + 1$, $Q_0 = (x_0, y_0)$	
$x_0 =$	0x026EB7A859923FBC82189631F8103FE4AC9CA2970012D5D46024804801 841CA44370958493B205E647DA304DB4CEB08CBBD1BA39494776FB988 B47174DCA88C7E2945283A01C8972
$y_0 =$	0x0349DC807F4FBF374F4AEADE3BCA95314DD58CEC9F307A54FFC61E FC006D8A2C9D4979C0AC44AEA74FBEBBB9F772AEDCB620B01A7BA7 AF1B320430C8591984F601CD4C143EF1C7A3
t_1	0x26EB7A859923FBC82189631F8103FE4AC9CA2970012D5D460248048018 41CA44370958493B205E647DA304DB4CEB08CBBD1BA39494776FB988B4 7174DCA88C7E2945283A01C8972
v_1	0x2BF4AB0A0654BCC72510BA7C97DE64A1AE0751E2026B571B207ED40B A71667E4E8D88ED0A7687C20E786092A0294F91246B0B76338CD70EC3803 B75A92F06BBD9314CE03131BCA0
u_1	0x2BCCA12217DE9277B0B2011E225EBA18027DDE7E54A78221DF115074 3866EE6BD3A301D14243961C0694AF2A124E2DF0889112E9D9809D 9BAE9B7B41AFA4C39C7E33100BC1E6A6E
t_2	0x16E7B7EF519ADF86BF01ED25CCFC6CABD4933D1BFEF9B6ADE7818 AFB872580F2C0A2D07A5533568596888DCAFCA4C627C14697BCC4BD599 F40F62C1952916F4B20C9943FE59ECA7
v_2	0x13BEE3A1BF46B5DEBD2D827F158FB4205CDBBD0B37670FD4D249C C9776C6E7475D4C58ECB7003E1464AA655B176564DF251B223642D965E 546EA2028A35700AC5A1CE1C25833E20
u_2	0xE19CBC6A4D57A6E1465C2A9E87F34207EC3C4FE70B69D1B1A83CD 55E6A02D8978215F4AD2BBFB14BD9F444A2FB169502D8114D47D9FE 4582FA470F1EA7CF73700D7D66EC5FACE6

Table A4. *Cont.*

t_3	0x705810F304A19E653B1DB8A1451F3F6296CB174243A86AFDDA06C1E74 62EF1D3FE6AE540FA775BF61E2B5D4B3CC5C7E77818B24A1E88BF3CA 43C793F358BFFF6DD70292113EBA0D
v_3	0x5BCF313A3AFC4D794C9D0366461F019BC343BC25AF970EBC81E3CD B42B4E221C771B70C4B76D89DE5472FBB67973B22EA76112AD3F63A8F D0DB845970466D1401CE97EFAED1906C
u_3	0x3768E085616E1041183FC92AB605B4D66A5906561BB66AD2283DC6B 2BC8026699AF02C9B9996ED727B1B5E2DBBE62D6C5923A33205D23A011 693DE482988480ECC227E76710AB9
t_4	0x12775DC1600EBA8175A61AC35380F29868603C6803BD2F25FB5ABBEF 3C34E67EA50E983E1A265C3FBF30BCD1817B98A9F24AA1B18E04423B5 018A73710941EDEE3494B316CDBCD
v_4	0x16E89D4D47B7DDAAFC8F25CDD200F0FF3DAC8D687E17325C2594566 BAD586676E6E138D5A352DDB278D9D86BF1BDBA1A8E72D18C9F5E0226 10340AA8055B9CD03CF94312FFC215C
u_4	0xED089CC8CF79B26383A0082FE34EA885C6FF7EC123FAE8D8C3178 AF2792318011E71377D481BB784EE048DE9C0309AB1936ADA2A60C19 DA67C6663F3DEF1D61740F0D5E1F76883
t_4^{-1}	0x253E10151FF41A3EA108024F484D4C65AB81A3E49901BD2DC858F63C 87C865A28737A9BE47407ABD3166C39915E445AAB5B902B1009DB20E37 0A47F02EF03D29E5C071C8089D50F
λ_4	0x500477AFFF704DE6EF4846F7F4CAA9E48DB443466E6F8C2B85F1A75 2A31110DBC30E2491C17F308B248A57CC5E31794BBD7F2915B243053C 65045830F12D50581BA869AF7F09D24
x_4	0x2F04E2F7C2D35C1D42E68075890653DC3B65B112780C70521590A79E4 3288E7ACB0F03B5189825F11A64729F492668EBB67A7129A61DCD33E47 A4E36B8F51769439D8E82C4E77C8

Table A5. Recommended elliptic curve domain parameters over $GF(2^m)$.

$A = 0x1, m = 163, f(x) = x^{163} + x^7 + x^6 + x^3 + 1, f(x) + x^m = 0xc9$	
x_0	0x02FE13C0537BBC11ACAA07D793DE4E6D5E5C94EEE8
y_0	0x0289070FB05D38FF58321F2E800536D538CCDAA3D9
$A = 0x1, m = 233, f(x) = x^{233} + x^{74} + 1, f(x) + x^m = 0x4000000000000000001$	
x_0	0x017232BA853A7E731AF129F22FF4149563A419C26BF50A4C9D6EEFAD6126
y_0	0x01DB537DECE819B7F70F555A67C427A8CD9BF18AEB9B56E0C11056FAE6A3
$A = 0x1, m = 283, f(x) = x^{283} + x^{12} + x^7 + x^5 + 1, f(x) + x^m = 0x10a1$	
x_0	0x00FAC9DFCBAC8313BB2139F1BB755FEF65BC391F8B36F8F8EB7371FD558B
y_0	0x01006A08A41903350678E58528BEBF8A0BEFF867A7CA36716F7E01F8105
$A = 0x1, m = 409, f(x) = x^{409} + x^{87} + 1, f(x) + x^m = 0x20000000000000000000001$	
x_0	0x0060F05F658F49C1AD3AB1890F7184210EFD0987E307C84C27ACCFB8F9F67C C2C460189EB5AAAA62EE222EB1B35540CFE9023746
y_0	0x01E369050B7C4E42ACBA1DACBF04299C3460782F918EA427E6325165E9EA10E 3DA5F6C42E9C55215AA9CA27A5863EC48D8E0286B
$A = 0x1, m = 571, f(x) = x^{571} + x^{10} + x^5 + x^2 + 1, f(x) + x^m = 0x425$	
x_0	0x026EB7A859923FBC82189631F8103FE4AC9CA2970012D5D46024804801841CA4 4370958493B205E647DA304DB4CEB08CBBD1BA39494776FB988B47174DCA88C 7E2945283A01C8972
y_0	0x0349DC807F4FBF374F4AEADE3BCA95314DD58CEC9F307A54FFC61EFC006D 8A2C9D4979C0AC44AEA74FBEBBB9F772AEDCB620B01A7BA7AF1B320430C85 91984F601CD4C143EF1C7A3

References

1. Miller, V. Uses of elliptic curves in cryptography. In *Advances in Cryptology: Proceedings of Crypto'85*; Springer: Berlin/Heidelberg, Germany, 1986.
2. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209. [CrossRef]

3. Wang, C.C.; Truong, T.K.; Shao, H.M.; Deutsch, L.J.; Omura, J.K.; Reed, I.S. VLSI architectures for computing multiplications and inverses in GF(2^m). *IEEE Trans. Comput.* **1985**, *C-34*, 709–717. [CrossRef] [PubMed]
4. Bernstein, D.J. Batch Binary Edwards. In *Advances in Cryptology—CRYPTO 2009*; Halevi, S., Ed.; Springer: Berlin/Heidelberg, Germany, 2009; Volume 5677, pp. 317–333.
5. Chen, Y.H.; Huang, C.H. Efficient operations in large finite fields for elliptic curve cryptographic. *Int. J. Eng. Technol. Manag. Res.* **2020**, *7*, 141–151. [CrossRef]
6. Blake, F.; Murty, V.K.; Xu, G. A note on window $tau-$NAF algorithm. *Inf. Process. Lett.* **2005** *95*, 496–502. [CrossRef]
7. Al Saffar, N.F.H.; Said, M.R.M. High performance methods of elliptic curve scalar multiplication. *Int. J. Comput. Appl.* **2014**, *108*, 39–45. [CrossRef]
8. Pathak, H.K.; Sanghi, M. Speeding up computation of scalar multiplication in elliptic curve cryptosystem. *Int. J. Comput. Sci. Eng.* **2010**, *2*, 1024–1028.
9. Eid, W.; Turki, F.A.; Marius, C.S. Efficient elliptic curve operators for Jacobian coordinates. *Electonics* **2022**, *11*, 3123. [CrossRef]
10. Al Musa, S.; Xu, G. Fast scalar multiplication for elliptic curves over binary fields by efficiently computable formulas. In *Progress in Cryptology—INDOCRYPT 2017*; Springer: Cham, Switzerland, 2017.
11. Li, J.; Zhong, S.; Li, Z.; Cao, S.; Zhang, J.; Wang, W. Speed-oriented architecture for binary field point multiplication on elliptic curves. *IEEE Access* **2019**, *7*, 32048–32060. [CrossRef]
12. Li, J.; Wang, W.; Zhang, J.; Luo, Y.; Ren, S. Innovative dual-binary-field architecture for point multiplication of elliptic curve cryptography. *IEEE Access* **2021**, *9*, 12405–12419. [CrossRef]
13. Oudjida, A.K.; Liacha, A. Radix-2^w arithmetic for scalar multiplication in elliptic curve cryptography. *IEEE Trans. Circuits Syst. I Reg. Pap.* **2021**, *68*, 1979–1989. [CrossRef]
14. Bernstein, D.J.; Lange, T. Analysis and optimization of elliptic-curve single-scalar multiplication. In Proceedings of the Eighth International Conference on Finite Fields and Applications, Melbourne, Australia, 9–13 July 2007; pp. 1–20.
15. Ning, Y.D.; Chen, Y.H.; Shih, C.S.; Chu, S.I. Lookup table-based design of scalar multiplication for elliptic curve cryptography. *Cryptography* **2024**, *8*, 11. [CrossRef]
16. Cho, S.M.; Gwak, S.G.; Kim, C.H.; Hong, S. Faster elliptic curve arithmetic for triple-base chain by reordering sequences of field operations. *Multimed. Tools Appl.* **2016**, *75*, 14819–14831. [CrossRef]
17. Zhang, J.; Chen, Z.; Ma, M.; Jiang, R.; Li, H.; Wang, W. High-performance ECC scalar multiplication architecture based on comb method and low-latency window recoding algorithm. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2024**, *32*, 382–395. [CrossRef]
18. Matteo, S.D.; Baldanzi, L.; Crocetti, L.; Nannipieri, P.; Fanucci, L.; Saponara, S. Secure elliptic curve crypto-processor for real-time IoT applications. *Energies* **2021**, *14*, 4676. [CrossRef]
19. Pillutla, S.R.; Boppana, L. A high-throughput fully digit-serial polynomial basis finite field GF(2^m) multiplier for IoT applications. In Proceedings of the IEEE Region 10 International Conference (TENCON2019), Kochi, India, 17–20 October 2019; pp. 920–924.
20. Sabbry, N.H.; Levina, A.B. An optimized point multiplication strategy in elliptic curve cryptography for resource-constrained devices. *Mathematics* **2024**, *12*, 881. [CrossRef]
21. Shah, P.G.; Huang, X.; Sharma, D. Sliding window method with flexible window size for scalar multiplication on wireless sensor network nodes. In Proceedings of the International Conference on Wireless Communication and Sensor Computing (ICWCSC), Chennai, India, 2–4 January 2010; pp. 1–6.
22. Darrel, H.; Scott, V.; Alfred, M. *Guide to Elliptic Curve Cryptography*; Springer: New York, NY, USA, 2004.
23. Montgomery, P.L. Speeding the Pollard and elliptic curve methods of factorization. *Math. Comput.* **1987**, *48*, 243–264. [CrossRef]

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

mathematics

MDPI

Article

A Model of Effector–Tumor Cell Interactions Under Chemotherapy: Bifurcation Analysis

Rubayyi T. Alqahtani

Department of Mathematics and Statistics, College of Science, Imam Mohammad Ibn Saud Islamic University, Riyadh, Saudi Arabia; rtalqahtani@imamu.edu.sa

Abstract: This paper studies the dynamic behavior of a three-dimensional mathematical model of effector–tumor cell interactions that incorporates the impact of chemotherapy. The well-known logistic function is used to model tumor growth. Elementary concepts of singularity theory are used to classify the model steady-state equilibria. I show that the model can predict hysteresis, isola/mushroom, and pitchfork singularities. Useful branch sets in terms of model parameters are constructed to delineate the domains of such singularities. I examine the effect of chemotherapy on bifurcation solutions, and I discuss the efficiency of chemotherapy treatment. I also show that the model cannot predict a periodic behavior for any model parameters.

Keywords: cancer; effectors–tumor cells; chemotherapy; singularity; bifurcation; periodic; hysteresis; isola/mushroom; pitchfork

MSC: 37M05; 92C50

1. Introduction

Cancer is the primary cause of death worldwide and is regarded as a costly medical condition, particularly in developing nations. According to data from the World Health Organization, there were 20 million new cases of cancer globally in 2022 alone, accounting for 10 million deaths [1].

Quantitative models are commonly used in the field of mathematical oncology [2–4] to forecast tumor growth and treatment response. In cancer research, mathematical oncology has proven to be extremely helpful. Through it, we have been able to better understand the underlying biological interactions between tumors and effector cells [5], personalize cancer treatments [6], and understand drug efficiency and resistance [7,8].

Numerous mathematical models explaining the interactions between effectors–tumor cells have been proposed and investigated in the literature [9–22]. Many of these models were formulated based on the interactions between predators and preys [9–11,19–22]. Indeed, the relationship between cytotoxic immune cells and tumor cells can be compared to the dynamics observed in predator–prey interactions. Once activated, immune cells take on the role of predators, actively hunting for cells that exhibit their respective antigens, which are perceived as their prey. Upon identification of a target, these immune cells attach to and destroy the target cell. The comparison to predator–prey dynamics provides a framework for examining the variations in tumor cell populations during the immunoediting process, along with the immune system's responses to these changes [19,21,22].

However, this analogy does not completely convey the reality of the situation. Several assumptions and outcomes associated with predator–prey models are not observable in the interactions that take place between tumor cells and immune cells. First, classical predator–prey models assume that the biomass of prey consumed is directly transformed into the biomass of the predator. Although the predator depends on the prey for its survival, tumor cells and immune cells are in competition for critical shared resources, such as glucose and amino acids. Immune cells are required to compete with cancer cells for available resources, yet they do not gain any direct metabolic advantage from effectively targeting these malignant cells [22].

The second key area where the analogy between predator–prey relationships and tumor–immune interactions diverges is linked to the manifestation of oscillations. Continuous fluctuations in the populations of predators and preys are a natural aspect of predator–prey relationships [19,23], yet this type of oscillatory behavior has not been observed in interactions involving tumor and immune cells [19].

In another regard, the complexity of mathematical models that describe the interactions between tumors and immune cells can differ based on the particular types of immune cells involved, including CD8+ "killer" (cytotoxic) T cells and CD4+ "helper" T cells, among others [24]. However, it is recognized that the essential elements in the interactions between tumors and immune cells should include cancer cells, activated effector (cytotoxic) cells, and antigen-presenting cells (APCs) [19]. In numerous instances [9,19,20], it is reasonable to presume that the engagement with antigen-presenting cells, along with the ensuing activation of T cells, attains a quasi-steady state prior to influencing the dynamics of cancer cells. This enables the depiction of the interaction between immune cells and cancer cells through the use of just two populations: one representing cytotoxic cells and the other representing cancer cells, thus forming a predator–prey community model.

Some of the aforementioned research on mathematical modeling of tumor–immune interactions has specifically concentrated on the analysis of steady-state multiplicity occurrences within these interactions [9,20,25–28]. Notably, Kuznetsov et al. [9] put forth one of the earliest and most basic predator–prey models to explain the occurrence of multiple equilibria in tumor–immune cell interactions. Only two different cell types made up the model: effector cells, which were the predator, and tumor cells, which were the prey. The existence of "dormant cells", or regions with low concentrations of tumor cells, "active cells", and regions of coexistence, or domains where "dormant cells" can elude effector regulation and become active, were all predicted by the model [9]. De Pillis and Radunskaya [25] subsequently examined a model of tumor and immune cells that was experimentally validated, and they demonstrated the presence of bistability between the disease-free equilibrium and the unhealthy steady state. López et al. [26], on the other hand, formulated and examined a model of tumor–immune cell interactions under chemotherapy, demonstrating consistency with experimental data. The authors showed the existence of bistability between the disease-free state and the malignant state through a number of bifurcations mechanism such as saddle node and transcritical bifurcations. Recently, Bashkirtseva et al. [20] added the effect of chemotherapy treatment to the system examined in [9]. The authors uncovered steady-state multiplicity as well as periodic behavior in the studied model.

The aforementioned research applied numerical methods, specifically continuation techniques [29], to construct bifurcation diagrams that represent the dependence of the model state variables on a designated system parameter. These techniques, while advantageous, are constrained in their ability to deliver a full representation of all branching phenomena (singularities) that the model is capable of exhibiting. This is particularly true

when the model comprises a substantial number of parameters. In contrast, the singularity theory [30] is a considerably more effective mathematical tool for examining bifurcation solutions. The theory offers a structured approach to ascertain the number of topologically unique bifurcation diagrams present in a nonlinear dynamic system. Additionally, it facilitates the division of the model's multidimensional parameter space into distinct regions, with each corresponding to various types of bifurcation diagrams. This information serves to classify control parameters, which are vital in shaping system dynamics by managing transitions between various bifurcation diagrams. By condensing the steady-state equations of the model into a singular function, it becomes possible to identify the properties of numerous solutions of a bifurcation equation by investigating a number of derivatives associated with the singular function. Examples of the application of the theory in chemical reactors and bioreactors can be found in [31,32].

The motivation behind this study stems from the inquiry into whether a simple and classical model [9] of tumor–immune interactions to which we add chemotherapy effects can yield more intriguing dynamics than those previously documented in the literature [9,16,20,25–28]. For this purpose, I sought to provide a general framework for the analysis of bifurcation behavior in the model using elementary concepts of the singularity theory. The relative simplicity of the model allows for a description of the steady-state equilibria of the system in the form of a single nonlinear algebraic equation. The singularity theory can thus serve as an effective instrument for categorizing the various branching phenomena within the model. To my knowledge, no such analysis was used before for tumor–immune cell interaction models. I examine the existence of basic singularities such as hysteresis, isola/mushroom, and pitchfork singularities within the model. Additionally, I analyze the effect of the model's biological parameters and those associated with chemotherapy on these bifurcation solutions.

The second aim of this paper is to conduct an analytical examination of the model's capacity to forecast periodic behavior. I have successfully established general and noteworthy conditions for the presence of Hopf points within the model. The rest of the paper is organized as follows. In the next section, the model is presented, followed in Section 3 by the analysis of model equilibria. In Section 4, static analysis is carried out, followed by dynamic analysis in Section 5. Numerical simulations are carried out in Section 6, followed by the discussion and conclusions in the last sections.

2. The Mathematical Model

The model, based on the work of [9] and in which chemotherapy terms were added, consists of two types of cells: effector cells E (predator) and tumor cells T (prey). The equations of the model are the following:

$$\frac{dE}{dt} = s + \frac{pET}{g+T} - mET - dE - k_E ME. \tag{1}$$

$$\frac{dT}{dt} = \alpha T(1-\beta T) - nET - k_T MT. \tag{2}$$

$$\frac{dM}{dt} = -\gamma M + v. \tag{3}$$

The concentrations of effector and tumor cells are denoted by E (cells) and T (cells), respectively, while the concentration of chemotherapy drug is denoted by M (mg/m^2). The effector cells have a normal growth rate of s (cells/day) and a constant death rate of d (1/day). The decay of E cells as a result of their interactions with tumor cells is represented by the term mET, and it occurs at a rate of m (1/cells.day). Drugs used in

chemotherapy also kill effector cells at a rate of k_E (1/day). The Michaelis–Menten growth of effector cells in response to tumor cells is represented by the term $\frac{pET}{g+T}$, where g (cells) and p (1/day) are the parameters of the growth rate.

It is assumed in (Equation (2)) that tumor cells increase in accordance with the logistic function, where the model's coefficients for the isolated population of tumor cells are α (1/day) and β (1/cells). Tumor cell reduction owing to effector cells presence is denoted by the term nET, whereas tumor cell lysis is denoted by n (1/cells.day). Additionally, tumor cells are killed by chemotherapy at a rate of k_T (1/day).

The third equation (Equation (3)) represents the change in concentration of chemotherapy drug over time, with γ (1/day) being the rate of elimination of the drug from the body and v (mg/m^2.day) being the amount of drug administered to the body.

A note should be added regarding the selection of the tumor growth model. Various such models were proposed and scrutinized in the literature [12,14]. These include linear, logistic, Mendelsohn, exponential, Gompertz, Surface, and Bertalanffy models [12,14]. It is generally accepted that the selection of a suitable growth model is strongly dependent on the specific type of tumors involved [12,14]. In the context of general mathematical analysis, akin to the methodology employed in this paper, the literature, as referenced in [9,16,20,25–28], has predominantly favored the logistic growth rate. This preference is primarily due to the mathematical convenience offered by the logistic function when compared to other tumor growth models.

The following variables are used to render the model dimensionless:

$$\bar{E} = \frac{E}{E_0}, \bar{T} = \frac{T}{T_0}, \bar{M} = \frac{M}{M_0}, \bar{s} = \frac{s}{nE_0T_0}, \bar{p} = \frac{p}{nT_0}, \bar{g} = \frac{g}{T_0}, \bar{m} = \frac{m}{n}, \bar{d} = \frac{d}{nT_0}. \tag{4}$$

$$\bar{k}_E = \frac{k_E}{nT_0}, \bar{k}_T = \frac{k_T}{nT_0}, \bar{\alpha} = \frac{\alpha}{nT_0}, \bar{\beta} = \beta T_0, \bar{v} = \frac{v}{nT_0M_0}, \bar{\gamma} = \frac{\gamma}{nT_0}, \bar{t} = nT_0t. \tag{5}$$

E_0, T_0, and M_0 are reference concentrations for E, T, and M respectively.

The dimensionless model is

$$\frac{d\bar{E}}{d\bar{t}} = \bar{s} + \frac{\bar{p}\bar{E}\bar{T}}{\bar{g}+\bar{T}} - \bar{m}\bar{E}\bar{T} - \bar{d}\bar{E} - \bar{k}_E M_0 \bar{M}\bar{E}. \tag{6}$$

$$\frac{d\bar{T}}{d\bar{t}} = \bar{\alpha}\bar{T}(1-\bar{\beta}T) - \bar{E}\bar{T} - k_T M_0 \bar{M}\bar{T}. \tag{7}$$

$$\frac{d\bar{M}}{d\bar{t}} = -\bar{\gamma}\bar{M} + \bar{v}. \tag{8}$$

In the rest of this paper, the (bar) notation is dropped from all variables and parameters.

3. Analysis of Model Equilibria

The model always has a trivial steady-state solution obtained when $T = 0$, i.e.,

$$E = \frac{s}{d + k_E M_0 \frac{v}{\gamma}}, M = \frac{v}{\gamma}. \tag{9}$$

When $T \neq 0$, Equation (7) yields

$$E = \alpha(1-\beta T) - k_T M_0 \frac{v}{\gamma}. \tag{10}$$

Substituting Equation (10) into the steady-state form of Equation (6) yields the following cubic equation for T:

$$F := a_3T^3 + a_2T^2 + a_2T + a_0, \tag{11}$$

where

$$a_3 = \alpha\beta\gamma_1^2 m. \tag{12}$$

$$a_2 = \alpha\beta d\gamma^2 - \alpha\gamma^2 m + \alpha\beta g\gamma^2 m - \alpha\beta\gamma p + (\alpha\beta\gamma - k_E M_0 + \gamma k_T m M_0)v; \ (a_2 := a_{20} + a_{21}v). \tag{13}$$

$$a_1 = -\alpha d\gamma^2 + \alpha\beta dg\gamma^2 - \alpha g\gamma^2 m + \alpha\gamma^2 p + \gamma^2 s + (-\alpha\gamma k_E M_0 + \alpha\beta g\gamma k_E M_0 + d\gamma k_T M_0 + g\gamma k_T m M_0 - \gamma k_T M_0 p)v + k_E k_T M_0^2 v^2; \ (a_1 = a_{10} + a_{11}v + a_{12}v^2). \tag{14}$$

$$a_0 = g(-\alpha d\gamma^2 + \gamma^2 s + (-\alpha\gamma k_E M_0 + d\gamma k_T M_0)v + k_E k_T M_0^2 v^2); \ (a_0 := g(a_{00} + a_{01}v + a_{02}v^2)). \tag{15}$$

The coefficient a_3 is always positive, and the number of possible positive solutions of Equation (11) can be determined using Descartes rule, as shown in Table 1.

Table 1. Number of positive roots of Equations (11)–(15).

Case	a_3	a_2	a_1	a_0	Number of Sign Changes	Number of Positive Roots
1	+	+	+	+	0	0
2	+	+	+	−	1	1
3	+	+	−	+	2	2, 0
4	+	+	−	−	1	1
5	+	−	+	+	2	2, 0
6	+	−	+	−	3	3, 1
7	+	−	−	+	2	2, 0
8	+	−	−	−	1	1

Moreover, it can be seen from Equation (11) that the nontrivial steady state crosses the trivial steady state ($T = 0$) when $a_0 = 0$. The quadratic equation of $a_0 = 0$ in terms of v (Equation (15)) makes it possible to analytically solve for the critical value v_c where the two steady states cross. Beyond this critical value v_c, the tumor is completely suppressed.

4. Static Analysis

We start by carrying out a steady-state analysis of the system. The steady-states equations of the model were conveniently reduced to a single nonlinear equation in T (Equation (11)). The singularity theory can therefore be readily applied to analyze the system. The chemotherapy dose (v) is the most convenient parameter to vary and is selected as the main bifurcation parameter. The steady-state equation (Equation (11)) is cubic in T. Within the framework of this equation, singularity theory delineates two forms of codimension-one singularities: hysteresis, which describes the development of an isola that consists of a closed locus of a solution branch bordered by two fold points, and the evolution of this isola into mushroom singularities. Additionally, a pitchfork singularity, recognized as codimension two, is defined for a cubic single-scalar function. The fundamental singularities are depicted in Figure 1a–d. It is essential to note that even minor variations in model parameters can lead to the disintegration of the perfect pitchfork, resulting in the appearance of four additional bifurcation patterns, as illustrated in Figure 1e.

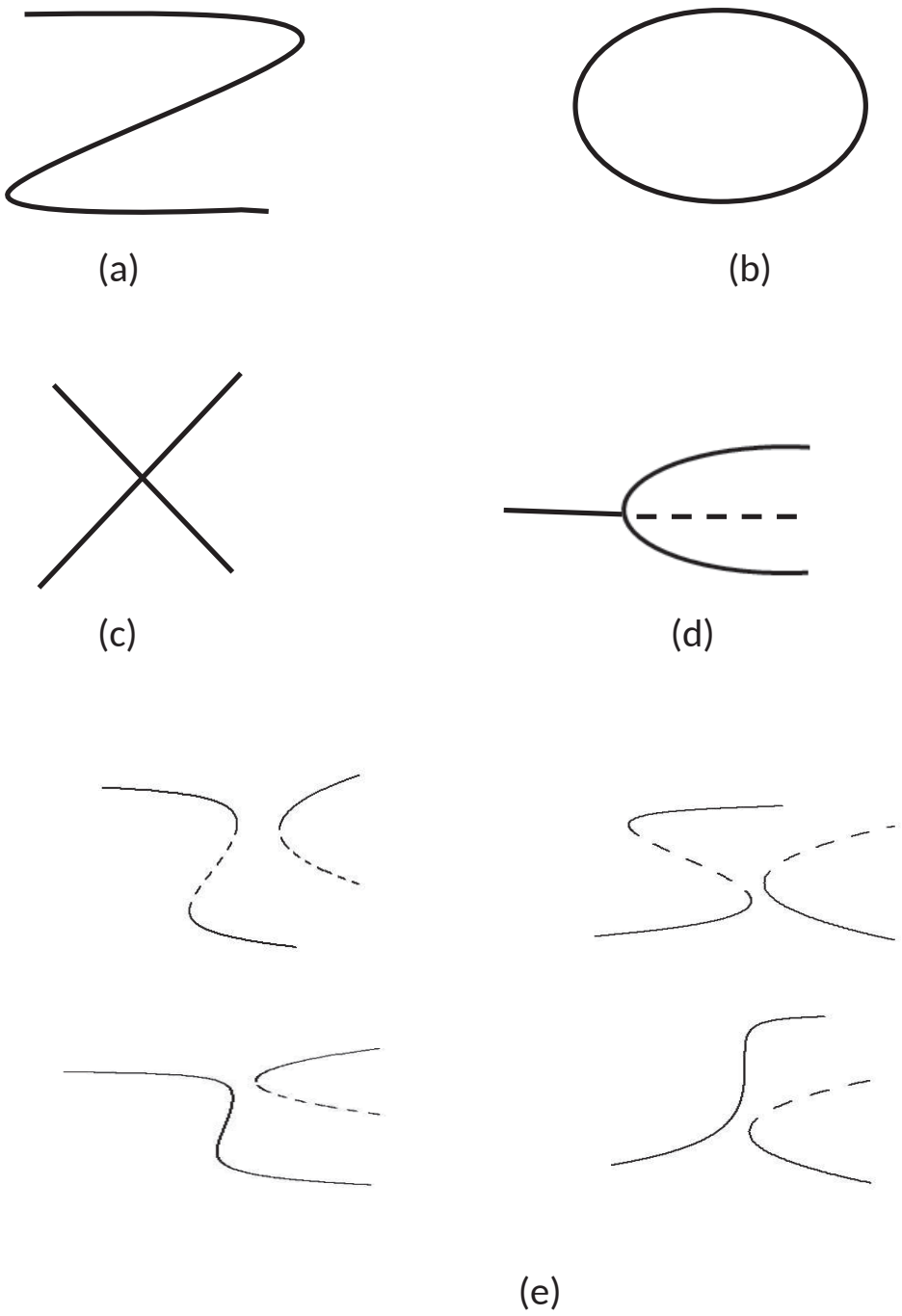


Figure 1. Basic singularities: (**a**) hysteresis; (**b**) isoal; (**c**) mushroom; (**d**) perfect pitchfork; (**e**) perturbed bifurcation diagrams for the pitchfork.

4.1. Hysteresis Singularity

The conditions for the appearance/disappearance of a hysteresis loop are the following:

$$F = F_T = F_{TT} = 0. \tag{16}$$

In addition, a number of other derivatives must remain nonzero, namely, F_v, F_{Tv}, and F_{TTT}. The hysteresis conditions for the system are the following:

$$F = a_3T^3 + a_2T^2 + a_1T + a_0 = 0. \tag{17}$$

$$F_T = 3a_3T^2 + 2a_2T + a_1 = 0. \tag{18}$$

$$F_{TT} = 6a_3T + 2a_2 = 0. \tag{19}$$

Equation (19) has one solution that is $T = -\frac{a_2}{3a_3}$. Substituting this solution in Equation (17) and in Equation (18) yields the following relations for the hysteresis singularity:

$$\frac{a_2^2}{3a_3} = a_1. \tag{20}$$

$$\frac{a_2^3}{27a_3^2} = -1. \tag{21}$$

These two equations are also equivalent to $a_2 = -3a_3^{\frac{2}{3}}$ and $a_1 = 3a_3^{\frac{1}{3}}$. Recasting the expressions of a_2 and a_1 from Equations (13) and (14) and using the last two equations yields the following two relations:

$$v = \frac{-3a_3^{\frac{2}{3}} - a_{20}}{a_{21}}. \tag{22}$$

$$a_{10} + a_{11}v + a_{12}v^2 = 3a_3^{\frac{1}{3}}. \tag{23}$$

These two equations form the hysteresis boundary. It remains to check that the other derivatives F_v, F_{Tv}, and F_{TTT} at these conditions remain nonzero. It can be noted that F_{TTT} cannot vanish for any values of strictly positive model parameters, since $F_{TTT} = 6a_3 \neq 0$. The rest of conditions will be evaluated numerically along the boundary.

4.2. Isola/Mushroom Singularity

The second possible qualitative change that can occur in the steady-state locus is the appearance of an isola and the growth of an isola into a mushroom. The requirements for these two changes are that

$$F = F_T = F_v = 0, \tag{24}$$

with the additional requirements that

$$F_{Tv} \neq 0, F_{TT} \neq 0, F_{vv} \neq 0. \tag{25}$$

The expression for F_v (Equations (11)–(15)) is

$$F_v = a_{21}T^2 + (a_{11} + 2a_{12}v)T + a_{01} + 2a_{02}v. \tag{26}$$

Solving for $F_v = 0$ yields

$$v = -\frac{a_{21}T^2 + a_{11}T + a_{01}}{2a_{12}T + 2a_{02}}. \tag{27}$$

Substituting Equation (27) into $F = 0$ (Equation (17)) and into $F_T = 0$ (Equation (18)) establishes the isola/mushroom boundary. The rest of the conditions Equations (25) will be evaluated numerically along the obtained boundary.

4.3. Pitchfork Singularity

The conditions for the single-scalar function to undergo a pitchfork bifurcation are

$$F = F_T = F_{TT} = 0. \tag{28}$$

$$F_v = 0. \tag{29}$$

and

$$F_{Tv} \neq 0, F_{TTT} \neq 0. \tag{30}$$

Equations (28) yields the hysteresis conditions of Equations (22) and (23). Equation (29), on the other hand, yields Equation (27). Therefore, the pitchfork singularity is represented by equating Equations (22) and (23) with Equation (27). The condition $F_{TTT} \neq 0$ is always satisfied, while $F_{Tv} \neq 0$ will be evaluated numerically along the boundary.

5. Dynamic Bifurcation

The conditions for the three dimensional model (Equations (6)–(8)) to predict a Hopf points are [29] as follows:

$$F_1 := S_1 S_2 - S_3 = 0. \tag{31}$$

$$S_2 > 0, \tag{32}$$

where S_1, S_2, and S_3 are given by

$$S_1 = j_{11} + j_{22} + j_{33}. \tag{33}$$

$$S_2 = det\begin{pmatrix} j_{11} & j_{12} \\ j_{21} & j_{22} \end{pmatrix} + det\begin{pmatrix} j_{22} & j_{23} \\ j_{32} & j_{33} \end{pmatrix} + det\begin{pmatrix} j_{11} & j_{13} \\ j_{31} & j_{33} \end{pmatrix}. \tag{34}$$

$$S_3 = det(J). \tag{35}$$

The $j_{11}, j_{12}, \cdots$ are the elements of the Jacobean J. The elements of J are given explicitly by taking the derivatives of Equations (6)–(8), yielding

$$j_{11} = -d + \frac{pT}{g+T} - k_E M_0 M - mT, \;\; j_{12} = \frac{pE}{g+T} - \frac{pET}{(g+T)^2} - mE, \;\; j_{13} = -k_E M_0 E. \tag{36}$$

$$j_{21} = -T, \;\; j_{22} = \alpha - 2\alpha\beta T - E - k_T M_0 M, \;\; j_{23} = -k_T M_0 T. \tag{37}$$

$$j_{31} = 0, \;\; j_{32} = 0, \;\; j_{33} = -\gamma. \tag{38}$$

The expressions for the terms S_i (i = 1, 3) (Equations (33)–(35)) are

$$S_1 = -\alpha\beta T + \alpha(1-\beta T) - d - E + \frac{pT}{g+T} - \gamma - k_E M M_0 - k_T M M_0 - mT. \tag{39}$$

$$\begin{aligned} S_2 = &-\gamma\left(-d + \frac{pT}{g+T} - k_E M M_0 - mT\right) - \gamma(-\alpha\beta T + \alpha(1-\beta T) - E - k_T M M_0) \\ &- \left(\frac{pE}{g+T} - \frac{pET}{(g+T)^2} - mE\right)\left(-d + \frac{pT}{g+T} - k_E M M_0 - mT\right) \\ &+ (-\alpha\beta T + \alpha(1-\beta T) - E - k_T M M_0)\left(-d + \frac{pT}{g+T} - k_E M M_0 - mT\right). \end{aligned} \tag{40}$$

$$S_3 = \gamma\left(\frac{pE}{g+T} - \frac{pET}{(g+T)^2} - mE\right)\left(-d + \frac{pT}{g+T} - k_E M M_0 - mT\right)$$
$$-\gamma(-\alpha\beta T + \alpha(1-\beta T) - E - k_T M M_0)\left(-d + \frac{pT}{g+T} - k_E M M_0 - mT\right). \quad (41)$$

The expressions of S_i ($i = 1, 3$) (Equations (39)–(41)) can be simplified using the steady states of Equations (6)–(8). In particular, Equations (6) and (7) at the steady state can be rewritten to yield, respectively,

$$E\left(-d + \frac{pT}{g+T} - k_E M M_0 - mT\right) = -s. \quad (42)$$

$$\alpha(1-\beta T) - k_E M_0 M - E = 0. \quad (43)$$

Substituting Equations (42) and (43) into Equations (39)–(41) yields

$$S_1 = -\gamma - \frac{s}{E} - \alpha\beta T. \quad (44)$$

$$S_2 = \frac{\gamma s}{E} + \alpha\beta\gamma T + \frac{\alpha\beta s T}{E} + s(-m + \frac{gp}{(g+T)^2}). \quad (45)$$

$$S_3 = -\frac{\alpha\beta\gamma s T}{E} - \gamma s(-m + \frac{gp}{(g+T)^2}). \quad (46)$$

Algebraic manipulations yield the following useful relation:

$$\tilde{\gamma} S_2 + S_3 = -\gamma^2(S_1 + \gamma). \quad (47)$$

Substituting Equation (47) in the first Hopf condition $F_1 := S_1 S_2 - S_3 = 0$ yields

$$F_1 := (S_1 + \gamma)(S_2 + \gamma^2) = 0. \quad (48)$$

Since S_2 is required to be positive, the Hopf conditions (Equations (31) and (32)) are reduced to

$$(S_1 + \gamma) = 0 \text{ and } S_2 > 0. \quad (49)$$

But, we have the following expression for $S_1 + \gamma$ (Equation (44)):

$$S_1 + \gamma = -\alpha\beta T - \frac{s}{E}, \quad (50)$$

which can never equal zero for positive values of α, β, s, E, and T. We conclude therefore that no Hopf points can occur for any model parameters.

Numerical Simulations

The model parameters' nominal values were carefully selected to represent realistic ranges [9,16]:

$$d = 0.0407\ \frac{1}{day},\ g = 2 \times 10^4\ cells, k_E = 0.6\ \frac{1}{day}, k_T = 0.6\ \frac{1}{day}, m = 5.505 \times 10^{-10}\ \frac{1}{cells.day}.$$

$$n = 1.101 \times 10^{-7}\ \frac{1}{cells.day}, p = 0.124\ \frac{1}{day},\ s = 1.321 \times 10^4\ \frac{cells}{day}, \alpha = 0.1801\ \frac{1}{day}.$$

$$\beta = 2 \times 10^{-9} \frac{1}{cells}, \gamma = 0.9 \frac{1}{day}, M_0 = 10^3 \frac{mg}{m^2}, T_0 = E_0 = 10^6 \, cells. \tag{51}$$

The dimensionless values (omitting the bar) are the following:

$$d = 0.37, g = 0.02, k_E = 0.45, k_T = 0.45, m = 0.005, n = 1, p = 1.13$$

$$s = 0.12, \alpha = 1.636, \beta = 2 \times 10^{-3}, \gamma = 8.18 \tag{52}$$

The branch sets for the different singularities are next constructed, for example, in the parameter space (m, d). Figure 2 shows the hysteresis and isola/mushroom boundaries for the nominal values of the parameters in Equation (52). Region (1) has a unique solution. Figure 3 shows an example of a bifurcation diagram in this region in terms of drug intensity, for example, for $(m = 0.002, d = 0.1)$. It can be seen that for values of v larger than the bifurcation point BR, the trivial solution, i.e., $T = 0$ is the sole stable steady state. For v values below the BR, the system settles on a unique stable steady state. A feature of the model for this case is that as the drug intensity increases, the tumor continues to grow until it reaches a maximum value; beyond that, the tumor decreases until its eradication.

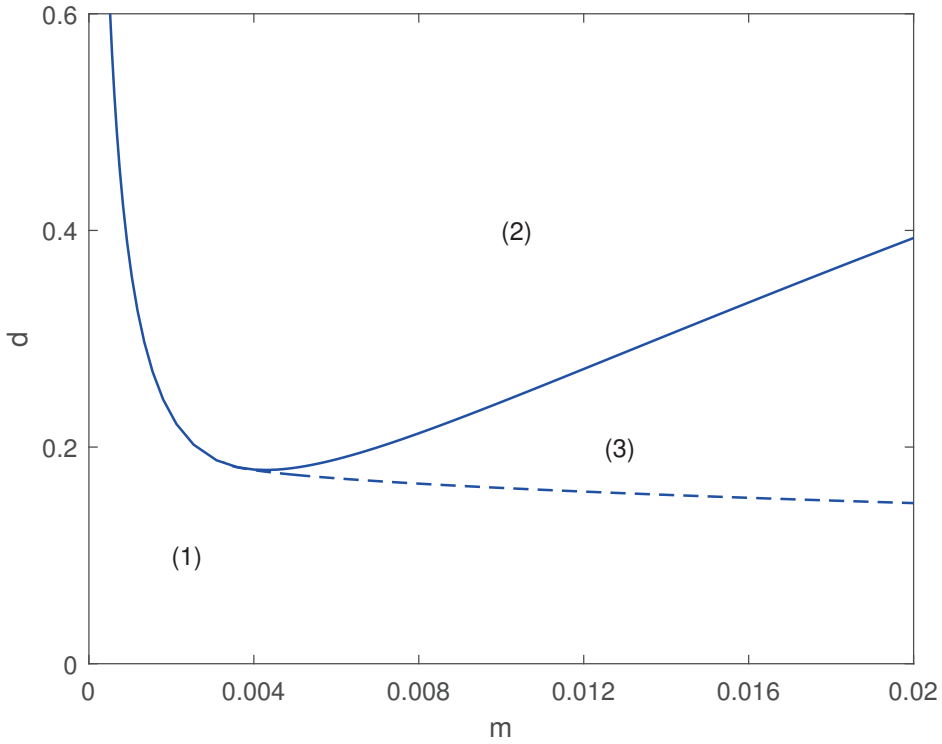


Figure 2. Hysteresis singularity (solid) and isola/mushroom singularity (dashed line) for model nominal values shown in Equation (52).

In region (2) of the branch set of Figure 2, two static limit points are born as a result of crossing the hysteresis line. An example of a bifurcation diagram is shown in Figure 4 for $(m = 0.002, d = 0.4)$. Two static limit points $LP1$ and $LP2$ occur at $v = 0.001048$ and $v = 0.001316$, respectively. The following regimes are therefore expected: For v smaller than $LP1$, the system settles on the low-concentration tumor cells (the trivial solution $T = 0$ is unstable). Between $LP1$ and $LP2$, there is bistability where the system can settle on low-tumor cells, but any changes in the initial conditions/external stimulations can push the system to sneak to a higher tumor concentration despite the administration of the drug.

As the value of v increases beyond $LP2$, the tumor cell concentration decreases with the drug intensity. Values of v larger than the BR lead to suppression of the tumor.

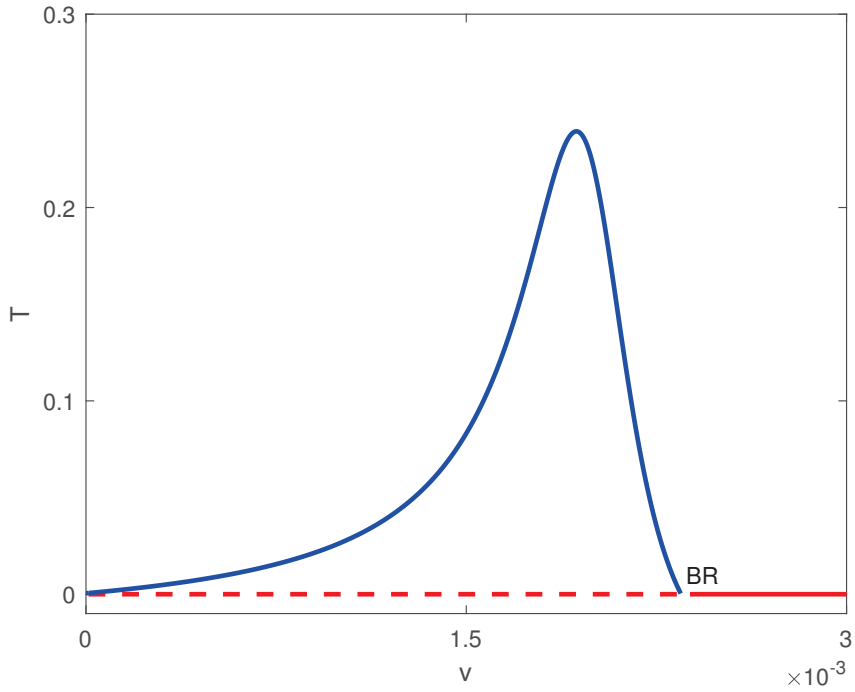


Figure 3. Bifurcation diagram for region (1) of Figure 2 for $(m = 0.002, d = 0.1)$; solid line (stable branch); dashed line (unstable branch); BR (bifurcation point). Blue color nontrivial steady-state; Red color trivial steady-state $(T = 0)$.

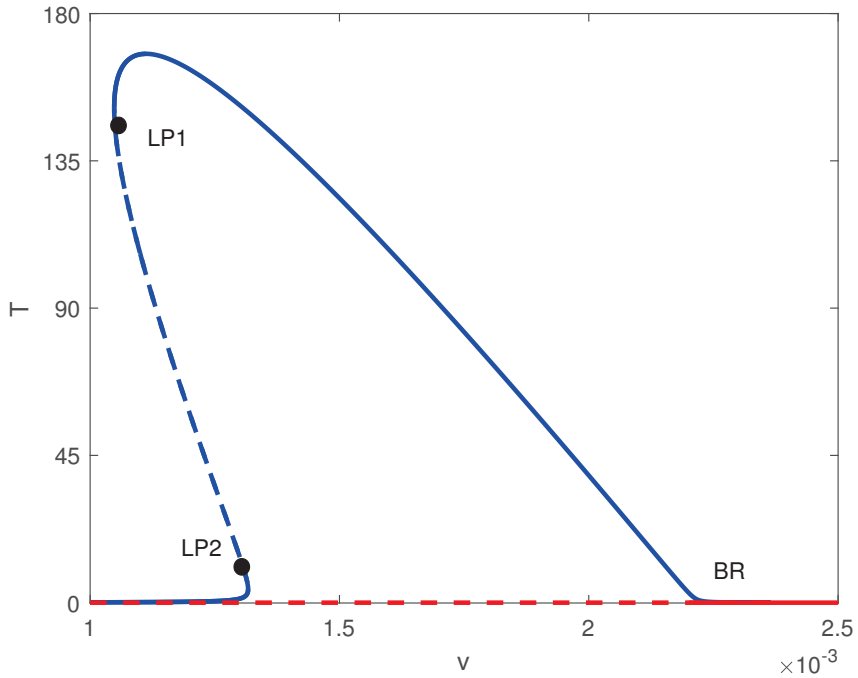


Figure 4. Bifurcation diagram for region (2) of Figure 2 for $(m = 0.002, d = 0.4)$: solid line (stable branch); dashed line (unstable branch); BR (bifurcation point). LP1 and LP2 limit points. Blue color is nontrivial steady state. Red color is trivial steady state $(T = 0)$.

For region 3 of the branch set of Figure 2, an example of a bifurcation diagram is shown in Figure 5, for example, for $(m = 0.0125, d = 0.2)$. The diagram is more complex and shows

the existence of a stable low-tumor concentration steady state (spiral), an unstable middle concentration steady state (saddle), and an upper stable steady state (node). Figure 5b shows a logarithmic plot (in y scale) for easier viewing. It can be seen that for v values smaller than $LP1, (v = 0.00171)$, the low-cell branch coexists with the high-cell branch. Values of v larger than $LP1$ and smaller than $LP2, (v = 0.00211)$ lead to high-tumor-cell concentration. In the small region between $LP2, (v = 0.00211)$ and $LP3, (v = 0.00218)$, the high-tumor-cell branch coexists with the no-tumor steady state. As v values increase past $LP3$, the tumor cell concentration continues to decrease. Values of v larger than the BR lead to the suppression of the tumor. Time variations showing bistability are shown in Figure 6 for the value of $v = 5 \times 10^{-4}$ and two sets of initial conditions. Start-up conditions $(E, T, M) = (1.3 \times 10^{-2}, 2.6 \times 10^{-2}, 6 \times 10^{-5})$ lead after some transient oscillations (because of the spiral nature of the steady state) to the low-tumor steady state. On the other hand, the initial conditions $(E, T, M) = (1.3 \times 10^{-2}, 10^{-1}, 6 \times 10^{-5})$ lead to the high-tumor steady state. Next, we examine the effect of the model parameters (other than m and d) on the hysteresis and isola/mushroom singularities. Figure 7 shows the results of the sensitivity analysis. In each case, a 25 percent change in the parameter is assumed. Figure 7a shows that an increase in the normal growth rate of effector cells from $s = 0.12$ to $s = 0.15$ increases the region of uniqueness as the regions of hysteresis and isola/mushroom move to higher values of d. Figure 7b shows that an increase in the value of p from 1.13 to 1.41 increases the region of uniqueness further compared to the effect of s. A decrease in the effect of k_E from 5.45 to 4.08 (Figure 7c) has the effect of increasing the region of unique solution, while an increase in the value of k_T from 5.45 to 6.81 (Figure 7d) has the same effect. Finally, the effect of g is the least pronounced, as shown in Figure 7e.

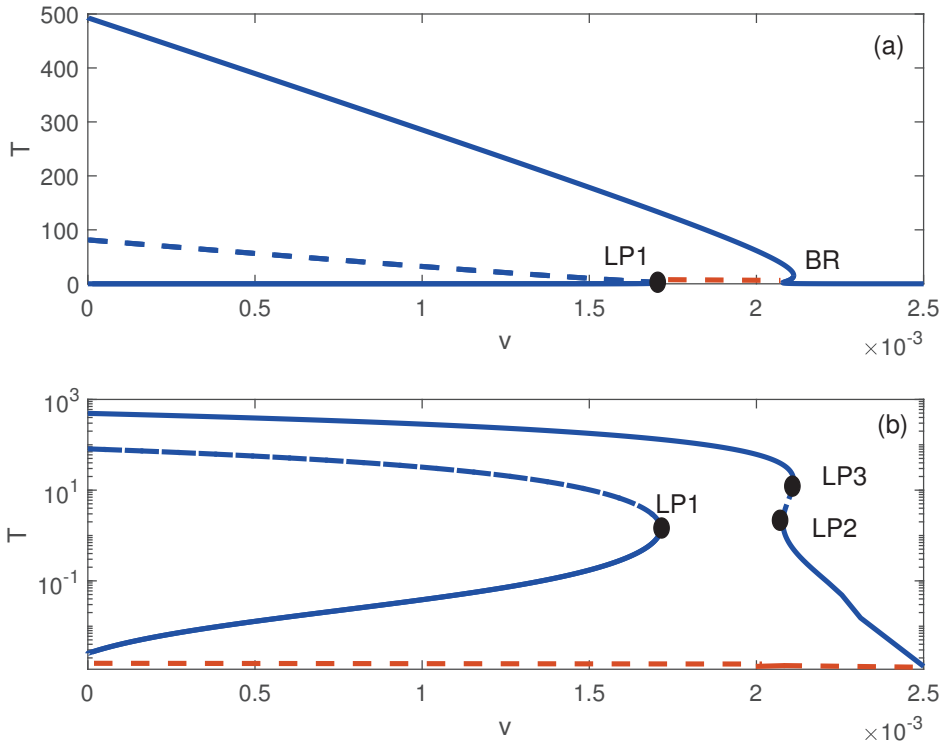


Figure 5. Bifurcation diagram for region (3) of Figure 2 for $(m = 0.0125, d = 0.2)$: (**a**) diagram in linear scale; (**b**) diagram in semi-logarithmic scale (in T); solid line (stable branch); dashed line (unstable branch); BR (bifurcation point); LP1, LP2, and LP3 limit points; blue color is nontrivial steady state; red color is trivial steady state $(T = 0)$.

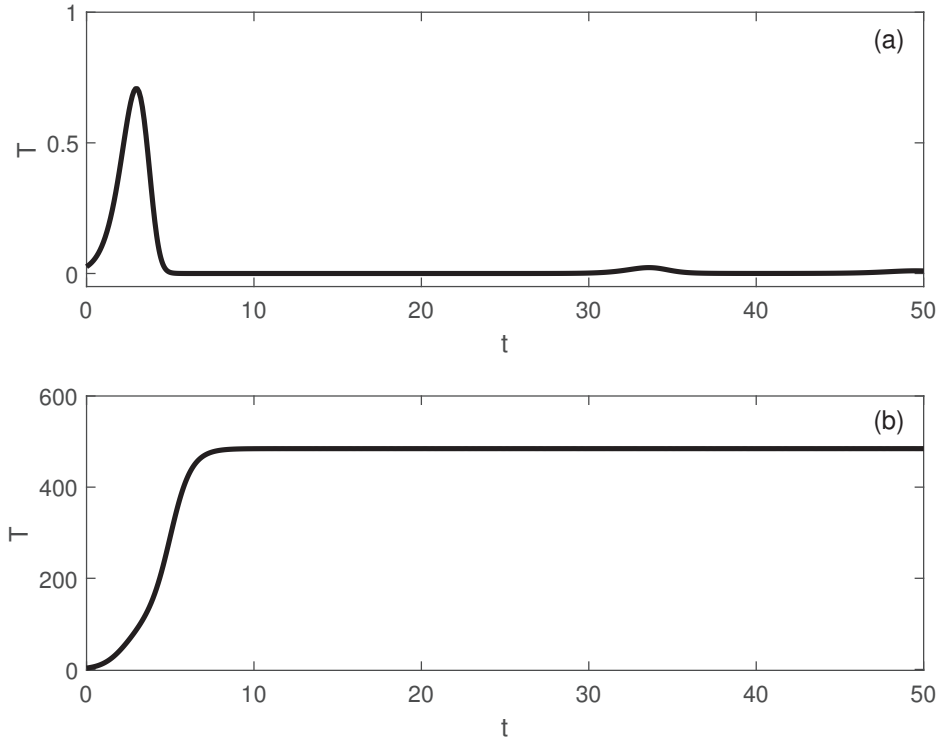


Figure 6. Simulations illustrating bistability in Figure 5, for example, for $v = 5 \times 10^{-4}$. (**a**) Initial conditions $(E, T, M) = (1.3 \times 10^{-2}, 2.6 \times 10^{-2}, 6 \times 10^{-5})$ lead to low-tumor steady state. (**b**) Initial conditions $(E, T, M) = (1.3 \times 10^{-2}, 3, 6 \times 10^{-5})$ lead to the high-tumor steady state.

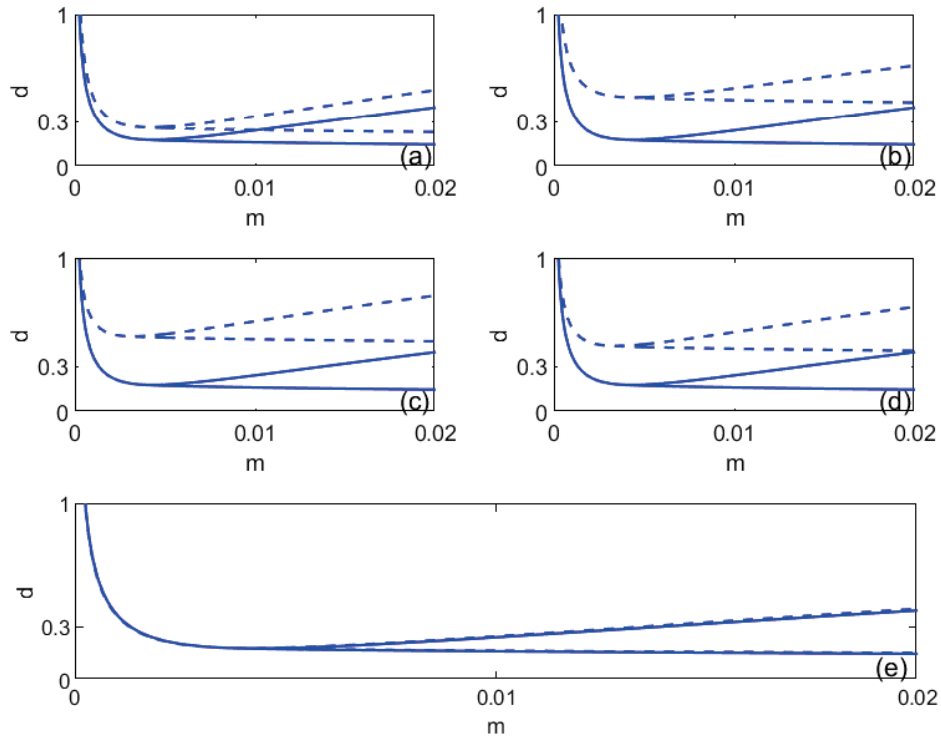


Figure 7. Effect of the different model parameters on the hysteresis and isola/mushroom singularities of Figure 2: (**a**) solid line ($s = 0.12$), dashed line (s = 0.15). (**b**) solid line (p = 1.13), dashed line (p = 1.41). (**c**) solid line ($k_E = 5.45$), dashed line ($k_E = 4.087$). (**d**) solid line ($k_T = 5.45$), dashed line ($k_T = 6.81$). (**e**) solid line ($g = 0.02$), dashed line ($g = 0.015$).

Next, we examine the occurrence of pitchfork singularity. Figure 8 shows the pitchfork boundary in the parameter space (m, d). Figure 9 shows an example of a bifurcation diagram on the boundary itself of Figure 8, for example, for $(m = 0.004, d = 0.1528)$. A perfect pitchfork can be observed. (Figure 9b shows a logarithmic plot on the y axis of Figure 9a). For values of v up to the LP, there is coexistence between the low-tumor-cell and the high-tumor-cell branch. Beyond the LP and up to the BR, the tumor cell concentration decreases steadily as v increases. Beyond the value of the BR, the tumor is suppressed.

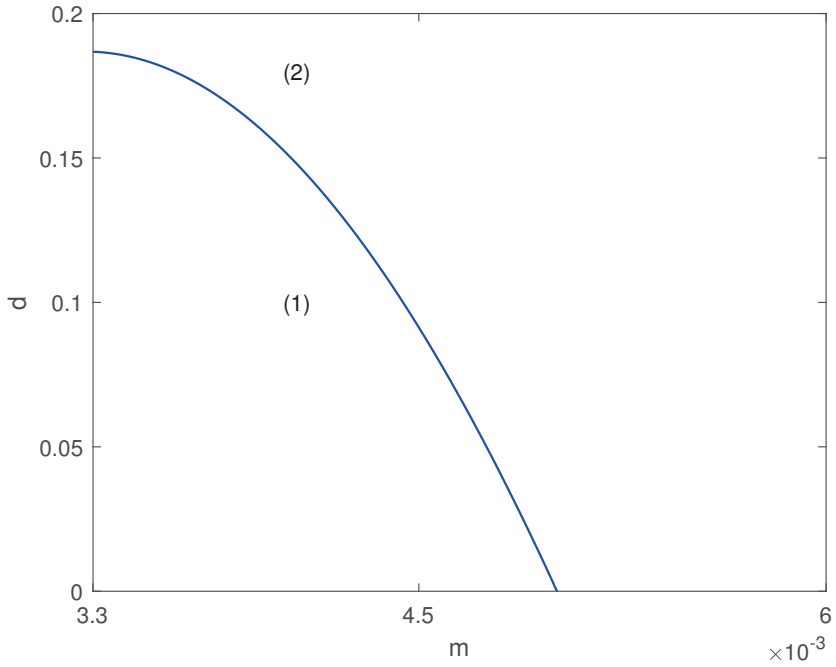


Figure 8. Pitchfork singularity for nominal values shown in Equation (52).

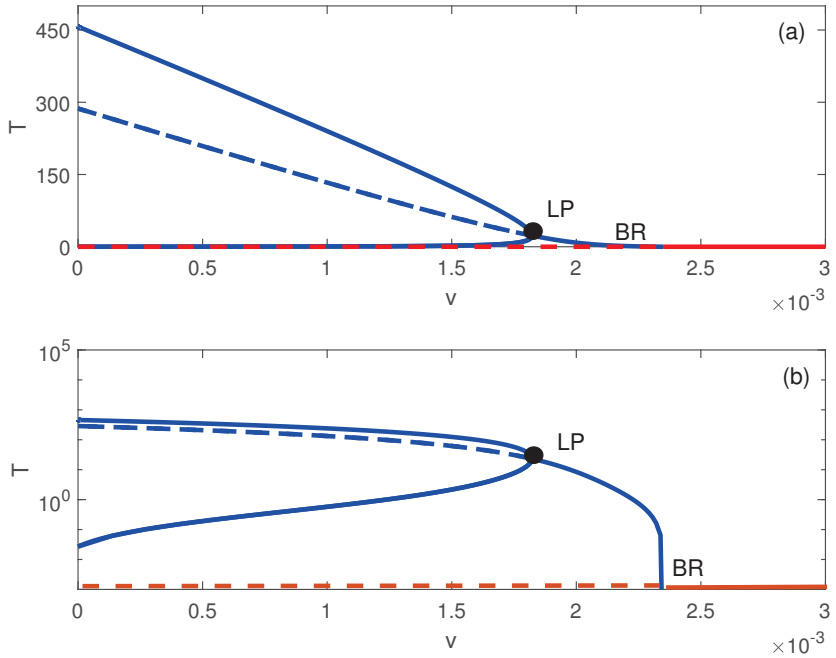


Figure 9. (**a**) Bifurcation diagram showing a perfect pitchfork for the model parameters $(m = 0.004, d = 0.1528)$ in the boundary between region (1) and (2) of Figure 8. (**b**) Diagram in semi-logarithmic scale (in T). Solid line (stable branch); dashed line (unstable branch); BR (bifurcation point); LP1 (limit point); blue color is nontrivial steady state; Red color is trivial steady state $(T = 0)$.

Figure 10 shows the behavior in region (1) of Figure 8, for example, for $(m = 0.004,$ $d = 0.11)$. The perfect pitchfork of Figure 9 is perturbed, and the lower and upper stable branches are no longer connected. Values of v smaller than the LP lead to bistability, while values larger than the LP and up to the BR lead to low-concentration stable steady state. Values of v larger than the BR lead to suppression of the tumor.

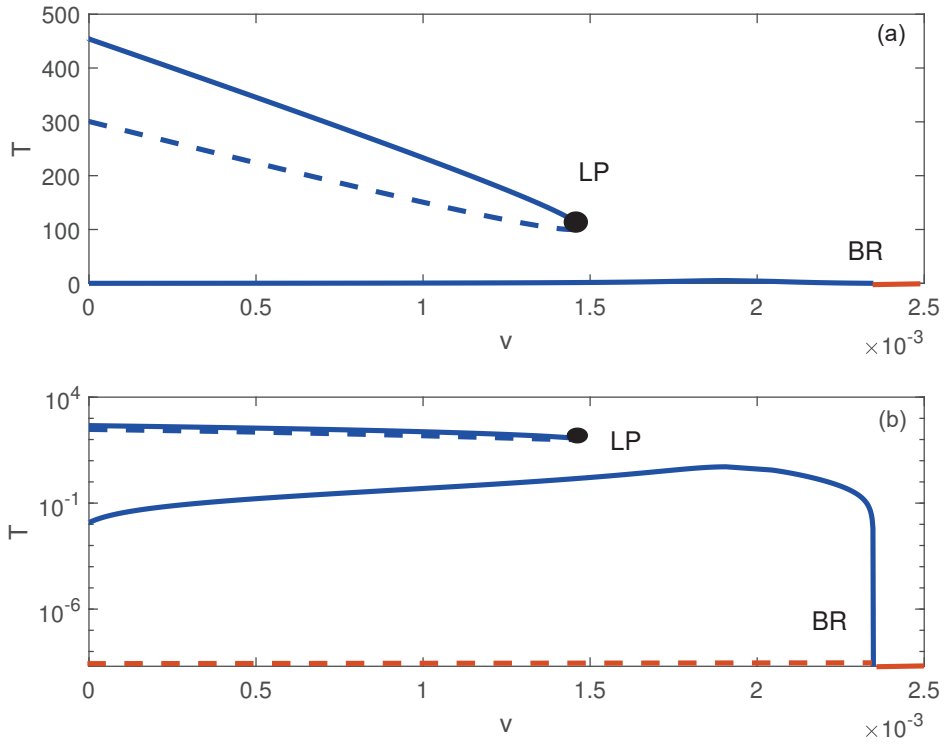


Figure 10. (**a**) Bifurcation diagram for the model parameters $(m = 0.004, d = 0.11)$ in region (1) of Figure 8. (**b**) Diagram in semi-logarithmic scale (in T). Solid line (stable branch); dashed line (unstable branch); BR (bifurcation point); LP (limit point); blue color is nontrivial steady state; red color is trivial steady state $(T = 0)$.

Figure 11 shows an example of a bifurcation diagram in region (2) of Figure 8, for example, for $(m = 0.004, d = 0.18)$. The perfect pitchfork is again perturbed. However, from a practical point of view, the behavior of Figure 11 is similar to that of Figure 10. The only difference between Figure 10 and Figure 11 is the presence of a maximum in Figure 10, while the tumor cell concentration in Figure 11 decreases steadily after the LP.

Figure 12 shows the effect of model parameters on the limits of pitchfork singularity. It can be seen that compared to the nominal case (blue line), an increase in the value of s by 25% (yellow line) increases the pitchfork boundary, as the boundary occurs at larger values of d. The effect of the increase in p by 25% (red line) is very pronounced. A decrease in the value of k_E (magenta line) or an increase in k_T (black line) increases the pitchfork boundary.

Finally, we saw in all the aforementioned bifurcation diagrams that if the drug intensity is increased past a critical point (vc) solution of $a_0 = 0$ (Equation (15)), the tumor is completely suppressed. Figure 13 shows the effect of the different model parameters on the critical value (v_c). It can be seen that as s is increased, v_c decreases, while the opposite can be seen for d. As k_E increases, v_c increases, while k_T has the opposite effect and is more pronounced. The value of v_c is independent of β, g, p, and m.

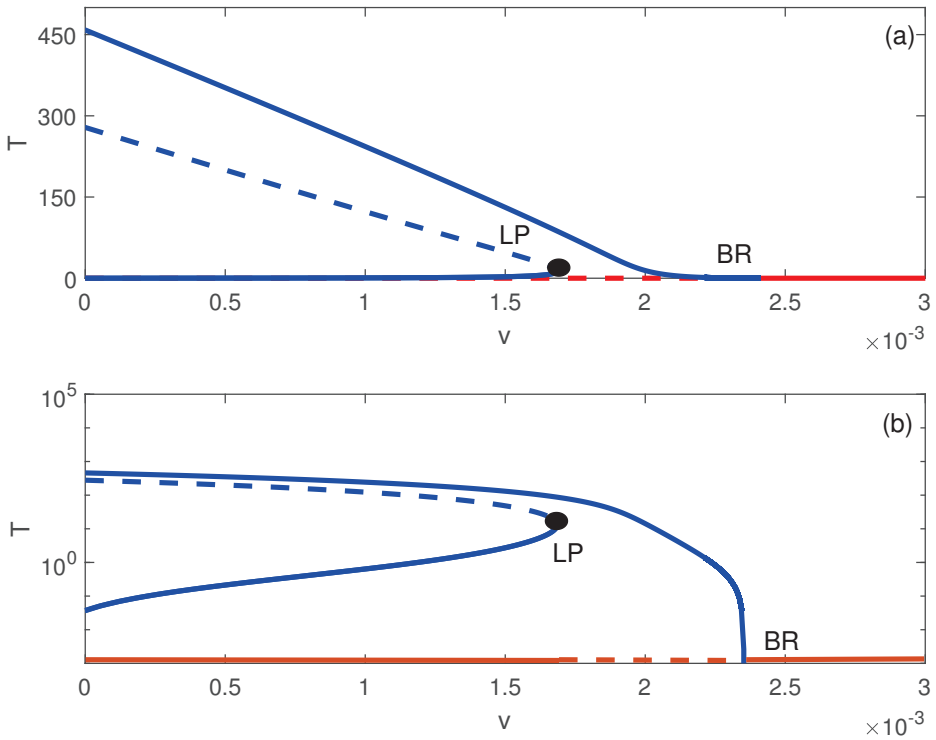


Figure 11. (**a**) Bifurcation diagram for the model parameters $(m = 0.004, d = 0.18)$ in region (2) of Figure 8. (**b**) Diagram in semi-logarithmic scale (in T). Solid line (stable branch); dashed line (unstable branch); BR (bifurcation point); LP (limit point); blue color is nontrivial steady state; red color is trivial steady state $(T = 0)$.

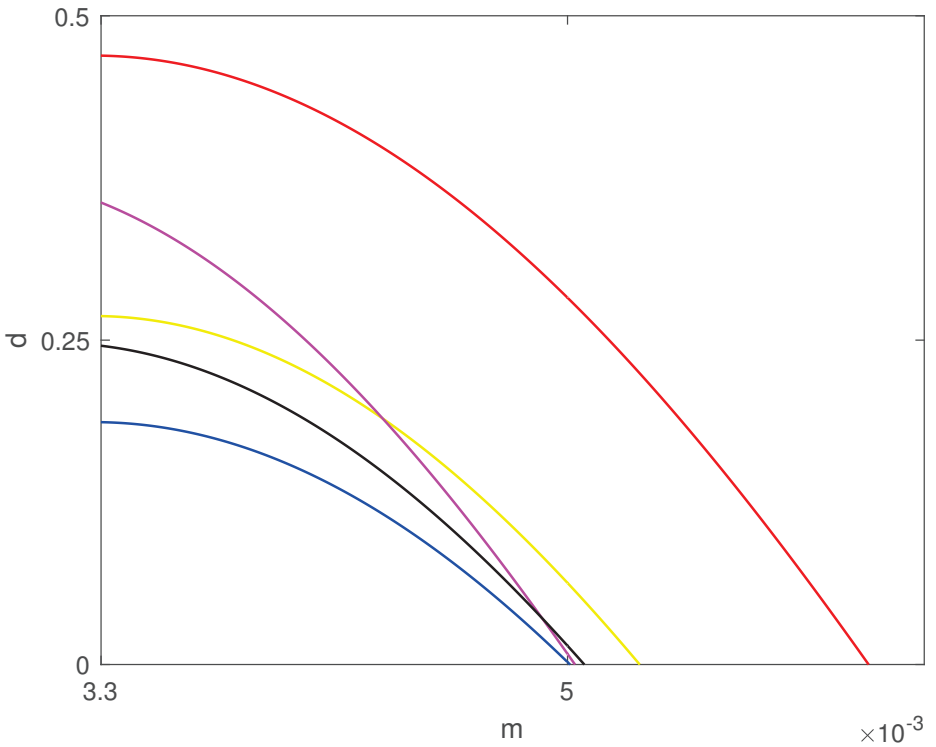


Figure 12. Effect of the different model parameters on the pitchfork singularity of Figure 8. Blue line nominal case $(s = 0.12, p = 1.13, k_E = 5.45, k_T = 5.45)$; yellow line (s = 0.15); red line (p = 1.41); magenta line (k_E = 4.63); black line (k_T = 5.72).

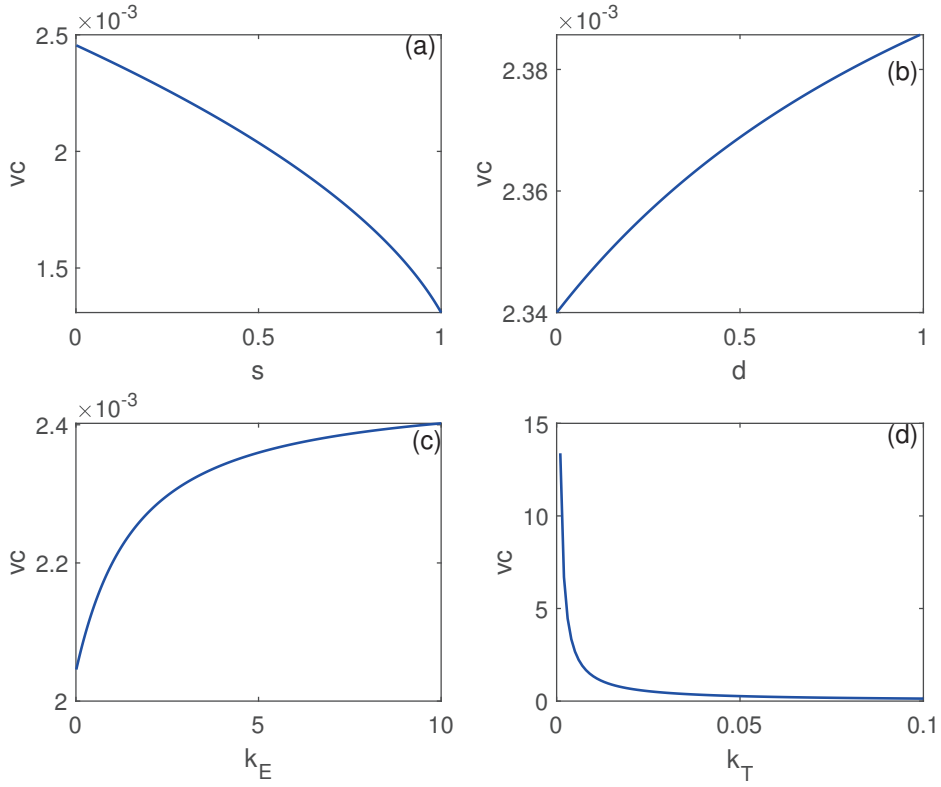


Figure 13. Effect model parameters on the critical value of drug intensity, past of which the tumor is suppressed. (**a**) Effect of s. (**b**) Effect of d. (**c**) Effect of k_E. (**d**) Effect of k_T.

6. Discussion

The theoretical analysis using the singularity theory has shown that small variations in the model's biological parameter values can lead to a number of bifurcation patterns. In real life, it is feasible for biological parameter values to vary given that both effector and tumor cell populations are heterogeneous, comprising different subpopulations that possess distinct parameter values influencing their behavior.

The model without chemotherapy was studied in [9], and the authors showed the presence of hysteresis, which is marked by the coexistence of areas with low-tumor-cell concentrations alongside regions with high concentrations as well as regions where "dormant cells" can evade regulatory effects from effectors and subsequently become active. Saddle node and transcritical bifurcations were also shown to exist [25,26] in such models.

With the introduction of chemotherapy, the analysis has revealed that the model is capable of predicting a greater range of behaviors than what was identified in earlier studies [9,25,26].

For some values of model biological parameters, the effector system is efficient, and the model can only predict a low-tumor-cell steady state. However, when the level of administrated drug is increased, the tumor cell concentration does not, as expected, decrease monotonically. Rather, and as result of the complex interactions between the model's biological parameters and those associated with the chemotherapy, the tumor cell concentration increases and reaches a maximum before decreasing. Only drug levels past a critical level (Equation (15)) can completely suppress the tumor. For this combination of biological and chemotherapy parameters, the administration of the drug at intermediate doses (relative to the critical value, Equation (15)) can be detrimental to the disease outcome.

For other values of model biological parameters, the model forecasts a hysteresis. It was observed that relatively low drug levels can stabilize the system at low-tumor-cell

populations. However, increasing the drug level may push the system into the zone of hysteresis, where a sneaking phenomenon can push the tumor from low- to high-tumor-cell populationss as a result of changes in the initial conditions and/or external stimulations. Again, only large values of drug administration past a critical threshold can eradicate the tumor. For this scenario, it is advised to avoid administrating the drug at levels within the hysteresis region.

Pitchfork singularities, either perfect or in perturbed forms, were found to also occur in the model for some range of parameters. In these cases, the effector system is less efficient, and even for small doses of chemotherapy, the system exhibits bistability between the low-tumor and high- (uncontrolled) tumor-cell populations. If the drug levels are increased past the bistability domain of the pitchfork, then either the tumor cells decrease monotonically or unexpectedly increase, reaching a maximum before decreasing. For these sets of conditions, it is advised to increase the drug past low values of bisability but below the peak of tumor cell concentration.

Together with chemotherapy, immunotherapy may act as a treatment that influences systemic parameters, such as the sustained elevation of the cytolytic potential of immune killer cells, which is indicated by the parameter (n) in Equation (2). If the systems parameters can be altered, then in order to avoid bistability, the region of unique steady state was found to be favored by an increase in the growth rate of the effector cells, a decrease in their death rate, an increase in the degree of recruitment of maximum immune effector cells in relation to cancer cells, an increase in the effect of chemotherapy on tumor cells, and a decrease in the effect of chemotherapy on effector cells. The degree of recruitment of maximum immune effector cells in relation to cancer cells had, on the other hand, the least pronounced effect.

Additionally, in all the uncovered bifurcation patterns, the tumor can be completely eradicated if the chemotherapy is increased past a critical value. A simple analytical expression (Equation (15)) was found for this value. This critical value decreases when the effector cells' normal growth rate is increased and/or death rate decreases or when the growth rate (α) of the tumor decreases. The critical value also depends on the chemotherapy drug parameters, where it can be decreased by an increase in the rate of response coefficient affecting the tumor cells or a decrease in the response coefficient affecting the effector cells.

In relation to the topic of oscillations, we have demonstrated that the proposed model does not have the capability to predict any oscillatory behavior, regardless of the model parameters. Population size fluctuations are an essential element of the interactions that occur between predators and prey. Established models of predator–prey interactions account for these oscillations by utilizing suitable functional responses or by introducing delays within the model, among other factors [19,23]. The lack of oscillatory behavior in the proposed competition model is worthy of discussion. It is probably advantageous for tumor cells to induce fluctuations in the human body, as this can enable a temporary avoidance of the immune system and lead to a higher rate of tumor spread. This has been demonstrated in certain diseases, such as malaria [33] and smallpox [34]. However, oscillations in the context of tumor–immune cell interactions have not yet been confirmed [19,35]. Several interpretations were suggested [19,21], but the most plausible explanation is the potential risk of autoimmune reactions, which may endanger the stability of healthy cells. Attaining a state of comprehensive homeostasis within the body amidst the complex interactions between tumor and immune cells poses significant challenges, as the dedication to generating immune cells can disrupt other physiological processes. For this reason, I maintain that a suitable model that effectively represents the dynamics of tumor–immune

interactions should refrain from predicting periodic or aperiodic behavior, at least within the acceptable range of model parameters.

Additionally, since the nominal values of the model's parameters were taken from realistic conditions [9,16], the objective of any future work is to validate the model by comparing all the uncovered bifurcation behavior to real-life outcomes. It should be noted that hysteresis, saddle node, and transcrtical bifurcation are known to occur experimentally [9,25,26]. The rest of the branching phenomena needs to be validated. But at this point, it is essential to highlight that the bifurcation phenomena observed in the model, as shown in the corresponding figures, occurred when the dimensionless drug intensity was less than approximately 3×10^{-3}. It is a well-established fact that the total amount of chemotherapy administered during treatment is limited, as it adversely affects both cancerous and normal cells. Guidelines indicate that chemotherapy doses can reach as high as 3500 mg/m^2 per day [36]. Utilizing the dimensionless variables in Equation (5), this corresponds to a dimensionless drug intensity of about 3.2×10^{-3}, which is greater than the maximum value used in this analysis. This observation confirms that the range selected for this study is realistic.

A final note should be made about the limitations of the current work. The chemotherapy effect on both effector and tumor cells was described by a simple mass action linear model through the parameters k_E and k_T (Equations (1) and (2)). Other expressions were also used in the literature such as a saturation type, e.g., $\frac{kT}{(1+hT)}$ [20], or the exponential kill model with a time-delayed concentration, e.g., $k(1 - e^{-M})$ [26]. The extension of the results of this paper to these models is worth investigating.

7. Conclusions

The purpose of this work was to analyze the different bifurcation solutions that can be displayed by a simple model describing effector–tumor cell interactions under chemotherapy. The mathematical analysis using singularity theory managed to delineate the complex interactions between the model's biological and chemotherapy parameters that results in a number of bifurcation phenomena.

For particular values of the model's biological parameters, there exists solely a stable equilibrium. The concentration of tumor cells escalates with increasing drug intensity, attains a peak, and then diminishes, culminating in their elimination at higher drug doses.

For other values of model biological parameters, a hysteresis occurs and implies that within specific ranges of drug dose, a sneaking phenomenon could lead to an increase in tumor cell numbers, driven by changes in the initial conditions and/or external influences.

In other cases, the system shows bistability at low doses of chemotherapy between a reduced number of tumor cells and those that are uncontrolled. If the dosage surpasses the bistability threshold, the tumor cells may either decrease in a consistent manner or, counterintuitively, increase to a peak before declining.

Finally, the relevance of these mathematical models is not confined to understanding how intrinsic parameters or chemotherapy affect the emergence of different bifurcations and bistability. After identifying these patterns, the next phase is to apply these models to explore various other issues, including the merit of periodic adjustment of drug doses and evaluating the optimal rate of drug administration during patient treatment by treating the medication as a control function $v(t)$. This is especially important for minimizing the risk of drug toxicity. Additionally, identifying the best timing for each drug injection represents another critical optimal control problem that requires further research.

Funding: This work was supported and funded by the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University (IMSIU) (grant number IMSIU-DDRSP2501).

Data Availability Statement: The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author.

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

References

1. Bray, F.; Laversanne, M.; Sung, H.; Ferlay, J.; Siegel, R.L.; Soerjomataram, I.; Jemal, A. Global cancer statistics 2022: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA Cancer J. Clin.* **2024**, *74*, 229–263. [CrossRef] [PubMed]
2. Rockne, R.C.; Scott, J.G. Introduction to Mathematical Oncology. *JCO Clin. Cancer Inform.* **2019**, *3*, 1–4.
3. Araujo, R.P.; McElwain, D.L.S. A history of the study of solid tumour growth: The contribution of mathematical modelling. *Bull. Math. Biol.* **2004**, *66*, 1039–1091. [CrossRef] [PubMed]
4. Enderling, H.; Wolkenhauer, O. Are all models wrong? *Comput. Syst. Oncol.* **2020**, *1*, e1008.
5. Batmani, Y.; Khaloozadeh, H. Optimal drug regimens in cancer chemotherapy: A multi-objective approach. *Comput. Biol. Med.* **2013**, *43*, 2089–2095.
6. Agur, Z.; Vuk-Pavlovic, S. Mathematical modeling in immunotherapy of cancer: Personalizing clinical trials. *Mol. Ther.* **2012**, *20*, 1–2.
7. Wang, Z.; Deisboeck, T.S. Mathematical modeling in cancer drug discovery. *Drug Discov. Today* **2014**, *19*, 145–150.
8. Yin, A.; Moes, D.J.; van Hasselt, J.G.; Swen, J.J.; Guchelaar, H.J. A review of mathematical models for tumor dynamics and treatment resistance evolution of solid tumors. *CPT Pharmacometrics Syst. Pharmacol.* **2019**, *8*, 720–737.
9. Kuznetsov, V.A.; Makalkin, I.A.; Taylor, M.A.; Perelson, A.S. Nonlinear dynamics of immunogenic tumors: Parameter estimation and global bifurcation analysis. *Bull. Math. Biol.* **1994**, *56*, 295–321.
10. Robertson-Tessi, M.; El-Kareh, A.; Goriely, A. A mathematical model of tumor-immune interactions. *J. Theor. Biol.* **2012**, *294*, 56–73. [CrossRef]
11. Kaur, G.; Ahmad, N. On study of immune response to tumor cells in prey-predator system. *Int. Sch. Res. Not.* **2014**, *2014*, 346597. [CrossRef]
12. Benzekry, S.; Lamont, C.; Beheshti, A.; Tracz, A.; Ebos, J.M.; Hlatky, L.; Hahnfeldt, P. Classical mathematical models for description and prediction of experimental tumor growth. *PLoS Comput. Biol.* **2014**, *10*, e1003800. [CrossRef] [PubMed]
13. Michor, F.; Beal, K. Improving cancer treatment via mathematical modeling: Surmounting the challenges is worth the effort. *Cell* **2015**, *163*, 1059–1063. [CrossRef] [PubMed]
14. Murphy, H.; Jaafari, H.; Dobrovolny, H.M. Differences in predictions of ODE models of tumor growth: A cautionary example. *BMC Cancer* **2016**, *16*, 163. [CrossRef]
15. Heesterman, B.L.; Bokhorst, J.M.; de Pont, L.M.; Verbist, B.M.; Bayley, J.P.; van der Mey, A.G.; Corssmit, E.P.; Hes, F.J.; van Benthem, P.P.; Jansen, J.C. Mathematical models for tumor growth and the reduction of overtreatment. *J. Neurol. Surg. B Skull Base* **2019**, *80*, 72–78. [CrossRef]
16. Lestari, D.; Sari, E.R.; Arifah, H. Dynamics of a mathematical model of cancer cells with Chemotherapy. *J. Phys. Conf. Ser.* **2019**, *1320*, 012026. [CrossRef]
17. Akhmetzhanov, A.R.; Kim, J.W.; Sullivan, R.; Beckman, R.A.; Tamayo, P.; Yeang, C.H. Modelling bistable tumour population dynamics to design effective treatment strategies. *J. Theor. Biol.* **2019**, *474*, 88–102. [CrossRef]
18. Beckman, R.A.; Kareva, I.; Adler, F.R. How should cancer models be constructed? *Cancer Control* **2020**, *27*, 1073274820962008. [CrossRef]
19. Kareva, I.; Luddy, K.A.; O'Farrelly, C.; Gatenby, R.A.; Brown, J.S. Predator-prey in tumor-immune interactions: A wrong model or just an incomplete one? *Front. Immunol.* **2021**, *12*, 668221. [CrossRef]
20. Bashkirtseva, I.; Chukhareva, A.; Ryashko, L. Modeling and analysis of nonlineartumor-immune interaction under chemotherapy and radiotherapy. *Math. Methods Appl. Sci.* **2022**, *45*, 7983–7991. [CrossRef]
21. Hamilton, P.T.; Anholt, B.R.; Nelson, B.H. Tumour immunotherapy: Lessons from predator-prey theory. *Nat. Rev. Immunol.* **2022**, *22*, 765–775. [CrossRef] [PubMed]
22. Kareva, I.; Berezovskaya, F. Cancer immunoediting: A process driven by metabolic competition as a predator-prey-shared resource type model. *J. Theor. Biol.* **2015**, *380*, 463–472. [PubMed]
23. Bazykin, A.D. *Nonlinear Dynamics of Interacting Populations*; World Scientific: Singapore, 1998.
24. Golubovskaya, V.; Wu, L. Different Subsets of T Cells, Memory, Effector Functions, and CAR-T Immunotherapy. *Cancers* **2016**, *8*, 36. [CrossRef] [PubMed]

25. De Pillis, L.G.; Radunskaya, A.E. Modeling tumor-immune dynamics, In *Mathematical Models of Tumor-Immune System Dynamics*; Eladdadi, A., Kim, P., Mallet, D., Eds.; Springer Proceedings in Mathematics & Statistics 107; Springer Science+Business Media: New York, NY, USA, 2014; pp. 59–108.
26. López, A.G.; Seoane, J.M.; Sanjuán, M.A. A validated mathematical model of tumor growth including tumor-host interaction, cell-mediated immune response and chemotherapy. *Bull. Math. Biol.* **2014**, *76*, 2884–2906.
27. Makhlouf, A.M.; El-Shennawy, L.; Elkaranshawy, H.A. Mathematical modelling for the role of CD4+T cells in tumor-immune interactions. *Comput. Math. Methods Med.* **2020**, *2020*, 7187602. [CrossRef]
28. Song, G.; Tian, T.; Zhang, X. A mathematical model of cell-mediated immune response to tumor. *Math. Biosci. Eng.* **2020**, *18*, 373.
29. Wiggins, S. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*; Springer: New York, NY, USA, 1990.
30. Golubitsky, M.; Stewart, I.; Schaeffer, D.G. *Singularities and Groups in Bifurcation Theory: Volume II*; Springer: Berlin/Heidelberg, Germany, 1988.
31. Alhumaizi, K.; Aris, R. *Surveying a Dynamical System: A Study of the Gray-Scott Reaction in a Two-Phase Reactor*; Research Notes in Mathematics Series; Chapman & Hall/CRC: London, UK, 1995.
32. Ajbar, A.; Alhumaizi, K. *Dynamics of the Chemostat: A Bifurcation Theory Approach*; Chapman and Hall/CRC: London, UK, 2011.
33. Smith, L.M.; Motta, F.C.; Chopra, G.; Moch, J.K.; Nerem, R.R.; Cummins, B.; Roche, K.E.; Kelliher, C.M.; Leman, A.R.; Harer, J.; et al. An intrinsic oscillator drives the blood stage cycle of the malaria parasite *Plasmodium falciparum*. *Science* **2020**, *368*, 754.
34. Greer, M.; Saha, R.; Gogliettino, A.; Yu, C.; Zollo-Venecek, K. Emergence of oscillations in a simple epidemic model with demographic data. *R. Soc. Open Sci.* **2020**, *7*, 191187.
35. Dorraki, M.; Fouladzadeh, A.; Salamon, S.J.; Allison, A.; Coventry, B.J.; Abbott, D. On detection of periodicity in C-reactive protein (CRP) levels. *Sci. Rep.* **2018**, *8*, 11979. [CrossRef]
36. Mazard, T.; Ychou, M.; Thezenas, S.; Poujol, S.; Pinguet, F.; Thirion, A.; Bleuse, J.P.; Portales, F.; Samalin, E.; Assenat, E. Feasibility of biweekly combination chemotherapy with capecitabine, irinotecan, and oxaliplatin in patients with metastatic solid tumors: Results of a two-step phase I trial: XELIRI and XELIRINOX. *Cancer Chemother. Pharmacol.* **2012**, *69*, 807–814.

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

mathematics

MDPI

Article

Managing the Risk via the Chi-Squared Distribution in VaR and CVaR with the Use in Generalized Autoregressive Conditional Heteroskedasticity Model

Fazlollah Soleymani [1,*], Qiang Ma [2] and Tao Liu [3]

[1] Department of Mathematics, Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137-66731, Iran
[2] Department of Mathematics, Harbin Institute of Technology, Harbin 150001, China; hitmaqiang@hit.edu.cn
[3] School of Mathematics and Statistics, Northeastern University at Qinhuangdao, Qinhuangdao 066004, China; liutao@neuq.edu.cn
* Correspondence: soleymani@iasbs.ac.ir

Abstract: This paper develops a framework for quantifying risk by integrating analytical derivations of Value at Risk (VaR) and Conditional VaR (CVaR) under the chi-squared distribution with empirical modeling via Generalized Autoregressive Conditional Heteroskedasticity (GARCH) processes. We first establish closed-form expressions for VaR and CVaR under the chi-squared distribution, leveraging properties of the inverse regularized gamma function and its connection to the quantile of the distribution. We evaluate the proposed framework across multiple time windows to assess its stability and sensitivity to market regimes. Empirical results demonstrate the chi-squared-based VaR and CVaR, when coupled with GARCH volatility forecasts, particularly during periods of heightened market volatility.

Keywords: chi-squared distribution; CVaR; risk; GARCH; confidence level

MSC: 91G70; 62M10; 62P20

1. Introductory Notes

Risk management in finance involves quantifying the potential losses in a portfolio [1]. Two commonly used measures are Value-at-Risk (VaR) [2] and Conditional VaR (CVaR), otherwise called Expected Shortfall (ES). These measures assess the extent of financial risk under a given probability distribution.

VaR is a widely used measure for risk that estimates the maximum loss in a portfolio's value within a defined time period, given a predetermined level of confidence. Formally, suppose that X is a random variable showing the losses, and let $F_X(x)$ be its cumulative distribution function (CDF). VaR measures the worst expected loss at a confidence level p on a given time horizon as follows [2,3]:

$$\mathrm{VaR}_p = \inf\{x \in \mathbb{R} \mid P(X \leq x) \geq p\} = F_X^{-1}(p). \tag{1}$$

This represents the loss threshold exceeded with probability $1-p$. For normally distributed returns, where $X \sim N(\mu, \sigma^2)$, the VaR is computed as follows:

$$\mathrm{VaR}_p = \mu + \sigma\Phi^{-1}(p) = \mu - \sqrt{2}\sigma\mathrm{erfc}^{-1}(2p), \tag{2}$$

where $\Phi^{-1}(p)$ represents the inverse CDF of the standard normal distribution. Several characteristics define this risk measure. First, it is not coherent due to its lack of subadditivity. Second, it is highly interpretable, offering an intuitive and easily understandable framework for risk assessment. However, a weakness of VaR is that it fails to account for the severity of losses beyond the specified threshold [4].

A more robust risk measure is CVaR. This measure considers the expected loss that the loss exceeds the VaR value. It is provided as follows [5]:

$$\mathrm{CVaR}_p = \mathbb{E}[X \mid X > \mathrm{VaR}_p]. \tag{3}$$

CVaR satisfies all four features of a coherent risk measure (translation invariance, positive homogeneity, subadditivity and monotonicity). Unlike VaR, CVaR accounts for losses beyond the threshold. Under the assumption of a normal distribution:

$$\mathrm{CVaR}_p = \mu + \sigma\frac{\phi(\Phi^{-1}(p))}{1-p} = \mu - \frac{\sigma e^{-\mathrm{erfc}^{-1}(2p)^2}}{\sqrt{2\pi}(p-1)}, \tag{4}$$

where $\phi(\cdot)$ is the standard normal density function.

The choice of probability distribution for modeling financial returns is crucial when calculating risk measures such as VaR and ES [6,7]. The distribution affects the accuracy of risk estimates, particularly in capturing tail risk, skewness, and kurtosis. Different distributions lead to varying risk quantifications, which can impact risk management decisions.

The normal distribution is frequently used due to its analytical tractability and simplicity [8]. It allows for the straightforward computation of VaR and ES using closed-form solutions. However, it fails to capture heavy tails and skewness, leading to an underestimation of extreme losses. The Student's t-distribution addresses this issue by incorporating heavier tails, making it more suitable for financial data. Nevertheless, it assumes symmetric tails, which may not align with real-world financial return distributions where negative shocks tend to be more pronounced. The generalized extreme value (GEV) distribution and generalized Pareto distribution (GPD) are tailored for modeling extreme losses [9]. These distributions provide accurate estimates of tail risk, making them highly effective for ES calculations. However, they require a careful selection of threshold values, and their estimation procedures can be complex and data-intensive.

When modeling extreme losses, we often use the GEV distribution:

$$H_{\chi,h,\mu}(x) = \begin{cases} \exp\left(-\left(1+\chi\frac{-\mu+x}{h}\right)^{-1/\chi}\right), & \chi \neq 0, \\ \exp\left(-e^{-(x-\mu)/h}\right), & \chi = 0. \end{cases} \tag{5}$$

The parameter χ determines the tail heaviness and the location parameter is μ, and the scale parameter is h. The GPD is used for modeling exceedances over a high threshold:

$$G_{\chi,h}(y) = -\left(\chi\frac{y}{h}+1\right)^{-1/\chi} + 1, \quad \text{for } y > 0. \tag{6}$$

Using GPD, we obtain the following analytical expressions for risk measures (N_p is the quantity of observations after the threshold p) [10]:

$$\mathrm{VaR}_p = p + \frac{h}{\chi}\left(\left(\frac{n}{N_p}p\right)^{-\chi} - 1\right), \tag{7}$$

$$\mathrm{CVaR}_p = \frac{\mathrm{VaR}_p}{1-\chi} + \frac{h-\chi p}{1-\chi}. \tag{8}$$

Further related discussions and background can be found in [11,12]. It should be highlighted that Norton et al. recently conducted an in-depth study of the ES for several widely used distributions, as presented in [13]. They derived a generalized expression for the GEV distribution, focusing on a specific case where the parameter of shape is fixed at zero.

Risk management in financial markets involves assessing and controlling the uncertainty associated with asset returns [14]. The Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model provides a framework for modeling and forecasting time-varying volatility, making it an essential tool for financial risk management [15].

This study underscores the critical role of deriving explicit, closed-form expressions for VaR and CVaR within the framework of the chi-squared distribution. The motivation stems from historical evidence demonstrating that inadequate risk management can result in financial losses over short periods. Within this context, the research evaluates the accuracy of these risk measures, systematically comparing their computed values and advocating for the chi-squared distribution as a viable and robust alternative to the conventional normal distribution in financial risk assessment. Furthermore, the derived formulations are utilized to simulate and forecast stock prices and returns across various time horizons using the GARCH process, as elaborated in [16]. This work bridges theoretical probability theory with practical risk management, providing a unified tool for regulators and financial institutions to enhance risk assessment protocols.

While the normal distribution remains a popular choice for modeling returns due to its analytical tractability, it often underestimates tail risk because of its thin tails and symmetry assumption. In contrast, the chi-squared distribution, being asymmetric and positively skewed, offers an alternative framework particularly well-suited for modeling non-negative financial quantities such as losses [17]. This skewness better captures the empirical reality that extreme losses in financial markets tend to be more pronounced than extreme gains. Furthermore, unlike symmetric heavy-tailed alternatives such as the Student's t-distribution, which may still misrepresent downside risk, or extreme value distributions like the GEV, which can be complex to estimate and sensitive to threshold selection, the chi-squared distribution provides a relatively simple and interpretable model for the tail behavior of financial losses. In this work, we derive closed-form expressions for VaR and CVaR under the chi-squared assumption and compare them against traditional models in a GARCH framework. Empirical analysis using historical stock return data reveals that the chi-squared distribution yields risk estimates that are competitive with or superior to those from Student's t and GEV models, particularly in terms of VaR exceedance rates and the accuracy of CVaR predictions during volatile periods. Evaluation metrics such as the Kupiec and Christoffersen backtesting procedures, applied in the numerical section, support the reliability of the chi-squared-based risk measures, indicating their effectiveness as a viable and robust alternative for capturing asymmetry and tail risk in financial return distributions.

The GARCH(1,1) model is employed in this study due to its well-established effectiveness in capturing time-varying volatility in financial time series, particularly through its parsimonious structure and empirical success across a wide range of asset classes [18]. Its formulation balances model complexity and explanatory power, making it a standard benchmark in volatility modeling. The choice of GARCH(1,1) is further justified by its ability to accommodate volatility clustering, a common feature in financial returns, where periods of high volatility tend to be followed by similar periods. To validate the adequacy of the GARCH(1,1) specification, we perform model diagnostic checks, including the Ljung-Box Q-test on the standardized residuals and squared residuals to confirm the

absence of serial correlation and remaining ARCH effects. Moreover, the estimated model parameters are statistically significant and satisfy the stationarity condition with $\alpha + \beta < 1$ indicating a stable volatility process. To examine model robustness across different market regimes, we apply the GARCH(1,1) model to datasets spanning both tranquil and turbulent market periods, including the 2008 financial crisis and the COVID-19 pandemic. The results demonstrate that the model maintains consistent volatility forecasts and captures shifts in market dynamics effectively. These findings affirm that the GARCH(1,1) framework provides a sufficiently accurate and stable structure for the integration of alternative distributional assumptions—such as the chi-squared distribution—within the computation of risk measures like VaR and CVaR.

The remainder of the manuscript is organized as follows. Section 2 furnishes a discussion of the fundamental principles of GARCH models. In Section 3, an overview of the chi-squared distribution is presented, followed by the derivation of an analytical formula for the VaR measure based on this distribution. Special emphasis is placed on the role of innovation distribution. Section 4 is dedicated to deriving the CVaR measure. In Section 5, the developed formulations are applied to calculate VaR and CVaR within the framework of a GARCH(1,1) model. Lastly, Section 6 wraps up the study with a summary of the key findings and offers critical perspectives on the implications of this work.

2. Definition of the GARCH(1,1) Process

The GARCH model, proposed by Bollerslev, extends the ARCH model developed by Engle [19]. GARCH models are broadly utilized in finance and econometrics to model volatility clustering in economic time series. The GARCH(1,1) model is defined by the following system of equations:

$$X_t = \sigma_t e_t, \quad e_t \sim \text{i.i.d.}(0,1), \tag{9}$$

$$\sigma_t^2 = \omega + \alpha X_{t-1}^2 + \beta \sigma_{t-1}^2, \tag{10}$$

where

- X_t represents the returns (or log-returns) at time t.
- σ_t^2 is the conditional variance at time t.
- $\omega > 0$ ensures positive variance.
- $\alpha, \beta \geq 0$ are model parameters.
- e_t is a sequence of i.i.d. standard normal innovations.

For the GARCH(1,1) process to be weakly stationary (i.e., to have a finite and constant variance over time), the following condition must hold [20]:

$$\mathbb{E}[\sigma_t^2] = \frac{\omega}{1 - \alpha - \beta}, \quad \text{if } \alpha + \beta < 1. \tag{11}$$

If $\alpha + \beta \geq 1$, the process exhibits long-memory effects, and σ_t^2 does not converge to a finite unconditional variance. The existence of higher-order moments requires additional restrictions. The second moment exists if

$$\mathbb{E}[X_t^2] = \frac{\omega}{1 - \alpha - \beta}, \quad \text{if } \alpha + \beta < 1. \tag{12}$$

For the fourth moment to exist, we require the following:

$$\mathbb{E}[X_t^4] < \infty \quad \text{if} \quad \mathbb{E}[(\alpha e_t^2 + \beta)^2] < 1. \tag{13}$$

For normal innovations, this condition simplifies to the following:

$$3\alpha^2 + 2\alpha\beta + \beta^2 < 1. \tag{14}$$

The GARCH(1,1) model is a fundamental tool in financial econometrics for modeling time-varying volatility. Its mathematical properties provide useful insights into financial risk modeling, including volatility clustering and persistence. Noting that if a GARCH-based model systematically underestimates risk, adjustments such as incorporating extreme value theory (EVT) or switching to a heavy-tailed distribution may be necessary [21]. The application of GARCH models in risk management provides an effective approach to estimating financial risk by capturing volatility clustering and heavy tails. When coupled with VaR and CVaR methodologies, GARCH models enhance financial decision-making by providing time-varying risk estimates. However, accurate implementation requires careful distributional assumptions to ensure reliable risk forecasts.

It is recalled that financial institutions use GARCH models to assess portfolio risk by estimating the time-varying covariance matrix of asset returns. A multivariate GARCH (MGARCH) model can be used to compute dynamic portfolio VaR:

$$\mathrm{VaR}_{\alpha,t}^{\text{portfolio}} = \boldsymbol{w}'\boldsymbol{\mu}_t + \sqrt{\boldsymbol{w}'\boldsymbol{\Sigma}_t\boldsymbol{w}}q_\alpha, \tag{15}$$

wherein $\boldsymbol{w}$ stands for the vector of portfolio weights, $\boldsymbol{\mu}_t$ is the vector of conditional mean returns, and $\boldsymbol{\Sigma}_t$ is the conditional covariance matrix.

The stationarity of the GARCH(1,1) model is governed by the condition $\alpha + \beta < 1$, which guarantees weak stationarity and the existence of a finite unconditional variance. This condition ensures that the conditional variance σ_t^2 does not diverge over time and that the influence of past shocks on future volatility decays exponentially [18,20]. When $\alpha + \beta$ is close to one, the model exhibits strong volatility persistence, implying that the effects of shocks remain significant over a long horizon, a phenomenon often referred to as long-memory behavior in volatility. Although GARCH(1,1) is not strictly a long-memory model in the formal statistical sense (which typically requires a hyperbolic rate of decay), high values of $\alpha + \beta$ in practice can mimic long-memory dynamics, which is frequently observed in empirical financial time series. This property is particularly relevant when forecasting risk measures like VaR and CVaR over different horizons, as it underscores the importance of accounting for volatility persistence.

Although the current analysis focuses on univariate volatility modeling and risk assessment using the GARCH(1,1) model and chi-squared distribution, it is important to recognize the relevance of portfolio theory in this context. In modern portfolio theory, risk is typically evaluated through the variance–covariance structure of asset returns, which becomes dynamic when modeled by multivariate extensions of GARCH models, such as the BEKK, DCC, or VECH formulations. These models allow for time-varying estimation of the portfolio's risk profile, where the volatility and correlation structures evolve over time. When the innovations in the GARCH model are assumed to follow a chi-squared distribution, the resulting risk forecasts can capture greater asymmetry and heavier tail behavior compared to the normal assumption, thereby offering a more conservative and realistic estimation of downside risk in portfolio settings. In particular, incorporating the chi-squared distribution into the innovation structure of multivariate GARCH models allows for improved sensitivity to tail risks across diversified assets, which directly impacts the estimation of portfolio VaR and CVaR. This integration provides a meaningful advancement in portfolio risk management by aligning statistical modeling assumptions with empirical features of financial return distributions.

3. Chi-Squared Distribution for VaR

The chi-squared distribution, denoted as $\chi^2(m)$, is a probability distribution characterized by a single positive parameter m, representing the degrees of freedom. It is defined as the distribution of a sum of squared terms of m independent standard normal variables, i.e., if $Z_i \sim N(0,1)$, then the stochastic variable $X = \sum_{i=1}^{m} Z_i^2$ follows a $\chi^2(m)$ distribution [22]. The PDF of the chi-squared distribution is furnished via

$$f(x;m) = \frac{x^{\frac{m}{2}-1}e^{-x/2}}{2^{m/2}\Gamma(m/2)}, \quad x > 0, \tag{16}$$

wherein $\Gamma(\cdot)$ is the Euler Gamma function as follows

$$\Gamma[z] = \int_0^\infty e^{-t}t^{z-1}dt.$$

The shape of the distribution depends on m where, for small values, the distribution is highly skewed with a peak near zero, while for a larger m, it approximates a normal distribution because of the Central Limit Theorem. The expected value and variance of the chi-squared distribution are given by $\mathbb{E}[X] = m$ and $\mathrm{Var}(X) = 2m$, respectively, demonstrating its direct dependence on the degrees of freedom. Some of the features of such a distribution are as follows:

$$\mathrm{Median}[X] = 2Q^{-1}\left(\frac{m}{2}, 0, \frac{1}{2}\right),$$

$$\mathrm{Skewness}[X] = 2\sqrt{2}\sqrt{\frac{1}{m}},$$

$$\mathrm{Kurtosis}[X] = \frac{3(m+4)}{m}.$$

The PDF and CDF of the chi-squared distribution are given in Figure 1. Now let us have the following random variable

$$X \sim \text{Chi squared distribution}(m). \tag{17}$$

This distribution has applications in financial risk management. In risk management, the distribution is crucial for backtesting VaR models, where the distribution of test statistics follows a chi-squared law under the null hypothesis. Financial institutions must evaluate the potential impact of extreme market conditions on portfolio risk, and one method involves using chi-squared-based test statistics to assess the significance of tail events.

Theorem 1. *Let $X \in L^p$ denote a random variable that characterizes loss behavior based on the chi-squared distribution with parameter m. The VaR associated with X can be explicitly formulated in a closed-form expression, as presented in Equation (18).*

Proof. The random variable X is an element of the L^p space. To derive the VaR for $X \sim \chi^2(m)$, we proceed as follows. The VaR at level p is the smallest z such that the CDF of X exceeds p as in (1):

$$\mathrm{VaR}_p(X) = \min\{z \in \mathbb{R} \mid P(X \le z) \ge p\}.$$

The CDF is expressed via the regularized lower incomplete gamma function:

$$P(X \le z) = \frac{\gamma\left(\frac{m}{2}, \frac{z}{2}\right)}{\Gamma\left(\frac{m}{2}\right)},$$

where $\gamma(a,x) = \int_0^x t^{a-1}e^{-t}dt$ is the lower incomplete gamma function. The equation $P(X \leq z) = p$ becomes the following:

$$\frac{\gamma\left(\frac{m}{2}, \frac{z}{2}\right)}{\Gamma\left(\frac{m}{2}\right)} = p.$$

Equivalently, in terms of the generalized regularized gamma function:

$$\text{GammaRegularized}\left(\frac{m}{2}, 0, \frac{z}{2}\right) = p,$$

where $\text{GammaRegularized}(a, z_0, s) = \frac{\Gamma(a,z_0)-\Gamma(a,s)}{\Gamma(a)}$. The solution $\frac{z}{2}$ is obtained by inverting the above relationship:

$$\frac{z}{2} = Q^{-1}\left(\frac{m}{2}, 0, p\right),$$

where $Q^{-1}(a, z_0, s)$ denotes the inverse of $\text{GammaRegularized}(a, z_0, s)$. This yields the following:

$$z = 2 \cdot Q^{-1}\left(\frac{m}{2}, 0, p\right).$$

The scaling factor 2 arises from the chi-squared distribution's parametrization as $\chi^2(m) = \text{Gamma}\left(\frac{m}{2}, 2\right)$. The inverse function $Q^{-1}\left(\frac{m}{2}, 0, p\right)$ computes the value s satisfying the following:

$$\frac{\Gamma\left(\frac{m}{2}, 0\right) - \Gamma\left(\frac{m}{2}, s\right)}{\Gamma\left(\frac{m}{2}\right)} = p,$$

which simplifies to $\Gamma\left(\frac{m}{2}, s\right) = \Gamma\left(\frac{m}{2}\right)(1-p)$. Thus, $s = Q^{-1}\left(\frac{m}{2}, 1-p\right)$ in standard notation. Combining these steps, the VaR is as follows:

$$\text{VaR}_p(X) = 2 \cdot Q^{-1}\left(\frac{m}{2}, 0, p\right). \tag{18}$$

This completes the proof. □

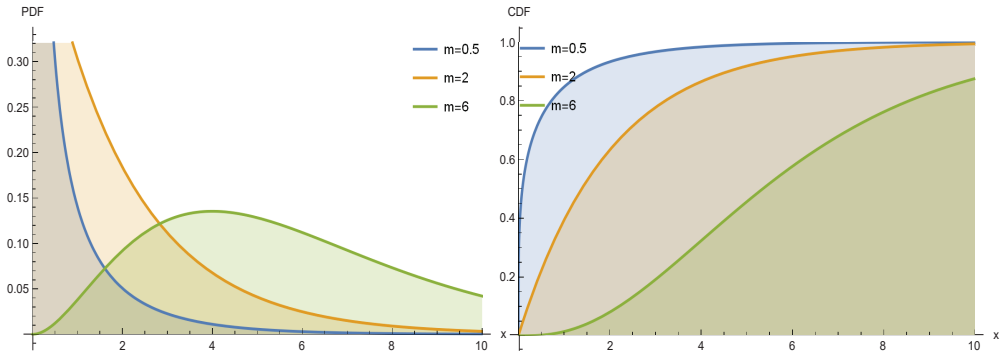


Figure 1. The PDF and CDF of the chi-squared distribution using (16) in left and right for various degrees of freedom, respectively.

To justify the use of higher-order moments in both the GARCH process and the chi-squared distribution for VaR, we can analyze the properties and implications of these moments in the context of financial risk management. In the GARCH(1,1) process, higher-order moments (such as skewness and kurtosis) play a significant role in capturing the distributional characteristics of financial returns. Specifically, higher moments help account for the fat tails and asymmetry often observed in real market data. The existence of these moments depends on the parameter restrictions discussed earlier. In the chi-squared distribution, higher-order moments such as skewness and kurtosis provide valuable insights

into the distribution's shape and its potential to model extreme tail events. The skewness ($2\sqrt{2}/\sqrt{m}$) and kurtosis ($\frac{3(m+4)}{m}$) are directly related to the degrees of freedom, influencing the risk measures derived from the distribution. These moments enhance the accuracy of risk assessments by offering a more detailed understanding of the distribution's behavior at the tails, which is crucial for robust VaR estimation. Therefore, the inclusion of higher-order moments in both the GARCH process and the chi-squared distribution is essential for capturing the complexities of financial risk, especially in environments characterized by volatility clustering and heavy tails.

4. Chi-Squared Distribution for CVaR

Unlike VaR, which only provides a quantile estimate, CVaR takes into account the entire tail distribution, making it a more coherent and informative risk measure [23].

Theorem 2. *Assuming the criteria outlined in Theorem 1, the CVaR for the chi-squared distribution could be expressed in an analytical way as provided in (19).*

Proof. By following a similar line of reasoning as in Theorem 1 and utilizing the definition of CVaR given in (3), we proceed as follows:

$$\begin{aligned}\mathrm{CVaR}_p(X) &= \mathbb{E}\big[X \mid X \geq \mathrm{VaR}_p(X)\big],\\ &= \mathbb{E}\Big[X \mid X \geq 2 \cdot Q^{-1}\Big(\frac{m}{2}, 0, p\Big)\Big],\\ &= \frac{1}{1-p}\int_{2\cdot Q^{-1}\left(\frac{m}{2},0,p\right)}^{\infty} x f(x)\, dx,\end{aligned}$$

where $f(x) = \frac{1}{2^{m/2}\Gamma(m/2)} x^{m/2-1} e^{-x/2}$ is the PDF of $\chi^2(m)$. Substituting $t = x/2$, $x = 2t$, and $dx = 2dt$, the integral becomes the following:

$$\int_{2\cdot Q^{-1}\left(\frac{m}{2},0,p\right)}^{\infty} x f(x)\, dx = \frac{2^{m/2+1}}{\Gamma(m/2)} \int_{Q^{-1}\left(\frac{m}{2},0,p\right)}^{\infty} t^{m/2} e^{-t}\, dt.$$

This simplifies to the following:

$$\frac{2}{\Gamma(m/2)}\Gamma\Big(\frac{m}{2}+1, Q^{-1}\Big(\frac{m}{2}, 0, p\Big)\Big).$$

The survival function $1-p$ is given by the following:

$$1-p = \frac{\Gamma\left(\frac{m}{2}, Q^{-1}\left(\frac{m}{2}, 0, p\right)\right)}{\Gamma(m/2)}.$$

Combining these results, the CVaR is as follows:

$$\mathrm{CVaR}_p(X) = \frac{2\Gamma\left(\frac{m}{2}+1, Q^{-1}\left(\frac{m}{2}, 0, p\right)\right)}{\Gamma\left(\frac{m}{2}, Q^{-1}\left(\frac{m}{2}, 0, p\right)\right)}. \tag{19}$$

This completes the proof. □

The relationship between VaR and CVaR has been furnished in Figure 2. Risk managers rely on distributions that accurately capture tail risk for CVaR estimation, whereas the chi-squared distribution serves as an auxiliary component in financial econometrics and regulatory assessments.

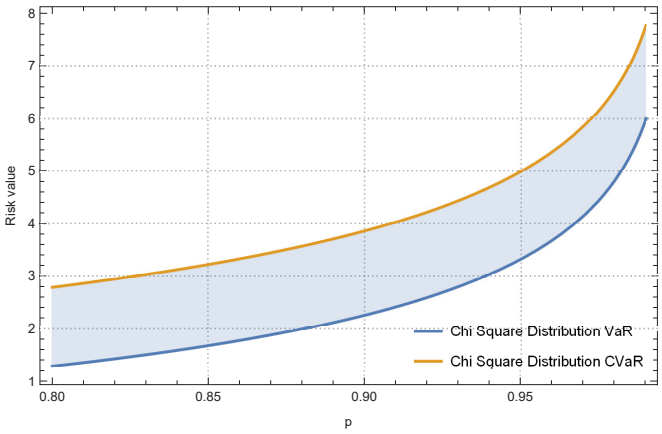


Figure 2. Comparisons of VaR and CVaR under Theorems 1 and 2 and $m = 0.8$.

5. Simulation Results

The objective of this section is to assess the predictive performance of VaR and CVaR within a risk management framework by employing the GARCH model to analyze trading days in an equity market primarily composed of stocks from the S&P500 index. The dataset has been carefully selected to represent a diverse range of stocks. To estimate these risk measures, a methodology based on one-day-ahead volatility forecasting is implemented. All numerical computations have been conducted using Mathematica 14 [24] with machine precision.

This study considers multiple stocks to evaluate the proposed approach. The first experimental case examines the stock ticker "NYSE:VZ," while the second analysis focuses on "NASDAQ:VABK.". A detailed overview of the selected tickers and the corresponding dataset is presented in Table 1. Besides, the features for the considered stocks in terms of the trading volumes and their daily prices within the time windows are given in Figures 3 and 4 for NYSE:VZ and in Figures 5 and 6 for NASDAQ:VABK. Noting that the volume of the trades for the specific tocker is given in Figure 3 only to highlight the volume of the trades of the time window. We only focus on the Please explain the business day prices in the model due to the presence of the associated prices to the stocks in the market. After extracting the initial data, their corresponding daily returns (fractional changes) are used in the GARCH process.

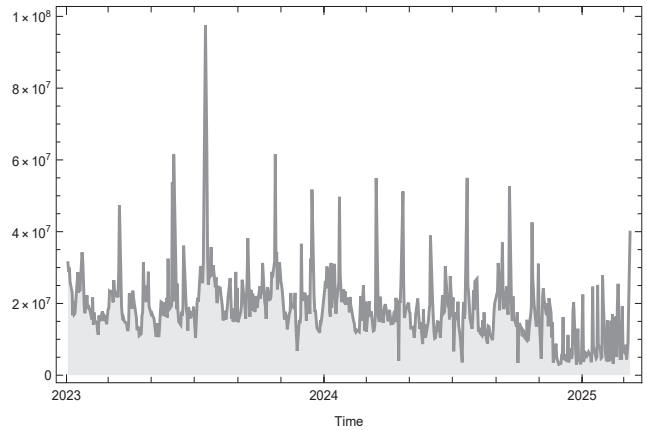


Figure 3. Volume of the trades over the considered time window for the stock NYSE:VZ.

Table 1. The distinctive attributes and defining features of the chosen financial market tickers.

Stock	Tickers	Market	Section	Start	End	Data Size
Verizon Communications	VZ	NYSE	Diversified Telecommunication Services	1 January 2023	10 March 2025	546
Virginia National Bankshares	VABK	NASDAQ	Regional Banks	1 January 2023	10 March 2025	546

The fractional changes in this study have been extracted in Wolfram as follows:

```
return1 =
 FinancialData["NYSE:VZ",
  "FractionalChange", {{2023, 01, 01}, {2025, 03, 10}, "Daily"}]
```

For the stock NYSE:VZ, and

```
return2 =
 FinancialData["NASDAQ:VABK",
  "FractionalChange", {{2023, 01, 01}, {2025, 03, 10}, "Daily"}]
```

For the stock NASDAQ:VABK. The selection of the stocks for this study is driven by the need to examine a diverse range of companies, particularly those with different market behaviors and risk profiles, which is essential for evaluating the robustness of the proposed risk measures. Specifically, "NYSE:VZ" represents a large, established telecommunications company, while "NASDAQ:VABK" is a smaller bank with a potentially different risk profile, providing a useful contrast. By considering these two stocks, we aim to assess the proposed methodology across different sectors, which is crucial for understanding its applicability in diverse market settings. Furthermore, the selection of these stocks is based on their availability of high-quality data, which ensures the reliability of the results in forecasting volatility and risk measures. The rationale behind this selection is to capture a broad spectrum of risk dynamics, which can be generalized to a larger set of assets in future studies.

The use of the chi-squared distribution in modeling financial return series introduces potential operational risk if the distribution fails to accurately represent key empirical features, such as tail heaviness, asymmetry, or volatility persistence. In such cases, the resulting VaR and CVaR estimates may suffer from systematic bias, potentially leading to under- or overestimation of financial risk. To mitigate this concern, the model's performance was evaluated across two distinct datasets exhibiting different volatility profiles.

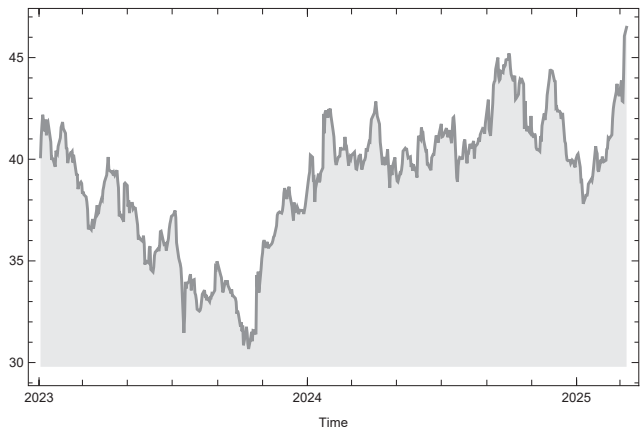


Figure 4. Business day prices in USD over the considered time window for the stock NYSE:VZ.

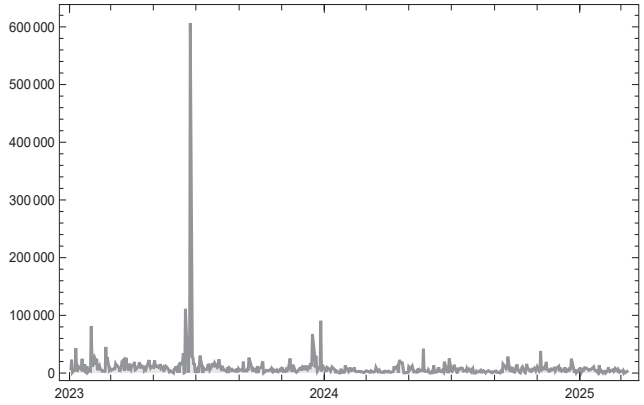


Figure 5. Volume of the trades over the considered time window for the stock NASDAQ:VABK.

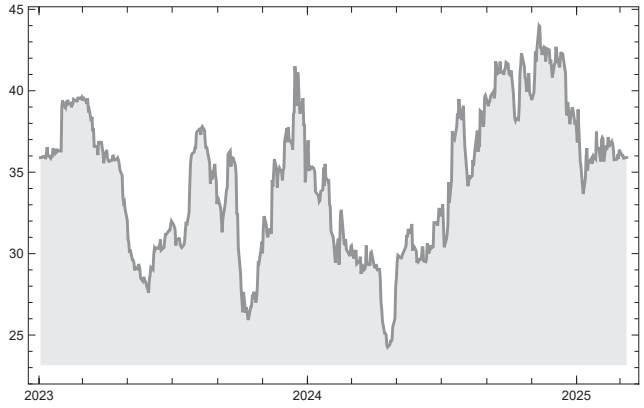


Figure 6. Business day prices in USD over the considered time window for the stock NASDAQ:VABK.

The stock return analysis is conducted by applying the model specified in (9) and employing a time series fitting methodology. The statistical properties derived from this approach are summarized in Tables 2 and 3, where the estimation of the process parameters is carried out utilizing the maximum likelihood method. The sample selection in this study is motivated by the objective of evaluating risk measures within a representative and diversified equity market environment. The time period from January 2023 to March 2025 was selected to capture recent post-pandemic market dynamics, ensuring the applicability of the findings to contemporary financial risk management practices. This choice provides a comprehensive basis for assessing the performance of the proposed chi-squared-based VaR and CVaR measures under real-world conditions.

Table 2. The estimation of parameters under (9) and (10) resulting from the dataset utilized in the first experiment.

w	α	β	**Error Variance**
0.580849	0.129969	0.536188	24.4537

Table 3. The estimation of parameters under (9) and (10) resulting from the dataset utilized in the second experiment.

w	α	β	**Error Variance**
1.80771	0.425899	0.24823	114.126

The numerical results obtained from the simulations, as illustrated in Figures 7 and 8 for the initial experiment, lead to the following observations. The application of the chi-squared distribution gives us this upper hand so that not so many tight values of the confidence level are required and $p = 90\%$ would also be enough to have appropriate values for the VaR and CVaR without over- or under-estimations of the risk values for very high volatile stock returns under the GARCH process. It is important to highlight that a considerable number of prior approaches have been based on the assumption of normality or log-normality. Although the chi-squared distribution is characterized by an asymmetrical behavior in its PDF, adopting a confidence level of 80% or 90% facilitates robust risk evaluation and yields reliable scalar estimates within financial markets. For the second experiment, Figures 9 and 10 are furnished.

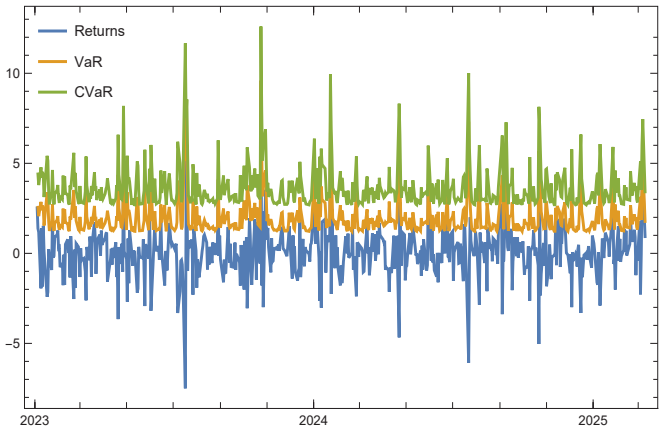


Figure 7. The comparative analysis of the risk quantifiers under the chi-squared distribution is conducted for the pre-specified tail levels, with $p = 80\%$ for the stock NYSE:VZ.

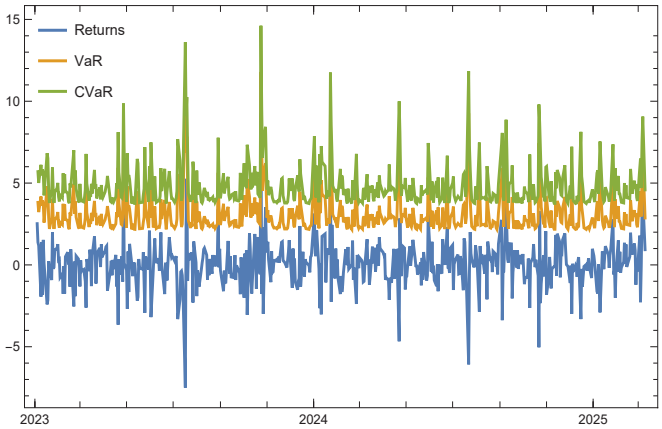


Figure 8. The comparative analysis of the risk quantifiers under the chi-squared distribution is conducted for the pre-specified tail levels, with $p = 90\%$ depicted on the right, for the stock NYSE:VZ.

The simulation results presented in this section provide valuable insights into the effectiveness of the chi-squared-based VaR and CVaR measures within a GARCH(1,1) framework for financial risk assessment. The empirical analysis, conducted using stock return data from the S&P 500 index, shows that the chi-squared distribution offers a flexible and robust alternative to traditional normality-based assumptions. Notably, the results indicate that setting the confidence level at 80% or 90% is sufficient to capture the essential risk characteristics of highly volatile stock returns, mitigating the risk of overestimation or underestimation. This observation is particularly relevant for financial risk management, where conventional approaches often impose stricter confidence levels, potentially leading to excessive capital requirements or inadequate risk buffers. Additionally, the numerical

computations reinforce the asymmetric nature of the chi-squared distribution, which aligns well with the observed skewness in financial return distributions. The obtained results suggest that adopting a chi-squared-based risk framework may enhance portfolio risk evaluation, particularly in markets exhibiting volatility clustering. Future research may extend this analysis by incorporating noncentral chi-squared distributions or alternative heavy-tailed models to further refine risk quantification methodologies.

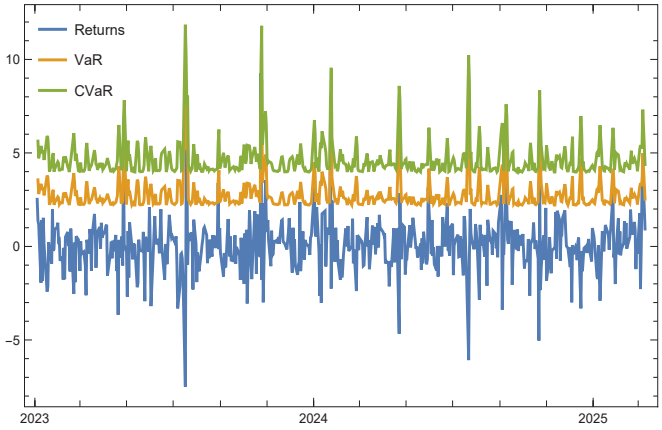


Figure 9. The comparative analysis of the risk quantifiers under the chi-squared distribution is conducted for the pre-specified tail levels, with $p = 80\%$ for the stock NASDAQ:VABK.

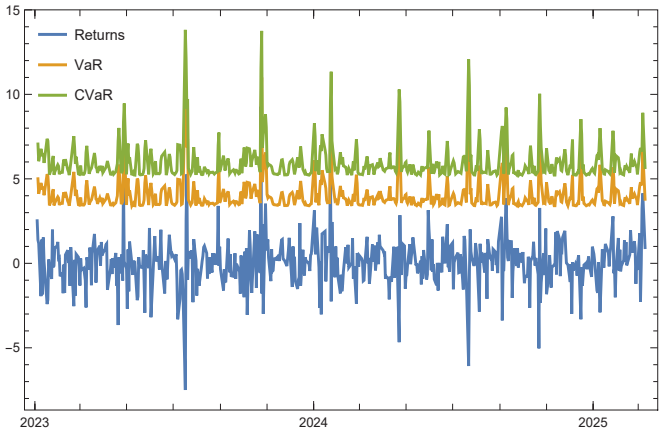


Figure 10. The comparative analysis of the risk quantifiers under the chi-squared distribution is conducted for the pre-specified tail levels, with $p = 90\%$ depicted on the right, for the stock NASDAQ:VABK.

The selection of lower confidence levels, such as 80% and 90% in this study, is motivated by the empirical behavior observed in the simulation results, where these levels provided adequate coverage without overestimating risk. The chi-squared distribution, due to its heavier tails and asymmetry, captures the risk characteristics of volatile stock returns more effectively than the normal distribution. Nonetheless, we acknowledge the importance of quantitative validation; therefore, a future extension of this work will involve conducting a thorough backtesting analysis to assess the coverage accuracy and forecasting performance of these confidence levels through statistical measures such as Kupiec's POF test and Christoffersen's independence test.

To strengthen the analysis, a comprehensive model evaluation is included, assessing the predictive performance of the proposed chi-squared-based VaR and CVaR measures. The evaluation is conducted using several established metrics, including the error variance and the likelihood ratio test. These metrics provide a quantitative measure of the model's ability to predict risk accurately, ensuring that the assumptions underlying the

GARCH(1,1) process and the chi-squared distribution are valid for the chosen dataset. Additionally, the backtesting procedure involves comparing the model's risk estimates with the actual observed outcomes to assess the accuracy of the risk predictions. This is conducted by comparing the exceedance rates of the VaR and CVaR estimates with the chosen confidence levels (e.g., 80% or 90%) over the test period. The model's performance is then validated by checking if the observed violations align with the expected frequency as per the confidence levels.

While the proposed framework based on the chi-squared distribution and GARCH(1,1) modeling offers a robust alternative to traditional normality-based risk measures, several limitations should be acknowledged. Firstly, the assumption of fixed degrees of freedom in the chi-squared distribution may not fully capture the dynamic nature of financial return distributions, especially during periods of market stress. Secondly, the framework does not account for potential leverage effects or asymmetries in volatility, which may be better addressed using GJR-GARCH or EGARCH extensions. Thirdly, the application is limited to univariate time series analysis, whereas multivariate extensions could enhance risk evaluation in diversified portfolios. Additionally, the use of a single distribution family restricts the exploration of other heavy-tailed or skewed distributions that may offer superior tail risk modeling in specific contexts.

6. Concluding Remarks

The selection of an appropriate distribution is essential for accurately estimating the quantiles of financial return distributions. Financial returns often exhibit skewness, excess kurtosis, and fat tails, making standard normal distribution assumptions problematic. If an incorrect distribution is used, the estimated risk measures may underestimate or overestimate the actual risk, leading to either excessive capital reserves or insufficient risk coverage. In this work, we have derived the following:

$$\mathrm{VaR}_p(X) = 2 \cdot Q^{-1}\Big(\frac{m}{2}, 0, p\Big), \quad \mathrm{CVaR}_p(X) = \frac{2\Gamma\big(\frac{m}{2}+1, Q^{-1}\big(\frac{m}{2}, 0, p\big)\big)}{\Gamma\big(\frac{m}{2}, Q^{-1}\big(\frac{m}{2}, 0, p\big)\big)},$$

where $Q^{-1}(\cdot)$ denotes the inverse generalized regularized incomplete gamma function. These results are then operationalized in a risk management context by modeling time-varying volatility in stock returns using the GARCH process. The chi-squared distribution arises as a result of summing the squared values of m independent Gaussian random variables, each possessing a mean of zero and a unit variance. More generalized forms of this distribution can be derived by considering the sum of squares of Gaussian random variables with different statistical properties. An extension of this concept is the noncentral chi-squared distribution, which is attained when summing the squared values of independent Gaussian random variables that maintain a unit variance but have nonzero means. Exploring these extended distributions for the computation of risk measures, like VaR and CVaR, and analyzing their practical relevance in portfolio risk management within the framework of GARCH models presents directions for future research. However, empirical validation of this assumption through statistical goodness-of-fit tests—such as the Kolmogorov–Smirnov or Anderson-Darling tests—was not performed in the current version. Incorporating such tests to compare the chi-squared distribution against alternative models like the Student-t or generalized error distribution is a promising direction for future work to strengthen the empirical basis of the proposed methodology.

The simulation results presented in this study largely align with the theoretical expectations. The use of the chi-squared distribution allowed for capturing the skewness and excess kurtosis present in the financial return distributions, with minimal discrepancies in the VaR and CVaR measures when using confidence levels of 80% or 90%. These

results demonstrate that the chi-squared distribution, when combined with the GARCH model, provides a robust approach for risk quantification, particularly in volatile markets. However, slight deviations in the second experimental case suggest that further refinement, such as incorporating noncentral chi-squared distributions or more advanced GARCH models, could yield even more precise estimates for stocks with extreme volatility.

Author Contributions: Conceptualization, F.S.; Methodology, F.S.; Software, F.S.; Validation, F.S. and T.L.; Formal analysis, F.S.; Investigation, Q.M.; Resources, Q.M.; Data curation, Q.M.; Writing—original draft, Q.M.; Writing—review & editing, Q.M.; Visualization, T.L.; Supervision, T.L.; Project administration, T.L.; Funding acquisition, T.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Scientific Research Project of Jilin Provincial Department of Education (JJKH20251638KJ), the Open Fund Project of Marine Ecological Restoration and Smart Ocean Engineering Research Center of Hebei Province (HBMESO2321), the Technical Service Project of Eighth Geological Brigade of Hebei Bureau of Geology and Mineral Resources Exploration (KJ2022-021), and the Technical Service Project of Hebei Baodi Construction Engineering Co., Ltd. (KJ2024-012).

Data Availability Statement: No special data were utilized.

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

References

1. Albrecher, H.; Binder, A.; Lautscham, V.; Mayer, P. *Introduction to Quantitative Methods for Financial Markets*; Springer: Basel, Switzerland, 2013.
2. Markowitz, H. Portfolio selection. *J. Financ.* **1952**, *7*, 77–91.
3. Halkos, G.E.; Tsirivis, A.S. Value-at-risk methodologies for effective energy portfolio risk management. *Econ. Anal. Policy* **2019**, *62*, 197–212. [CrossRef]
4. Gao, C.-T.; Zhou, X.-H. Forecasting VaR and ES using dynamic conditional score models and skew Student distribution. *Econ. Model.***2016**, *53*, 216–223. [CrossRef]
5. Sarykalin, S.; Serraino, G.; Uryasev, S. Value-at-Risk vs. Conditional Value-at-Risk in Risk Management and Optimization, Tutorials in Operations Research. In *State-of-the-Art Decision-Making Tools in the Information-Intensive Age*; INFORMS: Catonsville, MD, USA, 2008; pp. 270–294.
6. Viet Long, H.; Bin Jebreen, H.; Dassios, I.; Baleanu, D. On the statistical GARCH model for managing the risk by employing a fat-tailed distribution in finance. *Symmetry* **2020**, *12*, 1698. [CrossRef]
7. Braione, M.; Scholtes, N.K. Forecasting value-at-risk under different distributional assumptions. *Econometrics* **2016**, *4*, 3. [CrossRef]
8. Lechner, L.A.; Ovaert, T.C. Value-at-risk: Techniques to account for leptokurtosis and asymmetric behavior in returns distributions. *J. Risk Financ.* **2010**, *11*, 464–480. [CrossRef]
9. De-Graft, E.; Owusu-Ansah, J.; Barnes, B.; Donkoh, E.K.; Appau, J.; Effah, B.; Mccall, M. Quantifying economic risk: An application of extreme value theory for measuring fire outbreaks financial loss. *Financ. Math. Appl.* **2019**, *4*, 1–12.
10. Gilli, M.; Kellezi, E. An application of extreme value theory for measuring financial risk. *Comput. Econ.* **2006**, *27*, 207–228. [CrossRef]
11. Ahmed, D.; Soleymani, F.; Ullah, M.Z.; Hasan, H. Managing the risk based on entropic value-at-risk under a normal-Rayleigh distribution. *Appl. Math. Comput.* **2021**, *402*, 126129. [CrossRef]
12. Wang, G.-J.; Zhu, C.-L. BP-CVaR: A novel model of estimating CVaR with back propagation algorithm. *Econ. Lett.* **2021**, *209*, 110125. [CrossRef]
13. Norton, M.; Khokhlov, V.; Uryasev, S. Calculating CVaR and bPOE for common probability distributions with application to portfolio optimization and density estimation. *Ann. Oper. Res.* **2021**, *299*, 1281–1315. [CrossRef]
14. Wang, Z.-R.; Chen, Y.-H.; Jin, Y.-B.; Zhou, Y.-J. Estimating risk of foreign exchange portfolio: Using VaR and CVaR based on GARCH-EVT-Copula model. *Physica A* **2010**, *389*, 4918–4928. [CrossRef]
15. Jafar, S.H.; Akhtar, S.; El-Chaarani, H.; Khan, P.A.; Binsaddig, R. Forecasting of NIFTY 50 Index Price by Using Backward Elimination with an LSTM Model. *J. Risk Financ. Manag.* **2023**, *16*, 423. [CrossRef]
16. Williams, B. *GARCH*(1, 1) *Models*; Ruprecht-Karls-Universität Heidelberg: Heidelberg, Germany, 2011.
17. Martín, J.; Isabel Parra, M.; Pizarro, M.M.; Sanjuán, E.L. A new Bayesian method for estimation of value at risk and conditional value at risk. *Empir. Econ.* **2024**, *68*, 1171–1189.

18. Herwartz, H. Stock return prediction under GARCH—An empirical assessment. *Int. J. Forecast.* **2017**, *33*, 569–580. [CrossRef]
19. Goode, J.; Kim, Y.S.; Fabozzi, F.J. Full versus quasi MLE for ARMA-GARCH models with infinitely divisible innovations. *Appl. Econ.* **2015**, *47*, 5147–5158. [CrossRef]
20. Bollerslev, T. Generalized autoregressive conditional heteroskedasticity. *J. Econometr.* **1986**, *31*, 307–327. [CrossRef]
21. Davis, R.A.; Mikosch, T. Extreme value theory for GARCH processes. In *Handbook of Financial Time Series*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 187–200.
22. Westfall, P.H. *Understanding Advanced Statistical Methods*; CRC Press: Boca Raton, FL, USA, 2013.
23. Karmakar, M.; Paul, S. Intraday portfolio risk management using VaR and CVaR: A CGARCH-EVT-Copula approach. *Int. J. Forecast.* **2019**, *35*, 699–709. [CrossRef]
24. Georgakopoulos, N.L. *Illustrating Finance Policy with Mathematica*; Springer International Publishing: London, UK, 2018.

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

mathematics

MDPI

Article

The Numerical Approximation of Caputo Fractional Derivatives of Higher Orders Using a Shifted Gegenbauer Pseudospectral Method: A Case Study of Two-Point Boundary Value Problems of the Bagley–Torvik Type

Kareem T. Elgindy [1,2]

[1] Department of Mathematics and Sciences, College of Humanities and Sciences, Ajman University, Ajman P.O. Box 346, United Arab Emirates; k.elgindy@ajman.ac.ae
[2] Nonlinear Dynamics Research Center (NDRC), Ajman University, Ajman P.O. Box 346, United Arab Emirates

Abstract: This paper introduces a novel Shifted Gegenbauer Pseudospectral (SGPS) method for approximating Caputo fractional derivatives (FDs) of an arbitrary positive order. The method employs a strategic variable transformation to express the Caputo FD as a scaled integral of the mth-derivative of the Lagrange interpolating polynomial, thereby mitigating singularities and improving numerical stability. Key innovations include the use of shifted Gegenbauer (SG) polynomials to link mth-derivatives with lower-degree polynomials for precise integration via SG quadratures. The developed fractional SG integration matrix (FSGIM) enables efficient, pre-computable Caputo FD computations through matrix–vector multiplications. Unlike Chebyshev or wavelet-based approaches, the SGPS method offers tunable clustering and employs SG quadratures in barycentric forms for optimal accuracy. It also demonstrates exponential convergence, achieving superior accuracy in solving Caputo fractional two-point boundary value problems (TPBVPs) of the Bagley–Torvik type. The method unifies interpolation and integration within a single SG polynomial framework and is extensible to multidimensional fractional problems.

Keywords: Bagley–Torvik; Caputo; fractional derivative; Gegenbauer polynomials; pseudospectral

MSC: 41A10; 65D30; 65L60

1. Introduction

Fractional calculus generalizes calculus by allowing differentiation and integration to arbitrary real orders. This framework provides a powerful tool for modeling memory effects, long-range interactions, and anomalous diffusion—phenomena commonly observed in scientific and engineering applications. Unlike classical integer-order models, which assume purely local and instantaneous interactions, fractional-order models naturally incorporate non-locality and history dependence. This feature allows them to more accurately represent real-world processes such as viscoelasticity, dielectric polarization, electrochemical reactions, and subdiffusion in disordered media [1–3]. Moreover, fractional-order models often require fewer parameters to match or exceed the accuracy of classical models in representing complex dynamics, making them both efficient and descriptive [4].

The key advantage of FDs over classical derivatives lies in their capacity to capture hereditary characteristics and long-range temporal correlations, which are particularly relevant in biological systems [5], control systems [6], and viscoelastic materials [3]. For

instance, traditional damping models use exponential kernels that decay too quickly to accurately capture certain relaxation behaviors. In contrast, fractional models employ power-law kernels, enabling them to describe slower and more realistic decay rates [7].

Among the various definitions of FDs, the Caputo FD is particularly popular due to its compatibility with classical initial and boundary conditions, which allows seamless integration with standard numerical and analytical techniques for solving fractional differential equations. Unlike the Riemann–Liouville FD, the Caputo FD defines the FD of a constant as zero, simplifying the mathematical treatment of steady-state solutions and improving the applicability of collocation methods. A prominent example of its application is the Bagley–Torvik equation, a well-known fractional differential equation involving a Caputo derivative of order 1.5. This equation models the motion of a rigid plate immersed in a viscous fluid, where the FD term represents a damping force that depends on the history of the plate's motion. Such damping—referred to as fractional or viscoelastic damping—is commonly used to model materials exhibiting memory effects.

Recent advances in finite-time stability analysis for fractional systems (e.g., [8]) underscore the growing demand for robust numerical methods. In particular, the numerical approximation of FDs and the solution to equations such as the Bagley–Torvik equation remain active and challenging areas of research. These developments highlight the necessity of stable, efficient, and highly accurate numerical methods capable of capturing the complex dynamics inherent in fractional-order models. Several numerical studies have demonstrated that classical methods struggle to maintain accuracy or stability when adapted to fractional settings due to the singular kernel behavior of fractional integrals, especially near the origin [9]. Hence, developing dedicated fractional methods that respect the non-local structure of the problem is crucial for realistic simulations. In the following, we mention some of the key contributions to the numerical solution to the Bagley–Torvik equation using the Caputo FD: **Spectral Methods:** Saw and Kumar [10] proposed a Chebyshev collocation scheme for solving the fractional Bagley–Torvik equation. The Caputo FD was handled through a system of algebraic equations formed using Chebyshev polynomials and specific collocation points. Ji et al. [11] presented a numerical solution using SC polynomials. The Caputo derivative was expressed using an operational matrix of FDs, and the fractional-order differential equation was reduced to a system of algebraic equations that was solved using Newton's method. Hou et al. [12] solved the Bagley–Torvik equation by converting the differential equation into a Volterra integral equation, which was then solved using Jacobi collocation. Ji and Hou [13] applied Laguerre polynomials to approximate the solution to the Bagley–Torvik equation. The Laplace transform was first used to convert the problem into an algebraic equation, and then, Laguerre polynomials were used for numerical inversion. **Wavelet-Based Methods:** Kaur et al. [14] developed a hybrid numerical method using non-dyadic wavelets for solving the Bagley–Torvik equation. Dincel [15] employed sine–cosine wavelets to approximate the solution to the Bagley–Torvik equation, where the Caputo FD was computed using the operational matrix of fractional integration. Rabiei and Razzaghi [16] introduced a wavelet-based technique, utilizing the Riemann–Liouville integral operator to transform the fractional Bagley–Torvik equation into algebraic equations. **Operational Matrix Methods:** Abd-Elhameed and Youssri [17] formulated an operational matrix of FDs in the Caputo sense using Lucas polynomials, and applied Tau and collocation methods to solve the Bagley–Torvik equation. Youssri [18] introduced an operational matrix approach using Fermat polynomials for solving the fractional Bagley–Torvik equation in the Caputo sense. A spectral tau method was employed to transform the problem into algebraic equations. **Galerkin Methods:** Izadi and Negar [19] used a local discontinuous Galerkin scheme with upwind fluxes for solving the Bagley–Torvik equation. The Caputo derivative was approximated by discretizing

elementwise systems. Chen [20] proposed a fast multiscale Galerkin algorithm using orthogonal functions with vanishing moments. **Spline and Finite Difference Methods:** Tamilselvan et al. [21] used a second-order spline approximation for the Caputo FD and a central difference scheme for the second-order derivative term in solving the Bagley–Torvik equation. **Artificial Intelligence-Based Methods:** Verma and Kumar [22] employed an artificial neural network method with Legendre polynomials to approximate the solution to the Bagley–Torvik equation, where the Caputo derivative was handled through an optimization-based training process.

This work introduces a novel framework for approximating Caputo FDs of any positive orders using an SGPS method. Unlike traditional approaches, our method employs a strategic change of variables to transform the Caputo FD into a scaled integral of the mth-derivative of the Lagrange interpolating polynomial, where m is the ceiling of the fractional order α. This transformation mitigates the singularity inherent in the Caputo derivative near zero, thereby improving numerical stability and accuracy. The numerical approximation of the Caputo FD is finally furnished by linking the mth-derivative of SG polynomials with another set of SG polynomials of lower degrees and higher parameter values whose integration can be recovered within excellent accuracies using SG quadratures. By employing orthogonal collocation and SG quadratures in barycentric form, we achieve a highly accurate, computationally efficient, and stable scheme for solving fractional differential equations under optimal parameter settings compared to classical PS methods. Furthermore, we provide a rigorous error analysis showing that the SGPS method is convergent when implemented within a semi-analytic framework, where all necessary integrals are computed analytically, and is conditionally convergent with an exponential rate of convergence for sufficiently smooth functions when performed using finite-precision arithmetic. This exponential convergence generally leads to superior accuracy compared to existing wavelet-based, operational matrix, and finite difference methods. We conduct rigorous error and convergence analyses to derive the total truncation error bound of the method and study its asymptotic behavior within double-precision arithmetic. The SGPS is highly flexible in the sense that the SG parameters associated with SG interpolation and quadratures allow for flexibility in adjusting the method to suit different types of problems. These parameters influence the clustering of collocation and quadrature points and can be tuned for optimal performance. A key contribution of this work is the development of the FSGIM. This matrix facilitates the direct computation of Caputo FDs through efficient matrix–vector multiplications. Notably, the FSGIM is constant for a given set of points and parameter values. This allows for pre-computation and storage, significantly accelerating the execution of the SGPS method. The SGPS method avoids the need for extended precision arithmetic, as it remains within the limits of double-precision computations, making it computationally efficient compared to methods that require high-precision arithmetic. The current approach is designed to handle any positive fractional order α, making it more flexible than some existing methods that are constrained to specific fractional orders. Unlike Chebyshev polynomials (fixed clustering) or wavelets (local support), SG polynomials offer tunable clustering via their index λ, optimizing accuracy for smooth solutions, while their derivative properties enable efficient FD computation, surpassing finite difference methods in convergence rate. The efficacy of our approach is demonstrated through its application to Caputo fractional TPBVPs of the Bagley–Torvik type, where it outperforms existing numerical schemes. The method's framework supports extension to multidimensional and time-dependent fractional PDEs through the tensor products of FSGIMs. By integrating interpolation and integration into a cohesive SG polynomial-based approach, it provides a unified solution framework for fractional differential equations.

The remainder of this paper is structured as follows. Section 2 introduces the SGPS method, providing a detailed exposition of its theoretical framework and numerical implementation. The computational complexity of the derived FSGIM is discussed in Section 3. A comprehensive error analysis of the method is carried out in Section 4, establishing its convergence properties and providing insights into its accuracy. In Section 5, we demonstrate the effectiveness of the SGPS method through a case study, focusing on its application to Caputo fractional TPBVPs of the Bagley–Torvik type. Section 6 presents a series of numerical examples, demonstrating the superior performance of the SGPS method in comparison to existing techniques. Section 7 conducts a sensitivity analysis to investigate the impact of the SG parameters on the numerical stability of the SGPS method, providing practical insights into parameter selection for relatively small interpolation and quadrature mesh sizes. Finally, Section 8 concludes the paper with a summary of our key findings and a discussion of potential future research directions. Table 1 and the list of acronyms display the symbols and acronyms used in the paper and their meanings. A pseudocode for the SGPS method to solve Bagley–Torvik TPBVPs is provided in Appendix A. Appendix B supports the error analysis conducted in Section 4 by providing rigorous mathematical justifications for the asymptotic order of some key terms in the error bound.

Table 1. Table of symbols and their meanings.

Symbol	Meaning	Symbol	Meaning	Symbol	Meaning
$\forall$	for all	$\forall_a$	for any	$\forall_{aa}$	for almost all
$\forall_e$	for each	$\forall_s$	for some	$\forall_{rs}$	for (a) relatively small
$\forall_{rl}$	for (a) relatively large	$\ll$	much less than	$\not\ll$	not much less than
$\gg$	much greater than	$\exists$	there exist(s)	$\sim$	asymptotically equivalent
$\precsim$	asymptotically less than	$\eqsim$	asymptotically less than or equal to	$\not\approx$	not sufficiently close to
$\mathfrak{C}$	set of all complex-valued functions	$\mathfrak{F}$	set of all real-valued functions	$\mathbb{C}$	set of complex numbers
$\mathbb{R}$	set of real numbers	$\mathbb{R}_0$	set of non-negative real numbers	$\mathbb{R}_\theta$	set of nonzero real numbers
$\mathbb{R}^-_{-1/2}$	$\{x \in \mathbb{R} : -1/2 < x < 0\}$	$\mathbb{Z}$	set of integers	$\mathbb{Z}^+$	set of positive integers
$\mathbb{Z}^+_0$	set of non-negative integers	$\mathbb{Z}^+_e$	set of positive even integers	$i{:}j{:}k$	list of numbers from i to k with increment j
$i{:}k$	list of numbers from i to k with increment 1	$y_{1:n}$ or $y_i\big\vert_{i=1:n}$	list of symbols $y_1, y_2, \ldots, y_n$	$\{y_{1:n}\}$	set of symbols $y_1, y_2, \ldots, y_n$
$\mathbb{J}_n$	$\{0 : n-1\}$	$\mathbb{J}^+_n$	$\mathbb{J}_n \cup \{n\}$	$\mathbb{N}_n$	$\{1 : n\}$
$\mathbb{N}_{m,n}$	$\{m : n\}$	$\mathbb{G}^\lambda_n$	set of GG zeros of the $(n+1)$st-degree Gegenbauer polynomial with index $\lambda > -1/2$	$\hat{\mathbb{G}}^\lambda_n$	set of SGG points in the interval $[0, 1]$
$\mathbf{\Omega}_{a,b}$	closed interval $[a, b]$	$\mathbf{\Omega}^\circ$	interior of the set $\mathbf{\Omega}$	$\mathbf{\Omega}_T$	specific interval $[0, T]$
$\mathbf{\Omega}_{L\times T}$	Cartesian product $\mathbf{\Omega}_L \times \mathbf{\Omega}_T$	$\Gamma(\cdot)$	Gamma function	$\Gamma(\cdot,\cdot)$	upper incomplete gamma function
$\lceil . \rceil$	ceiling function	$\mathfrak{I}_{j\geq k}$	indicator (characteristic) function $\begin{cases}1 & \text{if } j \geq k,\\ 0 & \text{otherwise.}\end{cases}$	$E_{\alpha,\beta}(z)$	two-parameter Mittag–Leffler function
$(\cdot)_n$	Pochhammer symbol	$\mathrm{supp}(f)$	support of function f	f^*	complex conjugate of f
f_n	$f(t_n)$	$f_{N,n}$	$f_N(t_n)$	$\mathcal{I}^{(t)}_b h$	$\int_0^b h(t)\,dt$
$\mathcal{I}^{(t)}_{a,b} h$	$\int_a^b h(t)\,dt$	$\mathcal{I}^{(t)}_t h$	$\int_0^t h(.)\,d(.)$	$\mathcal{I}^{(t)}_b h\{u(t)\}$	$\int_0^b h(u(t))\,dt$
$\mathcal{I}^{(t)}_{a,b} h\{u(t)\}$	$\int_a^b h(u(t))\,dt$	$\mathcal{I}^{(x)}_{\mathbf{\Omega}_{a,b}} h$	$\int_a^b h(x)\,dx$	∂_x	d/dx
∂^n_x	d^n/dx^n	${}^cD^\alpha_x f$	αth-order Caputo FD of f at x given by ${}^cD^\alpha_x f = \frac{1}{\Gamma(\lceil\alpha\rceil-\alpha)}\int_0^x \frac{f^{(\lceil\alpha\rceil)}(t)}{(x-t)^{\alpha-\lceil\alpha\rceil+1}}\,dt$	$\mathrm{Def}(\mathbf{\Omega})$	space of all functions defined on $\mathbf{\Omega}$
$C^k(\mathbf{\Omega})$	space of k times continuously differentiable functions on $\mathbf{\Omega}$	$L^p(\mathbf{\Omega})$	Banach space of measurable functions $u \in \mathrm{Def}(\mathbf{\Omega})$ with $\|u\|_{L^p} = \left(\mathcal{I}_{\mathbf{\Omega}}|u|^p\right)^{1/p} < \infty$	$L^\infty(\mathbf{\Omega})$	space of all essentially bounded measurable functions on $\mathbf{\Omega}$
$\|f\|_{L^\infty(\mathbf{\Omega})}$	L^∞ norm: $\sup_{x\in\mathbf{\Omega}}\|f(x)\| = \inf\{M \geq 0 : \|f(x)\| \leq M\ \forall_{aa}\, x \in \mathbf{\Omega}\}$	$\|\cdot\|_1$	l_1-norm	$\|\cdot\|_2$	Euclidean norm

Table 1. *Cont.*

Symbol	Meaning	Symbol	Meaning	Symbol	Meaning
$\mathcal{H}^{k,p}(\mathbf{\Omega})$	Sobolev space of weakly differentiable functions with integrable weak derivatives up to order k	$\boldsymbol{t}_N$	$[t_{N,0}, t_{N,1}, \ldots, t_{N,N}]^\top$	$g_{0:N}$	$[g_0, g_1, \ldots, g_N]^\top$
$g^{(0:N)}$	$[g, g', \ldots, g^{(N)}]^\top$	$c^{0:N}$	$[1, c, c^2, \ldots, c^N]$	$\boldsymbol{t}_N^\top$ or $[t_{N,0:N}]$	$[t_{N,0}, t_{N,1}, \ldots, t_{N,N}]$
$h(\boldsymbol{y})$	vector with i-th element $h(y_i)$	$\boldsymbol{h}(\boldsymbol{y})$ or $h_{1:m}[\boldsymbol{y}]$	$[h_1(\boldsymbol{y}), \ldots, h_m(\boldsymbol{y})]^\top$	$\boldsymbol{y}^{\div}$	vector of reciprocals of the elements of $\boldsymbol{y}$
$\mathbf{O}_n$	zero matrix of size n	$\mathbf{1}_n$	all-ones matrix of size n	$\mathbf{I}_n$	identity matrix of size n
$\mathbf{C}_{n,m}$	matrix $\mathbf{C}$ of size $n \times m$	$\mathbf{C}_n$	n-th row of matrix $\mathbf{C}$	$\boldsymbol{1}_n$	n-dimensional all ones column vector
$\boldsymbol{0}_n$	n-dimensional all-zeros column vector	$\mathbf{A}^\top$ or $\mathrm{trp}(\mathbf{A})$	transpose of matrix $\mathbf{A}$	$\mathrm{diag}(\boldsymbol{v})$	diagonal matrix with $\boldsymbol{v}$ on the diagonal
$\mathrm{resh}_{m,n}(\mathbf{A})$	reshape $\mathbf{A}$ into an $m \times n$ matrix	$\mathrm{resh}_n(\mathbf{A})$	reshape $\mathbf{A}$ into a square matrix of size n	$\kappa(\mathbf{A})$	condition number of $\mathbf{A}$
$\otimes$	Kronecker product	$\odot$	Hadamard product	$\mathbf{A}_{(r)}$	r-times Hadamard product of $\mathbf{A}$
$\mathbf{A}^{\circ m}$	each entry in $\mathbf{A}$ raised to the power m	$f(n) = O(g(n)) \nexists\, n_0, c > 0 : 0 \le f(n) \le cg(n)\ \forall n \ge n_0$		$f(n) = o(g(n))$	$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$

Remark : A vector is represented in print by a bold italicized symbol, while a two-dimensional matrix is represented by a bold symbol, except for a row vector whose elements form a certain row of a matrix, which is represented by a bold symbol.

2. The SGPS Method

This section introduces the SGPS method for approximating Caputo FDs. Readers interested in obtaining a deeper understanding of Gegenbauer and SG polynomials, as well as their associated quadratures, are encouraged to consult [23–26].

Let $\alpha \in \mathbb{R}^+ \backslash \mathbb{Z}^+, m = \lceil \alpha \rceil, f \in \mathcal{H}^{m,2}(\mathbf{\Omega}_1), \left\{ \hat{x}^\lambda_{n,0:n} \right\} = \hat{\mathbb{G}}^\lambda_n$, and consider the following SGPS interpolant of f:

$$I_n f(x) = f^\top_{0:n} \mathcal{L}^\lambda_{0:n}[x], \tag{1}$$

where $\mathcal{L}^\lambda_k(x)$ is the nth-degree Lagrange interpolating polynomial in modal form defined by

$$\mathcal{L}^\lambda_k(x) = \hat{\omega}^\lambda_k \,\mathrm{trp}\left(\hat{\lambda}^{\lambda\,\div}_{0:n}\right)\left(\hat{G}^\lambda_{0:n}[\hat{x}^\lambda_{n,k}] \odot \hat{G}^\lambda_{0:n}[x]\right), \quad \forall k \in \mathbb{J}^+_n; \tag{2}$$

$\hat{\lambda}^\lambda_{0:n}$ and $\hat{\omega}^\lambda_{0:n}$ are the normalization factors for SG polynomials and the Christoffel numbers associated with their quadratures, respectively,

$$\hat{\lambda}^\lambda_j = \frac{\pi 2^{1-4\lambda}\Gamma(j+2\lambda)}{j!\Gamma^2(\lambda)(j+\lambda)},$$
$$\hat{\omega}^\lambda_k = 1/\left[\mathrm{trp}\left(\hat{\lambda}^{\lambda\,\div}_{0:n}\right)\left(\hat{G}^\lambda_{0:n}[\hat{x}^\lambda_{n,k}]\right)_{(2)}\right];$$

and $\forall j, k \in \mathbb{J}^+_n$ (cf. Equations (2.6), (2.7), (2.10), and (2.12) in [23]). The matrix form of Equation (2) can be stated as

$$\mathcal{L}^\lambda_{0:n}[x] = \mathrm{diag}\left(\hat{\omega}^\lambda_{0:n}\right)\left(\hat{G}^\lambda_{0:n}[x\boldsymbol{1}_{n+1}] \odot \hat{G}^\lambda_{0:n}[\hat{\boldsymbol{x}}^\lambda_n]\right)^\top \hat{\lambda}^{\lambda\,\div}_{0:n}. \tag{3}$$

Equation (1) allows us to approximate the Caputo FD of f:

$${}^cD^\alpha_x f \approx {}^cD^\alpha_x I_n f = f^\top_{0:n}\, {}^cD^\alpha_x \mathcal{L}^\lambda_{0:n}. \tag{4}$$

To accurately evaluate ${}^cD^\alpha_x \mathcal{L}^\lambda_{0:n}$, we apply the following m-dependent change of variables:

$$\tau = x\left(1 - y^{\frac{1}{m-\alpha}}\right),$$

which reduces ${}^{c}D_{x}^{\alpha}f$ to a scalar multiple of the integral of the mth-derivative of f on the fixed interval $\boldsymbol{\Omega}_1$, denoted by ${}^{E}D_{x}^{\alpha}f$, and defined by

$$ {}^{E}D_{x}^{\alpha}f \;=\; \frac{x^{m-\alpha}}{\Gamma(m-\alpha+1)}\mathcal{I}_1^{(y)}f^{(m)}\{x\left(1-y^{\frac{1}{m-\alpha}}\right)\}. \tag{5} $$

It is easy here to show that the value of $x\left(1-y^{\frac{1}{m-\alpha}}\right)$ will always lie in the range $\boldsymbol{\Omega}_x \,\forall\, 0 \leq x, y \leq 1$. Combining Equations (4) and (5) gives

$$ {}^{c}D_{x}^{\alpha}f \approx \frac{x^{m-\alpha}}{\Gamma(m-\alpha+1)}f_{0:n}^{\top}\mathcal{I}_1^{(y)}\mathcal{L}_{0:n}^{\lambda,m}\{x\left(1-y^{\frac{1}{m-\alpha}}\right)\}, \tag{6} $$

where $\mathcal{L}_j^{\lambda,m}$ denotes the mth-derivative of $\mathcal{L}_j^{\lambda}\,\forall j \in \mathbb{J}_n^{+}$. Substituting Equation (3) into Equation (6) yields

$$ {}^{c}D_{x}^{\alpha}f \approx \frac{x^{m-\alpha}}{\Gamma(m-\alpha+1)}\left[\operatorname{trp}\left(\hat{\lambda}_{m:n}^{\lambda^{\div}}\right)\times\right. $$

$$ \left.\left(\mathcal{I}_1^{(y)}\hat{G}_{m:n}^{\lambda,m}\{\left(x-xy^{\frac{1}{m-\alpha}}\right)\boldsymbol{1}_{n+1}\}\odot\hat{G}_{m:n}^{\lambda}\{\hat{\boldsymbol{x}}_n^{\lambda}\}\right)\operatorname{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right)\right]f_{0:n}, \tag{7} $$

where $\hat{G}_j^{\lambda,m}$ denotes the mth-derivative of $\hat{G}_j^{\lambda}\,\forall j \in \mathbb{N}_{m,n}$.

To efficiently evaluate Caputo FDs at arbitrary points $z_{0:M} \in \boldsymbol{\Omega}_1 \;\forall\, M \in \mathbb{Z}_0^{+}$, Formula (7) can be applied iteratively within a loop. While direct implementation using a loop over the vector's elements of $\boldsymbol{z}_M$ is possible, employing matrix operations is highly recommended for substantial performance gains. To this end, notice first that Equation (3) can be rewritten at $\boldsymbol{z}_M$ as

$$ \mathcal{L}_{0:n}^{\lambda}\{\boldsymbol{z}_M\} \;=\; \operatorname{resh}_{n+1,M+1}\left[\operatorname{trp}\left(\hat{\lambda}_{0:n}^{\lambda^{\div}}\right)\times\left(\hat{G}_{0:n}^{\lambda}\{\boldsymbol{z}_M\otimes\boldsymbol{1}_{n+1}\}\odot\hat{G}_{0:n}^{\lambda}\{\boldsymbol{1}_{M+1}\otimes\hat{\boldsymbol{x}}_n^{\lambda}\}\right)\times\right. $$

$$ \left.\left(\mathbf{I}_{M+1}\otimes\operatorname{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right)\right)\right]. \tag{8} $$

Equation (8) together with (6) yield:

$$ {}^{c}D_{\boldsymbol{z}_M}^{\alpha}f \approx \frac{1}{\Gamma(m-\alpha+1)}\left[{\boldsymbol{z}_M}^{\circ(m-\alpha)}\odot\left({}^{E}\hat{\mathbf{Q}}_n^{\alpha}f_{0:n}\right)\right], \tag{9} $$

where

$$ {}^{E}\hat{\mathbf{Q}}_n^{\alpha} \;=\; \operatorname{resh}_{n+1,M+1}^{\top}\left[\operatorname{trp}\left(\hat{\lambda}_{m:n}^{\lambda^{\div}}\right)\times\right. $$
$$ \left(\mathcal{I}_1^{(y)}\hat{G}_{m:n}^{\lambda,m}\{\boldsymbol{z}_M\otimes\left(\left(1-y^{\frac{1}{m-\alpha}}\right)\boldsymbol{1}_{n+1}\right)\}\odot\hat{G}_{m:n}^{\lambda}\{\boldsymbol{1}_{M+1}\otimes\hat{\boldsymbol{x}}_n^{\lambda}\}\right)\times $$
$$ \left.\left(\mathbf{I}_{M+1}\otimes\operatorname{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right)\right)\right]. $$

With simple algebraic manipulation, we can further show that Equation (9) can be rewritten as

$$ {}^{c}D_{\boldsymbol{z}_M}^{\alpha}f \approx {}^{E}\mathbf{Q}_n^{\alpha}f_{0:n}, \tag{10} $$

where

$$ {}^{E}\mathbf{Q}_n^{\alpha} \;=\; \frac{1}{\Gamma(m-\alpha+1)}\operatorname{diag}\left({\boldsymbol{z}_M}^{\circ(m-\alpha)}\right){}^{E}\hat{\mathbf{Q}}_n^{\alpha}. \tag{11} $$

We refer to the $(M+1)\times(n+1)$ matrix ${}^{E}\mathbf{Q}_n^{\alpha}$ as *"the αth-order FSGIM,"* which approximates Caputo FD at the points $z_{0:M}$ using an nth-degree SG interpolant. We also refer to ${}^{E}\hat{\mathbf{Q}}_n^{\alpha}$ as

the *"αth-order FSGIM Generator"* for an obvious reason. Although the implementation of Formula (10) is straightforward, Formula (9) is slightly more stable numerically, with fewer arithmetic operations, particularly because it avoids constructing a diagonal matrix and directly applies elementwise multiplication after the matrix–vector product. Note that for $M = 0$, Formulas (9) and (10) reduce to (7).

It remains now to show how to compute

$$\mathcal{I}_1^{(y)} \hat{G}_j^{\lambda,m}[x\left(1 - y^{\frac{1}{m-\alpha}}\right)], \quad \forall_a j \in \mathbb{N}_{m:n}, \quad x \in \mathbf{\Omega}_1,$$

effectively. Notice first that although the integrand is defined in terms of a polynomial in x, the integrand itself is not a polynomial in y, since $1/(m-\alpha)$ is not an integer for $\alpha \in \mathbb{R}^+ \backslash \mathbb{Z}^+$. Therefore, when trying to evaluate the integral symbolically, the process can be very challenging and slow. Numerical integration, on the other hand, is often more practical for such integrals because it can achieve any specified accuracy by evaluating the integrand at discrete points without requiring closed-form antiderivatives or algebraic complications. Our reliable tool for this task is the SGIM; cf. [23,25] and the references therein. The SGIM utilizes the barycentric representation of shifted Lagrange interpolating polynomials and their associated barycentric weights to approximate definite integrals effectively through matrix–vector multiplications. The SGPS quadratures constructed by these operations extend the classical Gegenbauer quadrature methods and can improve their performance in terms of convergence speed and numerical stability. An efficient way to construct the SGIM is to premultiply the corresponding GIM by half, rather than shifting the quadrature nodes, weights, and Lagrange polynomials to the target domain $\mathbf{\Omega}_1$, as shown earlier in [23]. In the current work, we only need the GIRV, $\mathbf{P}$, which extends the applicability of the barycentric GIM to include the boundary point 1 (cf. [24] Algorithm 6 or 7). The associated SGIRV, $\hat{\mathbf{P}}$, can be directly generated through the formula

$$\hat{\mathbf{P}} = \frac{1}{2}\mathbf{P}.$$

Given that the construction of $\hat{\mathbf{P}}$ is independent of the SGPS interpolant (1), we can define $\hat{\mathbf{P}}$ using any set of SGG quadrature nodes $\hat{\mathbb{G}}_{n_q}^{\lambda_q} \ \forall_s n_q \in \mathbb{Z}_0^+, \lambda_q > -1/2$. This flexibility enables us to improve the accuracy of the required integrals without being constrained by the resolution of the interpolation grid. With this strategy, the SGIRV provides a convenient way to approximate the required integral through the following matrix–vector multiplication:

$$\mathcal{I}_1^{(y)} \hat{G}_j^{\lambda,m}[x - xy^{\frac{1}{m-\alpha}}] \approx \hat{\mathbf{P}}\, \hat{G}_j^{\lambda,m}\left(x\left(1 - \left(\hat{\boldsymbol{x}}_{n_q}^{\lambda_q}\right)^{\circ \frac{1}{m-\alpha}}\right)\right), \tag{12}$$

$\forall_a j \in \mathbb{N}_{m:n}, x \in \mathbf{\Omega}_1$. We refer to a quadrature of the form (12) as the (n_q, λ_q)-SGPS quadrature. A remarkable property of Gegenbauer polynomials (and their shifted counterparts) is that their derivatives are essentially other Gegenbauer polynomials, albeit with different degrees and parameters, as shown by the following theorem.

Theorem 1. *The mth-derivatives of the nth-degree, λ-indexed, Gegenbauer and SG polynomials are given by*

$$G_n^{\lambda,m}(x) = \chi_{n,m}^{\lambda}\, G_{n-m}^{\lambda+m}(x), \tag{13a}$$

$$\hat{G}_n^{\lambda,m}(\hat{x}) = \hat{\chi}_{n,m}^{\lambda}\, \hat{G}_{n-m}^{\lambda+m}(\hat{x}), \tag{13b}$$

where

$$\chi_{n,m}^{\lambda} \;=\; \frac{2^m\, n!\,\Gamma(2\lambda)\,(\lambda)_m\,\Gamma(m+n+2\lambda)}{(n-m)!\,\Gamma(2(m+\lambda))\,\Gamma(n+2\lambda)},$$

$$\hat{\chi}_{n,m}^{\lambda} \;=\; 2^m\,\chi_{n,m}^{\lambda} \;=\; \frac{n!\,\Gamma(\lambda+1/2)\,\Gamma(n+m+2\lambda)}{(n-m)!\,\Gamma(n+2\lambda)\,\Gamma(m+\lambda+1/2)}, \tag{14}$$

$\forall n \geq m, x \in \mathbf{\Omega}_{-1,1}$, *and* $\hat{x} \in \mathbf{\Omega}_1$.

Proof. Let $C_n^{\lambda}(x)$ be the nth-degree, λ-indexed Gegenbauer polynomial standardized by Szegö [27]. We shall first prove that

$$C_n^{\lambda,m}(x) \;=\; 2^m\,(\lambda)_m\,C_{n-m}^{\lambda+m}(x), \quad \forall n \geq m, \tag{15}$$

where $C_j^{\lambda,m}$ denotes the mth-derivative of $C_j^{\lambda}\ \forall j \in \mathbb{N}_{m,n}$. To this end, we shall use the well-known derivative formula of this polynomial given by the following recurrence relation:

$$C_n^{\lambda,1}(x) \;=\; 2\,\lambda\,C_{n-1}^{\lambda+1}(x), \quad n \geq 1.$$

We will prove Equation (15) through mathematical induction on m. The base case $m \;=\; 1$ holds true due to the given recurrence relation for the first derivative. Assume now that Equation (15) holds true for $m \;=\; k$, where k is an arbitrary integer such that $1 < k \leq n-1$. That is,

$$C_n^{\lambda,k}(x) \;=\; 2^k\,(\lambda)_k\,C_{n-k}^{\lambda+k}(x).$$

We need to show that it also holds true for $m = k+1$. Differentiating both sides of the induction hypothesis with respect to x gives

$$\begin{aligned} C_n^{\lambda,k+1}(x) \;&=\; \frac{d}{dx}\left[C_n^{\lambda,k}(x)\right] \;=\; \frac{d}{dx}\left[2^k(\lambda)_k\,C_{n-k}^{\lambda+k}(x)\right] \\ = 2^k\,(\lambda)_k\,\frac{d}{dx}\left[C_{n-k}^{\lambda+k}(x)\right] \;&=\; 2^k\,(\lambda)_k\cdot 2(\lambda+k)\,C_{n-k-1}^{\lambda+k+1}(x) \\ &= 2^{k+1}\,(\lambda)_{k+1}\,C_{n-k-1}^{\lambda+k+1}(x). \end{aligned}$$

This shows that if the formula holds for $m = k$, it also holds for $m = k+1$. Through mathematical induction, Equation (15) holds true for all integers $m : 0 \leq m \leq n$. Formula ([28] (A.5)) and the fact that

$$C_n^{\lambda}(1) \;=\; \frac{\Gamma(n+2\lambda)}{\Gamma(n+1)\,\Gamma(2\lambda)},$$

immediately show that

$$G_n^{\lambda,m}(x) \;=\; 2^m(\lambda)_m\frac{C_{n-m}^{\lambda+m}(1)}{C_n^{\lambda}(1)}G_{n-m}^{\lambda+m}(x), \quad \forall n \geq m,$$

from which Equation (13a) is derived. Formula (13b) follows from (13a) through successive application of the Chain Rule. □

Equations (12) and (13b) bring to light the sought formula

$$\mathcal{I}_1^{(y)}\hat{G}_j^{\lambda,m}[x - xy^{\frac{1}{m-\alpha}}] \approx \hat{\chi}_{j,m}^{\lambda}\left[\hat{\mathbf{P}}\,\hat{G}_{j-m}^{\lambda+m}\left(x\left(1-\left(\hat{\boldsymbol{x}}_{n_q}^{\lambda_q}\right)^{\circ\frac{1}{m-\alpha}}\right)\right)\right], \tag{16}$$

where $\forall j \in \mathbb{N}_{m:n}, x \in \mathbf{\Omega}_1$. Figure 1 illustrates the key polynomial transformations in the SGPS method, where lower-degree SG polynomials serve as scaled transformations of the derivative terms. We denote the approximate αth-order Caputo FD of a function at point x, computed using Equation (16) in conjunction with either Equations (9) or (10), as ${}^{n,\lambda,E}_{n_q,\lambda_q}D^{\alpha}_{x}$. It is interesting to notice here that the quadrature nodes involved in the computations of the necessary integrals (16), which are required for the construction of the FSGIM ${}^{n,\lambda,E}_{n_q,\lambda_q}D^{\alpha}_{x}$, are independent of the SGG points associated with the SGPS interpolant (1), and therefore, any set of SGG quadrature nodes can be used. This flexibility allows for improving the accuracy of the required integrals without being constrained by the resolution of the interpolation grid.

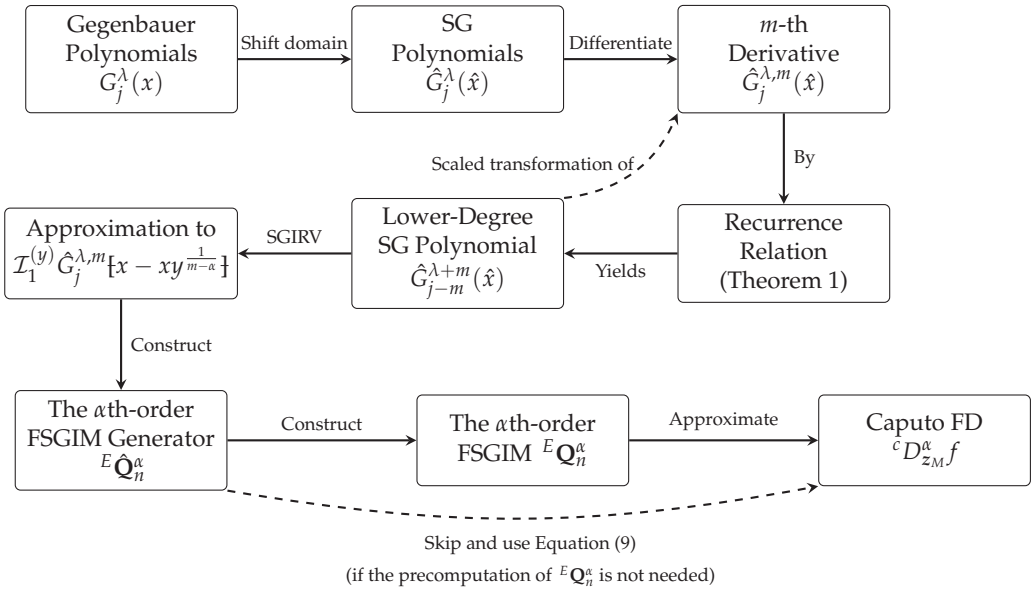


Figure 1. Key relationships in the SGPS method showing the polynomial transformations and their computational roles. The lower-degree SG polynomial $\hat{G}_{j-m}^{\lambda+m}(\hat{x})$ serves as a scaled transformation of the derivative term $\hat{G}_j^{\lambda,m}\left(x - xy^{\frac{1}{m-\alpha}}\right)$ through Theorem 1, and can be numerically integrated with high precision to approximate the necessary integrals of the mth-derivatives of higher-degree SG polynomials. The approximation is then used to construct the αth-order FSGIM generator, which directly generates the αth-order FSGIM. The FSGIM is finally used to approximate the Caputo FD at the required nodes. The required Caputo FD approximation can also be obtained directly by using the generator matrix through Equation (9).

Figure 2 illustrates the logarithmic absolute errors of Caputo FD approximations for $f_1(t) = t^N$. These approximations utilize SG interpolants of varying parameters but consistent degrees, in conjunction with a $(15, 0.5)$-SGPS quadrature. The exact Caputo FD of f_1 is given below:

$${}^{c}D_t^{\alpha}f_1 = \begin{cases} \frac{N!}{\Gamma(N+1-\alpha)}t^{N-\alpha}, & N > \alpha - 1, \\ 0, & N \leq \alpha - 1, \end{cases} \quad \forall N \in \mathbb{Z}_0^+, \alpha \in \mathbb{R}^+.$$

In all plots of Figure 2, the rapid convergence of the PS approximations is evident. Given that the SG interpolants share the same polynomial degree as the power function, and since $f_1^{(n+1)} \equiv 0$, the interpolation error vanishes, as we demonstrate later with Theorem 2 in Section 4. Consequently, the quadrature error becomes the dominant component. Theorem 4 in Section 4 further indicates that the quadrature error vanishes for $n < n_q + m + 1$, which elucidates the high accuracy achieved by the SGPS method in all four plots when n is sufficiently less than $n_q + m + 1$ in many cases, leading to a near-machine epsilon level

of the total error. While the error analysis in Section 4 predicts the collapse of the total error when $n \leq n_q + m$ under exact arithmetic, the limitations of finite-digit arithmetic often prevent this, frequently necessitating an increase in n_q by one unit or more, especially when varying λ_q, for effective total error collapse. In Subplot 1, with $n_q = 4$, an nth-degree SG interpolant sufficiently approximates the Caputo FD of the power function t^n to within machine precision for $2 \leq n \leq 5$. The error curves exhibit plateaus in this range, with slight fluctuations for specific λ values, attributed to accumulated round-off errors as the approximation approaches machine precision. For $6 \leq n \leq 10$, the total error becomes predominantly the quadrature error and remains relatively stable around 10^{-4}. Notably, the error profiles remain consistent for $6 \leq n \leq 10$ despite variations in λ. Altering λ_q while keeping λ constant can significantly impact the error, as shown in the upper right plot. Specifically, the error generally decreases with decreasing λ_q values, with the exception of $\lambda_q = 0.5$, where the error reaches its minimum. The lower left plot demonstrates the exponential decay of the error with increasing values of n_q, with the error decreasing by approximately two orders of magnitude for every two-unit increase in n_q. The lower right plot presents a comparison between the SGPS method and MATLAB's "integral" function, employing the tolerance parameters RelTol = AbsTol = 10^{-15}. The SGPS method achieves near-machine-precision accuracy with the parameter values $\lambda = \lambda_q = 0.5$ and $n_q = 12$, outperforming MATLAB's integral function by nearly two orders of magnitude in certain cases. The method achieves near-machine-epsilon precision with relatively coarse grids, demonstrating notable stability through consistent error trends.

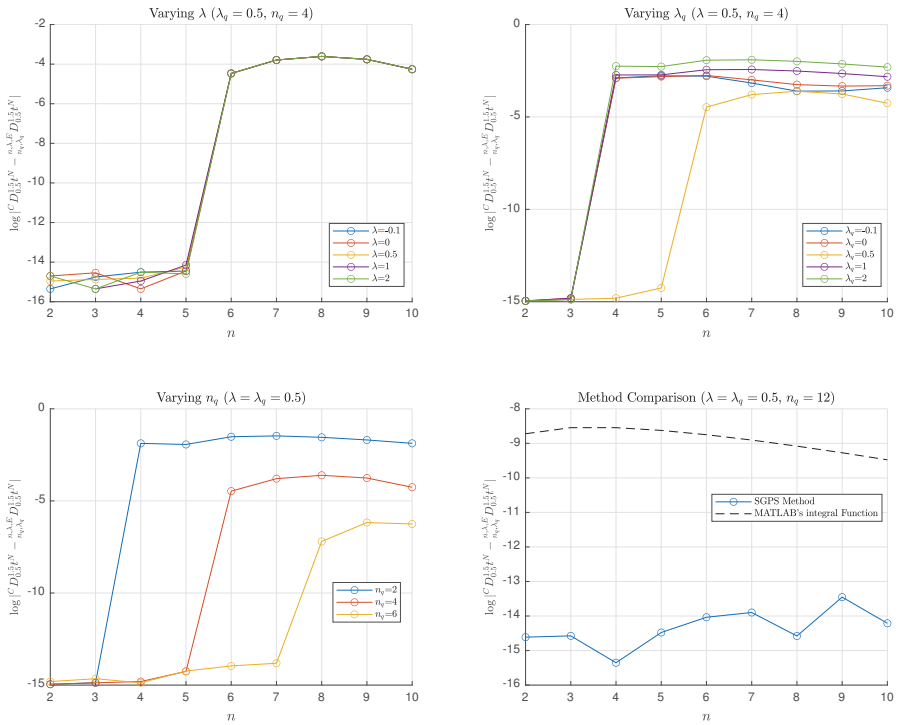


Figure 2. The logarithmic absolute errors of Caputo FD approximations of the power function f_1, computed using the SGPS method. The fractional order is set to $\alpha = 1.5$, and the approximations are evaluated at $t = 0.5$. The SG interpolant degrees range from $n = 2$ to 10. The figure presents errors under different conditions: (**Top-left**): Varying λ with fixed $\lambda_q = 0.5$ and $n_q = 4$. (**Top-right**): Varying λ_q with fixed $\lambda = 0.5$ and $n_q = 4$. (**Bottom-left**): Varying n_q with $\lambda = \lambda_q = 0.5$. (**Bottom-right**): Comparison between the SGPS method (with $n_q = 12$ and $\lambda = \lambda_q = 0.5$) and MATLAB's integral function. The top figures include comparisons with SC and SL interpolants and quadrature cases, where $\lambda = 0$ and $\lambda = 0.5$ correspond to the standard SC and SL cases, respectively.

Figure 3 further shows the logarithmic absolute errors of the Caputo FD approximations of the function $f_2(t) = e^{\beta t} : \beta \in \mathbb{R}^+$ using SG interpolants of various parameters and a $(15, 0.5)$-SGPS quadrature. The exact Caputo FD of f_2 is given below:

$$ {}^{c}D_t^{\alpha} f_2 = \sum_{k=0}^{\infty} \frac{\beta^{k+m} t^{-\alpha+k+m}}{\Gamma(k+m-\alpha+1)} = \beta^{\alpha} t^{-\alpha} E_{1,\alpha-m+1}(\beta t). $$

The figure illustrates the rapid convergence of the proposed PS approximations. Specifically, across the parameter range $\lambda \in \{-0.2, -0.1, 0, 0.5, 1, 2\}$, the logarithmic absolute errors exhibit a consistent decrease as the degree of the Gegenbauer interpolant increases. This trend underscores the improved accuracy of higher-degree interpolants in approximating the Caputo FD up to a defined precision threshold. For lower degrees (n), the error reduction is more enunciated as λ decreases, indicating that other members of the SG polynomial family, associated with negative λ values, exhibit superior convergence rates in these cases. For higher degrees (n), the errors converge to a stable accuracy level irrespective of the λ value, highlighting the robustness of higher-degree interpolants in accurately approximating the Caputo FD. The near-linear error profiles observed in the plots confirm the exponential convergence of the PS approximations, with convergence rates modulated by the parameter selections, as detailed in Section 4.

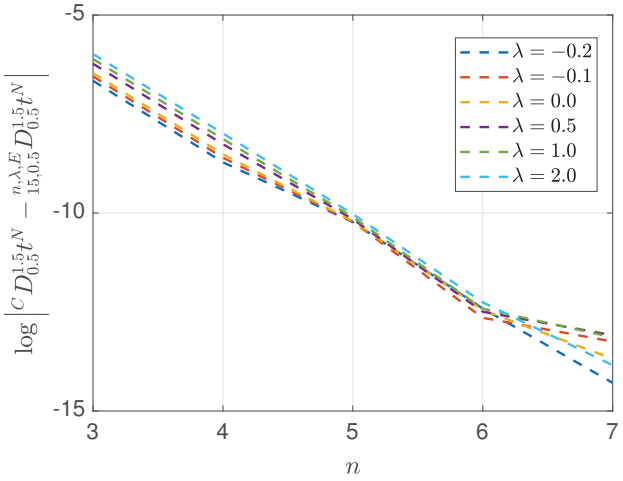


Figure 3. The logarithmic absolute errors of Caputo FD approximations of f_2 at $t = 0.5$ for $\beta = 0.1, \alpha = 1.5$, comparing Gegenbauer interpolants (degrees $n = 3$–7) across five parameter values $\lambda \in \{-0.2, -0.1, 0, 0.5, 1, 2\}$, using a $(15, 0.5)$-SGPS quadrature. The figure includes comparisons with SC and SL interpolants cases.

3. Computational Complexity

In this section, we provide a computational complexity analysis of constructing ${}^{E}\mathbf{Q}_n^{\alpha}$, incorporating the quadrature approximation (16). The analysis is based on the key matrix operations involved in the construction process, which we analyze individually as follows: Observe from Equation (11) that the term ${z_M}^{\circ(m-\alpha)}$ involves raising each element of an $(M+1)$-dimensional vector to the power $(m-\alpha)$, which requires $O(M)$ operations. Constructing ${}^{E}\mathbf{Q}_n^{\alpha}$ from ${}^{E}\hat{\mathbf{Q}}_n^{\alpha}$ involves diagonal scaling by $\mathrm{diag}\left({z_M}^{\circ(m-\alpha)}\right)$, which requires another $O(Mn)$ operation. The matrix ${}^{E}\hat{\mathbf{Q}}_n^{\alpha}$ is constructed using several matrix multiplications and elementwise operations. For each entry of z_M, the dominant steps include the following:

- The computation of $\hat{G}_{m:n}^{\lambda}$. Using the three-term recurrence equation

$$ (n+2\alpha)\hat{G}_{n+1}^{(\lambda)}(\hat{x}) = 2(n+\alpha)(2\hat{x}-1)\hat{G}_n^{(\lambda)}(\hat{x}) - n\hat{G}_{n-1}^{(\lambda)}(\hat{x}), $$

$\forall n \in \mathbb{Z}^+$, starting with $\hat{G}_0^{(\lambda)}(\hat{x}) = 1$ and $\hat{G}_1^{(\lambda)}(\hat{x}) = 2\hat{x} - 1$, we find that each polynomial evaluation requires $O(1)$ per point, as the number of operations remains constant regardless of the value of n. Since the polynomial evaluation is required for polynomials up to degree n, this requires $O(n)$ operations per point. The computations of $\hat{G}^{\lambda}_{m:n}[\hat{\boldsymbol{x}}^{\lambda}_n]$ therefore require $O(n^2)$ operations.

- The quadrature (16) involves evaluating a polynomial at transformed nodes. The cost of calculating $\hat{\chi}^{\lambda}_{j,m}$ depends on the chosen methods for computing factorials and the Gamma function. It can be considered a constant overhead for each evaluation of the Equation (14). The computation of $\left(\hat{\boldsymbol{x}}^{\lambda_q}_{n_q}\right)^{\circ \frac{1}{m-\alpha}}$ involves raising each element of the column vector $\hat{\boldsymbol{x}}^{\lambda_q}_{n_q}$ to the power $1/(m-\alpha)$. The cost here is linear in (n_q+1), as each element requires a single exponentiation operation. Since we need to evaluate the polynomial at n_q+1 points, the total cost for this step is $O(n_q)$. The cost of the matrix–vector multiplication is also linear in n_q+1. Therefore, the computational cost of this step is $O(n_q)$ for each $j \in \mathbb{N}_{m:n}$. The overall cost, if we consider all polynomial functions involved in this step, is thus $O(nn_q)$.
- The Hadamard product introduces another $O(n^2)$ operations.
- The evaluation of $\hat{\tilde{\lambda}}^{\lambda^{\div}}_{m:n}$ requires $O(n)$ operations, and the product of $\mathrm{trp}\left(\hat{\tilde{\lambda}}^{\lambda^{\div}}_{m:n}\right)$ according to the result from the Hadamard product requires $O(n^2)$ operations.
- The final diagonal scaling $\mathrm{diag}\left(\hat{\omega}^{\lambda}_{0:n}\right)$ contributes $O(n)$.

Summing the dominant terms, the overall computational complexity of constructing ${}^{E}\mathbf{Q}^{\alpha}_{n}$ is of $O\big(n(n+n_q)\big)$ per entry of $\boldsymbol{z}_M$. We therefore expect the total number of operations required to construct the matrix ${}^{E}\mathbf{Q}^{\alpha}_{n}$ for all entries of $\boldsymbol{z}_M$ to be of $O\big(Mn(n+n_q)\big)$.

Remark 1. *The construction runtime of the FSGIM matrix ${}^{E}\mathbf{Q}^{\alpha}_{n}$ (size $(M+1)\times(n+1)$) used by the SGPS method scales as $O(Mn(n+n_q))$, where n is the interpolant degree, M is the number of evaluation points, and n_q is the highest degree of the Gegenbauer polynomial used to construct the quadrature rule. For large n and M, the FSGIM requires $O(Mn)$ storage. While this remains manageable in double-precision arithmetic, precomputation of the FSGIM offsets runtime costs, making the method practical for moderate-scale problems. For sufficiently smooth solutions, the chosen quadrature parameter n_q can often be smaller than n without sacrificing accuracy, as the integrands are well approximated by low-degree polynomials. This reduces the dominant $O(Mnn_q)$ term in the runtime and further improves the efficiency.*

4. Error Analysis

The following theorem defines the truncation error of the αth-order SGPS quadrature (10) associated with the αth-order FSGIM ${}^{E}\mathbf{Q}^{\alpha}_{n}$ in closed form.

Theorem 2. *Let $n \geq m-1$, and suppose that $f \in C^{n+1}(\boldsymbol{\Omega}_1)$ is approximated by the SGPS interpolant* (1)*. Also, assume that the integrals*

$$\mathcal{I}_1^{(y)} \hat{G}^{\lambda,m}_{m:n}\text{ł}x\Big(1-y^{\frac{1}{m-\alpha}}\Big)\text{ł}, \tag{17}$$

are computed exactly $\forall_{\text{ł}} x \in \boldsymbol{\Omega}_1$. Then, $\exists\, \xi = \xi(x) \in \boldsymbol{\Omega}_1^{\circ}$ such that the truncation error, ${}^{\alpha}\mathcal{T}^{\lambda}_n(x,\xi)$, in the Caputo FD approximation (7) *is given by*

$${}^{\alpha}\mathcal{T}^{\lambda}_n(x,\xi) = {}^{\alpha}\eta^{\lambda}_n\, f^{(n+1)}(\xi)\, \mathcal{I}_x^{(\tau)} \frac{\hat{G}^{\lambda+m}_{n+1-m}}{(x-\tau)^{\alpha+1-m}}, \tag{18}$$

where

$$ {}^{\alpha}\eta_n^{\lambda} \;=\; \frac{\sqrt{\pi}2^{-2\lambda-2n-1}\Gamma(m+n+2\lambda+1)}{(n-m+1)!\,\Gamma(m-\alpha)\Gamma\left(m+\lambda+\frac{1}{2}\right)\Gamma(n+\lambda+1)}. $$

Proof. The Lagrange interpolation error associated with the SGPS interpolation (1) is given below:

$$ f(x) \;=\; I_n f(x) + \frac{f^{(n+1)}(\xi)}{(n+1)!\,\hat{K}_{n+1}^{\lambda}}\,\hat{G}_{n+1}^{\lambda}(x), $$

where $\hat{K}_n^{\lambda}$ is the leading coefficient of the nth-degree, λ-indexed SG polynomial (cf. Equation (4.12) in [23]). Applying Caputo FD on both sides of the equation gives the truncation error associated with Formula (7) in the following form:

$$ {}^{\alpha}\mathcal{T}_n^{\lambda}(x,\xi) \;=\; \frac{f^{(n+1)}(\xi)}{(n+1)!\,\hat{K}_{n+1}^{\lambda}}\;{}^{c}D_x^{\alpha}\hat{G}_{n+1}^{(\lambda)} $$

$$ = \frac{f^{(n+1)}(\xi)}{(n+1)!\,\hat{K}_{n+1}^{\lambda}\Gamma(m-\alpha)}\,\mathcal{I}_x^{(\tau)}\frac{\hat{G}_{n+1}^{(\lambda,m)}}{(x-\tau)^{\alpha+1-m}}. \tag{19} $$

The proof is established by substituting Formula (13b) into (19). □

For the theoretical truncation error in Equation (18), we assume that the integrals in Equation (17) are evaluated exactly. In practice, however, these integrals are approximated using SGPS quadratures, with the corresponding quadrature errors analyzed in Theorems 4 and 5, as discussed later in this section.

The following theorem marks the truncation error bound associated with Theorem 2.

Theorem 3. *Suppose that the assumptions of Theorem 2 hold true. Then, the truncation error* ${}^{\alpha}\mathcal{T}_n^{\lambda}(x,\xi)$ *is asymptotically bounded above by*

$$ \left|{}^{\alpha}\mathcal{T}_n^{\lambda}(x,\xi)\right| \lesssim \mathcal{A}_{n+1}\hat{\vartheta}_{m,\lambda}2^{-2\lambda-2n}n^{\lambda+m} \quad \forall_{rl}\, n, \tag{20} $$

where $\mathcal{A}_n \;=\; \left\|f^{(n)}\right\|_{L^{\infty}(\mathbf{\Omega}_1)}$ *and*

$$ \hat{\vartheta}_{m,\lambda} \;=\; \frac{1}{\sqrt{e}}\left(\lambda+m-\frac{1}{2}\right)^{-\lambda-m}\left(\left(\lambda+m-\frac{1}{2}\right)\sinh\left(\frac{1}{\lambda+m-\frac{1}{2}}\right)\right)^{\frac{1}{4}(-2\lambda-2m+1)}. $$

Proof. Since $\lambda+m>3/2>0$, Equation (4.29a) in [23] shows that $\left\|\hat{G}_{n+1-m}^{\lambda+m}\right\|_{L^{\infty}(\mathbf{\Omega}_1)} = 1$. Thus,

$$ \left|\mathcal{I}_x^{(\tau)}\frac{\hat{G}_{n+1-m}^{\lambda+m}}{(x-\tau)^{\alpha+1-m}}\right| \le \mathcal{I}_x^{(\tau)}(x-\tau)^{m-\alpha-1} \;=\; \frac{x^{m-\alpha}}{m-\alpha} \le \frac{1}{m-\alpha}, \tag{21} $$

according to the Mean Value Theorem for Integrals. Notice also that $\Gamma(z)>1/z\;\forall z\in\mathbf{\Omega}_1^{\circ}$. Combining this elementary inequality with the sharp inequalities of the Gamma function ([25] Inequality (96)) implies that

$$ \left|{}^{\alpha}\eta_n^{\lambda}\right| < \frac{1}{\sqrt{e}}(m-\alpha)\left(\lambda+m-\frac{1}{2}\right)^{-\lambda-m}2^{-2\lambda-2n-\frac{3}{2}}(\lambda+n)^{-\lambda-n-\frac{1}{2}}\times $$

$$\left(\left(\lambda+m-\frac{1}{2}\right)\sinh\left(\frac{1}{\lambda+m-\frac{1}{2}}\right)\right)^{\frac{1}{4}(-2\lambda-2m+1)}(2\lambda+m+n)^{2\lambda+m+n+\frac{1}{2}}\times$$

$$\left(\frac{1}{1620(2\lambda+m+n)^5}+1\right)\left((\lambda+n)\sinh\left(\frac{1}{\lambda+n}\right)\right)^{\frac{1}{2}(-\lambda-n)}\times$$

$$\left((2\lambda+m+n)\sinh\left(\frac{1}{2\lambda+m+n}\right)\right)^{\lambda+\frac{m+n}{2}}\sim\vartheta_{\alpha,\lambda}2^{-2\lambda-2n-\frac{3}{2}}n^{\lambda+m}\ \forall_{rl}\,n,\tag{22}$$

where $\vartheta_{\alpha,\lambda}=(m-\alpha)\hat{\vartheta}_{m,\lambda}$. The required asymptotic Formula (20) is derived by combining the asymptotic Formula (22) with inequality (21). □

Since the dominant term in the asymptotic bound (20) is $2^{-2\lambda-2n}$, the truncation error exhibits exponential decay as $n\to\infty$. Notice also that increasing α while keeping λ fixed and keeping n sufficiently large leads to an increase in m, which, in turn, affects two factors: (i) the polynomial term $n^{\lambda+m}$ grows, which slightly slows convergence, and (ii) the prefactor $\hat{\vartheta}_{m,\lambda}\sim e^{-1/2}\,m^{-(\lambda+m)}\ \forall_{rl}\,m$, which decreases exponentially, reducing the error; cf. Figure 4. Despite the polynomial growth of the former factor, the exponential decay term 2^{-2n} dominates. Now, let us consider the effect of changing λ while keeping α fixed and n large enough. If we increase λ gradually, the term $2^{-2\lambda}$ will exhibit exponential decay, and the prefactor $\hat{\vartheta}_{m,\lambda}\sim e^{-1/2}\lambda^{-(\lambda+m)}\ \forall_{rl}\,\lambda$ will also decrease exponentially, further reducing the error. The polynomial term $n^{\lambda+m}$, on the other hand, will increase, slightly increasing the error. Although the polynomial term $n^{\lambda+m}$ grows and slightly increases the error, the dominant exponential decay effects from both $2^{-2\lambda}$ and the prefactor $\hat{\vartheta}_{m,\lambda}$ ensure that the truncation error decreases significantly as λ increases. Hence, increasing λ leads to faster decay of the truncation error. This analysis shows that for $\forall_{rl}\,n$, increasing α slightly increases the error bound due to polynomial growth but does not affect exponential convergence. Furthermore, increasing λ generally improves convergence, since the exponential decay dominates the polynomial growth. In fact, one can see this last remark from two other viewpoints:

(i) $\forall_{rl}\,n/(m-1), \mathrm{supp}\left(\hat{G}^{\lambda+m}_{n+1-m}\right)\to\{0,1\}\ \forall_{rl}\,\lambda$, and the truncation error ${}^{\alpha}\mathcal{T}^{\lambda}_{n}\to 0$ accordingly.

(ii) $\forall\lambda\in\mathbb{R}^{+}, \mathrm{supp}\left(\hat{G}^{\lambda,m}_{j}\right)\to\{0,1\}$, as $j/m\to\infty$. Consequently, the integrals (17) collapse $\forall\hat{G}^{\lambda,m}_{k}: m<k\leq n, k\gg m$, indicating faster convergence rates in the Caputo FD approximation (7).

In all cases, choosing a sufficiently large n ensures overall exponential convergence. It is important to note that these observations are based on the asymptotic behavior of the error upper bound as $n\to\infty$, assuming the SGPS quadrature is computed exactly.

Beyond the convergence considerations mentioned above, we highlight two important numerical stability issues related to this analysis:

(i) A small buffer parameter ε is often introduced to offset the instability of the SG interpolation near $\lambda=-1/2$, where SG polynomials grow rapidly for increasing orders [23].

(ii) As λ increases, the SGG nodes $\hat{x}^{\lambda}_{n,0:n}$ cluster more toward the center of the interval. This means that the SGPS interpolation rule (1) relies more on extrapolation than interpolation, making it more sensitive to perturbations in the function values and amplifying numerical errors. This consideration reveals that, although increasing λ theoretically improves the convergence rate, it can introduce numerical instability

due to increased extrapolation effects. Therefore, when selecting λ, one must balance convergence speed against numerical stability considerations to ensure accurate interpolation computations. This aligns well with the widely accepted understanding that, for sufficiently smooth functions and sufficiently large spectral expansion terms, the truncated expansion in the SC quadrature (corresponding to $\lambda = 0$) is optimal in the L^∞-norm for definite integral approximations; cf. [28] and the references therein.

In the following, we study the truncation error of the quadrature Formula (16) and how its outcomes add up to the above analysis.

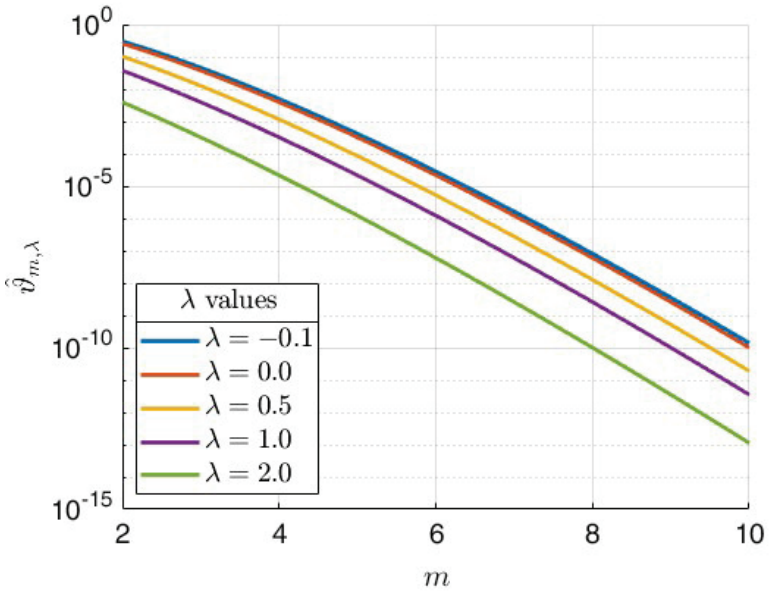


Figure 4. Log-lin plots of $\hat{\vartheta}_{m,\lambda}$ for $\lambda = -0.1, 0, 0.5, 1, 2$, and $m = 2:10$.

Theorem 4. *Let $j \in \mathbb{N}_{m:n}$, $x \in \mathbf{\Omega}_1$, and assume that $\hat{G}_{j-m}^{\lambda+m}\left(x - xy^{\frac{1}{m-\alpha}}\right)$ is interpolated by the SG polynomials with respect to the variable y at the SGG nodes $\hat{x}_{n_q,0:n_q}^{\lambda_q}$. Then, $\exists\, \eta = \eta(y) \in \mathbf{\Omega}_1^\circ$ such that the truncation error, $\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta)$, in the quadrature approximation* (16) *is given by*

$$\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta) = \frac{(-1)^{n_q+1}\hat{\chi}_{j-m,n_q+1}^{\lambda+m}}{(n_q+1)!\hat{K}_{n_q+1}^{\lambda_q}}\left(\frac{x}{m-\alpha}\right)^{n_q+1}\eta^{\frac{(n_q+1)(1-m+\alpha)}{m-\alpha}}\times$$

$$\hat{G}_{j-m-n_q-1}^{\lambda+m+n_q+1}\left(x - x\eta^{\frac{1}{m-\alpha}}\right)\mathcal{I}_1^{(y)}\hat{G}_{n_q+1}^{\lambda_q}\cdot\mathfrak{I}_{j\geq m+n_q+1}. \tag{23}$$

Proof. Theorem 4.1 in [23] immediately shows that

$$\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta) = \frac{1}{(n_q+1)!\hat{K}_{n_q+1}^{\lambda_q}}\left[\partial_y^{n_q+1}\hat{G}_{j-m}^{\lambda+m}\left(x - xy^{\frac{1}{m-\alpha}}\right)\right]_{y=\eta}\mathcal{I}_1^{(y)}\hat{G}_{n_q+1}^{\lambda_q}$$

$$= \frac{(-1)^{n_q+1}}{(n_q+1)!\hat{K}_{n_q+1}^{\lambda_q}}\left(\frac{x}{m-\alpha}\right)^{n_q+1}\eta^{\frac{(n_q+1)(1-m+\alpha)}{m-\alpha}}\hat{G}_{j-m}^{\lambda+m,n_q+1}\left(x - x\eta^{\frac{1}{m-\alpha}}\right)\mathcal{I}_1^{(y)}\hat{G}_{n_q+1}^{\lambda_q}, \tag{24}$$

according to the Chain Rule. The error bound (23) is accomplished by substituting Formula (13b) into (24). The proof is completed by further realizing that

$$\hat{G}_{j-m}^{\lambda+m,n_q+1}\Big(x - x\eta^{\frac{1}{m-\alpha}}\Big) \;=\; \partial_\tau^{n_q+1}\hat{G}_{j-m}^{\lambda+m}(\tau)\Big|_{\tau\,=\,x-x\eta^{\frac{1}{m-\alpha}}} \;=\; 0,$$

$\forall j < m + n_q + 1$. □

The truncation error analysis of the quadrature approximation (16) hinges on understanding the interplay between the parameters j, n_q, m, λ and λ_q. While Theorem 4 provides an exact error expression, the next theorem establishes a rigorous asymptotic upper bound, revealing how the error scales with these parameters.

Theorem 5. *Let the assumptions of Theorem 4 hold true. Then, the truncation error, $\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta)$, in the quadrature approximation* (16) *is bounded above by*

$$\Big|\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta)\Big| \lesssim B_m^{\lambda,\lambda_q} 2^{-2n_q}\big(j-m-n_q\big)^{-j+m+n_q+\frac{1}{2}} j^{-2\lambda-2m+1}\times$$

$$\big(j+n_q\big)^{j+2\lambda+m+n_q+\frac{1}{2}} n_q^{-2n_q-m-\lambda+\lambda_q-\frac{5}{2}}\left(\frac{x}{m-\alpha}\right)^{n_q+1}\eta^{\frac{(n_q+1)(1-m+\alpha)}{m-\alpha}}\times$$

$$\mathrm{Y}_{D^{\lambda_q}}(n_q)\,\mathfrak{I}_{j\geq m+n_q+1}, \tag{25}$$

$\forall_{rl}\, n_q$, *where*

$$\mathrm{Y}_{D^{\lambda_q}}(n_q) \;=\; \begin{cases} 1, & \lambda_q \in \mathbb{R}_0^+, \\ D^{\lambda_q} n_q^{-\lambda_q}, & \lambda_q \in \mathbb{R}_{-1/2}^-, \end{cases}$$

where $D^{\lambda_q} > 1$ is a constant dependent on λ_q, and B_m^{λ,λ_q} is a constant dependent on m, λ, and λ_q.

Proof. Lemma 5.1 in [26] shows that

$$\big\|\hat{G}_k^{\gamma}\big\|_{L^\infty(\mathbf{\Omega}_1)} = \begin{cases} 1, & k \in \mathbb{Z}_0^+,\ \gamma \in \mathbb{R}_0^+, \\ \sigma^\gamma k^{-\gamma}, & \gamma \in \mathbb{R}_{-1/2}^-,\ k \to \infty, \end{cases} \tag{26}$$

where $\sigma^\gamma > 1$ is a constant dependent on γ. Therefore,

$$\Big|\hat{G}_{j-m-n_q-1}^{\lambda+m+n_q+1}\Big(x - x\eta^{\frac{1}{m-\alpha}}\Big)\Big| \leq 1,$$

since $\lambda + m + n_q + 1 > 0$. Moreover, Formula (14) and the definition of $\hat{K}_{n_q}^{\lambda_q}$ (see p.g. 103 of [26]) show that

$$\frac{\hat{\chi}_{j-m,n_q+1}^{\lambda+m}}{(n_q+1)!\hat{K}_{n_q+1}^{\lambda_q}} \;=\; \frac{2^{-2n_q-1}\Gamma\big(\lambda_q+1\big)(j-m)!}{\Gamma\big(2\lambda_q+1\big)\Gamma\big(n_q+2\big)\Gamma\big(n_q+\lambda_q+1\big)}\times$$

$$\frac{\Gamma\Big(m+\lambda+\frac{1}{2}\Big)\Gamma\big(n_q+2\lambda_q+1\big)\Gamma\big(j+m+n_q+2\lambda+1\big)}{\Gamma(j+m+2\lambda)\Gamma\big(j-m-n_q\big)\Gamma\big(m+n_q+\lambda+\frac{3}{2}\big)}. \tag{27}$$

The proof is established by applying the sharp inequalities of the Gamma function (Inequality (96) of [25]) to (27). □

When $m \ll n_q$, the analysis of Theorem 5 bifurcates into the following two essential cases:

Case I ($j \sim n_q$) : Let $j = m + n_q + k + 1 : k = o(n_q)$. The first few error factors in (25) can be simplified as follows:

$$\begin{aligned}
&2^{-2n_q}\left(j-m-n_q\right)^{-j+m+n_q+\frac{1}{2}} j^{-2\lambda-2m+1}\left(j+n_q\right)^{j+2\lambda+m+n_q+\frac{1}{2}} \times \\
&n_q^{-2n_q-m-\lambda+\lambda_q-\frac{5}{2}} \sim 2^{-2n_q}(k+1)^{-k-1/2} n_q^{-2\lambda-2m+1}\left(2n_q\right)^{2n_q+2\lambda+m+\frac{1}{2}} \times \\
&n_q^{-2n_q-m-\lambda+\lambda_q-\frac{5}{2}} \sim 2^{\frac{1}{2}+m+2\lambda}(k+1)^{-k-\frac{1}{2}} n_q^{-1-2m-\lambda+\lambda_q}.
\end{aligned}$$

The dominant exponential decay factor in $\sup\left|\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta)\right|$ is therefore

$$\Lambda_{n_q,m}^{\alpha}(x,\eta) = \left(\frac{x}{m-\alpha}\right)^{n_q+1} \eta^{\frac{(n_q+1)(1-m+\alpha)}{m-\alpha}}.$$

This shows that the error bound decays exponentially with n_q if

$$\frac{x\,\eta^{\frac{1-m+\alpha}{m-\alpha}}}{m-\alpha} < 1, \tag{28}$$

is satisfied. Observe that increasing λ accelerates the algebraic decay, driven by the polynomial term $n_q^{-1-2m-\lambda+\lambda_q}$. While increasing λ_q can counteract this acceleration, the exponential term eventually dictates the convergence rate. Practically, to improve the algebraic decay in this case, we can increase λ and choose $\lambda_q : \lambda_q \leq \lambda + 2m + 1$ to prevent polynomial term growth.

Case II ($j \gg n_q$) : According to Lemma A1, the dominant terms involving j are approximately $e^{\frac{2\lambda n_q}{j}} j^{2n_q+2} \approx j^{2n_q+2}$. This result can also be derived from the asymptotic error bound in (25) by observing that $j - n_q \sim j \sim j + n_q$. Thus, the dominant terms involving j,

$$\left(j-n_q\right)^{-j+m+n_q+\frac{1}{2}} j^{-2\lambda-2m+1}\left(j+n_q\right)^{j+2\lambda+m+n_q+\frac{1}{2}},$$

reduce to approximately j^{2n_q+2}. Consequently, the error bound becomes

$$\begin{aligned}
\left|\mathfrak{T}_{j,n_q}^{\lambda_q}(\eta)\right| &\lesssim B_m^{\lambda,\lambda_q} 2^{-2n_q} j^{2n_q+2} n_q^{-2n_q-m-\lambda+\lambda_q-\frac{5}{2}} \left(\frac{x}{m-\alpha}\right)^{n_q+1} \eta^{\frac{(n_q+1)(1-m+\alpha)}{m-\alpha}} \Upsilon_{D^{\lambda_q}}(n_q) \\
&= B_m^{\lambda,\lambda_q} \left(\frac{j^2}{4n_q^2}\right)^{n_q} \frac{j^2}{n_q^{m+\lambda-\lambda_q+\frac{5}{2}}} \left(\frac{x}{m-\alpha}\right)^{n_q+1} \eta^{\frac{(n_q+1)(1-m+\alpha)}{m-\alpha}} \Upsilon_{D^{\lambda_q}}(n_q).
\end{aligned}$$

The exponential decay is now governed by

$$\Theta_{n_q,m,j}^{\alpha}(x,\eta) = \left(\frac{j^2 x \eta^{\frac{1-m+\alpha}{m-\alpha}}}{4n_q^2(m-\alpha)}\right)^{n_q}, \tag{29}$$

which requires that either

$$j < n_q, \quad \text{or} \tag{30a}$$

$$j = n_q \quad \text{and} \quad \frac{x\eta^{\frac{1-m+\alpha}{m-\alpha}}}{4(m-\alpha)} < 1. \tag{30b}$$

However, both conditions contradict the assumption $j \gg n_q$. Therefore, error convergence occurs only if

$$\frac{x\eta^{\frac{1-m+\alpha}{m-\alpha}}}{4(m-\alpha)} \ll 1: \quad \frac{j^2 x\eta^{\frac{1-m+\alpha}{m-\alpha}}}{4n_q^2(m-\alpha)} < 1. \tag{31}$$

In practice, to improve the algebraic decay in this case, we can choose $\lambda_q : \lambda_q \leq \lambda + m + \frac{5}{2}$ to prevent the polynomial term growth.

Given $m \ll n_q$, the quadrature truncation error in Theorem 5 converges when $n_q \leq n - m - k$, under the conditions that $1 \leq k \ll n_q$ and Condition (28) are met, or if $k \not\ll n_q$ and Condition (31) are satisfied. In the special case when $n_q > n - m - 1$, the quadrature truncation error totally collapses due to Theorem 4. In all cases, the parameter λ always serves as a decay accelerator, whereas λ_q functions as a decay brake. Notably, the observed slower convergence rate with increasing λ_q aligns well with the earlier finding in [28] that selecting relatively large positive values of $\lambda_q > 2$ causes the Gegenbauer weight function associated with the GIM to diminish rapidly near the boundaries $x = \pm 1$. This effect shifts the focus of the Gegenbauer quadrature toward the central region of the interval, increasing sensitivity to errors and making the quadrature more extrapolatory. Extensive prior research by the author on the application of Gegenbauer and SG polynomials for interpolation and collocation informs the selection of the Gegenbauer index γ within the interval

$$\mathcal{T}_{c,r} = \left\{\gamma \mid -\frac{1}{2} + \varepsilon \leq \gamma \leq r,\, 0 < \varepsilon \ll 1,\, r \in [1,2]\right\}, \tag{32}$$

designated as the "Gegenbauer parameter collocation interval of choice" in [29]. Specifically, investigations utilizing GG, SG, flipped-GG-Radau, and related nodal sets demonstrate that these configurations yield optimal numerical performance within this interval, consistently producing stable and accurate schemes for problems with smooth solutions; cf. [26,28,29] and the references therein, for example.

The following theorem provides an upper bound for the asymptotic total error, encompassing both the series truncation error and the quadrature approximation error in light of Theorems 3 and 5.

Theorem 6 (Asymptotic total truncation error bound)**.** *Let $m \ll n_q$, and suppose that the assumptions of Theorems 2 and 4 hold true. Then, the total truncation error, denoted by ${}^{\alpha}\mathcal{E}_{n,n_q}^{\lambda,\lambda_q}(x,\xi)$, arising from both the series truncation* (1) *and the quadrature approximation* (16)*, is asymptotically bounded above by:*

$$\begin{aligned}
\left|{}^{\alpha}\mathcal{E}_{n,n_q}^{\lambda,\lambda_q}(x,\xi,\eta)\right| &\lesssim \mathcal{A}_{n+1}\hat{\vartheta}_{m,\lambda}2^{-2\lambda-2n}n^{\lambda+m} \\
&+\frac{\mathcal{A}_0\varpi^{upp}B_m^{\lambda,\lambda_q}x^{m-\alpha}}{\lambda_{\max}^{\lambda}\,\Gamma(m-\alpha+1)}2^{-2n_q}n^{2(1-m-\lambda)}n_q^{-2n_q-m-\lambda+\lambda_q-\frac{5}{2}}\times \\
&\left(n-n_q\right)^{-n+m+n_q+\frac{3}{2}}\left(n+n_q\right)^{n+2\lambda+m+n_q+\frac{1}{2}}\Lambda_{n_q,m}^{\alpha}(x,\eta)\times \\
&{}_2\mathrm{Y}_{\sigma^{\lambda},D^{\lambda_q}}(n,n_q)\,\mathfrak{I}_{n\geq m+n_q+1},
\end{aligned}$$

$\forall_{rl}\, n, n_q$, *where*

$$\varpi^{upp} = \begin{cases} \varpi^{upp,+}, & \lambda \in \mathbb{R}_0^+, \\ \varpi^{upp,-}, & \lambda \in \mathbb{R}_{-1/2}^-, \end{cases}$$

$$\frac{1}{\bar{\lambda}^{\lambda}_{\max}} = \begin{cases} \dfrac{1}{\bar{\lambda}^{\lambda}_{n}}, & \lambda \in \mathbb{R}^{+}_{0}, \\ \dfrac{1}{\bar{\lambda}^{\lambda}_{m+n_q+1}}, & \lambda \in \mathbb{R}^{-}_{-1/2'} \end{cases}$$

$${}_2\mathrm{Y}_{\sigma^{\lambda},D^{\lambda_q}}(n,n_q) = \begin{cases} 1, & \lambda \in \mathbb{R}^{+}_{0}, \quad \lambda_q \in \mathbb{R}^{+}_{0}, \\ \sigma^{\lambda} n^{-\lambda}, & \lambda \in \mathbb{R}^{-}_{-1/2'}, \quad \lambda_q \in \mathbb{R}^{+}_{0}, \\ D^{\lambda_q} n_q^{-\lambda_q}, & \lambda \in \mathbb{R}^{+}_{0}, \quad \lambda_q \in \mathbb{R}^{-}_{-1/2'}, \\ \sigma^{\lambda} D^{\lambda_q} n^{-\lambda} n_q^{-\lambda_q}, & \lambda \in \mathbb{R}^{-}_{-1/2'}, \quad \lambda_q \in \mathbb{R}^{-}_{-1/2'}, \end{cases}$$

$\mathcal{A}_{n+1}$, $\hat{\vartheta}_{m,\lambda}$, D^{λ_q}, and σ^{λ} are constants with the definitions and properties outlined in Theorems 3 and 5, as well as in Equation (26).

Proof. The total truncation error is the sum of the truncation error associated with Caputo FD approximation (7), ${}^{\alpha}\mathcal{T}^{\lambda}_{n}(x,\xi)$, and the accumulated truncation errors associated with the quadrature approximation (16), for $j = m:n$, arising from Formula (7):

$$\begin{aligned} {}^{\alpha}\mathcal{E}^{\lambda,\lambda_q}_{n,n_q}(x,\xi,\eta) &= {}^{\alpha}\mathcal{T}^{\lambda}_{n}(x,\xi) \\ &+ \frac{x^{m-\alpha}}{\Gamma(m-\alpha+1)} \sum_{k\in\mathbb{J}^{+}_{n}} \hat{\omega}^{\lambda}_{k} f_k \sum_{j\in\mathbb{N}_{m:n}} \left(\hat{\lambda}^{\lambda}_{j}\right)^{-1} \mathfrak{T}^{\lambda_q}_{j,n_q}(\eta)\, \hat{G}^{\lambda}_{j}\left(\hat{x}^{\lambda}_{n,k}\right) \\ &= {}^{\alpha}\mathcal{T}^{\lambda}_{n}(x,\xi) + \frac{x^{m-\alpha}}{\Gamma(m-\alpha+1)} \sum_{k\in\mathbb{J}^{+}_{n}} \omega^{\lambda}_{k} f_k \sum_{j\in\mathbb{N}_{m:n}} \left(\lambda^{\lambda}_{j}\right)^{-1} \mathfrak{T}^{\lambda_q}_{j,n_q}(\eta)\, \hat{G}^{\lambda}_{j}\left(\hat{x}^{\lambda}_{n,k}\right), \end{aligned}$$

where $\mathfrak{T}^{\lambda_q}_{j,n_q}(\eta)$ is the truncation error associated with the quadrature approximation (16) $\forall_{\mathscr{C}}\, j$, and $\bar{\lambda}^{\lambda}_{0:n}$ and $\varpi^{\lambda}_{0:n}$ are the normalization factors for Gegenbauer polynomials and the Christoffel numbers associated with their quadratures. The key upper bounds on these latter factors were recently derived in Lemmas B.1 and B.2 of [30]:

$$\begin{aligned} \omega^{\lambda}_{j} \lesssim \omega^{\mathrm{upp},+} &= \frac{\pi}{n+1} \quad \forall (j,\lambda) \in \mathbb{J}^{+}_{n} \times \mathbb{R}^{+}_{0}, \\ \omega^{\lambda}_{j} < \omega^{\mathrm{upp},-} &= \frac{\Gamma^2(\lambda+1/2)}{2\, n^{1+2\lambda}} \quad \forall (j,\lambda) \in \mathbb{J}^{+}_{n} \times \mathbb{R}^{-}_{-1/2'} \\ \max_{j\in\mathbb{J}^{+}_{n}} \frac{1}{\bar{\lambda}^{\lambda}_{j}} &= \begin{cases} \dfrac{1}{\bar{\lambda}^{\lambda}_{n}}, & \lambda \in \mathbb{R}^{+}_{0}, \\ \dfrac{1}{\bar{\lambda}^{\lambda}_{0}}, & \lambda \in \mathbb{R}^{-}_{-1/2'} \end{cases} \end{aligned}$$

where $\bar{\lambda}^{\lambda}_{0} = \dfrac{\sqrt{\pi}\,\Gamma(1/2+\lambda)}{\Gamma(1+\lambda)}$. By combining these results with Equation (26), we can bound the total truncation error by

$$\left|{}^{\alpha}\mathcal{E}^{\lambda,\lambda_q}_{n,n_q}(x,\xi,\eta)\right| \lesssim \mathcal{A}_{n+1}\hat{\vartheta}_{m,\lambda} 2^{-2\lambda-2n} n^{\lambda+m}$$

$$+ \frac{\mathcal{A}_0\, \omega^{\mathrm{upp}}\, x^{m-\alpha}}{\bar{\lambda}^{\lambda}_{\max}\,\Gamma(m-\alpha+1)} (n+1)(n-m-n_q) \max_{j\in\mathbb{N}_{m+n_q+1:n}} \left|\mathfrak{T}^{\lambda_q}_{j,n_q}(\eta)\right| \mathrm{Y}_{\sigma^{\lambda}}(n), \tag{33}$$

where $\forall_{rl}\, n$. Since the j-dependent polynomial factor

$$(j-m-n_q)^{-j+m+n_q+\frac{1}{2}}\, j^{-2\lambda-2m+1} (j+n_q)^{j+2\lambda+m+n_q+\frac{1}{2}},$$

is maximized at $j = n$ by Lemma A2, the proof is accomplished by applying the asymptotic inequality (25) to (33) after replacing j with n. □

Under the assumptions of Theorem 6, exponential error decay dominates the overall error behavior if $n_q \leq n - m - k$, provided that $k \ll n_q$ and Condition (28) hold, or if $k \not\ll n_q$ and Condition (31) are satisfied. In the special case when $n_q > n - m - 1$, the total truncation error reduces to pure interpolation error, as the quadrature truncation error vanishes. The rigorous asymptotic analysis presented in this section leads to the following practical guideline for selecting λ and λ_q:

Rule of Thumb *(Selection of λ and λ_q Parameters).* $\forall_{rl}\, n$ and n_q:

- *High-precision computations*: Consider $\lambda \in \mathbf{\Omega}_2$ with appropriately adjusted λ_q:

$$-1/2 + \varepsilon \leq \lambda_q \leq 2. \tag{34}$$

- *General-purpose computations*: Consider $\lambda = \lambda_q = 0$ (SC interpolation and quadrature). This latter choice is motivated by the fact that the truncated expansion in the SC quadrature is known to be optimal in the L^∞-norm for definite integral approximations of smooth functions.

Remark 2. *The recommended range* (34) *for λ_q is derived by combining two key observations:*

1. ***Polynomial term growth prevention**: To control the quadrature truncation error bound:*
   - *Choose λ_q such that*
   $$\lambda_q \leq \lambda + 2m + 1 \quad \forall_{rs}\, m,$$
   for $n_q \leq n - m - k : k = o(n_q)$.
   - *Choose λ_q such that*
   $$\lambda_q \leq \lambda + m + \frac{5}{2} \quad \forall_{rs}\, m,$$
   for $n_q \leq n - m - k : k \neq o(n_q)$.
2. ***Stability and accuracy**: The Gegenbauer index should lie within the interval $\mathcal{T}_{c,r}$ to ensure higher stability and accuracy.*

Since $m \geq 1$, the inequalities $\lambda + 2m + 1 > 2$ and $\lambda + m + \frac{5}{2} > 2$ hold. To maintain stability (as indicated by Observation 2), we enforce $\lambda_q \leq 2$.

Remark 3. *It is important to note that the observations made in this section rely on asymptotic results $\forall_{rl}\, n, n_q$. However, since the integrand is smooth when $\alpha \not\approx m$, the SG quadrature often achieves high accuracy with relatively few nodes. Smooth integrands may exhibit spectral convergence before asymptotic effects takes place, as we demonstrate later in Section 6.*

Remark 4. *The truncation errors in the SGPS method's quadrature strategy are not negligible in general but can be made negligible by choosing a sufficiently large n_q, especially when $n_q > n - m - 1$, as demonstrated in this section. Aliasing errors, while less severe than in Fourier-based methods on equi-spaced grids, can still arise in the SGPS method due to undersampling in interpolation or quadrature, particularly for non-smooth functions or when n and n_q are not sufficiently large. These errors are mitigated by the use of non-equispaced SGG nodes, barycentric forms, and the flexibility to increase n_q independently of n. To ensure robustness, we may (i) increase n_q for complex integrands or higher fractional orders α, (ii) follow this study's guidelines for λ and λ_q to optimize node clustering and stability, (iii) monitor solution smoothness and consider adaptive methods for non-smooth cases, and (iv) utilize the precomputable FSGIM to efficiently test the convergence of the SGPS method for different n_q values. The numerical simulations in Section 6*

suggest that, for smooth problems, these errors are already well controlled, with modest n and n_q, achieving near-machine precision. However, for more challenging problems, careful parameter tuning and validation are essential to minimize error accumulation.

Remark 5. *The SGPS method assumes sufficient smoothness of the solution to exploit the rapid convergence properties of PS approximations. For less smooth functions, alternative specialized methods may be more appropriate. In particular, filtering techniques (e.g., modal filtering) can be integrated to dampen spurious high-frequency oscillations without significantly degrading the overall accuracy. Adaptive interpolation strategies, such as local refinement near singularities or moving-node approaches, may also be employed to capture localized features more accurately. Furthermore, domain decomposition techniques, where the computational domain is partitioned into subdomains with potentially different resolutions or spectral parameters, offer another viable pathway to accommodate irregularities while preserving the advantages of SGPS approximations within each smooth subregion.*

To provide empirical support for our theoretical claims on the convergence rate of the SGPS method, we analyze the error in computing the Caputo FD as a function of the number of interpolation points for various parameter values. We estimate the rate of convergence based on a semi-log regression of the error. Specifically, we assume that the error follows an exponential decay model of the form $E_n \approx c \cdot e^{-pn}$, where p is the exponential decay rate and c is a positive constant. Taking the natural logarithm of this expression yields $\ln E_n \approx -pn + \ln c$. We can estimate p by performing a linear regression of $\ln E_n$ against n. The magnitude of the slope of the resulting line provides an estimate for the decay rate p. As an illustration, reconsider Test Function f_2, previously examined in Section 2, with its error plots shown in Figure 3. Under the same data settings, Figure 5 depicts the variation in the estimated exponential decay rate (p) and coefficient (c) with respect to λ. The decay rate p remains relatively consistent across different λ values, fluctuating slightly between 4 and 4.6, indicating that the SGPS method sustains a stable exponential convergence rate under variations in λ. The coefficient c varies smoothly between approximately 0.1 and 1, reflecting a stable baseline magnitude of the approximation error. The bounded variation in c further suggests that the method's accuracy is largely insensitive to the choice of λ within the considered range.

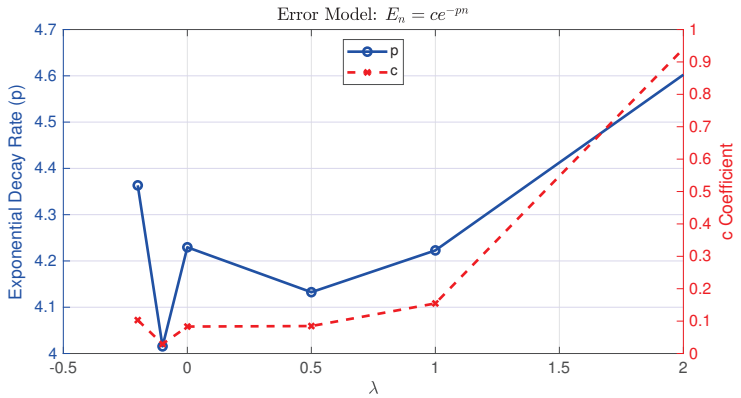


Figure 5. The empirical convergence analysis of the fractional operator approximation showing the relationship between λ parameters and error model components obtained through regression. The left axis (blue circles) displays the exponential decay rate p from the error model $E_n = ce^{-pn}$, while the right axis (red crosses) shows the corresponding coefficient c values. The dual-axis visualization demonstrates how different λ values in the approximation scheme affect both the convergence rate and magnitude of approximation errors.

5. Case Study: Caputo Fractional TPBVP of the Bagley–Torvik Type

In this section, we consider the application of the proposed method on the following Caputo fractional TPBVP of the Bagley–Torvik type, defined as follows:

$$a\,{}^{c}D_{x}^{\alpha}u + b\,{}^{c}D_{x}^{1.5}u + cu(x) \;=\; f(x), \quad x \in \mathbf{\Omega}_1, \tag{35a}$$

with the given Dirichlet boundary conditions

$$u(0) \;=\; \gamma_1, \quad u(1) \;=\; \gamma_2, \tag{35b}$$

where $\alpha > 1, \{a, b, c, \gamma_{1:2}\} \subset \mathbb{R}$, and $f \in L^2(\mathbf{\Omega}_1)$. With the derived numerical instrument for approximating Caputo FDs, determining an accurate numerical solution to the TPBVP is rather straightforward. Indeed, collocating System (35a) at the SGG set $\left\{\hat{x}_{n,0:n}^{\lambda}\right\} \;=\; \hat{\mathbb{G}}_n^{\lambda}$ in conjunction with Equation (10) yields

$$a\,{}^{E}\mathbf{Q}_n^{\alpha}\,u_{0:n} + b\,{}^{E}\mathbf{Q}_n^{1.5}\,u_{0:n} + cu_{0:n} \;=\; f_{0:n}. \tag{36a}$$

Since $\hat{G}_k^{\lambda}(0) \;=\; (-1)^k$ and $\hat{G}_k^{\lambda}(1) \;=\; 1\,\forall k \in \mathbb{J}_n^{+}$, according to the properties of SG polynomials, substituting the boundary conditions (35b) into Equation (1) gives the following system of equations:

$$\left[\mathrm{trp}\left(\hat{\lambda}_{0:n}^{\lambda^{\div}}\right)\left(\left((-1)^{0:n}\otimes \mathbf{1}_{n+1}\right)^{\top}\odot \hat{G}_{0:n}^{\lambda}[\hat{\boldsymbol{x}}_n^{\lambda}]\right)\mathrm{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right)\right]u_{0:n} \;=\; \gamma_1, \tag{36b}$$

$$\left[\mathrm{trp}\left(\hat{\lambda}_{0:n}^{\lambda^{\div}}\right)\hat{G}_{0:n}^{\lambda}[\hat{\boldsymbol{x}}_n^{\lambda}]\mathrm{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right)\right]u_{0:n} \;=\; \gamma_2. \tag{36c}$$

Therefore, the linear system described by Equations (36a), (36b) and (36c) can now be compactly written in the following form:

$$\mathbf{A}u_{0:n} \;=\; \boldsymbol{F}, \tag{37}$$

where

$$\mathbf{A} \;=\; \begin{bmatrix} a\,{}^{E}\mathbf{Q}_n^{\alpha} + b\,{}^{E}\mathbf{Q}_n^{1.5} + c\mathbf{I}_{n+1} \\ \mathrm{trp}\left(\hat{\lambda}_{0:n}^{\lambda^{\div}}\right)\left(\left((-1)^{0:n}\otimes \mathbf{1}_{n+1}\right)^{\top}\odot \hat{G}_{0:n}^{\lambda}[\hat{\boldsymbol{x}}_n^{\lambda}]\right)\mathrm{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right) \\ \mathrm{trp}\left(\hat{\lambda}_{0:n}^{\lambda^{\div}}\right)\hat{G}_{0:n}^{\lambda}[\hat{\boldsymbol{x}}_n^{\lambda}]\,\mathrm{diag}\left(\hat{\omega}_{0:n}^{\lambda}\right) \end{bmatrix},$$

is the collocation matrix, and

$$\boldsymbol{F} \;=\; [f_{0:n}^{\top}, \gamma_{1:2}]^{\top}.$$

The solution to the linear system (37) provides the approximate solution values at the SGG points. The solution values at any non-collocated point in $\mathbf{\Omega}_1$ can further be estimated with excellent accuracy via the interpolation Formula (1).

When $\alpha \in \mathbb{Z}^{+}$, Caputo FD reduces to the classical integer-order derivative of the same order. In this case, we can use the first-order GDM in barycentric form, $\mathbf{D}^{(1)}$, of Elgindy and Dahy [31]. This matrix enables the approximation of the function's derivative at the GG nodes using the function values at those nodes by employing matrix–vector multiplication. The entries of the differentiation matrix are computed based on the barycentric weights and GG nodes. The associated differentiation formula exhibits high accuracy, often exhibiting exponential convergence for smooth functions. This rapid convergence is a hallmark of PS methods and makes the GDM highly accurate for approximating derivatives. Furthermore, the utilization of barycentric forms improves the numerical stability of the differentiation matrix and leads to efficient computations. Using the properties of PS differentiation

matrices, higher-order differentiation matrices can be readily generated through successive multiplication by the first-order GDM:

$$\mathbf{D}^{(k)} \;=\; \mathbf{D}_{(k)}^{(1)}, \quad \forall k > 1.$$

The SGDM of any order k, $\hat{\mathbf{D}}^{(k)}$, based on the SGG point set $\hat{\mathbb{G}}_n^{\lambda}$, can be generated directly from $\mathbf{D}^{(1)}$ using the following formula:

$$\hat{\mathbf{D}}^{(k)} \;=\; 2^k\, \mathbf{D}_{(k)}^{(1)}, \quad \forall k \geq 1.$$

Figure 6 outlines the complete solution workflow for applying the SGPS method to Bagley–Torvik TPBVPs. The process begins with constructing the FSGIMs and, when necessary, the SGDM for integer orders. These are used to discretize the governing fractional differential equations via collocation at SGG nodes. The resulting system is assembled into a linear algebraic system, which is solved to obtain the numerical solution at collocation points. Finally, the global numerical solution is recovered by interpolating these discrete values using the SGPS interpolant.

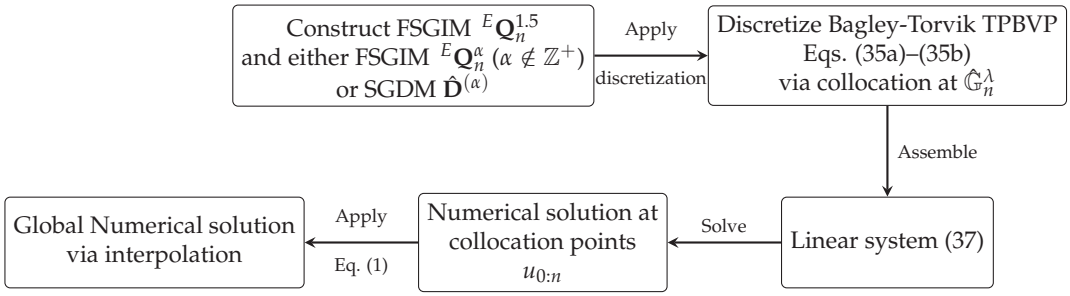


Figure 6. The solution workflow for Bagley–Torvik TPBVPs using the SGPS method. The process begins with problem discretization using FSGIMs and the SGDM (if necessary), followed by collocation at SGG points to form a linear system. After solving the system, the solution is obtained at collocation points and can be interpolated to arbitrary points.

6. Numerical Examples

In this section, we present numerical experiments conducted on a personal laptop equipped with an AMD Ryzen 7 4800H processor (2.9 GHz, 8 cores/16 threads) and 16GB of RAM, and running Windows 11. All simulations were performed using MATLAB R2023b. The accuracy of the computed solutions was assessed using absolute errors and maximum absolute errors, which provide quantitative measures of the pointwise and worst-case discrepancies between the exact and numerical solutions, respectively.

Example 1. *Consider the Caputo fractional TPBVP of the Bagley–Torvik type*

$${}^{c}D_x^2 u + {}^{c}D_x^{1.5} u + u(x) \;=\; x^2 + 2 + 4\sqrt{\frac{x}{\pi}}, \quad x \in \mathbf{\Omega}_1,$$

with the given Dirichlet boundary conditions

$$u(0) \;=\; 0, \quad u(1) \;=\; 1.$$

The exact solution is $u(x) \;=\; x^2$. This problem was solved by Al-Mdallal et al. [32] using a method that combines conjugating collocation, spline analysis, and the shooting technique. Their reported error norm was 3.78×10^{-12}; cf. [33]. Later, Batool et al. [33] addressed the same problem using integral operational matrices based on Chelyshkov

polynomials, transforming the problem into solvable Sylvester-type equations. They reported an error norm of 2.3388×10^{-25}, obtained using approximate solution terms with significantly more than 16 digits of precision. Specifically, the three terms used to derive this error included 32, 47, and 47 digits after the decimal point, indicating that the method utilizes extended or arbitrary-precision arithmetic, rather than being constrained to standard double precision. For a more fair comparison, since all components of our computational algorithm adhere to double-precision representations and computations, we recalculated their approximate solution using Equation (92) of [33] on the MATLAB platform with double-precision arithmetic. Our results indicate that the maximum absolute error in their approximate solution, evaluated at 50 equally spaced points in $\mathbf{\Omega}_1$, was approximately 2.22×10^{-16}. The SGPS method produced this same result using the parameters $n = n_q = 4$ and $\lambda = \lambda_q = 1.1$. The elapsed time required to run the SGPS method was 0.004732 s. Figure 7 illustrates the exact solution, the approximate solution obtained using the SGPS method, and the absolute errors at the SGG collocation points.

Example 2. *Consider the Caputo fractional TPBVP of the Bagley–Torvik type*

$$ {}^{c}D_x^2 u + {}^{c}D_x^{1.5} u + u(x) = 1 + x, \quad x \in \mathbf{\Omega}_1, $$

with the given Dirichlet boundary conditions

$$ u(0) = 1, \quad u(1) = 2. $$

The exact solution is $u(x) = 1 + x$. Yüzbaşı [34] solved this problem using a numerical technique based on collocation points, matrix operations, and a generalized form of Bessel functions of the first kind. The maximum absolute error reported in [34] (at $M = 6$) was 4.6047×10^{-8}. Our SGPS method produced near-exact solution values within a maximum absolute error of 4.44×10^{-16} using $n = n_q = \lambda = \lambda_q = 2$; cf. Figure 8. The elapsed time required to run the SGPS method was 0.004142 s.

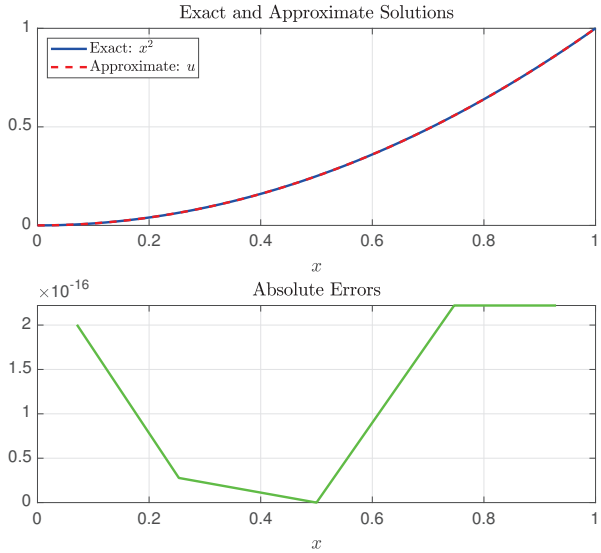


Figure 7. The exact solution to Example 1 and its approximation on $\mathbf{\Omega}_1$ (upper) and the absolute errors at the collocation points (lower). The approximate solution was obtained using the SGPS method with parameters $n = n_q = 4$ and $\lambda = \lambda_q = 1.1$.

Example 3. *Consider the Caputo fractional TPBVP of the Bagley–Torvik type*

$$ {}^{c}D_{x}^{1.5}u + u(x) \;=\; \frac{2}{\Gamma(3/2)}\sqrt{x} + x(x-1), \quad x \in \mathbf{\Omega}_1, $$

with the given Dirichlet boundary conditions

$$ u(0) \;=\; u(1) \;=\; 0. $$

The exact solution is $u(x) \;=\; x^2 - x$. Our SGPS method produced near-exact solution values within a maximum absolute error of 1.94×10^{-16} using $n \;=\; n_q \;=\; 3$ and $\lambda \;=\; \lambda_q \;=\; 1$; cf. Figure 9. The elapsed time required to run the SGPS method was 0.004160 s.

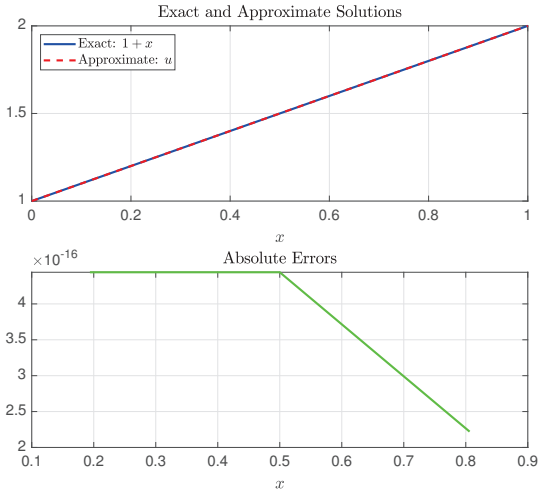


Figure 8. The exact solution to Example 2 and its approximation on $\mathbf{\Omega}_1$ (upper) and the absolute errors at the collocation points (lower). The approximate solution was obtained using the SGPS method with parameters $n \;=\; n_q \;=\; \lambda \;=\; \lambda_q \;=\; 2$.

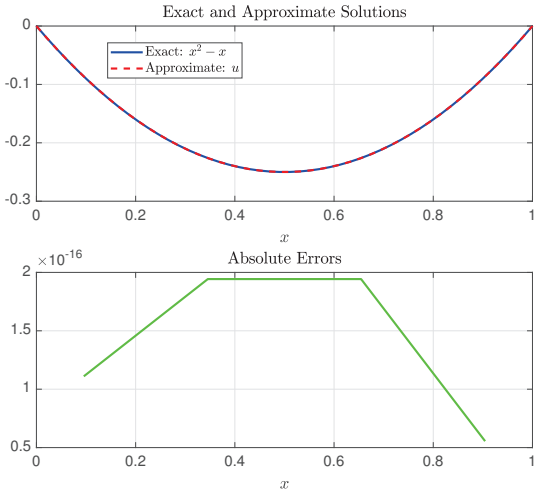


Figure 9. The exact solution to Example 3 and its approximation on $\mathbf{\Omega}_1$ (upper) and the absolute errors at the collocation points (lower). The approximate solution was obtained using the SGPS method with parameters $n \;=\; n_q \;=\; 3$ and $\lambda \;=\; \lambda_q \;=\; 1$.

7. Sensitivity Analysis of SG Parameters $\forall_{rs}\, n$ and n_q

Optimizing the performance of the SGPS method requires a thorough understanding of how the Gegenbauer parameters λ and λ_q influence numerical stability and accuracy. These parameters govern the clustering of collocation and quadrature points, directly

affecting the condition number of the collocation matrix $\mathbf{A}$ and the overall robustness of the method. In this section, we present a sensitivity analysis to quantify the impact of varying λ and λ_q on the stability of the SGPS method, as measured by the condition number $\kappa(\mathbf{A})$. The analysis specifically examines the behavior of the method when solving the Caputo Fractional TPBVP of the Bagley–Torvik Type with smaller interpolation and quadrature mesh sizes. The numerical examples from Section 6 serve as the basis for this analysis, and the results are visualized using surface plots, contour plots, and semilogarithmic plots to illustrate the condition number's behavior across the parameter space.

Figure 10 illustrates the influence of varying the parameters λ and λ_q on the condition number of collocation matrix $\mathbf{A}$ associated with Example 1. Higher condition numbers indicate increased sensitivity to perturbations in the input data, potentially leading to instability in the numerical solution. The results show that $\forall_{rs}\, n$ and n_q, the condition number is influenced by λ; as λ increases, the condition number tends to grow linearly. Conversely, the condition number exhibits minimal sensitivity to changes in λ_q within the range specified by the "Rule of Thumb." This suggests that the stability of the method for low-degree SG interpolants and small quadrature mesh sizes is primarily dependent on the appropriate selection of λ. Specifically, choosing a λ that is relatively large and positive can compromise stability. Conversely, the figure indicates that λ values closer to $-1/2$ (while maintaining a sufficient distance to prevent excessive growth in SG polynomial values), combined with λ_q values within the interval $\mathcal{T}_{c,r}$, particularly near its endpoints, yield lower condition numbers. We notice, however, that $\kappa(\mathbf{A})$ remains in the order of 10^2 for $-0.49 \leq \lambda, \lambda_q \leq 1.9$, indicating that the SGPS method is numerically stable for this range of parameters. Moreover, for double-precision arithmetic, this observation implies a potential loss of two significant digits in the worst case. However, the actual error observed in the numerical experiments is much smaller, indicating that the method is highly accurate in practice. Figures 11 and 12 present further sensitivity analyses of the SGPS method's numerical stability for Examples 2 and 3. The condition number in both examples is in the order of 10 for the parameter range considered (up to nearly 2), indicating high numerical stability for that parameter range. These figures consistently indicate that stability is influenced by λ $\forall_{rs}\, n$ and n_q, but minimally impacted by λ_q.

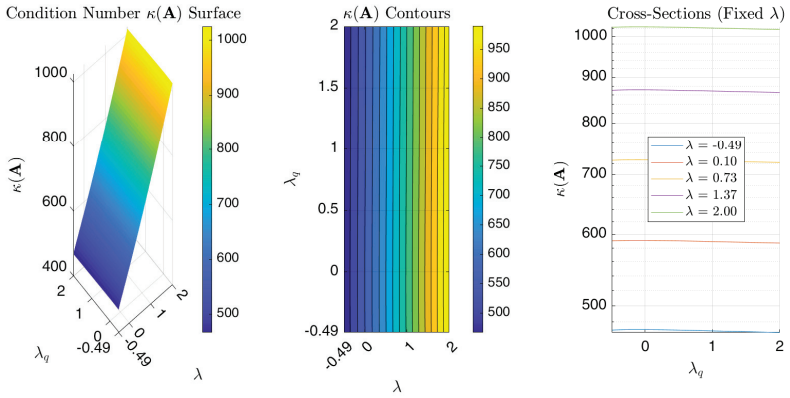


Figure 10. The sensitivity analysis of collocation matrix $\mathbf{A}$'s numerical stability for Example 1 using the SGPS method. The panels illustrate the following: (**left**) a surface plot depicting the condition number $\kappa(\mathbf{A})$ as a function of the parameters λ and λ_q; (**center**) a contour plot showing the distribution of the condition number across the parameter space; and (**right**) semilogarithmic plots of the condition number $\kappa(\mathbf{A})$ as a function of λ_q for selected fixed values of λ. The parameters used in the analysis are $\alpha = 1.5$, $n = n_q = 4$, and $\lambda, \lambda_q \in [-0.49, 2]$.

The sensitivity analysis conducted in this section reveals an important decoupling in parameter effects: while λ primarily governs the numerical stability through its lin-

ear relationship with the condition number of the collocation matrix, λ_q predominantly controls the accuracy of Caputo FD approximations, as seen earlier in Figures 2 and 3, without significantly affecting system conditioning. This decoupling allows for the independent optimization of stability and accuracy. In particular, we can select λ to ensure well-conditioned systems while tuning λ_q to achieve the desired precision in derivative computations. The recommended parameter ranges ($\lambda, \lambda_q \in \mathcal{T}_{c,r}$) provide a practical balance. Negative values of λ and λ_q close to -0.49 can improve stability and accuracy when using smaller interpolation and quadrature grids, while excellent quadrature accuracy is often achieved at $\lambda_q = 0.5$. This separation of concerns simplifies parameter selection and enables robust implementations across diverse problem configurations.

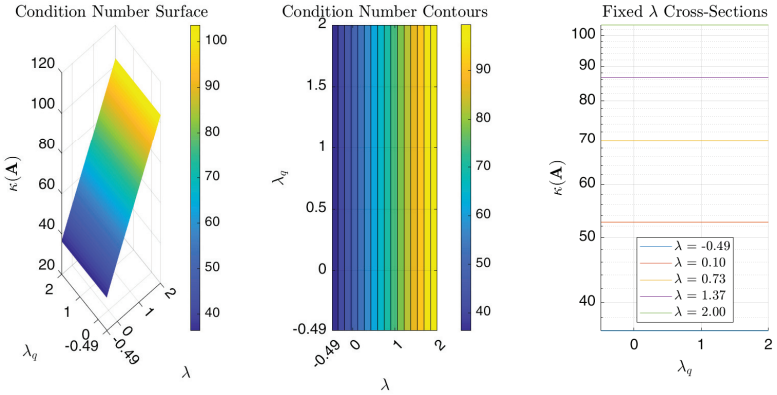


Figure 11. The sensitivity analysis of collocation matrix **A**'s numerical stability for Example 2 using the SGPS method. The panels illustrate the following: (**left**) a surface plot depicting the condition number $\kappa(\mathbf{A})$ as a function of the parameters λ and λ_q; (**center**) a contour plot showing the distribution of the condition number across the parameter space; and (**right**) semilogarithmic plots of the condition number $\kappa(\mathbf{A})$ as a function of λ_q for selected fixed values of λ. The parameters used in the analysis are $\alpha = 1.5$, $n = n_q = 2$, and $\lambda, \lambda_q \in [-0.49, 2]$.

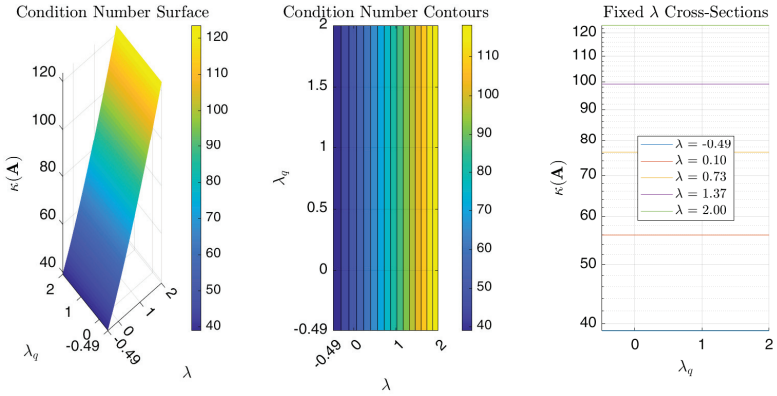


Figure 12. The sensitivity analysis of collocation matrix **A**'s numerical stability for Example 3 using the SGPS method. The panels illustrate the following: (**left**) a surface plot depicting the condition number $\kappa(\mathbf{A})$ as a function of the parameters λ and λ_q; (**center**) a contour plot showing the distribution of the condition number across the parameter space; and (**right**) semilogarithmic plots of the condition number $\kappa(\mathbf{A})$ as a function of λ_q for selected fixed values of λ. The parameters used in the analysis are $\alpha = 1.5$, $n = n_q = 3$, and $\lambda, \lambda_q \in [-0.49, 2]$.

8. Conclusions and Discussion

This study pioneers a unified SGPS framework that seamlessly integrates interpolation and integration for approximating higher-order Caputo FDs and solving TPBVPs of the Bagley–Torvik type, offering significant advancements in numerical methods for fractional differential equations through the following: (i) The development of FSGIMs

that accurately and efficiently approximate Caputo FDs at any random set of points using SG quadratures generalizes traditional PS differentiation matrices to the fractional-order setting, which we consider a significant theoretical advancement. (ii) The use of FSGIMs allows for pre-computation and storage, significantly accelerating the execution of the SGPS method. (iii) The method applies an innovative change of variables that transforms the Caputo FD into a scaled integral of an integer-order derivative. This transformation simplifies computations, facilitates error analysis, and mitigates singularities in the Caputo FD near zero, which improves both stability and accuracy. (iv) The method can produce approximations withing near-full machine precision at an exponential rate using relatively coarse mesh grids. (v) The method generally improves numerical stability and attempts to avoid issues related to ill conditioning in classical PS differentiation matrices by using SG quadratures in barycentric form. (vi) The proposed methodology can be extended to multidimensional fractional problems, making it a strong candidate for future research in high-dimensional fractional differential equations. (vii) Unlike traditional methods that treat interpolation and integration separately, the current method unifies these operations into a cohesive framework using SG polynomials. Numerical experiments validated the superior accuracy of the proposed method over existing techniques, achieving near-machine precision results in many cases. The current study also highlighted critical guidelines for selecting the parameters λ and λ_q to optimize the performance of the SGPS method $\forall_{rs}\, \alpha$. In particular, for large interpolation and quadrature mesh sizes, and for high-precision computations, λ should be selected within the range $\mathbf{\Omega}_2$, while λ_q should be adjusted to satisfy $-1/2 + \varepsilon \leq \lambda_q \leq 2$. This ensures a balance between convergence speed and numerical stability. For general-purpose computations, setting $\lambda = \lambda_q = 0$ (corresponding to the SC interpolant and quadrature) is recommended, as it provides optimal L^∞-norm accuracy for smooth functions. The analysis also revealed that increasing λ accelerates theoretical convergence but may introduce numerical instability due to extrapolation effects, while larger λ_q values can slow convergence. $\forall_{rs}\, n$ and n_q, the sensitivity analysis in this study reveals that the conditioning of the linear system of equations produced by the SGPS method when treating a Caputo Fractional TPBVP of the Bagley–Torvik Type increases approximately linearly with λ. This indicates that smaller values of λ in this case can lead to improved numerical stability. In particular, it is advisable to choose negative λ values, especially in the neighborhood of -0.49, as evidenced by the numerical simulations, but not too close to $-1/2$, to avoid the rapid growth of SG polynomials. The conditioning of the linear system is less sensitive to variations in λ_q compared to λ $\forall_{rs}\, n$ and n_q, with minimal effect on stability. However, to maintain accuracy, it is still recommended to keep λ_q within the recommended interval $\mathcal{T}_{c,r}$, with excellent quadrature accuracy often attained at $\lambda_q = 0.5$. These insights ensure robust and efficient implementations of the SGPS method across diverse problem settings. The SGPS method's computational efficiency is further underscored by its predictable runtime and storage costs, as summarized in Table 2. For practitioners, these estimates provide clear guidelines for resource allocation. The table also highlights recommended parameter ranges to balance accuracy and stability.

The current work assumes sufficient smoothness of the solution to achieve exponential convergence. For fractional problems involving weakly singular or non-smooth solutions, where derivatives may be unbounded, future research may investigate adaptive techniques—such as graded meshes or hybrid spectral–finite element approaches—to extend the method's applicability. The robust approximation of Caputo derivatives achieved by the SGPS method creates opportunities for modeling viscoelasticity in smart materials, anomalous transport in heterogeneous media, and non-local dynamics in control theory. Future directions could include adaptive parameter tuning to capture singularities in viscoelastic models or coupling the method with machine learning to optimize fractional-order

controllers. These applications would improve the method's interdisciplinary relevance while preserving its mathematical rigor. Additionally, the SGPS approach could be extended to multidimensional fractional problems, where tensor products of one-dimensional FSGIMs can be employed. The inherent parallelizability of FSGIM matrix–vector operations makes the method particularly suitable for GPU acceleration or distributed computing. For time-dependent fractional PDEs, like fractional diffusion equations, the SGPS method can employ the FSGIM for spatial discretization, transforming the problem into a system of ODEs in time. Standard time-stepping schemes, such as Runge–Kutta or fractional linear multistep methods, can then be applied. The precomputation and reuse of the FSGIM for spatial discretization at each time step can yield significant efficiency gains in time-marching schemes.

Table 2. Computational costs and typical parameters for the SGPS method.

Aspect	Cost/Parameter	Typical Values
Runtime	Construction of FSGIM: $\mathcal{O}(Mnn_q)$ Application of FSGIM: $\mathcal{O}(Mn)$	**Small:** $n, n_q = 2, 3, 4$; $M = n$ **Large:** $n, n_q \geq 10$; $M = n$
Storage	FSGIM: $\mathcal{O}(Mn)$	Same as above
Parameter Ranges	**Small** n, n_q: $\lambda, \lambda_q \in [-1/2 + \varepsilon, 2]$ **Large** n, n_q: $\lambda \in \mathbf{\Omega}_2, \lambda_q \in [-\frac{1}{2} + \varepsilon, 2]$	**Suggested:** $\lambda \approx -0.49$, $\lambda_q = 0.5$ **Suggested:** $\lambda = \lambda_q = 0$

Funding: The Article Processing Charges (APCs) for this publication were funded by Ajman University, United Arab Emirates.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: The author declares that the data supporting the findings of this study are available within the article.

Conflicts of Interest: The author declares that there are no conflicts of interests.

Abbreviations

Acronym	Meaning
FD	Fractional derivative
FSGIM	Fractional-order shifted Gegenbauer integration matrix
GDM	Gegenbauer differentiation matrix
GG	Gegenbauer–Gauss
GIM	Gegenbauer integration matrix
GIRV	Gegenbauer integration row vector
PS	Pseudospectral
SC	Shifted Chebyshev
SGDM	Shifted Gegenbauer differentiation matrix
SGIM	Shifted Gegenbauer integration matrix
SGIRV	Shifted Gegenbauer integration row vector
SGPS	Shifted Gegenbauer pseudospectral
SG	Shifted Gegenbauer
SGG	Shifted Gegenbauer–Gauss
SL	Shifted Legendre
TPBVP	Two-point boundary value problem

Appendix A. SGPS Algorithm for Bagley–Torvik TPBVPs

Algorithm A1 SGPS_BAGLEY_TORVIK

procedure SGPS_BAGLEY_TORVIK($f, \alpha, a, b, c, n, \lambda, n_q, \lambda_q, z$)
  // Step 1: Generate SGG nodes and weights
  $(\xi, w, \tilde{w}) \leftarrow$ GG nodes, Christoffel numbers, and barycentric weights on $[-1, 1]$ with $n+1$ points and parameter λ
  $x \leftarrow 0.5 \cdot (\xi + 1)$ ▷ SGG nodes on $[0, 1]$
  // Step 2: Construct Caputo FSGIM
  $Q_\alpha \leftarrow$ FSGIM for Caputo derivative of order α on $[0, 1]$
  $(G, \bar{\lambda}) \leftarrow$ Gegenbauer basis evaluated at ξ and their squared norms
  // Step 3: Assemble linear system from Equations (35a)–(35b)
  $I \leftarrow$ identity matrix of size $(n+1) \times (n+1)$
  if $\alpha \notin \mathbb{Z}^+$ **then**
    $A_{\text{colloc}} \leftarrow a \cdot Q_\alpha + b \cdot Q_{1.5} + c \cdot I$
  else
    // Step 4: Construct PS differentiation matrices
    $D \leftarrow$ barycentric GDM on $[-1, 1]$
    $m \leftarrow \lceil \alpha \rceil$
    $D_m \leftarrow 2^m \cdot D_{(m)}$ ▷ mth-order SGDM
    $A_{\text{colloc}} \leftarrow a \cdot D_m + b \cdot Q_{1.5} + c \cdot I$
  $A_{\text{BC1}} \leftarrow \text{trp}\left(\bar{\lambda}^{\div}\right) \left(\left((-1)^{0:n} \otimes \mathbf{1}_{n+1}\right)^\top \odot \hat{G}(x)\right) \text{diag}(w)$ ▷ $\hat{G}$ is the SG polynomial
  $A_{\text{BC2}} \leftarrow \text{trp}\left(\bar{\lambda}^{\div}\right) \hat{G}(x)\, \text{diag}(w)$
  $A \leftarrow \begin{bmatrix} A_{\text{colloc}} \\ A_{\text{BC1}} \\ A_{\text{BC2}} \end{bmatrix}$
  // Step 5: Assemble right-hand side
  $F \leftarrow \begin{bmatrix} f(x) \\ \gamma_1 \\ \gamma_2 \end{bmatrix}$ ▷ γ_1, γ_2: boundary conditions
  // Step 6: Solve and interpolate
  $u \leftarrow$ approximate solution of $Au = F$ at the collocation nodes x
  $v(z) \leftarrow$ barycentric interpolation of u at target points $z \in [0, 1]$
  return $u, v(z)$

Appendix B. Mathematical Proof

Lemma A1. *Let $\lambda > -\frac{1}{2}, m \geq 1$, and $j \geq m + n_q + 1$. Then, the j-dependent factor in Equation* (27),

$$\frac{(j-m)!\,\Gamma(j+m+n_q+2\lambda+1)}{\Gamma(j+m+2\lambda)\,\Gamma(j-m-n_q)}, \tag{A1}$$

has the asymptotic order $O\left(j^{2n_q+2}\right)$ as $j \to \infty$, $\forall_{rl}\, n_q$.

Proof. We analyze the asymptotic behavior of the expression as $j \to \infty$ using Stirling's approximation for the Gamma function:

$$\Gamma(z) \approx \sqrt{2\pi}\, z^{z-\frac{1}{2}} e^{-z} \quad \forall_{rl}\, z.$$

By also realizing that $(j-m)! = \Gamma(j-m+1)$, we have

$$\begin{aligned}
\Gamma(j-m+1) &\approx \sqrt{2\pi}\, j^{j-m+\frac{1}{2}} e^{-j}, \\
\Gamma(j+m+n_q+2\lambda+1) &\approx \sqrt{2\pi}\, (j+n_q)^{j+m+n_q+2\lambda+\frac{1}{2}} e^{-j-n_q}, \\
\Gamma(j+m+2\lambda) &\approx \sqrt{2\pi}\, j^{j+m+2\lambda-\frac{1}{2}} e^{-j}, \\
\Gamma(j-m-n_q) &\approx \sqrt{2\pi}\, (j-n_q)^{j-m-n_q-\frac{1}{2}} e^{n_q-j}.
\end{aligned}$$

Since $(j \pm n_q)^k \approx j^k (1 \pm \frac{n_q}{j})^k \approx j^k e^{\pm k n_q / j}\ \forall_{rl}\, j$, we can write the key ratio (A1) as follows:

$$\begin{aligned}
&\frac{\Gamma(j-m+1)\Gamma(j+m+n_q+2\lambda+1)}{\Gamma(j+m+2\lambda)\Gamma(j-m-n_q)} \\
&\approx \frac{j^{j-m+\frac{1}{2}} (j+n_q)^{j+m+n_q+2\lambda+\frac{1}{2}}}{j^{j+m+2\lambda-\frac{1}{2}} (j-n_q)^{j-m-n_q-\frac{1}{2}}} e^{-2n_q} \\
&\approx \frac{j^{j-m+\frac{1}{2}} j^{j+m+n_q+2\lambda+\frac{1}{2}} e^{(j+m+n_q+2\lambda+\frac{1}{2})n_q/j}}{j^{j+m+2\lambda-\frac{1}{2}} j^{j-m-n_q-\frac{1}{2}} e^{-(j-m-n_q-\frac{1}{2})n_q/j}} e^{-2n_q} \\
&= e^{\frac{2\lambda n_q}{j}} j^{2+2n_q} = O\left(j^{2n_q+2}\right), \quad \text{as } j \to \infty,\ \forall_{rl}\, n_q.
\end{aligned}$$

□

The following lemma is useful in analyzing the error bound of Theorem 5.

Lemma A2. *Let $\lambda > -\frac{1}{2}$ and $m \geq 1$ be an integer. The function*

$$E(j) = j^{-2\lambda-2m+1} (j-m-n_q)^{-j+m+n_q+\frac{1}{2}} (j+n_q)^{j+2\lambda+m+n_q+\frac{1}{2}},$$

is strictly increasing with $j\, \forall j \geq m+n_q+1\ \forall_{rl}\, n_q$.

Proof. Suppose that the assumptions of the lemma hold true. We show first that the logarithmic derivative of $E(j)$ is positive $\forall j \geq m+n_q+1$. To this end, take the natural logarithm

$$\ln E(j) = A \ln j + B \ln(j-m-n_q) + C \ln(j+n_q),$$

where

$$\begin{aligned}
A &= -2\lambda - 2m + 1, \\
B &= -j + m + n_q + \tfrac{1}{2}, \\
C &= j + 2\lambda + m + n_q + \tfrac{1}{2}.
\end{aligned}$$

Differentiating with respect to j yields

$$\begin{aligned}
\partial_j \ln E(j) &= \frac{A}{j} - \ln(j-m-n_q) + \frac{B}{j-m-n_q} + \ln(j+n_q) + \frac{C}{j+n_q} \\
&= \ln\left(\frac{j+n_q}{j-m-n_q}\right) + \frac{A}{j} + \frac{B}{j-m-n_q} + \frac{C}{j+n_q}.
\end{aligned}$$

For $j \geq m+n_q+1$, we have

- $\ln\left(\frac{j+n_q}{j-m-n_q}\right) > 0$, since $\frac{j+n_q}{j-m-n_q} > 1$.
- $\frac{A}{j} \to 0^-$.

- $\dfrac{B}{j-m-n_q} = -1 + \dfrac{1}{2(j-m-n_q)} \in (-1, -1/2]$.
- $\dfrac{C}{j+n_q} = 1 + \dfrac{2\lambda+m+1/2}{j+n_q} = 1^+$.

The rational terms combine to give a positive quantity. Thus, the logarithmic derivative, $\partial_j \ln E(j)$, is positive $\forall j \geq m + n_q + 1$. Since the natural logarithm is strictly increasing, it follows that $E(j)$ itself must be strictly increasing with j in that range. □

References

1. Podlubny, I. *Fractional Differential Equations: An Introduction to Fractional Derivatives, Fractional Differential Equations, to Methods of Their Solution and Some of Their Applications*; Elsevier: Amsterdam, The Netherlands, 1998; Volume 198.
2. Kilbas, A.A.; Srivastava, H.M.; Trujillo, J.J. *Theory and Applications of Fractional Differential Equations*; Elsevier: Amsterdam, The Netherlands, 2006; Volume 204.
3. Mainardi, F. *Fractional Calculus and Waves in Linear Viscoelasticity*; Imperial College Press: London, UK, 2010.
4. Das, S. *Functional Fractional Calculus*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 1.
5. Magin, R. Fractional calculus in bioengineering, part 1. *Crit. Rev. Biomed. Eng.* **2004**, *32* .
6. Monje, C.A.; Chen, Y.; Vinagre, B.M.; Xue, D.; Feliu-Batlle, V. *Fractional-Order Systems and Controls: Fundamentals and Applications*; Springer Science & Business Media: Basel, Switzerland, 2010.
7. Caputo, M. Linear models of dissipation whose Q is almost frequency independent—II. *Geophys. J. Int.* **1967**, *13*, 529–539. [CrossRef]
8. Momani, S.; Batiha, I.M.; Bendib, I.; Ouannas, A.; Hioual, A.; Mohamed, D. Examining finite-time behaviors in the fractional Gray–Scott model: Stability, synchronization, and simulation analysis. *Int. J. Cogn. Comput. Eng.* **2025**, *6*, 380–390. [CrossRef]
9. Diethelm, K., Ford, N.J., and Freed, A.D. A detailed error analysis for a fractional Adams method. *Numer. Algorithms* **2004**, *36*, 31–52. [CrossRef]
10. Saw, V.; Kumar, S. Numerical solution of fraction Bagley–Torvik boundary value problem based on Chebyshev collocation method. *Int. J. Appl. Comput. Math.* **2019**, *5*, 68. [CrossRef]
11. Ji, T.; Hou, J.; Yang, C. Numerical solution of the Bagley–Torvik equation using shifted Chebyshev operational matrix. *Adv. Differ. Equations* **2020**, *2020*, 648. [CrossRef]
12. Hou, J.; Yang, C.; Lv, X. Jacobi collocation methods for solving the fractional Bagley–Torvik equation. *Int. J. Appl. Math* **2020**, *50*, 114–120.
13. Ji, T.; Hou, J. Numerical solution of the Bagley–Torvik equation using Laguerre polynomials. *SeMA J.* **2020**, *77*, 97–106. [CrossRef]
14. Kaur, H.; Kumar, R.; Arora, G. Non-dyadic wavelets based computational technique for the investigation of Bagley–Torvik equations. *Int. J. Emerg. Technol.* **2019**, *10*, 1–14.
15. Dincel, A.T. A sine-cosine wavelet method for the approximation solutions of the fractional Bagley–Torvik equation. *Sigma J. Eng. Nat. Sci.* **2021**, *40*, 150–154.
16. Rabiei, K.; Razzaghi, M. The Numerical Solution of the Fractional Bagley–Torvik Equation by the Boubaker Wavelets. In *Acoustics and Vibration of Mechanical Structures–AVMS-2021: Proceedings of the 16th AVMS, Timişoara, Romania, 28–29 May 2021*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 27–37.
17. Abd-Elhameed, W.; Youssri, Y. Spectral solutions for fractional differential equations via a novel Lucas operational matrix of fractional derivatives. *Rom. J. Phys* **2016**, *61*, 795–813.
18. Youssri, Y.H. A new operational matrix of Caputo fractional derivatives of Fermat polynomials: An application for solving the Bagley–Torvik equation. *Adv. Differ. Equations* **2017**, *2017*, 1–17. [CrossRef]
19. Izadi, M.; Negar, M.R. Local discontinuous Galerkin approximations to fractional Bagley–Torvik equation. *Math. Methods Appl. Sci.* **2020**, *43*, 4798–4813. [CrossRef]
20. Chen, J. A fast multiscale Galerkin algorithm for solving boundary value problem of the fractional Bagley–Torvik equation. *Bound. Value Probl.* **2020**, *2020*, 1–13. [CrossRef]
21. Tamilselvan, A. Second order spline method for fractional Bagley–Torvik equation with variable coefficients and Robin boundary conditions. *J. Math. Model.* **2023**, *11*, 117–132.
22. Verma, A.; Kumar, M. Numerical solution of Bagley–Torvik equations using Legendre artificial neural network method. *Evol. Intell.* **2021**, *14*, 2027–2037. [CrossRef]
23. Elgindy, K.T. High-order numerical solution of second-order one-dimensional hyperbolic telegraph equation using a shifted Gegenbauer pseudospectral method. *Numer. Methods Partial. Differ. Equ.* **2016**, *32*, 307–349. [CrossRef]
24. Elgindy, K.T. High-order, stable, and efficient pseudospectral method using barycentric Gegenbauer quadratures. *Appl. Numer. Math.* **2017**, *113*, 1–25. [CrossRef]

25. Elgindy, K.T. Optimal control of a parabolic distributed parameter system using a fully exponentially convergent barycentric shifted Gegenbauer integral pseudospectral method. *J. Ind. Manag. Optim.* **2018**, *14*, 473. [CrossRef]
26. Elgindy, K.T.; Refat, H.M. High-order shifted Gegenbauer integral pseudo-spectral method for solving differential equations of Lane–Emden type. *Appl. Numer. Math.* **2018**, *128*, 98–124. [CrossRef]
27. Szegö, G. *Orthogonal Polynomials*; American Mathematical Society Colloquium Publication: Seattle, WA, USA, 1975; Volume 23.
28. Elgindy, K.T.; Smith-Miles, K.A. Optimal Gegenbauer quadrature over arbitrary integration nodes. *J. Comput. Appl. Math.* **2013**, *242*, 82–106. [CrossRef]
29. Elgindy, K.T.; Karasözen, B. Distributed optimal control of viscous Burgers' equation via a high-order, linearization, integral, nodal discontinuous Gegenbauer-Galerkin method. *Optim. Control. Appl. Methods* **2020**, *41*, 253–277. [CrossRef]
30. Elgindy, K.T.; Refat, H.M. Direct integral pseudospectral and integral spectral methods for solving a class of infinite horizon optimal output feedback control problems using rational and exponential Gegenbauer polynomials. *Math. Comput. Simul.* **2024**, *219*, 297–320. [CrossRef]
31. Elgindy, K.T.; Dahy, S.A. High-order numerical solution of viscous Burgers' equation using a Cole-Hopf barycentric Gegenbauer integral pseudospectral method. *Math. Methods Appl. Sci.* **2018**, *41*, 6226–6251. [CrossRef]
32. Al-Mdallal, Q.M.; Syam, M.I.; Anwar, M. A collocation-shooting method for solving fractional boundary value problems. *Commun. Nonlinear Sci. Numer. Simul.* **2010**, *15*, 3814–3822. [CrossRef]
33. Batool, A.; Talib, I.; Riaz, M.B. Fractional-order boundary value problems solutions using advanced numerical technique. *Partial. Differ. Equations Appl. Math.* **2025**, p. 101059. [CrossRef]
34. Yüzbaşı, Ş. Numerical solution of the Bagley–Torvik equation by the Bessel collocation method. *Math. Methods Appl. Sci.* **2013**, *36*, 300–312. [CrossRef]

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

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Mathematics Editorial Office
E-mail: mathematics@mdpi.com
www.mdpi.com/journal/mathematics

Disclaimer/Publisher's Note: The title and front matter of this reprint are at the discretion of the Guest Editors. The publisher is not responsible for their content or any associated concerns. The statements, opinions and data contained in all individual articles are solely those of the individual Editors and contributors and not of MDPI. MDPI disclaims responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-4586-6