AIMS Mathematics, 9(8): 21125–21143.
DOI: 10.3934/math.20241026
Received: 19 March 2024
Revised: 10 June 2024
Accepted: 14 June 2024
Published: 01 July 2024

https://www.aimspress.com/journal/Math

Research article

A third-order numerical method for solving fractional ordinary differential equations

Xiaopeng Yi[1], Chongyang Liu[2,3,*], Huey Tyng Cheong[1], Kok Lay Teo[1] and Song Wang[4]

[1] School of Mathematical Sciences, Sunway University, Kuala Lumpur 47500, Malaysia
[2] School of Mathematics and Information Science, Shandong Technology and Business University, Yantai 264005, China
[3] Yantai Key Laboratory of Big Data Modeling and Intelligent Computing, Yantai 264005, China
[4] School of Electrical Engineering, Computing and Mathematical Sciences, Curtin University, Perth 6845, Australia

* **Correspondence:** Email: chongyangliu@aliyun.com.

Abstract: In this paper, we developed a novel numerical method for solving general nonlinear fractional ordinary differential equations (FODEs). First, we transformed the nonlinear FODEs into the equivalent Volterra integral equations. We then developed a time-stepping algorithm for the numerical solution of the Volterra integral equations based on the third-order Taylor expansion for approximating the integrands in the Volterra integral equations on a chosen mesh with the mesh parameter h. This approximation led to implicit nonlinear algebraic equations in the unknowns at each given mesh point, and an iterative algorithm based on Newton's method was developed to solve the resulting implicit equations. A convergence analysis of this numerical scheme showed that the error between the exact solution and numerical solution at each mesh point is $O(h^3)$, independent of the fractional order. Finally, four numerical examples were solved to verify the theoretical results and demonstrate the effectiveness of the proposed method.

Keywords: fractional ordinary differential equations; time-stepping discretization; implicit scheme; Newton's method; convergence analysis

1. Introduction

Fractional calculus, as the name suggests, extends traditional integer-order calculus to fractional-order differential and integral calculus [1]. In recent decades, applications of fractional differential equations (FDEs) have received widespread attention in various disciplines, including physics [2], engineering [3] and biology [4]. Obtaining analytical solutions to FDEs is often challenging due to

the nonlocal and complex nature of fractional derivatives. Therefore, it is crucial to develop efficient numerical methods for solving FDEs.

In the literature, two main numerical approximation strategies for solving fractional ordinary differential equations (FODEs) have been extensively studied. The first strategy involves a direct approximation of the fractional derivatives in the FODEs, while the second strategy focuses on solving the equivalent Volterra integral equations. For the first strategy, a second-order numerical method for solving FODEs was proposed in [5], where the fractional derivatives are approximated by a weighted sum of function values at specified points. A finite difference method of order $(2-\alpha)$ with *L1*, *L2*, and *L2C* formulas was established in [6], where $\alpha \in (0,1)$ is a scalar representing the fractional order of the derivative. The *L1-2* formula, an enhanced version of the *L1* formula, was proposed in [7] to achieve a convergence rate of $3-\alpha$. Other approximation methods with the same convergence rate as obtained in [7] can be found in [8, 9]. It is worth noting that the direct discretization method for fractional derivatives described in [8] is specifically designed for linear FODEs. Also, the convergence rates of most of these methods depend on the fractional order α.

For the second strategy for solving FODEs, the classic $(1+\alpha)$th-order predicting-correction method was introduced in [10], and a detailed error analysis of this method was carried out in [11]. In [12], a method with the convergence order of $\min\{2, 1+2\alpha\}$ was developed. In [13], a one-step numerical integration method with second-order convergence rate, which is independent of α, was proposed. The second method in [8] used piecewise quadratic interpolation polynomials to approximate the Volterra integral equation, achieving a convergence rate of $1+2\alpha$. In addition, the perturbed quadratic predictor-corrector (Q-PCP) and decomposed quadratic predictor-corrector (Q-PCD) methods were developed in [14], where the convergence rates of $3-\alpha$ and $3-\frac{\alpha}{2}$ were achieved, respectively. However, the convergence rates of almost all of the aforementioned methods depend on the fractional order α.

In this paper, we present a third-order numerical scheme for the following general nonlinear FODEs:

$$\begin{cases} {}_{t_0}^{C}D_t^{\alpha}\boldsymbol{x}(t) = \boldsymbol{f}(t,\boldsymbol{x}(t)), & t \in (t_0, t_f], \\ \boldsymbol{x}(t_0) = \boldsymbol{x}^0, \end{cases} \tag{1}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)^{\top} \in (0,1]^n$ is a given vector of fractional orders; $\boldsymbol{x}(t) = (x_1(t), x_2(t), \ldots, x_n(t))^{\top} \in \mathbb{R}^n$ is the state vector at time t; $\boldsymbol{f} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$ is a given thrice continuously differentiable function with respect to all its arguments; t_0 is a given initial time; t_f is a given terminal time; $\boldsymbol{x}^0 = (x_1(t_0), x_2(t_0), \ldots, x_n(t_0))^{\top} \in \mathbb{R}^n$ is a given initial state vector; and ${}_{t_0}^{C}D_t^{\alpha}\boldsymbol{x}(t) = ({}_{t_0}^{C}D_t^{\alpha_1}x_1(t), {}_{t_0}^{C}D_t^{\alpha_2}x_2(t), \ldots, {}_{t_0}^{C}D_t^{\alpha_n}x_n(t))^{\top}$ with ${}_{t_0}^{C}D_t^{\alpha_i}x_i(t)$ denoting the α_ith Liouville-Caputo fractional derivative of $x_i(t)$.

We will develop a novel time-stepping numerical method to solve FODEs (1). Specifically, we first transform the FODEs into a set of equivalent Volterra integral equations. Then, we approximate these integral equations by using a third-order Taylor expansion (for the first two mesh points by a second-order Taylor expansion). This approximation leads to implicit nonlinear algebraic equations at each given mesh point. Then, Newton's method is employed to iteratively solve the nonlinear equations. A rigorous convergence analysis of the proposed method is carried out, which shows that the convergence rate from the numerical solution to the exact solution is third order and independent of the fractional order. Finally, the effectiveness and convergence results of the proposed method are illustrated through solving four numerical examples.

The organization of the rest of the paper is as follows. Section 2 proposes the numerical method for solving nonlinear FODEs. Section 3 provides the convergence and error analysis. Section 4 gives numerical results of four numerical examples. Finally, conclusions are drawn in Section 5.

2. Numerical method

Applying the Riemann-Liouville integral to both sides of FODEs (1) yields the following Volterra integral equations [1]:

$$x_i(t) = x_i^0 + \frac{1}{\Gamma(\alpha_i)} \int_{t_0}^{t} (t-\tau)^{\alpha_i - 1} f_i(\tau, \boldsymbol{x}(\tau)) d\tau, \quad t \in [t_0, t_f], \tag{2}$$

for $i = 1, 2, \ldots, n$, where $x_i^0 = x_i(t_0)$. For a given positive integer N, define a mesh on $[t_0, t_f]$ with $N+1$ mesh points $t_q, q = 0, 1, \ldots, N$, satisfying $t_0 < t_1 < \cdots < t_N = t_f$. For any $i = 1, 2, \ldots, n$, and $q = 0, 1, \ldots, N$, Eq (2) can be rewritten as

$$x_i(t_q) = x_i^0 + \frac{1}{\Gamma(\alpha_i)} \sum_{j=0}^{q-1} \int_{t_j}^{t_{j+1}} (t_q - \tau)^{\alpha_i - 1} f_i(\tau, \boldsymbol{x}(\tau)) d\tau. \tag{3}$$

We now consider the approximation of the integrand $f_i(\tau, \boldsymbol{x}(\tau))$ on the RHS of Eq (3) in $(t_0, t_1]$ by the following Taylor expansion:

$$f_i(\tau, \boldsymbol{x}(\tau)) = a_{i0} + b_{i0}(\tau - t_1) + c_{i0}(\tau - t_1)^2, \tag{4}$$

where a_{i0} and b_{i0} are coefficients to be determined, and $c_{i0}(\tau - t_1)^2$ is the remainder. To determine a_{i0} and b_{i0}, we omit the remainder and force the following equations to hold:

$$\begin{cases} f_i(t_0, \boldsymbol{x}^0) = a_{i0} + b_{i0}(t_0 - t_1), \\ f_i(t_1, \boldsymbol{x}(t_1)) = a_{i0}. \end{cases}$$

Solving these coupled equations gives

$$\begin{aligned} \triangle^0 f_i^1 &:= a_{i0} = f_i(t_1, \boldsymbol{x}(t_1)), \\ \triangle^1 f_i^1 &:= b_{i0} = \frac{f_i(t_1, \boldsymbol{x}(t_1)) - f_i(t_0, \boldsymbol{x}^0)}{h_1}, \end{aligned}$$

where $h_1 = t_1 - t_0$. Using these divided differences, Eq (4) can be expressed as

$$f_i(\tau, \boldsymbol{x}(\tau)) = \triangle^0 f_i^1 + \triangle^1 f_i^1 (\tau - t_1) + c_{i0}(\tau - t_1)^2. \tag{5}$$

Substituting the RHS of Eq (5) into the first term of the sum in Eq (3) yields

$$\begin{aligned} &\frac{1}{\Gamma(\alpha_i)} \int_{t_0}^{t_1} (t_q - \tau)^{\alpha_i - 1} \big[\triangle^0 f_i^1 + \triangle^1 f_i^1 (\tau - t_1) + c_{i0}(\tau - t_1)^2 \big] d\tau \\ &\quad = -\frac{I_{0,1}^i}{h_1} f_i(t_0, \boldsymbol{x}^0) + \Big(I_{0,0}^i + \frac{I_{0,1}^i}{h_1} \Big) f_i(t_1, \boldsymbol{x}(t_1)) + R_{i0}, \end{aligned} \tag{6}$$

where

$$I_{0,1}^{i} = \frac{-h_1(t_q - t_0)^{\alpha_i}}{\Gamma(\alpha_i + 1)} + \frac{(t_q - t_0)^{\alpha_i+1} - (t_q - t_1)^{\alpha_i+1}}{\Gamma(\alpha_i + 2)}, \tag{7}$$

$$I_{0,0}^{i} = \frac{(t_q - t_0)^{\alpha_i} - (t_q - t_1)^{\alpha_i}}{\Gamma(\alpha_i + 1)}, \tag{8}$$

$$R_{i0} = \frac{c_{i0}}{\Gamma(\alpha_i)} \int_{t_0}^{t_1} (t_q - \tau)^{\alpha_i - 1}(\tau - t_1)^2 d\tau.$$

For any $j = 1, 2, \ldots, q-1$, and $q = 2, 3, \ldots, N$, we approximate $f_i(\tau, \boldsymbol{x}(\tau))$ on $(t_j, t_{j+1}]$ by the following third-order Taylor expansion:

$$f_i(\tau, \boldsymbol{x}(\tau)) = a_{ij} + b_{ij}(\tau - t_{j+1}) + c_{ij}(\tau - t_{j+1})^2 + d_{ij}(\tau - t_{j+1})^3, \tag{9}$$

where a_{ij}, b_{ij} and c_{ij} are coefficients to be determined, and $d_{ij}(\tau - t_{j+1})^3$ is the remainder. Omitting the remainder, we can use the values of $f_i(\tau, \boldsymbol{x}(\tau))$ at points t_{j-1}, t_j and t_{j+1} to determine a_{ij}, b_{ij} and c_{ij}, yielding the following divided differences:

$$\begin{aligned}
\triangle^0 f_i^{j+1} &:= a_{ij} = f_i(t_{j+1}, \boldsymbol{x}(t_{j+1})),\\
\triangle^1 f_i^{j+1} &:= b_{ij} = \frac{h_{j+1} f_i(t_{j-1}, \boldsymbol{x}(t_{j-1}))}{h_j(h_j + h_{j+1})} - \frac{(h_j + h_{j+1}) f_i(t_j, \boldsymbol{x}(t_j))}{h_j h_{j+1}} + \frac{(h_j + 2h_{j+1}) f_i(t_{j+1}, \boldsymbol{x}(t_{j+1}))}{h_{j+1}(h_j + h_{j+1})},\\
\triangle^2 f_i^{j+1} &:= c_{ij} = \frac{f_i(t_{j-1}, \boldsymbol{x}(t_{j-1}))}{h_j(h_j + h_{j+1})} - \frac{f_i(t_j, \boldsymbol{x}(t_j))}{h_j h_{j+1}} + \frac{f_i(t_{j+1}, \boldsymbol{x}(t_{j+1}))}{h_{j+1}(h_j + h_{j+1})},
\end{aligned}$$

where $h_{j+1} = t_{j+1} - t_j$; and $h_j = t_j - t_{j-1}$. Using these differences, we rewrite Eq (9) as

$$f_i(\tau, \boldsymbol{x}(\tau)) = \triangle^0 f_i^{j+1} + \triangle^1 f_i^{j+1}(\tau - t_{j+1}) + \triangle^2 f_i^{j+1}(\tau - t_{j+1})^2 + d_{ij}(\tau - t_{j+1})^3. \tag{10}$$

Substituting the RHS of Eq (10) into the $(j+1)$th term of the sum in Eq (3) gives

$$\begin{aligned}
&\frac{1}{\Gamma(\alpha_i)} \int_{t_j}^{t_{j+1}} (t_q - \tau)^{\alpha_i - 1}[\ \triangle^0 f_i^{j+1} + \triangle^1 f_i^{j+1}(\tau - t_{j+1}) + \triangle^2 f_i^{j+1}(\tau - t_{j+1})^2 + d_{ij}(\tau - t_{j+1})^3]d\tau\\
&\quad = \frac{h_{j+1} I_{j,1}^{i} + I_{j,2}^{i}}{h_j(h_j + h_{j+1})} f_i(t_{j-1}, \boldsymbol{x}(t_{j-1})) - \frac{(h_j + h_{j+1}) I_{j,1}^{i} + I_{j,2}^{i}}{h_j h_{j+1}} f_i(t_j, \boldsymbol{x}(t_j))\\
&\qquad + \Big[I_{j,0}^{i} + \frac{(h_j + 2h_{j+1}) I_{j,1}^{i} + I_{j,2}^{i}}{h_{j+1}(h_j + h_{j+1})}\Big] f_i(t_{j+1}, \boldsymbol{x}(t_{j+1})) + R_{ij},
\end{aligned} \tag{11}$$

where

$$I_{j,2}^{i} = \frac{h_{j+1}^2 (t_q - t_j)^{\alpha_i}}{\Gamma(\alpha_i + 1)} - \frac{2h_{j+1}(t_q - t_j)^{\alpha_i + 1}}{\Gamma(\alpha_i + 2)} - 2\frac{(t_q - t_{j+1})^{\alpha_i + 2} - (t_q - t_j)^{\alpha_i + 2}}{\Gamma(\alpha_i + 3)}, \tag{12}$$

$$I_{j,1}^{i} = \frac{-h_{j+1}(t_q - t_j)^{\alpha_i}}{\Gamma(\alpha_i + 1)} - \frac{(t_q - t_{j+1})^{\alpha_i + 1} - (t_q - t_j)^{\alpha_i + 1}}{\Gamma(\alpha_i + 2)}, \tag{13}$$

$$I_{j,0}^{i} = \frac{(t_q - t_j)^{\alpha_i} - (t_q - t_{j+1})^{\alpha_i}}{\Gamma(\alpha_i + 1)}, \tag{14}$$

$$R_{ij} = \frac{d_{ij}}{\Gamma(\alpha_i)} \int_{t_j}^{t_{j+1}} (t_q - \tau)^{\alpha_i - 1} (\tau - t_{j+1})^3 d\tau.$$

Combining Eqs (3), (6), and (11), we obtain the following equations, which are equivalent to the equations appearing in Eq (3),

$$\begin{aligned} x_i(t_q) = x_i^0 - \frac{I_{0,1}^i}{h_1} f_i(t_0, \boldsymbol{x}^0) + \Big(I_{0,0}^i + \frac{I_{0,1}^i}{h_1}\Big) f_i(t_1, \boldsymbol{x}(t_1)) + \sum_{j=1}^{q-1} \Bigg\{ \frac{h_{j+1} I_{j,1}^i + I_{j,2}^i}{h_j (h_j + h_{j+1})} \\ \times f_i(t_{j-1}, \boldsymbol{x}(t_{j-1})) - \frac{(h_j + h_{j+1}) I_{j,1}^i + I_{j,2}^i}{h_j h_{j+1}} f_i(t_j, \boldsymbol{x}(t_j)) \\ + \Big[I_{j,0}^i + \frac{(h_j + 2h_{j+1}) I_{j,1}^i + I_{j,2}^i}{h_{j+1}(h_j + h_{j+1})} \Big] f_i(t_{j+1}, \boldsymbol{x}(t_{j+1})) \Bigg\} + R_i^q, \end{aligned} \tag{15}$$

for $i = 1, 2, \ldots, n$, and $q = 1, 2, \ldots, N$, where $R_i^q = \sum_{j=0}^{q-1} R_{ij}$ is the cumulative truncation error up to the point t_q.

Omitting the truncation error R_i^q in Eq (15), we define the following numerical scheme for Eq (3):

$$\begin{aligned} x_i^q = x_i^0 - \frac{I_{0,1}^i}{h_1} f_i(t_0, \boldsymbol{x}^0) + \Big(I_{0,0}^i + \frac{I_{0,1}^i}{h_1}\Big) f_i(t_1, \boldsymbol{x}^1) + \sum_{j=1}^{q-1} \Bigg\{ \frac{h_{j+1} I_{j,1}^i + I_{j,2}^i}{h_j (h_j + h_{j+1})} \\ \times f_i(t_{j-1}, \boldsymbol{x}^{j-1}) - \frac{(h_j + h_{j+1}) I_{j,1}^i + I_{j,2}^i}{h_j h_{j+1}} f_i(t_j, \boldsymbol{x}^j) \\ + \Big[I_{j,0}^i + \frac{(h_j + 2h_{j+1}) I_{j,1}^i + I_{j,2}^i}{h_{j+1}(h_j + h_{j+1})} \Big] f_i(t_{j+1}, \boldsymbol{x}^{j+1}) \Bigg\}, \end{aligned} \tag{16}$$

for $i = 1, 2, \ldots, n$, and $q = 1, 2, \ldots, N$, where x_i^q represents an approximation of $x_i(t_q)$ for each feasible i and q.

We comment that Eq (16) defines an implicit time-stepping scheme for the numerical solution of Eq (1). Since Eq (16) is implicit, we need to use a technique, such as a Newton's method, to solve the nonlinear algebraic equations at each point $t_q, q = 1, 2, \ldots, N$.

3. Convergence analysis

In this section, our focus is on the convergence analysis of the proposed numerical method.

Note that Eq (16) is a nonlinear system in $\boldsymbol{x}^q$ that will be solved by a Newton's iterative method. Thus, we need to show that the Jacobian matrix of this nonlinear system is invertible for any feasible q. For $i = 1, 2, \ldots, n$, and $q = 1, 2, \ldots, N$, let

$$\begin{aligned} F_i(\boldsymbol{x}^q) = x_i^q - x_i^0 + \frac{I_{0,1}^i}{h_1} f_i(t_0, \boldsymbol{x}^0) - \Big(I_{0,0}^i + \frac{I_{0,1}^i}{h_1}\Big) f_i(t_1, \boldsymbol{x}^1) \\ - \sum_{j=1}^{q-1} \Bigg\{ \frac{h_{j+1} I_{j,1}^i + I_{j,2}^i}{h_j (h_j + h_{j+1})} f_i(t_{j-1}, \boldsymbol{x}^{j-1}) - \frac{(h_j + h_{j+1}) I_{j,1}^i + I_{j,2}^i}{h_j h_{j+1}} f_i(t_j, \boldsymbol{x}^j) \end{aligned}$$

$$+\left[I^i_{j,0}+\frac{(h_j+2h_{j+1})I^i_{j,1}+I^i_{j,2}}{h_{j+1}(h_j+h_{j+1})}\right]f_i(t_{j+1},\boldsymbol{x}^{j+1})\Bigg\}.$$

We then have the following theorem.

Theorem 3.1. *Let $\boldsymbol{F}(\boldsymbol{x}^q)=(F_1(\boldsymbol{x}^q),F_2(\boldsymbol{x}^q),\ldots,F_n(\boldsymbol{x}^q))^\top$ for $q\in\{1,2,\ldots,N\}$. Then, the Jacobian matrix of $\boldsymbol{F}(\boldsymbol{x}^q)$ is invertible when $h_q>0$ is sufficiently small.*

Proof. From direct computation, we see that the Jacobian matrix of $\boldsymbol{F}(\boldsymbol{x}^q)$, denoted as $\boldsymbol{F}'(\boldsymbol{x}^q)$, can be rewritten as

$$\boldsymbol{F}'(\boldsymbol{x}^q)=\boldsymbol{I}_n-[b^q_{il}]_{n\times n},$$

for $i=1,2,\ldots,n, l=1,2,\ldots,n$, and $q=1,2,\ldots,N$, where $\boldsymbol{I}_n$ is the $n\times n$ identity matrix; and $[b^q_{il}]_{n\times n}$ is an $n\times n$ matrix with

$$b^q_{il}=\begin{cases}\left(I^i_{0,0}+\dfrac{I^i_{0,1}}{h_1}\right)\dfrac{\partial f_i}{\partial x_l}\Big|_{(t_1,\boldsymbol{x}^1)}, & \text{if } q=1,\\ \left[I^i_{q-1,0}+\dfrac{(h_{q-1}+2h_q)I^i_{q-1,1}+I^i_{q-1,2}}{h_q(h_{q-1}+h_q)}\right]\dfrac{\partial f_i}{\partial x_l}\Big|_{(t_q,\boldsymbol{x}^q)}, & \text{if } q=2,3,\ldots,N.\end{cases}\tag{17}$$

Since $\boldsymbol{f}$ is thrice continuously differentiable, $\dfrac{\partial f_i}{\partial x_l}$ is bounded on $[t_0,t_f]$ for $i=1,2,\ldots,n$, and $l=1,2,\ldots,n$. Let

$$M=\max_{\substack{i\in\{1,\ldots,n\}\\ l\in\{1,\ldots,n\}}}\left\{\left|\frac{\partial f_i}{\partial x_l}\right|\right\},\tag{18}$$

where $|\cdot|$ is the Euclidean norm.

Combining Eqs (7), (8), and (17), we obtain

$$|b^1_{il}|\le\left|\left(I^i_{0,0}+\frac{I^i_{0,1}}{h_1}\right)\right|M=\frac{h_1^{\alpha_i}M}{\Gamma(\alpha_i+2)},\tag{19}$$

for $i=1,2,\ldots,n$, and $l=1,2,\ldots,n$. Furthermore, combining Eqs (12)–(14) and (17), we obtain

$$|b^q_{il}|\le\left|I^i_{q-1,0}+\frac{(h_{q-1}+2h_q)I^i_{q-1,1}+I^i_{q-1,2}}{h_q(h_{q-1}+h_q)}\right|M<\frac{4h_q^{\alpha_i}M}{\Gamma(\alpha_i+1)},\tag{20}$$

for $i=1,2,\ldots,n$, $l=1,2,\ldots,n$, and $q=2,3,\ldots,N$. Note that a key step in the derivation of Eq (20) is given in Appendix.

For given constants $\sigma_i\in(0,1), i=1,2,\ldots,n$, define

$$\hat{h}=\min_{i\in\{1,\ldots,n\}}\left\{\left(\frac{\Gamma(\alpha_i+1)(1-\sigma_i)}{4nM}\right)^{\frac{1}{\alpha_i}}\right\}.\tag{21}$$

Select $h_1\le\hat{h}$ such that for $i=1,2,\ldots,n$,

$$\sum_{l=1}^{n} |b_{il}^1| \leq \frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)} \leq \frac{n\hat{h}^{\alpha_i} M}{\Gamma(\alpha_i + 2)} \leq \frac{\Gamma(\alpha_i + 1)(1 - \sigma_i)}{4\Gamma(\alpha_i + 2)}$$
$$< 1 - \sigma_i \leq \max_{1\leq\iota\leq n}\{1 - \sigma_\iota\} = 1 - \min_{1\leq\iota\leq n}\{\sigma_\iota\}. \tag{22}$$

Choose $h_q \leq \hat{h}$ such that for $i = 1, 2, \ldots, n$, and $q = 2, 3, \ldots, N$,

$$\sum_{l=1}^{n} |b_{il}^q| < \frac{4nh_q^{\alpha_i} M}{\Gamma(\alpha_i + 1)} \leq \frac{4n\hat{h}^{\alpha_i} M}{\Gamma(\alpha_i + 1)} = 1 - \sigma_i \leq \max_{1\leq\iota\leq n}\{1 - \sigma_\iota\} = 1 - \min_{1\leq\iota\leq n}\{\sigma_\iota\}. \tag{23}$$

For $i = 1, 2, \ldots, n$, and $q = 1, 2, \ldots, N$, combining Eqs (22) and (23), we obtain

$$0 < \min_{1\leq\iota\leq n}\{\sigma_\iota\} < 1 - \sum_{l=1}^{n} |b_{il}^q| = 1 - |b_{ii}^q| - \sum_{\substack{l=1 \\ l\neq i}}^{n} |b_{il}^q| \leq |1 - b_{ii}^q| - \sum_{\substack{l=1 \\ l\neq i}}^{n} |b_{il}^q|. \tag{24}$$

It follows from Eq (24) that $\boldsymbol{F}'(\boldsymbol{x}^q)$ is strictly diagonally dominant. According to the Levy-Desplanques theorem in [15], $\boldsymbol{F}'(\boldsymbol{x}^q)$ is non-singular. Therefore, $\boldsymbol{F}'(\boldsymbol{x}^q)$ is invertible, which completes the proof. □

Theorem 3.2. *Let $\boldsymbol{x}(t_q)$ be the exact solution of Eq (3), and let $\boldsymbol{x}^q$ be the solution of Eq (16) for $q = 1, 2, \ldots, N$. Then, there exists a positive constant C, independent of $\boldsymbol{\alpha}$, such that when h_1 is chosen to satisfy $h_1 \leq h^{3/2}$, it holds that*

$$\|\boldsymbol{x}(t_q) - \boldsymbol{x}^q\|_\infty \leq Ch^3, \tag{25}$$

where $\|\cdot\|_\infty$ is the infinity norm; $h = \max_{\ell\in\{2,\ldots,N\}}\{h_\ell\}$ satisfies the condition $h \leq \hat{h}$ ($\hat{h}$ is as defined in Eq (21)).

Proof. The proof is carried out in three steps.

Step 1. The truncation error $\boldsymbol{R}^q$ is of order $O(h^3)$

Let $\boldsymbol{R}^q = (R_1^q, R_2^q, \ldots, R_n^q)^\top$ for $q \in \{1, 2, \ldots, N\}$. From the definition of R_i^q given by Eq (15) and the thrice continuous differentiability of $\boldsymbol{f}$, we deduce that for $i = 1, 2, \ldots, n$, and $q = 1, 2, \ldots, N$,

$$|R_i^q| \leq |R_{i0}| + \sum_{j=1}^{q-1} |R_{ij}|$$
$$\leq \frac{|c_{i0}|h_1^2}{\Gamma(\alpha_i)} \int_{t_0}^{t_1} (t_q - \tau)^{\alpha_i - 1} d\tau + \sum_{j=1}^{q-1} \frac{|d_{ij}|h_{j+1}^3}{\Gamma(\alpha_i)} \int_{t_j}^{t_{j+1}} (t_q - \tau)^{\alpha_i - 1} d\tau$$
$$\leq \frac{dh^3}{\Gamma(\alpha_i)} \sum_{j=0}^{q-1} \int_{t_j}^{t_{j+1}} (t_q - \tau)^{\alpha_i - 1} d\tau = \frac{dh^3}{\Gamma(\alpha_i)} \frac{(t_q - t_0)^{\alpha_i}}{\alpha_i} \leq \frac{dh^3 t_f^{\alpha_i}}{\Gamma(\alpha_i + 1)},$$

because $h_1 \leq h^{3/2}$, where $d = \max_{\substack{i\in\{1,\ldots,n\} \\ j\in\{1,\ldots,q-1\}}} \{|c_{i0}|, |d_{ij}|\}$; and $h = \max_{\ell\in\{2,\ldots,N\}}\{h_\ell\}$.

Recall that $\alpha_i \in (0, 1]$ for $i = 1, 2, \ldots, n$. Thus, $2^{\alpha_i - 1} \leq \Gamma(\alpha_i + 1) \leq 1$, as established in Theorem 4.1 of [16]. Also, it is obvious that $t_f^{\alpha_i} < \max\{1, t_f\}$. Therefore, choosing $\hat{C} = 2d \max\{1, t_f\}$, we obtain

$$\|\boldsymbol{R}^q\|_\infty = O(h^3), \tag{26}$$

for $q = 1, 2, \ldots, N$.

Step 2. The initial step of the mathematical induction process involving Eq (25)

We first show that Eq (25) holds for the case $q = 1$.

Subtracting Eq (16) from Eq (15) and applying the Lagrange mean value theorem, we obtain

$$x_i(t_1) - x_i^1 = [u_{il}^1]_{1\times n}(\boldsymbol{x}(t_1) - \boldsymbol{x}^1) + R_i^1, \tag{27}$$

for $i = 1, 2, \ldots, n$, and $l = 1, 2, \ldots, n$, where u_{il}^1 is defined as

$$u_{il}^1 = \left(I_{0,0}^i + \frac{I_{0,1}^i}{h_1}\right)\frac{\partial f_i}{\partial x_l}\bigg|_{(t_1,\xi_l^1)}. \tag{28}$$

Here, for $l = 1, 2, \ldots, n$, $\xi_l^1 = x_l(t_1) + \lambda_l^1(x_l^1 - x_l(t_1))$ with $\lambda_l^1 \in (0, 1)$.

Based on Eq (27), we obtain

$$|x_i(t_1) - x_i^1| \le \left|[u_{il}^1]_{1\times n}(\boldsymbol{x}(t_1) - \boldsymbol{x}^1)\right| + |R_i^1| \le n\left\|[u_{il}^1]_{1\times n}\right\|_\infty \left\|\boldsymbol{x}(t_1) - \boldsymbol{x}^1\right\|_\infty + \|\boldsymbol{R}^1\|_\infty, \tag{29}$$

for $i = 1, 2, \ldots, n$, and $l = 1, 2, \ldots, n$.

Similar to the derivation of Eq (19), it follows that $\left\|[u_{il}^1]_{1\times n}\right\|_\infty \le \frac{h_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)}$, where M is as defined in Eq (18). Thus, Eq (29) can be rewritten as

$$|x_i(t_1) - x_i^1| \le \frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)}\left\|\boldsymbol{x}(t_1) - \boldsymbol{x}^1\right\|_\infty + \|\boldsymbol{R}^1\|_\infty \le \max_{1\le i\le n}\left\{\frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)}\right\}\left\|\boldsymbol{x}(t_1) - \boldsymbol{x}^1\right\|_\infty + \|\boldsymbol{R}^1\|_\infty,$$

for all feasible i, which implies that

$$\left(1 - \max_{1\le i\le n}\left\{\frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)}\right\}\right)\left\|\boldsymbol{x}(t_1) - \boldsymbol{x}^1\right\|_\infty \le \|\boldsymbol{R}^1\|_\infty. \tag{30}$$

According to the derivation of Eq (22), if we select $h_1 \le \hat{h}$ (recall that $\hat{h}$ is as defined in Eq (21)), then $1 - \frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)} > \min_{1\le\iota\le n}\{\sigma_\iota\} > 0$ is satisfied for all feasible i. Thus, $1 - \max_{1\le i\le n}\left\{\frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)}\right\} > \min_{1\le\iota\le n}\{\sigma_\iota\} > 0$ holds. From Eqs (26) and (30), we obtain

$$\left\|\boldsymbol{x}(t_1) - \boldsymbol{x}^1\right\|_\infty \le \frac{\|\boldsymbol{R}^1\|_\infty}{1 - \max_{1\le i\le n}\left\{\frac{nh_1^{\alpha_i} M}{\Gamma(\alpha_i + 2)}\right\}} < \frac{\|\boldsymbol{R}^1\|_\infty}{\min_{1\le\iota\le n}\{\sigma_\iota\}} = O(h^3). \tag{31}$$

Step 3. Inductive step for Eq (25)

We now consider the case $q \ge 2$. In Step 2, we have shown that Eq (31), i.e., the case $q = 1$, is valid. To apply the mathematical induction, we assume that

$$\|\boldsymbol{x}(t_j) - \boldsymbol{x}^j\|_\infty = O(h^3), \tag{32}$$

for $j = 1, 2, \ldots, q - 1$. Then, we need to show that $\|\boldsymbol{x}(t_q) - \boldsymbol{x}^q\|_\infty = O(h^3)$.

Subtracting Eq (16) from Eq (15) and applying the Lagrange mean value theorem, we obtain

$$x_i(t_q) - x_i^q = [u_{il}^1]_{1\times n}(\boldsymbol{x}(t_1) - \boldsymbol{x}^1) + \sum_{j=1}^{q-1}\left\{[v_{il}^{j-1}]_{1\times n}(\boldsymbol{x}(t_{j-1}) - \boldsymbol{x}^{j-1}) - [w_{il}^{j}]_{1\times n}(\boldsymbol{x}(t_j) - \boldsymbol{x}^j)\right\}$$
$$+ \sum_{j=1}^{q-2}\left\{[z_{il}^{j+1}]_{1\times n}(\boldsymbol{x}(t_{j+1}) - \boldsymbol{x}^{j+1})\right\} + [z_{il}^{q}]_{1\times n}(\boldsymbol{x}(t_q) - \boldsymbol{x}^q) + R_i^q, \tag{33}$$

for $i = 1, 2, \ldots, n$, and $l = 1, 2, \ldots, n$, where u_{il}^1 is as defined in Eq (28), while v_{il}^{j-1}, w_{il}^j, and z_{il}^{j+1} are defined as

$$v_{il}^{j-1} = \frac{h_{j+1}I_{j,1}^i + I_{j,2}^i}{h_j(h_j + h_{j+1})}\frac{\partial f_i}{\partial x_l}\bigg|_{(t_{j-1},\xi_l^{j-1})},$$
$$w_{il}^{j} = \frac{(h_j + h_{j+1})I_{j,1}^i + I_{j,2}^i}{h_j h_{j+1}}\frac{\partial f_i}{\partial x_l}\bigg|_{(t_j,\xi_l^{j})},$$
$$z_{il}^{j+1} = \left[I_{j,0}^i + \frac{(h_j + 2h_{j+1})I_{j,1}^i + I_{j,2}^i}{h_{j+1}(h_j + h_{j+1})}\right]\frac{\partial f_i}{\partial x_l}\bigg|_{(t_{j+1},\xi_l^{j+1})}.$$

Here, for $l = 1, 2, \ldots, n$, and $k = 0, 1, \ldots, q$, $\xi_l^k = x_l(t_k) + \lambda_l^k(x_l^k - x_l(t_k))$ with $\lambda_l^k \in (0, 1)$.

Based on Eq (33), we can deduce that

$$|x_i(t_q) - x_i^q| \le \left|[u_{il}^1]_{1\times n}(\boldsymbol{x}(t_1) - \boldsymbol{x}^1) + \sum_{j=1}^{q-1}\left\{[v_{il}^{j-1}]_{1\times n}(\boldsymbol{x}(t_{j-1}) - \boldsymbol{x}^{j-1}) - [w_{il}^{j}]_{1\times n}(\boldsymbol{x}(t_j) - \boldsymbol{x}^j)\right\}\right.$$
$$\left. + \sum_{j=1}^{q-2}\left\{[z_{il}^{j+1}]_{1\times n}(\boldsymbol{x}(t_{j+1}) - \boldsymbol{x}^{j+1})\right\}\right| + n\left\|[z_{il}^{q}]_{1\times n}\right\|_\infty\|\boldsymbol{x}(t_q) - \boldsymbol{x}^q\|_\infty + \|\boldsymbol{R}^q\|_\infty, \tag{34}$$

for $i = 1, 2, \ldots, n$, and $l = 1, 2, \ldots, n$.

Similar to the derivation of Eq (20), we can show that $\left\|[z_{il}^{q}]_{1\times n}\right\|_\infty < \dfrac{4h_q^{\alpha_i}M}{\Gamma(\alpha_i + 1)}$. Thus, Eq (34) can be expressed as

$$|x_i(t_q) - x_i^q| < \left|[u_{il}^1]_{1\times n}(\boldsymbol{x}(t_1) - \boldsymbol{x}^1) + \sum_{j=1}^{q-1}\left\{[v_{il}^{j-1}]_{1\times n}(\boldsymbol{x}(t_{j-1}) - \boldsymbol{x}^{j-1}) - [w_{il}^{j}]_{1\times n}(\boldsymbol{x}(t_j) - \boldsymbol{x}^j)\right\}\right.$$
$$\left. + \sum_{j=1}^{q-2}\left\{[z_{il}^{j+1}]_{1\times n}(\boldsymbol{x}(t_{j+1}) - \boldsymbol{x}^{j+1})\right\}\right| + \max_{1\le i\le n}\left\{\frac{n4h_q^{\alpha_i}M}{\Gamma(\alpha_i + 1)}\right\}\|\boldsymbol{x}(t_q) - \boldsymbol{x}^q\|_\infty + \|\boldsymbol{R}^q\|_\infty,$$

for all feasible i and l, which implies that

$$\left(1 - \max_{1\le i\le n}\left\{\frac{n4h_q^{\alpha_i}M}{\Gamma(\alpha_i + 1)}\right\}\right)\|\boldsymbol{x}(t_q) - \boldsymbol{x}^q\|_\infty$$
$$< \left|[u_{il}^1]_{1\times n}(\boldsymbol{x}(t_1) - \boldsymbol{x}^1) + \sum_{j=1}^{q-1}\left\{[v_{il}^{j-1}]_{1\times n}(\boldsymbol{x}(t_{j-1}) - \boldsymbol{x}^{j-1}) - [w_{il}^{j}]_{1\times n}(\boldsymbol{x}(t_j) - \boldsymbol{x}^j)\right\}\right.$$

$$+\sum_{j=1}^{q-2}\left\{[z_{il}^{j+1}]_{1\times n}(\boldsymbol{x}(t_{j+1})-\boldsymbol{x}^{j+1})\right\}\Bigg|+\|\boldsymbol{R}^q\|_\infty. \tag{35}$$

According to the derivation of Eq (23), if we select $h_q \le \hat{h}$, then $1-\dfrac{n4h_q^{\alpha_i}M}{\Gamma(\alpha_i+1)} \ge \min\limits_{1\le\iota\le n}\{\sigma_\iota\} > 0$ is satisfied for all feasible i. Thus, $1-\max\limits_{1\le i\le n}\left\{\dfrac{n4h_q^{\alpha_i}M}{\Gamma(\alpha_i+1)}\right\} \ge \min\limits_{1\le\iota\le n}\{\sigma_\iota\} > 0$ holds. From Eqs (26), (32), and (35) and the boundedness of u_{il}^1, v_{il}^{j-1}, w_{il}^j and z_{il}^{j+1} for $i = 1,2,\ldots,n$, $l = 1,2,\ldots,n$, and $j = 1,2,\ldots,q-1$, we obtain

$$\|\boldsymbol{x}(t_q)-\boldsymbol{x}^q\|_\infty = \frac{O(h^3)+\|\boldsymbol{R}^q\|_\infty}{1-\max\limits_{1\le i\le n}\left\{\dfrac{n4h_q^{\alpha_i}M}{\Gamma(\alpha_i+1)}\right\}} = O(h^3).$$

This completes the proof. □

Remark 1. *The choice of $h_1 \le h^{3/2}$ aligns with the Rannacher time-stepping technique as discussed in [17, 18]. This ensures that the overall accuracy of the proposed numerical method is not affected by the first time step.*

Theorem 3.3. *Let $\hat{\boldsymbol{x}}^q$ be the numerical solution of Eq (16) obtained by Newton's method for $q = 1,2,\ldots,N$. Then, there exists a positive constant $\tilde{C}$, independent of both h (as defined in Theorem 3.2) and $\boldsymbol{\alpha}$, such that*

$$\|\boldsymbol{x}(t_q)-\hat{\boldsymbol{x}}^q\|_\infty \le \tilde{C}h^3. \tag{36}$$

Proof. From Theorem 2.1 in [19], we note that there exists a well-defined number of iterations such that

$$\|\boldsymbol{x}^q-\hat{\boldsymbol{x}}^q\|_\infty = \begin{cases} O(h_1^2), & \text{if } q=1,\\ O(h_q^3), & \text{if } q=2,3,\ldots,N,\end{cases} \tag{37}$$

where $h_1 \le h^{3/2}$ is as defined in Theorem 3.2. Then, Eq (37) can be expressed as

$$\|\boldsymbol{x}^q-\hat{\boldsymbol{x}}^q\|_\infty = O(h^3), \tag{38}$$

where h is as defined in Theorem 3.2.

Combining Eqs (25) and (38), we obtain

$$\|\boldsymbol{x}(t_q)-\hat{\boldsymbol{x}}^q\|_\infty \le \|\boldsymbol{x}(t_q)-\boldsymbol{x}^q\|_\infty + \|\boldsymbol{x}^q-\hat{\boldsymbol{x}}^q\|_\infty = O(h^3).$$

Therefore, we can infer that Eq (36) holds. The proof is complete. □

Remark 2. *The error estimate (36) shows that for $q = 1,2,\ldots,N$, the numerical solution $\hat{\boldsymbol{x}}^q$ obtained by Newton's method converges to the exact solution $\boldsymbol{x}(t_q)$ with third-order accuracy. Note that this convergence rate is independent of $\boldsymbol{\alpha}$, which is a significant distinction from the methods reported in [7–9, 14].*

4. Numerical examples

In this section, we will solve four numerical examples to verify the effectiveness and convergence rate of the numerical method proposed in Section 3. Here, the stopping criterion for Newton's method is set to be 10^{-12}, and all computations are conducted in the MATLAB 2022b environment on a PC equipped with a 2.80 GHz Intel Core i7-1165G7 CPU and 16.0 GB RAM.

For all test examples, the following graded mesh is used:

$$t_q = \Big(\frac{q}{N}\Big)^2 t_f, \quad q = 0, 1, \ldots, N,$$

where N is a positive integer. This graded mesh satisfies the condition in Theorem 3.2. The computed maximum error and convergence order are defined as

$$E_{\max} := E_h = \max_{\substack{i\in\{1,\ldots,n\}\\ q\in\{1,\ldots,N\}}} \Big\{\Big|x_i(t_q) - \hat{x}_i^q\Big|\Big\},$$
$$\text{Order} := \log\Big(\frac{E_h}{E_{h'}}\Big)\Big/\log\Big(\frac{h}{h'}\Big),$$

respectively, where h is as defined in Theorem 3.2 under the partition number N; and h' is the maximum step size under the partition number N'.

Remark 3. *Note that other strategies can also be used in our calculations, as long as they comply with Remark 1.*

4.1. Example 1

Consider the following nonlinear FODE [14]:

$$\begin{aligned} {}_0^C D_t^\alpha x(t) &= \frac{\Gamma(9)}{\Gamma(9-\alpha)} t^{8-\alpha} - 3\frac{\Gamma(5+\frac{\alpha}{2})}{\Gamma(5-\frac{\alpha}{2})} t^{4-\frac{\alpha}{2}} + (t^8 - 3t^{4+\frac{\alpha}{2}})^3 - x^3(t), \quad t \in (0, 1], \\ x(0) &= 0. \end{aligned}$$

The exact solution of this nonlinear FODE is $x(t) = t^8 - 3t^{4+\frac{\alpha}{2}}$. Our proposed numerical method is applied to solve this example. For comparison, we also solve this example by the one-step method in [13]. The computed numerical results and those reported in [14] are listed in Table 1, from which we see that the maximum errors in all methods decrease as the number of mesh points N increases. Notably, the maximum error obtained by our method is significantly smaller than those obtained in [13] for different values of α. Compared with [14], the convergence order computed by our method is comparable to that of Q-PCP for $\alpha = 0.1$ and significantly higher than those of Q-PCD for $\alpha = 0.8$ and $\alpha = 0.9$. This is because the theoretical convergence order of Q-PCP deteriorates as α increases. It is clear from Table 1 that the convergence order obtained by our method approaches to the theoretical value of 3. More importantly, the convergence order is independent of both α and h. Furthermore, we perform 50 tests and take the average time as the computational time for all test examples. From Table 1, we see that the one-step method, utilizing a second-order Taylor expansion and an explicit numerical format, achieves slightly shorter computational time compared to our method at the expense of lower accuracy. Figure 1 illustrates the state trajectories for various α when $N = 640$. From

Figure 1, we can see that the numerical solutions obtained by our method closely match the exact solutions, which is consistent with the results in Table 1.

Table 1. Maximum error, convergence order, and computational time for Example 1.

$\alpha = 0.1$	Q-PCP [14]		One-step [13]			Our method		
N	$E_{\max}$	Order	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
10	4.9987E-04	-	8.7421E-02	-	3.8897E-04	7.5535E-04	-	1.2221E-03
20	7.9647E-05	2.6499	2.3923E-02	1.8696	1.1973E-03	1.8128E-04	2.1391	3.1013E-03
40	1.1982E-05	2.7328	6.0795E-03	1.9764	4.2962E-03	3.2801E-05	2.5126	8.9735E-03
80	1.7626E-06	2.7651	1.4810E-03	2.0374	1.6198E-02	5.1679E-06	2.6906	3.1299E-02
160	2.5521E-07	2.7879	3.5231E-04	2.0717	6.3558E-02	7.5681E-07	2.7842	1.1410E-01
320	3.6546E-08	2.8039	8.2776E-05	2.0896	2.5346E-01	1.0635E-07	2.8375	4.4582E-01
640	5.1822E-09	2.8181	1.9326E-05	2.0987	1.0010E+00	1.4574E-08	2.8706	1.7106E+00
$\alpha = 0.8$	Q-PCD [14]		One-step [13]			Our method		
N	$E_{\max}$	Order	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
10	9.8400E-02	-	2.8733E-02	-	3.9887E-04	1.3935E-02	-	1.2125E-03
20	9.8625E-03	3.3186	4.8370E-03	2.5705	1.1945E-03	2.4250E-03	2.6209	2.8512E-03
40	8.7791E-04	3.4898	8.7415E-04	2.4682	4.3209E-03	3.4682E-04	2.8583	9.2150E-03
80	1.0277E-04	3.0947	1.7301E-04	2.3370	1.6333E-02	4.6295E-05	2.9319	3.0611E-02
160	1.4646E-05	2.8108	3.7074E-05	2.2224	6.3507E-02	5.9772E-06	2.9668	1.1346E-01
320	2.2964E-06	2.6731	8.4136E-06	2.1396	2.5051E-01	7.5924E-07	2.9836	4.3508E-01
640	3.7444E-07	2.6166	1.9831E-06	2.0850	1.0118E+00	9.5666E-08	2.9919	1.7124E+00
$\alpha = 0.9$	Q-PCD [14]		One-step [13]			Our method		
N	$E_{\max}$	Order	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
10	6.5505E-02	-	2.4339E-02	-	4.0538E-04	1.6829E-02	-	1.2150E-03
20	7.2229E-03	3.1810	3.8287E-03	2.6683	1.2207E-03	2.8795E-03	2.6462	2.9877E-03
40	8.5280E-04	3.0823	6.5177E-04	2.5544	4.3127E-03	4.0702E-04	2.8755	9.2930E-03
80	1.2478E-04	2.7728	1.2335E-04	2.4016	1.6135E-02	5.3973E-05	2.9416	3.1094E-02
160	2.0420E-05	2.6113	2.5727E-05	2.2614	6.3355E-02	6.9457E-06	2.9715	1.1318E-01
320	3.4802E-06	2.5527	6.7752E-06	1.9249	2.5034E-01	8.8078E-07	2.9860	4.3424E-01
640	6.0180E-07	2.5318	1.8000E-06	1.9123	9.9944E-01	1.1088E-07	2.9932	1.7159E+00

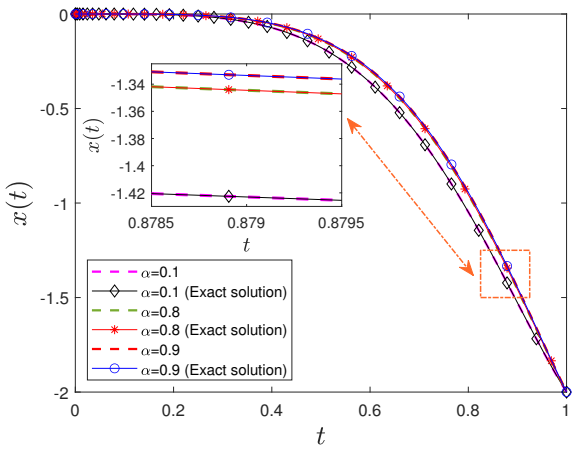

Figure 1. State trajectories when $N = 640$ for Example 1.

4.2. Example 2

Consider the following nonlinear FODE [20]:

$$ {}_0^C D_t^\alpha x(t) = -(x(t) - 0.01t^2 - 1)^2 - \cos^2(4\sqrt{t}) + 2\sqrt{\pi} J_0(4\sqrt{t}) + 1 + \frac{2t^{1.5}}{75\sqrt{\pi}}, \quad t \in (0, 10], $$

$$ x(0) = 1, $$

where $J_0(\cdot)$ is the zero-order Bessel function of the first kind [21].

This problem is solved again by using our proposed numerical scheme. For $\alpha = 0.5$, the exact solution of this example is $x(t) = \sin(4\sqrt{t}) + 0.01t^2 + 1$. For comparison, we also solve this example by using the implicit product integration trapezoidal rule (PI 2 Impl.) in [22]. The obtained numerical results are shown in Table 2. From Table 2, we see that the maximum error obtained by our method is much smaller than that obtained in [22] for various values of N. We also observe that the convergence order in [22] increases with the increasing value of N. In contrast, the convergence order computed by our method remains stable and is close to the theoretical value 3. This reconfirms that the convergence order of the proposed numerical method is independent of both α and h. Nevertheless, the computational time of the PI 2 Impl. method is significantly shorter compared to our method, attributed to a lower accuracy of the trapezoidal rule and the efficient treatment of persistent memory of fractional integrals. Figure 2 depicts the state trajectories when $N = 5120$. From Figure 2, it can be seen that the numerical solutions obtained by our method exhibit greater accuracy compared to those obtained using the method reported in [22].

Table 2. Maximum error, convergence order, and computational time for Example 2.

$\alpha = 0.5$	PI 2 Impl. [22]			Our method		
N	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
320	9.1818E-01	-	7.8216E-03	3.1282E-01	-	4.9170E-01
640	7.2655E-01	0.3377	1.3015E-02	2.1987E-02	3.8349	1.9347E+00
1280	5.0259E-01	0.5317	2.2718E-02	2.6433E-03	3.0580	7.7414E+00
2560	2.7916E-01	0.8483	3.8415E-02	3.3003E-04	3.0025	3.0429E+01
5120	1.1294E-01	1.3055	7.8790E-02	4.1351E-05	2.9970	1.2082E+02

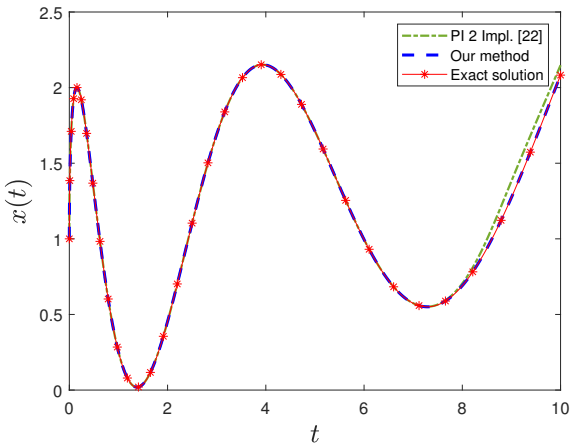

Figure 2. State trajectories when $N = 5120$ for Example 2.

4.3. Example 3

Consider the following nonlinear FODE [9]:

$$
\begin{aligned}
{}_0^C D_t^{\alpha} x(t) &= \frac{\Gamma(4+\alpha)}{6} t^3 + t^{6+2\alpha} - x^2(t), \quad t \in (0,1], \\
x(0) &= 0.
\end{aligned}
$$

The exact solution of this example is $x(t) = t^{3+\alpha}$. In contrast to the method proposed in [9] for directly approximating fractional derivatives and the method developed in [23], which employs the three-step Newton polynomial to approximate Volterra integrals, we use our proposed method and the MATLAB code provided in [23] to solve this problem for $\alpha = 0.3$, $\alpha = 0.6$, and $\alpha = 0.9$. The computed numerical results and those reported in [9] are listed in Table 3. From Table 3, we observe that the maximum error obtained by our method is significantly superior to those obtained in [9, 23]. Furthermore, we note that our method exhibits a consistent convergence order close to 3 for various fractional orders, indicating its stability and efficiency. We observe that the convergence order obtained in [23] is also close to 3. However, the computational complexity in [23] is higher compared to our method. Specifically, for calculating $\boldsymbol{x}^q$, the values of $\boldsymbol{x}^{q-3}$, $\boldsymbol{x}^{q-2}$, and $\boldsymbol{x}^{q-1}$, $q = 3, 4, \ldots, N$, are required in [23], while the one-step Euler method and the two-step Adams-Bashforth method are used to obtain the values of $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$, respectively. In contrast, our method only requires the values of $\boldsymbol{x}^{q-2}$ and $\boldsymbol{x}^{q-1}$ when calculating $\boldsymbol{x}^q$ for $q = 2, 3, \ldots, N$. The value of $\boldsymbol{x}^1$ can be calculated by using the second-order Taylor expansion. Thus, it can be seen from Table 3 that the computational time in [23] is higher than that in our method. The state trajectories for different fractional orders with $N = 2048$ are depicted in Figure 3. As it can be seen from Figure 3, our method shows satisfactory accuracy.

Table 3. Maximum error, convergence order, and computational time for Example 3.

$\alpha = 0.3$	Results from [9]		Results from [23]			Our method		
N	$E_{\max}$	Order	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
128	1.0154E-06	-	1.6389E-06	-	9.0903E-02	4.1874E-07	-	6.2773E-02
256	1.4941E-07	2.7647	1.9683E-07	3.0577	3.6583E-01	5.4184E-08	2.9585	2.3771E-01
512	2.3628E-08	2.6607	2.3785E-08	3.0488	1.5023E+00	6.9524E-09	2.9665	9.2097E-01
1024	3.7049E-09	2.6729	2.8896E-09	3.0411	6.1037E+00	8.8676E-10	2.9730	3.6671E+00
2048	5.6279E-10	2.7187	3.5272E-10	3.0343	2.4766E+01	1.1261E-10	2.9783	1.4613E+01
$\alpha = 0.6$	Results from [9]		Results from [23]			Our method		
N	$E_{\max}$	Order	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
128	1.0167E-05	-	1.8931E-06	-	9.0620E-02	7.2056E-07	-	6.2624E-02
256	1.9494E-06	2.3828	2.3429E-07	3.0144	3.6675E-01	9.0807E-08	2.9967	2.3662E-01
512	3.7176E-07	2.3906	2.9087E-08	3.0098	1.4719E+00	1.1405E-08	2.9974	9.2196E-01
1024	7.0707E-08	2.3944	3.6190E-09	3.0067	6.1504E+00	1.4297E-09	2.9980	3.6364E+00
2048	1.3497E-08	2.3891	4.5095E-10	3.0046	2.4819E+01	1.7901E-10	2.9987	1.4423E+01
$\alpha = 0.9$	Results from [9]		Results from [23]			Our method		
N	$E_{\max}$	Order	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
128	9.1822E-05	-	2.7547E-06	-	9.0707E-02	1.0554E-06	-	6.2696E-02
256	2.1549E-05	2.0911	3.4432E-07	3.0001	3.6541E-01	1.3237E-07	3.0036	2.3620E-01
512	5.0416E-06	2.0957	4.3036E-08	3.0001	1.4581E+00	1.6574E-08	3.0018	9.2368E-01
1024	1.1777E-06	2.0978	5.3791E-09	3.0001	6.2194E+00	2.0735E-09	3.0009	3.6373E+00
2048	2.7492E-07	2.0989	6.7236E-10	3.0001	2.4452E+01	2.5929E-10	3.0005	1.4573E+01

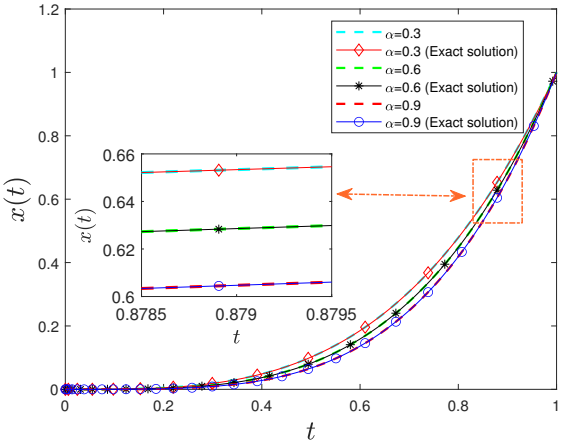

Figure 3. State trajectories when $N = 2048$ for Example 3.

4.4. Example 4

Consider the following nonlinear FODEs [24]:

$$\begin{cases} {}_0^C D_t^{\alpha_1} x_1(t) = \dfrac{1}{2} x_1(t), \\ {}_0^C D_t^{\alpha_2} x_2(t) = x_1^2(t) + x_2(t), \quad t \in (0, 1], \\ \boldsymbol{x}(0) = (1, 0)^\top. \end{cases}$$

The exact solutions of this example are $x_1(t) = e^{\frac{t}{2}}$ and $x_2(t) = te^t$ for $\alpha_1 = \alpha_2 = 1.0$. We solve this example for $(\alpha_1, \alpha_2)^\top = (0.7, 0.9)^\top$, $(\alpha_1, \alpha_2)^\top = (0.9, 0.7)^\top$, and $(\alpha_1, \alpha_2)^\top = (1.0, 1.0)^\top$ by our proposed method. Since the exact solutions for the cases $(\alpha_1, \alpha_2)^\top = (0.7, 0.9)^\top$ and $(\alpha_1, \alpha_2)^\top = (0.9, 0.7)^\top$ are unknown, we use the numerical solutions for $N = 2560$ as approximations to the exact solutions. The computed maximum error, convergence order, and computational time are presented in Table 4. It is observed from Table 4 that the value of the maximum error decreases as the value of N increases. For different fractional orders, the computed convergence order of our method is roughly 3. To visualize the numerical results, we plot the state trajectories corresponding to different fractional orders in Figure 4.

Table 4. Maximum error, convergence order, and computational time for Example 4.

$(\alpha_1, \alpha_2)^\top$	$(0.7, 0.9)^\top$			$(0.9, 0.7)^\top$			$(1.0, 1.0)^\top$		
N	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)	$E_{\max}$	Order	CPU(s)
10	3.6411E-03	-	1.9605E-03	4.9142E-03	-	1.9573E-03	1.4588E-03	-	2.0733E-03
20	5.0444E-04	2.9626	5.9179E-03	6.8817E-04	2.9465	6.2036E-03	1.9848E-04	2.9898	6.0395E-03
40	6.6335E-05	2.9816	2.0033E-02	9.0991E-05	2.9736	2.0026E-02	2.5849E-05	2.9959	1.9862E-02
80	8.5101E-06	2.9897	7.3699E-02	1.1702E-05	2.9861	7.3156E-02	3.2967E-06	2.9983	7.2059E-02
160	1.0751E-06	2.9983	2.8322E-01	1.4831E-06	2.9936	2.8699E-01	4.1619E-07	2.9993	2.8037E-01
320	1.3132E-07	3.0402	1.1216E+00	1.8506E-07	3.0094	1.1272E+00	5.2281E-08	2.9997	1.0886E+00
640	1.5305E-08	3.1045	4.5994E+00	2.1624E-08	3.1008	4.5519E+00	6.5512E-09	2.9998	4.3141E+00

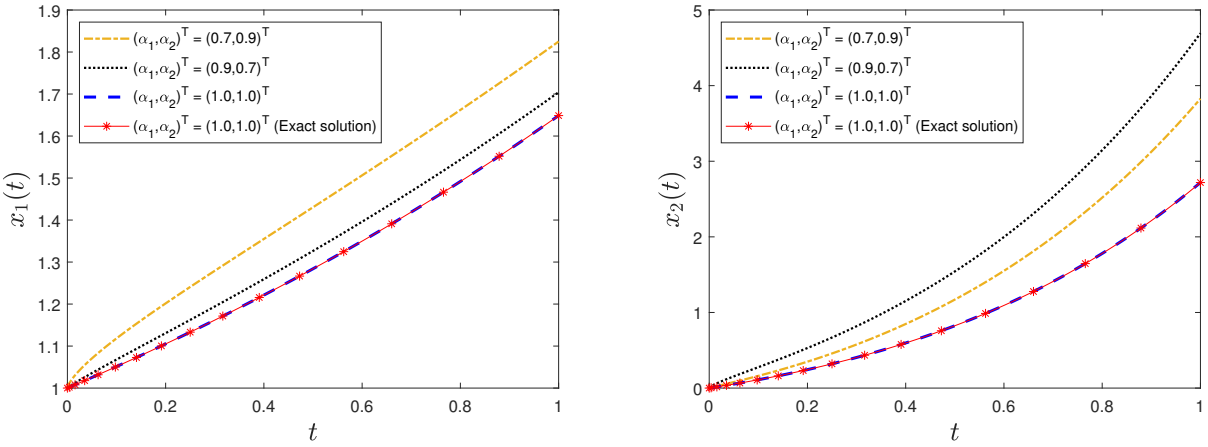

Figure 4. State trajectories when $N = 640$ for Example 4.

5. Conclusions

This paper has developed the third-order numerical method for solving nonlinear FODEs in the sense of Liouville-Caputo fractional derivatives. The fractional orders of these nonlinear FODEs can differ from each other. At each mesh point of a given mesh, we approximate the equivalent Volterra integral equations by using the third-order Taylor expansion (for the first subinterval, the second-order Taylor expansion is used). This approximation yields the implicit nonlinear algebraic equations that can be iteratively solved by the Newton's method. Furthermore, the convergence analysis and error estimate are performed, showing that the convergence rate of the proposed method is third order, independent of the fractional order. Finally, four non-trivial numerical examples are solved to illustrate the effectiveness and the convergence of the proposed method. In our future research, we will develop effective numerical methods for solving various fractional optimal control problems.

Author contributions

Xiaopeng Yi: Conceptualization, Investigation, Methodology, Software, Writing-original draft; Chongyang Liu: Conceptualization, Investigation, Methodology, Supervision, Software, Writing-review and editing; Huey Tyng Cheong: Conceptualization, Investigation, Methodology, Supervision, Writing-original draft; Kok Lay Teo: Conceptualization, Investigation, Methodology, Supervision, Writing-review and editing; Song Wang: Conceptualization, Investigation, Methodology, Writing-review and editing. All authors have read and agreed to the published version of the manuscript.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

This work is supported by the Fundamental Research Grant Scheme of Malaysia (grant number FRGS/1/2021/STG06/SYUC/03/1), the National Natural Science Foundation of China (No. 12271307), and the Shandong Province Natural Science Foundation of China (No. ZR2023MA054).

Conflict of interest

The authors declare that they have no competing interests.

References

1. K. Diethelm, *The analysis of fractional differential equations*, Berlin: Springer, 2010. https://doi.org/10.1007/978-3-642-14574-2
2. R. Hilfer, *Applications of fractional calculus in physics*, Singapore: World Scientific, 2000. https://doi.org/10.1142/3779
3. H. Sun, Y. Zhang, D. Baleanu, W. Chen, Y. Chen, A new collection of real world applications of fractional calculus in science and engineering, *Commun. Nonlinear Sci. Numer. Simul.*, **64** (2018), 213–231. https://doi.org/10.1016/j.cnsns.2018.04.019
4. C. Liu, X. Yi, Y. Feng, Modelling and parameter identification for a two-stage fractional dynamical system in microbial batch process, *Nonlinear Anal. Model. Control*, **27** (2022), 350–367. https://doi.org/10.15388/namc.2022.27.26234
5. Z. M. Odibat, Computational algorithms for computing the fractional derivatives of functions, *Math. Comput. Simulation*, **79** (2009), 2013–2020. https://doi.org/10.1016/j.matcom.2008.08.003
6. C. Li, F. Zeng, Finite difference methods for fractional differential equations, *Int. J. Bifurcat. Chaos*, **22** (2012), 1230014. https://doi.org/10.1142/S0218127412300145
7. G. H. Gao, Z. Z. Sun, H. W. Zhang, A new fractional numerical differentiation formula to approximate the Caputo fractional derivative and its applications, *J. Comput. Phys.*, **259** (2014), 33–50. https://doi.org/10.1016/j.jcp.2013.11.017
8. Y. Yan, K. Pal, N. J. Ford, Higher order numerical methods for solving fractional differential equations, *Bit Numer. Math.*, **54** (2014), 555–584. https://doi.org/10.1007/s10543-013-0443-3
9. X. Zhang, J. Cao, A high order numerical method for solving Caputo nonlinear fractional ordinary differential equations, *AIMS Math.*, **6** (2021), 13187–13209. https://doi.org/10.3934/math.2021762
10. K. Diethelm, N. J. Ford, A. D. Freed, A predictor-corrector approach for the numerical solution of fractional differential equations, *Nonlinear Dyn.*, **29** (2002), 3–22. https://doi.org/10.1023/A:1016592219341
11. K. Diethelm, N. J. Ford, A. D. Freed, Detailed error analysis for a fractional Adams method, *Numer. Algorithms*, **36** (2004), 31–52. https://doi.org/10.1023/B:NUMA.0000027736.85078.be

12. W. Deng, C. Li, Numerical schemes for fractional ordinary differential equations, In: P. Miidla, *Numerical modelling*, InTech, Rijeka, 2012, 355–374. https://doi.org/10.5772/34965

13. W. Li, S. Wang, V. Rehbock, A 2nd-order one-point numerical integration scheme for fractional ordinary differential equations, *Numer. Algebra Control Optim.*, **7** (2017), 273–287. https://doi.org/10.3934/naco.2017018

14. H. Kim, K. H. Kim, S. Lee, B. Jang, New explicit and accelerated techniques for solving fractional order differential equations, *Appl. Math. Comput.*, **379** (2020), 125228. https://doi.org/10.1016/j.amc.2020.125228

15. E. E. Tyrtyshnikov, *A brief introduction to numerical analysis*, Boston: Springer, 1997. https://doi.org/10.1007/978-0-8176-8136-4

16. A. Laforgia, P. Natalini, Exponential, gamma and polygamma functions: simple proofs of classical and new inequalities, *J. Math. Anal. Appl.*, **407** (2013), 495–504. https://doi.org/10.1016/j.jmaa.2013.05.045

17. R. Rannacher, Finite element solution of diffusion problems with irregular data, *Numer. Math.*, **43** (1984), 309–327. https://doi.org/10.1007/BF01390130

18. K. Zhang, X. Yang, S. Wang, K. L. Teo, Numerical performance of penalty method for American option pricing, *Optim. Methods Softw.*, **25** (2010), 737–752. https://doi.org/10.1080/10556780903051930

19. T. Yamamoto, A method for finding sharp error bounds for Newton's method under the Kantorovich assumptions, *Numer. Math.*, **49** (1986), 203–220. https://doi.org/10.1007/BF01389624

20. S. Wang, W. Li, C. Liu, On necessary optimality conditions and exact penalization for a constrained fractional optimal control problem, *Optim. Control Appl. Methods*, **43** (2022), 1096–1108. https://doi.org/10.1002/oca.2877

21. M. Abramowitz, I. A. Stegun, *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, United States: U.S. Department of Commerce, National Bureau of Standards, 1964. Available from: `https://archive.org/details/handbookofmathem1964abra`.

22. R. Garrappa, Numerical solution of fractional differential equations: a survey and a software tutorial, *Mathematics*, **6** (2018), 16. https://doi.org/10.3390/math6020016

23. A. Atangana, S. İ. Araz, *New numerical scheme with Newton polynomial: theory, methods, and applications*, London: Elsevier, 2021. https://doi.org/10.1016/C2020-0-02711-8

24. O. Postavaru, An efficient numerical method based on Fibonacci polynomials to solve fractional differential equations, *Math. Comput. Simulation*, **212** (2023), 406–422. https://doi.org/10.1016/j.matcom.2023.04.028

Appendix

In Eq (20), the following inequality holds:

$$\begin{aligned}
&\left|I^i_{q-1,0} + \frac{(h_{q-1}+2h_q)I^i_{q-1,1} + I^i_{q-1,2}}{h_q(h_{q-1}+h_q)}\right| \\
&= \left|\frac{h_q^{\alpha_i}}{\Gamma(\alpha_i+1)} + \frac{h_{q-1}h_q^{\alpha_i}}{(h_{q-1}+h_q)\Gamma(\alpha_i+2)} - \frac{h_{q-1}h_q^{\alpha_i}+h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)} + \frac{2h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+3)}\right| \\
&\le \left|\frac{h_q^{\alpha_i}}{\Gamma(\alpha_i+1)}\right| + \left|\frac{h_{q-1}h_q^{\alpha_i}}{(h_{q-1}+h_q)\Gamma(\alpha_i+2)}\right| + \left|\frac{h_{q-1}h_q^{\alpha_i}+h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)}\right| + \left|\frac{2h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+3)}\right| \\
&< \left|\frac{h_q^{\alpha_i}}{\Gamma(\alpha_i+1)}\right| + \left|\frac{h_{q-1}h_q^{\alpha_i}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)}\right| + \left|\frac{h_{q-1}h_q^{\alpha_i}+h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)}\right| + \left|\frac{2h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)}\right| \\
&= \frac{h_q^{\alpha_i}}{\Gamma(\alpha_i+1)} + \frac{2h_{q-1}h_q^{\alpha_i}+3h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)} = \frac{3h_{q-1}h_q^{\alpha_i}+4h_q^{\alpha_i+1}}{(h_{q-1}+h_q)\Gamma(\alpha_i+1)} < \frac{4h_q^{\alpha_i}}{\Gamma(\alpha_i+1)}.
\end{aligned}$$

AIMS Press