A Preconditioning Technique for All-at-Once System from the Nonlinear Tempered Fractional Diffusion Equation

Yong-Liang Zhao · Pei-Yong Zhu · Xian-Ming Gu · Xi-Le Zhao · Huan-Yan Jian

Received: 21 January 2019 / Revised: 30 September 2019 / Accepted: 30 December 2019 / Published online: 23 March 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
An all-at-once system of nonlinear algebra equations arising from the nonlinear tempered fractional diffusion equation with variable coefficients is studied. Firstly, both the nonlinear and linearized implicit difference schemes are proposed to approximate such the nonlinear equation with continuous/discontinuous coefficients. The stabilities and convergences of the two numerical schemes are proved under several assumptions. Numerical examples show that the convergence orders of these two schemes are 1 in both time and space. Secondly, the nonlinear all-at-once system is derived from the nonlinear implicit scheme. Newton’s method, whose initial value is obtained by interpolating the solution of the linearized implicit scheme on the coarse space, is chosen to solve such a nonlinear all-at-once system. To accelerate the speed of solving the Jacobian equations appeared in Newton’s method, a robust preconditioner is developed and analyzed. Numerical examples are reported to illustrate the effectiveness of our proposed preconditioner. Meanwhile, they also imply that our chosen initial guess for Newton’s method is feasible.

Keywords Nonlinear tempered fractional diffusion equation · All-at-once system · Newton’s method · Krylov subspace method · Toeplitz matrix · Banded Toeplitz preconditioner

Mathematics Subject Classification 65L05 · 65N22 · 65F10

Xian-Ming Gu
 guxianming@live.cn; guxm@swufe.edu.cn

Yong-Liang Zhao
 ylzhaofde@sina.com

Pei-Yong Zhu
 zpy6940@uestc.edu.cn

Xi-Le Zhao
 xlzhao122003@163.com

Huan-Yan Jian
 uestc_hyjian@sina.com

1 School of Mathematical Sciences, University of Electronic Science and Technology of China, Chengdu 611731, Sichuan, People’s Republic of China

2 School of Economic Mathematics/Institute of Mathematics, Southwestern University of Finance and Economics, Chengdu 611130, Sichuan, People’s Republic of China
1 Introduction

In this work, we mainly focus on solving the all-at-once system arising from the following nonlinear tempered fractional diffusion equation (NL-TFDE):

\[
\begin{align*}
\frac{\partial u(x,t)}{\partial t} &= d_+^\alpha(x) D_x^{\alpha,\lambda} u(x,t) + d_-^\alpha(x) D_x^{\alpha,\lambda} u(x,t) + f(u(x,t), x, t), \quad (x, t) \in [a, b] \times (0, T], \\
u(a, t) &= u(b, t) = 0, \\
u(x, 0) &= u_0(x), \\
\end{align*}
\]

where \( \alpha \in (1, 2), \lambda \geq 0, d_+(x) \geq d_-(x) > 0 \) and \( f(u(x,t), x, t) \) is a nonlinear source term which satisfies the Lipschitz condition, i.e. for some \( L > 0 \),

\[ | f(r_1, x, t) - f(r_2, x, t) | \leq L \cdot | r_1 - r_2 |, \text{ for all } r_1, r_2 \text{ over } [a, b] \times [0, T]. \]

The variants used in Eq. (1.1) of the left and right Riemann–Liouville tempered fractional derivatives are respectively defined as [1–3]

\[
\begin{align*}
a D_x^{\alpha,\lambda} u(x,t) &= a D_x^{\alpha,\lambda} u(x,t) - a \lambda \alpha^{\frac{1}{2}} \frac{\partial u(x,t)}{\partial x} - \lambda \alpha^2 u(x,t), \\
x D_b^{\alpha,\lambda} u(x,t) &= x D_b^{\alpha,\lambda} u(x,t) + a \lambda \alpha^{\frac{1}{2}} \frac{\partial u(x,t)}{\partial x} - \lambda \alpha^2 u(x,t),
\end{align*}
\]

where \( a D_x^{\alpha,\lambda} u(x,t) \) and \( x D_b^{\alpha,\lambda} u(x,t) \) are the left and right Riemann–Liouville tempered fractional derivatives defined respectively as [1,2]

\[
\begin{align*}
a D_x^{\alpha,\lambda} u(x,t) &= \frac{e^{-\lambda x}}{\Gamma(2 - \alpha)} \frac{\partial^2}{\partial x^2} \int_a^x \frac{e^{\lambda \xi} u(\xi,t)}{(x-\xi)^{1-\alpha}} d\xi, \\
x D_b^{\alpha,\lambda} u(x,t) &= \frac{e^{\lambda x}}{\Gamma(2 - \alpha)} \frac{\partial^2}{\partial x^2} \int_x^b \frac{e^{-\lambda \xi} u(\xi,t)}{(\xi-x)^{1-\alpha}} d\xi,
\end{align*}
\]

where \( \Gamma(\cdot) \) is the Gamma function. Noticing that if \( \lambda = 0 \), they will reduce to the Riemann–Liouville fractional derivatives [4].

Tempered fractional diffusion equations (TFDEs) are exponentially tempered extension of fractional diffusion equations. In recent several decades, the TFDEs are widely used across various fields, such as statistical physics [1,5,6], finance [7–10] and geophysics [2,11–14]. Unfortunately, it is difficult to obtain the analytical solutions of TFDEs, or the obtained analytical solutions are less practical. Hence, numerical methods such as finite difference method [15,16] and finite element method [17] become essential approaches to solve TFDEs. There are limited works addressing the finite difference schemes for the TFDEs. Baemura and Meerschaert [2] provided finite difference and particle tracking methods for solving the TFDEs on a bounded interval. The stability and second-order accuracy of the resulted schemes are discussed. Cartea and del-Castillo-Negrete [18] proposed a general finite difference scheme to numerically solve a Black–Merton–Scholes model with tempered fractional derivatives. Marom and Momoniat [19] compared the numerical solutions of three fractional differential equations (FDEs) with tempered fractional derivatives that occur in finance. However, the stabilities of their proposed schemes are not proved. Recently, Li and Deng [20] derived a series of high order difference approximations (called tempered-WSGD operators) for handling tempered fractional calculus. They also used such operators to numerically solve the TFDE, and the stability and convergence of the obtained numerical schemes are proved. Chen and Deng [21] proposed highly accurate numerical algorithms.
for the time-space fractional diffusion equation with time fractional substantial derivative and space tempered fractional derivative. Their algorithms can keep the high accuracy when handling with nonhomogeneous boundary and initial values, see [21] for a thoroughly discussion.

Similar to the fractional derivatives, the tempered fractional derivatives are nonlocal. Thus the discretized systems for TFDEs usually accompany a full (or dense) coefficient matrix. Traditional methods (e.g., Gaussian elimination) to solve such systems need computational cost is of $O(N^3)$ and storage requirement is of $O(N^2)$, where $N$ is the number of space grid points. Fortunately, the coefficient matrix of the discretized system always holds a Toeplitz-like structure. It is well known that Toeplitz matrices possess great structural properties, and their matrix-vector multiplications can be computed in $O(N \log N)$ operations via fast Fourier transforms (FFTs) [22,23]. With this fact, the memory requirement and computational cost of Krylov subspace methods are $O(N)$ and $O(N \log N)$, respectively. However, the convergence rate of the Krylov subspace methods will be slow, if the coefficient matrix is ill-conditioned. To address this problem, Wang et al. [9] proposed a circulant preconditioned generalized minimal residual method (PGMRES) to solve the discretized linear system, whose computational cost is of $O(N \log N)$. Lei et al. [24] proposed fast solution algorithms for solving TFDEs in one-dimensional (1D) and two-dimensional (2D). In their article, for 1D case, a circulant preconditioned iterative method and a fast-direct method are developed, and the computational complexity of both methods are $O(N \log N)$ in each time step. For 2D case, such two methods are extended to fast solve their alternating direction implicit (ADI) scheme, and the complexity of both methods are $O(N^2 \log N)$ in each time step. For many other studies about Toeplitz-like systems, see [25–28] and the references therein.

Actually, all the aforementioned fast implementations for TFDEs are developed from the time-stepping schemes, which are not suitable for parallel computations. If the solutions of all the time levels are stacked in a vector, the all-at-once system is obtained and it is often suitable for parallel computations, refer to [29,30]. To the best of our knowledge, such the system arising from the FDEs or the partial differential equations have been studied by many researchers [31–38]. However, the all-at-once system arising from the (nonlinear) TFDEs is less studied. In this work, the main contribution is that a preconditioning strategy is designed to solve the nonlinear all-at-once system arising from Eq. (1.1).

The rest of this paper is organized as follows: in Sect. 2, the nonlinear and linearized implicit difference schemes are derived to approximate the NL-TFDE (1.1). Then, the nonlinear all-at-once system is obtained from the nonlinear one. The stabilities and convergences of such two schemes are analyzed in Sect. 3. A preconditioning technique is designed in Sect. 4 to accelerate solving such the all-at-once system. In Sect. 5, numerical examples are provided to illustrate the first-order spatial and temporal convergences of the two implicit schemes and show the performance of our preconditioning strategy for solving such the system. Concluding remarks are given in Sect. 6.

### 2 Two Implicit Schemes and All-at-Once System

In this section, both the nonlinear and linearized implicit schemes are proposed to approach Eq. (1.1). Then, the all-at-once system is obtained from the nonlinear one.
2.1 Two Implicit Schemes

To derive the proposed schemes, we first introduce the mesh $\tilde{\omega}_{h\tau} = \tilde{\omega}_h \times \tilde{\omega}_\tau$, where $\tilde{\omega}_h = \{x_i = a + ih, \ i = 0, 1, \ldots, N; \ x_0 = a, x_N = b\}$ and $\tilde{\omega}_\tau = \{t_j = j\tau, \ j = 0, 1, \ldots, M; \ t_M = T\}$. Let $u^j_i$ represents the numerical approximation of $u(x_i, t_j)$. Then the variants of the Riemann–Liouville tempered fractional derivatives defined in Eqs. (1.2)–(1.3) at $(x, t) = (x_i, t_j)$ can be approximated respectively as [20,39]:

$$\begin{align*}
aD_x^{\alpha,h} u(x, t)|_{(x,t)=(x_i,t_j)} &\approx \frac{1}{h^\alpha} \sum_{k=0}^{i+1} g_k^{(\alpha)} u^j_{i-k+1} - \alpha \lambda^{\alpha-1} \delta_x u^j_i; \\
bD_t^{\alpha,\tau} u(x, t)|_{(x,t)=(x_i,t_j)} &\approx \frac{1}{h^\alpha} \sum_{k=0}^{N-j+1} g_k^{(\alpha)} u^j_{i+k-1} + \alpha \lambda^{\alpha-1} \delta_x u^j_i,
\end{align*}$$

(2.1)

where

$$\delta_x u^j_i = \frac{u^j_i - u^j_{i-1}}{h} \quad \text{and} \quad g_k^{(\alpha)} = \begin{cases} \tilde{g}_1^{(\alpha)} - \delta \lambda \left(1 - e^{-\delta \lambda}\right)^\alpha, & k = 1, \\
\tilde{g}_k^{(\alpha)} e^{-\delta \lambda (k-1)}, & k \neq 1 \end{cases}$$

with $\tilde{g}_k^{(\alpha)} = (-1)^k \left(\frac{\alpha}{k}\right)$ $(k \geq 0)$.

As for the time discretization, the backward Euler method is used. Combining Eqs. (2.1) and (2.2), the following first-order nonlinear implicit Euler scheme (NL-IES) is obtained:

$$\begin{align*}
u^j_i - w_1 \left( d_{+,i} \sum_{k=0}^{i+1} g_k^{(\alpha)} u^j_{i-k+1} + d_{-,i} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} u^j_{i+k-1} \right) + w_2 \left( d_{+,i} - d_{-,i} \right) \left( u^j_i - u^j_{i-1} \right) \\
= u^{j-1}_i + \tau f^j_{u,i},
\end{align*}$$

(2.3)

in which $w_1 = \frac{\tau}{h^\alpha}$, $w_2 = \frac{\alpha \lambda^{\alpha-1} \tau}{h}$, $d_{+,i} = d_+(x_i)$ and $f^j_{u,i} = f(u(x_i, t_j), x_i, t_j)$. Applying the formula $f(u(x_i, t_j), x_i, t_j) = f(u(x_i, t_{j-1}), x_i, t_{j-1}) + O(\tau)$ to Eq. (2.3) and omitting the small term, it gets the first-order linearized implicit Euler scheme (L-IES):

$$\begin{align*}
u^j_i - w_1 \left( d_{+,i} \sum_{k=0}^{i+1} g_k^{(\alpha)} u^j_{i-k+1} + d_{-,i} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} u^j_{i+k-1} \right) + w_2 \left( d_{+,i} - d_{-,i} \right) \left( u^j_i - u^j_{i-1} \right) \\
= u^{j-1}_i + \tau f^j_{u,i}.
\end{align*}$$

(2.4)

The stabilities and first-order convergences of schemes (2.3)–(2.4) will be discussed in Sect. 3.

2.2 The All-at-Once System

Several auxiliary notations are introduced before deriving the all-at-once system: $I$, $0$ and $O$ represent the identity matrix, zero vector and zero matrix with suitable orders, respectively.

$$\begin{align*}
u^j &= \begin{bmatrix} u^j_1, u^j_2, \ldots, u^j_{N-1} \end{bmatrix}^T, \\
f^j_u &= \begin{bmatrix} f^j_{u,1}, f^j_{u,2}, \ldots, f^j_{u,N-1} \end{bmatrix}^T, \\
D_\pm &= \text{diag}(d_\pm, d_\pm, \ldots, d_\pm, N-1), \quad B = \text{tridiag}(-1, 1, 0),
\end{align*}$$

$\copyright$ Springer
\[ u = \begin{bmatrix} u_1^1 \\ u_2^1 \\ \vdots \\ u_M^1 \end{bmatrix}, \quad f(u) = \begin{bmatrix} f_u^1 \\ f_u^2 \\ \vdots \\ f_u^M \end{bmatrix}, \quad v = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad G = \begin{bmatrix} g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \cdots & 0 \\ g_2^{(\alpha)} & g_1^{(\alpha)} & g_0^{(\alpha)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ g_{N-2}^{(\alpha)} & \cdots & \cdots & g_1^{(\alpha)} & g_0^{(\alpha)} \\ g_{N-1}^{(\alpha)} & g_{N-2}^{(\alpha)} & \cdots & g_2^{(\alpha)} & g_1^{(\alpha)} \end{bmatrix}. \]

In this work, the all-at-once system is derived based on Eq. (2.3), which can be expressed as:

\[ Au = \tau f(u) + v, \tag{2.5} \]

where \( A = \text{blktridiag}(-I, A, 0) \) is a bi-diagonal block matrix with \( A = I - w_1 (D_+ G + D_- G^T) + w_2 (D_+ - D_-) B \). Obviously, \( A \) is a Toeplitz-like matrix and its storage requirement is \( O(N) \).

For this nonlinear all-at-once system, we prefer to utilize Newton’s method [40]. As we know, this method requires to solve the equation with Jacobian matrix at each iterative step, and the computation of solving these equations is the main time-consuming part of Newton’s method. Before applying Newton’s method to solve the system (2.5), two problems affected the convergence of Newton’s method need to be addressed:

1. How to find a good enough initial value?
2. How to solve the Jacobian equations efficiently?

In this work, a strategy is provided to address these two problems. For the first problem, the initial value of Newton’s method is constructed by interpolating the solution of L-IES (2.4) on the coarse mesh. Numerical results in Sect. 5 show that it is a good enough initial value. For the second problem, the Jacobian matrix of (2.5) is a bi-diagonal block matrix. More precisely, such a matrix is the sum of a diagonal block matrix and a bi-diagonal block matrix, whose blocks are Toeplitz-like matrices. Based on this special structure, the preconditioned Krylov subspace methods, e.g., the preconditioned biconjugate gradient stabilized (PBiCGSTAB) method [41], are employed to solve the Jacobian equations appeared in Newton’s method. The details will be discussed in Sect. 4.

3 Stabilities and Convergences of (2.3) and (2.4)

Before proposing the strategy to solve Eq. (2.5), the stabilities and convergences of NL-IES (2.3) and L-IES (2.4) need to be studied first. Let \( U_i^j \) be the approximation solution of \( u_i^j \) in Eqs. (2.3) or (2.4) and \( e_i^j = U_i^j - u_i^j \) \( (i = 1, \ldots, N-1; \ j = 1, \ldots, M) \) be the error satisfying equations

\[
e_i^j - w_1 \left( \sum_{k=0}^{i+1} g_k^{(\alpha)} e_{i-k+1}^j + d_{-i} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} e_{i+k-1}^j \right) + w_2 (d_{+i} - d_{-i}) \left(e_i^j - e_{i-1}^j \right) \\
= e_i^{j-1} + \tau \left(f_u U_i^j - f_u u_i^j \right) \text{ (for NL-IES)}
\]
or
\[
e^{i_j} - w_1 \left( d_{+,i} \sum_{k=0}^{i+1} g^{(\alpha)}_k e_{i-k+1}^j + d_{-,i} \sum_{k=0}^{N-i+1} g^{(\alpha)}_k e_{i+k-1}^j \right) + w_2 \left( d_{+,i} - d_{-,i} \right) \left( e_i^j - e_{i-1}^j \right) \\
= e^{j-1}_i + \tau \left( f^j_{i,i} - f^{j-1}_{i,i} \right) \text{ (for L-IES)},
\]
in which \( f^j_{U,i} = f(U^k_i, x_i, t_k) \). To prove the stabilities and convergences of (2.3) and (2.4), the following results given in [39,42] are required.

**Lemma 3.1** [39] The coefficients \( g^{(\alpha)}_k \), for \( k = 0, 1, \ldots \), satisfy:
\[
\begin{cases}
ge^{(\alpha)}_1 < 0, & g^{(\alpha)}_k > 0 \quad \text{(for } k \neq 1); \\
\sum_{k=0}^{\infty} g^{(\alpha)}_k = 0, & \sum_{k=0}^{j} g^{(\alpha)}_k < 0 \quad \text{(for } j \geq 1).
\end{cases}
\]

**Lemma 3.2** (Discrete Gronwall inequality [42]) Suppose that \( \tilde{f}_k \geq 0, \eta_k \geq 0 \) (\( k = 0, 1, \ldots \)), and
\[
\eta_{k+1} \leq \rho \eta_k + \tau \tilde{f}_k, \quad \rho = 1 + C_0 \tau, \quad \eta_0 = 0,
\]
where \( C_0 \geq 0 \) is a constant, then
\[
\eta_{k+1} \leq \exp(C_0 \tau k) \sum_{j=0}^{k} \tau \tilde{f}_j.
\]

### 3.1 The Stabilities of (2.3) and (2.4)

Denote \( E^j = \begin{bmatrix} e^j_1, e^j_2, \ldots, e^j_{N-1} \end{bmatrix}^T \), and assume that
\[
\left\| E^j \right\|_\infty = \left| e_{\ell_j} \right| = \max_{1 \leq \ell \leq N-1} \left| e^j_{\ell} \right| \quad (0 \leq j \leq M, 1 \leq \ell_j \leq N - 1).
\]

Then, we can obtain the following conclusion about the stability of L-IES (2.4).

**Theorem 3.1** Suppose \( d_{+,x} \geq d_{-,x} > 0 \), then the L-IES (2.4) is stable, and we have
\[
\left\| E^j \right\|_\infty \leq \exp(TL) \left\| E^0 \right\|_\infty,
\]
for \( j = 1, \ldots, M \).
Proof From Lemma 3.1, we obtain \( \sum_{k=0}^{\ell_j+1} g_k^{(\alpha)} < 0 \) and \( \sum_{k=0}^{N-\ell_j+1} g_k^{(\alpha)} < 0 \). Then

\[
|e_j^{(\ell_j)}| \leq 1 - w_1 \left( d_{+,\ell_j} \sum_{k=0}^{\ell_j+1} g_k^{(\alpha)} + d_{-,\ell_j} \sum_{k=0}^{N-\ell_j+1} g_k^{(\alpha)} \right) |e_j^{(\ell_j)}| + w_2 \left( d_{+,\ell_j} - d_{-,\ell_j} \right) |e_j^{(\ell_j)}|
\]

\[
\leq e_j^{(\ell_j)} - w_1 \left( d_{+,\ell_j} g_1^{(\alpha)} e_j^{(\ell_j)} \right) + w_2 \left( d_{+,\ell_j} - d_{-,\ell_j} \right) e_j^{(\ell_j)}
\]

\[
\leq e_j^{(\ell_j)} - w_1 \left( d_{+,\ell_j} g_1^{(\alpha)} e_j^{(\ell_j)} + d_{-,\ell_j} g_1^{(\alpha)} e_j^{(\ell_j)} \right) + w_2 \left( d_{+,\ell_j} - d_{-,\ell_j} \right) e_j^{(\ell_j)}
\]

\[
\leq e_j^{(\ell_j)} - w_1 \left( d_{+,\ell_j} g_1^{(\alpha)} e_j^{(\ell_j)} + d_{-,\ell_j} g_1^{(\alpha)} e_j^{(\ell_j)} \right) + w_2 \left( d_{+,\ell_j} - d_{-,\ell_j} \right) e_j^{(\ell_j)}
\]

The above inequality implies \( \|E^j\|_{\infty} \leq (1 + \tau L) \|E^{j-1}\|_{\infty} \). Then

\[
\|E^j\|_{\infty} \leq (1 + \tau L)^j \|E^0\|_{\infty} \leq \exp(TL) \|E^0\|_{\infty}.
\]

From the above proof, it can be find that if \( \tau L < 1 \), the following result is true.

Theorem 3.2 Suppose \( d_{+}(x) \geq d_{-}(x) > 0 \) and \( \tau L < 1 \), then the NL-IES (2.3) is stable, and it obtains

\[
\|E^k\|_{\infty} \leq C_1 \|E^0\|_{\infty}, \quad \text{for } k = 1, \ldots, M,
\]

where \( C_1 \) is a positive constant.
Proof Based on the proof of Theorem 3.1, it yields

$$(1 - \tau L) \| E_j \|_\infty \leq \| E^{j-1} \|_\infty.$$  

Summing up for $j$ from 1 to $k$ and using Lemma 3.4 in [15], it gets

$$\| E_k \|_\infty \leq \exp(\frac{T L}{1 - \tau L}) \| E_0 \|_\infty.$$  

Note that

$$\lim_{\tau \to 0} \exp(\frac{T L}{1 - \tau L}) = \exp(T L).$$  

Hence, there is a positive constant $C_1$ such that

$$\| E_k \|_\infty \leq C_1 \| E_0 \|_\infty,$$

thereby $\| E_k \|_\infty \leq C_1 \| E_0 \|_\infty$, for $k = 1, \ldots, M$. □

In Theorem 3.2, it is worth to notice that the assumption $\tau < \frac{1}{L}$ is independent of the spatial size $h$. Actually, if the time step size $\tau$ becomes smaller, the easier such assumption can be satisfied.

3.2 The Convergences of (2.3) and (2.4)

In this subsection, the convergences of (2.3) and (2.4) are studied. Let $\xi^j_i = u(x_i, t_j) - u_i^j$ satisfies

$$\xi^j_i - w_1 \left( d_{+,i} \sum_{k=0}^{i+1} g_k^{(\alpha)} \xi^j_{i-k+1} + d_{-,i} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} \xi^j_{i+k-1} \right) + w_2 \left( d_{+,i} - d_{-,i} \right) \left( \xi^j_i - \xi^j_{i-1} \right)$$

$$= \xi^j_{i-1} + \tau \left( f(u(x_i, t_j), x_i, t_j) - f^{j-1}_{u,i} \right) + R^j_i \quad \text{(for NL-IES)}$$

or

$$\xi^j_i - w_1 \left( d_{+,i} \sum_{k=0}^{i+1} g_k^{(\alpha)} \xi^j_{i-k+1} + d_{-,i} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} \xi^j_{i+k-1} \right) + w_2 \left( d_{+,i} - d_{-,i} \right) \left( \xi^j_i - \xi^j_{i-1} \right)$$

$$= \xi^j_{i-1} + \tau \left( f(u(x_i, t_{j-1}), x_i, t_{j-1}) - f^{j-1}_{u,i} \right) + R^j_i \quad \text{(for L-IES)},$$

where $|R^j_i| \leq C_2 \left( \tau^2 + \tau h + \tau h^\beta \right)$ ($C_2$ is a positive constant) and $\beta > 0$ would be determined by the spatial regularity of the solution of Eq. (1.1).

Remark As proved recently by Ervin et al. [43], the fractional diffusion equation on a bounded domain always admits boundary layers near the Dirichlet boundaries. The boundary singularity always degrades the accuracy of a numerical scheme. However, that is not the emphasis of this current manuscript and we shall pursue that in the coming work. In this work, a numerical example in Sect. 5 will be provided to investigate the regularity of the solution of Eq. (1.1) and it suggests that $\beta \leq 1$.  

Springer
Denote $\xi^j = [\xi_1^j, \xi_2^j, \ldots, \xi_{N-1}^j]^T$ and $\|\xi^j\|_\infty = |\xi_{\ell_j}^j| = \max_{1 \leq \ell \leq N-1} |\xi^j_\ell| \ (0 \leq j \leq M, 1 \leq \ell_j \leq N - 1)$. Similar to the proof of Theorem 3.1, the following theorem about the convergence of L-IES can be established.

**Theorem 3.3** Assume that $d_+(x) \geq d_-(x) > 0$ and $u(x, t)$ is the sufficiently smooth solution of (1.1). $u^j_t$ is the numerical solution of (2.4). Then there is a positive constant $C$ such that

$$\|\xi^j\|_\infty \leq C(\tau + h + h^\beta), \quad j = 1, 2, \ldots, M.$$

**Proof** Same technique in Theorem 3.1 is utilized, then it yields

$$\|\xi^j\|_\infty = |\xi_{\ell_j}^j| \leq |\xi_{\ell_j}^{j-1}| + \tau \left( f(u(x_{\ell_j}, t_{j-1}), x_{\ell_j}, t_{j-1}) - f_{u, \ell_j}^{j-1} \right) + R_{\ell_j}^j$$

$$\leq (1 + \tau L) |\xi_{\ell_j}^{j-1}| + C_2 \left( \tau^2 + \tau h + \tau h^\beta \right)$$

$$\leq (1 + \tau L) \|\xi^{j-1}\|_\infty + C_2 \left( \tau^2 + \tau h + \tau h^\beta \right).$$

Using Lemma 3.2, it gets

$$\|\xi^j\|_\infty \leq \exp(TL)TC_2 \left( \tau + h + h^\beta \right) \leq C \left( \tau + h + h^\beta \right).$$

On the other hand, the convergence of NL-IES (2.3) is given as follows.

**Theorem 3.4** Assume that $d_+(x) \geq d_-(x) > 0$, $\tau L < 1$, and $u(x, t)$ is the sufficiently smooth solution of (1.1). $u^j_t$ is the numerical solution of (2.3). Then

$$\|\xi^j\|_\infty \leq C(\tau + h + h^\beta), \quad j = 1, 2, \ldots, M,$$

where $C$ is a positive constant.

**Proof** According to Theorem 3.2, we have

$$(1 - \tau L) \|\xi^j\|_\infty \leq \|\xi^{j-1}\|_\infty + C_2 \left( \tau^2 + \tau h + \tau h^\beta \right).$$

Similarly, it arrives

$$\|\xi^j\|_\infty \leq \frac{\exp(TL)}{1 - \tau L} TC_2 \left( \tau + h + h^\beta \right) \leq C \left( \tau + h + h^\beta \right), \quad j = 1, 2, \ldots, M,$$

in which $\frac{\exp(TL)}{1 - \tau L} \leq C_1$ is employed.

It is interesting to note that if $\xi_t^j$ represents the error between NL-IES (2.3) and L-IES (2.4), then it also satisfies $\|\xi_t^j\|_\infty \leq C(\tau + h + h^\beta) \ (j = 1, 2, \ldots, M).$ This can be proved easily through the condition $f_{u, i}^{j,i} = f_{u, i}^{j,i-1} + O(\tau)$. In the next section, fast implementations are designed to solve (2.5).

---

1 Numerical results can be found in our arXiv preprint https://arxiv.org/abs/1901.00635.
4 The Preconditioned Iterative Method

Since the Jacobian matrix of (2.5) can be treated as the sum of a diagonal block matrix and a block bi-diagonal matrix with Toeplitz-like blocks, then its matrix-vector multiplication can be done by FFT in $O(MN \log MN)$ operations. Such a technique truly reduces the computational cost of a Krylov subspace method, e.g., the biconjugate gradient stabilized (BiCGSTAB) method, but the convergence rate of this method is slow when the Jacobi matrix becomes very ill-conditioned. In order to accelerate the convergence of the Krylov subspace method, a preconditioner $P_\ell = \text{blktridiag}(-I, A_\ell, 0)$ ($\ell > 2$) is proposed and analyzed in this section, in which

$$A_\ell = I - w_1 (D_+G_\ell + D_-G_\ell^T) + w_2 (D_+ - D_-) B \text{ with } G_\ell = \begin{bmatrix} g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \cdots & 0 \\ g_2^{(\alpha)} & g_1^{(\alpha)} & g_0^{(\alpha)} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_\ell^{(\alpha)} & \cdots & g_2^{(\alpha)} & \cdots & g_1^{(\alpha)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ g_\ell^{(\alpha)} & \cdots & g_2^{(\alpha)} & \cdots & g_1^{(\alpha)} \end{bmatrix}.$$  

Noticing the properties of $g_k^{(\alpha)}$ given in Lemma 3.1, the following result about $P_\ell$ is true.

**Theorem 4.1** The preconditioner $P_\ell$ is a nonsingular matrix.

**Proof** Obviously, we only need to proof the nonsingularity of $A_\ell$. Let

$$H = \frac{A_\ell + A_\ell^T}{2} = I - \frac{\omega_1}{2} (D_+ + D_-) \left( G_\ell + G_\ell^T \right) + \frac{\omega_2}{2} (D_+ - D_-) \left( B + B^T \right)$$

and $\theta$ be the arbitrary eigenvalue of $H$. According to Lemma 3.1 and Gershgorin circle theorem [44], it arrives at

$$|\theta - \left[ 1 - \omega_1 (d_{+,i} + d_{-,i}) g_1^{(\alpha)} + \omega_2 (d_{+,i} - d_{-,i}) \right]|$$

$$\leq r_i = \left| -\omega_1 (d_{+,i} + d_{-,i}) \left( g_0^{(\alpha)} + g_2^{(\alpha)} \right) - \omega_2 (d_{+,i} - d_{-,i}) \right|$$

$$+ \sum_{k=3}^\ell \omega_1 (d_{+,i} + d_{-,i}) g_k^{(\alpha)}$$

$$\leq \omega_1 (d_{+,i} + d_{-,i}) \sum_{k=0, k \neq 1}^\ell g_k^{(\alpha)} + \omega_2 (d_{+,i} - d_{-,i}) < -\omega_1 (d_{+,i} + d_{-,i}) g_1^{(\alpha)}$$

$$+ \omega_2 (d_{+,i} - d_{-,i}).$$

This implies that all eigenvalues of $H$ are larger than 1. Then, the desired result is achieved.

Moreover, the following estimation shows that the preconditioner $P_\ell$ can be a feasible approximation for the matrix $A$.

**Theorem 4.2** Suppose $\omega_1 \leq \nu$ (a positive constant). For $\ell > 2$, $\|P_\ell - A\|_\infty = O(\ell^{-\alpha} e^{-(\ell-1)h\lambda})$. 

\[\square\] Springer
Algorithm 1 Solve $u$ from Eq. (2.5)

1: Given maximum iterative step maxit, tolerance tolout and initial vector $u^{(0)}$, which is obtained by interpolating the solution of L-IES (2.4) on the coarse grid (here $M = N = 16$)

2: for $k = 1, \ldots, \text{maxit}$ do

3: Solve $J_k z = -f(u^{(k)})$ via PBiCGSTAB method with preconditioner $P^\ell$ ($\ell = 8$ is chosen experimentally to balance the number of iterations and CPU time)

4: $u^{(k+1)} = u^{(k)} + z$

5: if $\|z\|_2 \leq \text{tolout}$ then

6: $u = u^{(k+1)}$

7: break

8: end if

9: end for

Proof According to the definition of $g^{(\alpha)}_\ell$ (see Sect. 2.1), it follows from $\tilde{g}^{(\alpha)}_\ell = O(\ell^{-(\alpha+1)})$ [45] that $g^{(\alpha)}_\ell = O(\ell^{-(\alpha+1)}e^{-(\ell-1)h\lambda})$. Then, combine with Lemma 3.1, we obtain

$$\|P^\ell - A\|_\infty \leq \nu \|D_+ (G - G^\ell) + D_-(G - G^\ell)^T\|_\infty = O(\ell^{-\alpha}e^{-(\ell-1)h\lambda}),$$

since $\|G - G^\ell\|_\infty = O(\ell^{-\alpha}e^{-(\ell-1)h\lambda})$. \hfill $\Box$

Moreover, Theorem 4.2 implies that the difference between $A$ and $P^\ell$ can become small, if $\ell$ and/or $\lambda$ become large enough. Meanwhile, it also suggests that $P^\ell$ can be regarded as a suitable preconditioner for the Jacobian systems in the Newton iteration method –cf. Line 3 of Algorithm 1. Unfortunately, it is difficult to theoretically investigate the eigenvalue distributions of the preconditioned Jacobian matrix, but we still can work out the figures to illustrate the clustering eigenvalue distributions of several specified preconditioned matrices in the next section. For convenience, let $u^{(k+1)}$ be the approximation of $u$ obtained in the $k$th Newton iterative step, the Jacobian matrix in the $k$th Newton iterative step is denoted as $J^k$. With these auxiliary notations, the preconditioned Newton’s method can be summarized in Algorithm 1. In fact, this algorithm can be viewed as a simple two-grid method, readers are suggested to refer to [46–48].

5 Numerical Examples

The first two numerical experiments presented in this section have a two-fold objective. On the one hand, they illustrate that the convergence orders of our two implicit schemes (2.3)–(2.4) are 1. On the other hand, they show the performance of the preconditioner $P^\ell$ proposed in Sect. 4. Example 3 is provided to investigate the regularity of the solution of Eq. (1.1), which is mentioned in Sect. 3.2. In Algorithm 1, for generating the initial guess $u^{(0)}$, the MATLAB build-in function “interp2” is employed in this work. The maxit and tolout in Algorithm 1 are fixed as 100 and $10^{-12}$, respectively. For the PBiCGSTAB method (or the BiCGSTAB method), it terminates if the relative residual error satisfies $\frac{\|r^k\|_2}{\|r^0\|_2} \leq 10^{-6}$ or the iteration number is more than 1000, where $r^k$ is the residual vector of the linear system after $k$ iterations, and the initial guess of the PBiCGSTAB method (or the BiCGSTAB method) is chosen as the zero vector. “P” represents that our proposed preconditioned iterative method in Sect. 4 is utilized to solve (2.5). “BS” (or “I”) means that the PBiCGSTAB method in Step 3 of Algorithm 1 is replaced by the MATLAB’s backslash method (or the BiCGSTAB
| α  | M       | L-IES (2, 4) | NL-IES (2, 3) | L-IES (2, 4) | NL-IES (2, 3) | L-IES (2, 4) | NL-IES (2, 3) | L-IES (2, 4) | NL-IES (2, 3) |
|----|---------|-------------|---------------|-------------|---------------|-------------|---------------|-------------|---------------|
|    |         | Err(τ, h)   | Order | Err(τ, h)   | Order | Err(τ, h)   | Order | Err(τ, h)   | Order |
| 1.1| 64      | 1.7040E−02  | –     | 1.6411E−02  | –     | 3.4870E−03  | –     | 3.3119E−03  | –     | 1.9582E−03  | –     | 1.8739E−03  | –     | 1.4530E−03  | –     |
| 128| 8.1919E−03 | 1.0567     |       | 7.8615E−03  | 1.0618 | 1.8406E−03  | 0.9218 | 1.7891E−03  | 0.8884 | 1.1232E−03  | 0.8019 | 1.0962E−03  | 0.7735 | 8.4050E−04  | 0.7897 | 8.2106E−04  | 0.7219 |
| 256| 3.5622E−03 | 1.2014     |       | 3.4131E−03  | 1.2037 | 8.8454E−04  | 1.0572 | 8.5870E−04  | 1.0590 | 5.6066E−04  | 1.0024 | 5.5332E−04  | 0.9863 | 4.3146E−04  | 0.9530 | 4.2879E−04  | 0.9372 |
| 512| 1.1960E−03 | 1.5746     |       | 1.1451E−03  | 1.5756 | 3.2057E−04  | 1.4643 | 3.1539E−04  | 1.4450 | 2.1087E−04  | 1.4108 | 2.0817E−04  | 1.4104 | 1.6352E−04  | 1.4088 | 1.6155E−04  | 1.4083 |
| 1.5| 64      | 1.5958E−02  | –     | 1.5646E−02  | –     | 7.3474E−03  | –     | 7.2273E−03  | –     | 3.0624E−03  | –     | 3.0226E−03  | –     | 1.8709E−03  | –     | 1.8409E−03  | –     |
| 128| 9.8784E−03 | 0.6919     |       | 9.7398E−03  | 0.6838 | 5.4494E−03  | 0.4311 | 5.3896E−03  | 0.4233 | 2.6863E−03  | 0.1890 | 2.6619E−03  | 0.1833 | 1.7165E−03  | 0.1243 | 1.7015E−03  | 0.1136 |
| 256| 5.3532E−03 | 0.8839     |       | 5.3054E−03  | 0.8764 | 3.3987E−03  | 0.6811 | 3.3755E−03  | 0.6751 | 1.9188E−03  | 0.4854 | 1.9081E−03  | 0.4803 | 1.3049E−03  | 0.3955 | 1.2980E−03  | 0.3905 |
| 512| 2.1501E−03 | 1.3160     |       | 2.1389E−03  | 1.3106 | 1.5152E−03  | 1.1655 | 1.5091E−03  | 1.1614 | 9.5571E−04  | 1.0056 | 9.5258E−04  | 1.0022 | 6.8923E−04  | 0.9209 | 6.8716E−04  | 0.9176 |
| 1.9| 64      | 1.3778E−02  | –     | 1.3882E−02  | –     | 1.3211E−02  | –     | 1.3292E−02  | –     | 1.1913E−02  | –     | 1.1978E−02  | –     | 1.1188E−02  | –     | 1.1237E−02  | –     |
| 128| 6.5646E−03 | 1.0696     |       | 6.5994E−03  | 1.0728 | 6.2890E−03  | 1.0708 | 6.3190E−03  | 1.0728 | 5.6615E−03  | 1.0733 | 5.6826E−03  | 1.0758 | 5.3106E−03  | 1.0750 | 5.3272E−03  | 1.0768 |
| 256| 2.8432E−03 | 1.2072     |       | 2.8560E−03  | 1.2083 | 2.7226E−03  | 1.2078 | 2.7334E−03  | 1.2090 | 2.4487E−03  | 1.2092 | 2.4562E−03  | 1.2101 | 2.2959E−03  | 1.2098 | 2.3014E−03  | 1.2109 |
| 512| 9.5277E−04 | 1.5773     |       | 9.5665E−04  | 1.5779 | 9.1215E−04  | 1.5776 | 9.1541E−04  | 1.5782 | 8.2000E−04  | 1.5783 | 8.2226E−04  | 1.5788 | 7.6866E−04  | 1.5786 | 7.7025E−04  | 1.5791 |
| 1.99| 64     | 1.6317E−02  | –     | 1.6444E−02  | –     | 1.6251E−02  | –     | 1.6374E−02  | –     | 1.6099E−02  | –     | 1.6220E−02  | –     | 1.6106E−02  | –     | 1.6141E−02  | –     |
| 128| 7.7962E−03 | 1.0655     |       | 7.8467E−03  | 1.0674 | 7.7646E−03  | 1.0655 | 7.8136E−03  | 1.0673 | 7.6921E−03  | 1.0655 | 7.7404E−03  | 1.0673 | 7.6525E−03  | 1.0655 | 7.7013E−03  | 1.0676 |
| 256| 3.3820E−03 | 1.2049     |       | 3.4010E−03  | 1.2061 | 3.3679E−03  | 1.2051 | 3.3867E−03  | 1.2061 | 3.3360E−03  | 1.2053 | 3.3547E−03  | 1.2062 | 3.3189E−03  | 1.2052 | 3.3374E−03  | 1.2064 |
| 512| 1.1342E−03 | 1.5762     |       | 1.1401E−03  | 1.5768 | 1.1294E−03  | 1.5763 | 1.1353E−03  | 1.5768 | 1.1187E−03  | 1.5756 | 1.1245E−03  | 1.5769 | 1.1129E−03  | 1.5764 | 1.1187E−03  | 1.5769 |

**Table 1** The maximum norm errors and convergence orders for Example 1 with $h = 2^{-10}$
Table 2  The maximum norm errors and convergence orders for Example 1 with $\tau = h$

| $\alpha$ | $\lambda$ | $\lambda = 0$ | $\lambda = 1$ |
|----------|-----------|---------------|---------------|
|          | $N$       | L-IES (2.4)   | NL-IES (2.3)  | L-IES (2.4)   | NL-IES (2.3)  |
|          |           | $\text{Err}(\tau, h)$ | $\text{Order}^2$ | $\text{Err}(\tau, h)$ | $\text{Order}^2$ | $\text{Err}(\tau, h)$ | $\text{Order}^2$ | $\text{Err}(\tau, h)$ | $\text{Order}^2$ |
| 1.1      | 64        | 1.5584E−01    | 1.5573E−01    | 1.5167E−01    | 1.5155E−01    | 1.5155E−01    | 1.5155E−01    | 1.5155E−01    | 1.5155E−01    |
|          | 128       | 1.3274E−01    | 0.2315        | 1.3270E−01    | 0.2309        | 1.0372E−01    | 0.5482        | 1.0369E−01    | 0.5475        |
|          | 256       | 1.0866E−01    | 0.2888        | 1.0865E−01    | 0.2885        | 6.3971E−02    | 0.6972        | 6.3968E−02    | 0.6969        |
|          | 512       | 7.0002E−02    | 0.6344        | 6.9998E−02    | 0.6343        | 2.9700E−02    | 1.1070        | 2.9700E−02    | 1.1069        |
| 1.5      | 64        | 2.5949E−02    | 2.5752E−02    | 2.8572E−02    | 2.9235E−02    | 2.9235E−02    | 2.9235E−02    | 2.9235E−02    | 2.9235E−02    |
|          | 128       | 1.4949E−02    | 0.7956        | 1.4903E−02    | 0.7891        | 1.3870E−02    | 1.0426        | 1.4127E−02    | 1.0492        |
|          | 256       | 8.3971E−03    | 0.8321        | 8.3863E−03    | 0.8295        | 6.4427E−03    | 1.1062        | 6.4410E−03    | 1.1331        |
|          | 512       | 3.6657E−03    | 1.1958        | 3.6634E−03    | 1.1949        | 2.6046E−03    | 1.3066        | 2.6042E−03    | 1.3064        |
| 1.9      | 64        | 1.2654E−02    | 1.2833E−02    | 8.5981E−03    | 8.9106E−03    | 8.9106E−03    | 8.9106E−03    | 8.9106E−03    | 8.9106E−03    |
|          | 128       | 6.0078E−03    | 1.0747        | 6.0826E−03    | 1.0771        | 4.0909E−03    | 1.0716        | 4.2426E−03    | 1.0706        |
|          | 256       | 2.5959E−03    | 1.2106        | 2.6269E−03    | 1.2113        | 1.7690E−03    | 1.2095        | 1.8359E−03    | 1.2085        |
|          | 512       | 8.6860E−04    | 1.5795        | 8.7876E−04    | 1.5798        | 5.9203E−04    | 1.5792        | 6.1474E−04    | 1.5784        |
| 1.99     | 64        | 1.6292E−02    | 1.6426E−02    | 1.4250E−02    | 1.4383E−02    | 1.4250E−02    | 1.4383E−02    | 1.4383E−02    | 1.4383E−02    |
|          | 128       | 7.7669E−03    | 1.0688        | 7.8202E−03    | 1.0707        | 6.8188E−03    | 1.0634        | 6.8761E−03    | 1.0647        |
|          | 256       | 3.3653E−03    | 1.2066        | 3.3857E−03    | 1.2078        | 2.9598E−03    | 1.2040        | 2.9826E−03    | 1.2050        |
|          | 512       | 1.1280E−03    | 1.5770        | 1.1343E−03    | 1.5777        | 9.9293E−04    | 1.5757        | 1.0004E−03    | 1.5760        |
| $\alpha$ | $N$ | $\lambda = 5$ | $\lambda = 10$ |
|------|-----|----------------|----------------|
|      |     | L-IES (2.4)   | NL-IES (2.3)   | L-IES (2.4) | NL-IES (2.3) |
|      |     | $\text{Err}(\tau, h)$ | $\text{Order}_2$ | $\text{Err}(\tau, h)$ | $\text{Order}_2$ | $\text{Err}(\tau, h)$ | $\text{Order}_2$ |
| 1.1  | 64  | 1.1031E−01  | −          | 1.1001E−01  | −          | 2.8818E−01  | −          | 2.8659E−01  | −          |
|     | 128 | 6.9002E−02 | 0.6769     | 6.8934E−02 | 0.6743     | 2.1162E−01 | 0.4455     | 2.1100E−01 | 0.4417     |
|     | 256 | 3.4589E−02 | 0.9963     | 3.4590E−02 | 0.9949     | 1.1957E−01 | 0.8236     | 1.1944E−01 | 0.8210     |
|     | 512 | 1.4117E−02 | 1.2929     | 1.4120E−02 | 1.2926     | 4.5635E−02 | 1.3896     | 4.5637E−02 | 1.3880     |
| 1.5  | 64  | 3.3391E−02 | −          | 3.2306E−02 | −          | 1.5219E−01 | −          | 1.5281E−01 | −          |
|     | 128 | 1.5028E−02 | 1.1518     | 1.4496E−02 | 1.1561     | 7.6948E−02 | 0.9839     | 7.7233E−02 | 0.9844     |
|     | 256 | 6.3534E−03 | 1.2421     | 6.1213E−03 | 1.2437     | 3.5850E−02 | 1.1019     | 3.5988E−02 | 1.1017     |
|     | 512 | 2.1102E−03 | 1.5901     | 2.0322E−03 | 1.5908     | 1.2908E−02 | 1.4737     | 1.2952E−02 | 1.4743     |
| 1.9  | 64  | 9.8016E−03 | −          | 1.0259E−02 | −          | 2.9712E−02 | −          | 3.0369E−02 | −          |
|     | 128 | 4.7515E−03 | 1.0446     | 4.9612E−03 | 1.0481     | 1.2665E−02 | 1.2302     | 1.2947E−02 | 1.2300     |
|     | 256 | 2.0794E−03 | 1.1922     | 2.1659E−03 | 1.1957     | 5.1522E−03 | 1.2976     | 5.2688E−03 | 1.2971     |
|     | 512 | 7.0070E−04 | 1.5693     | 7.2863E−04 | 1.5717     | 1.6714E−03 | 1.6241     | 1.7097E−03 | 1.6237     |
| 1.99 | 64  | 1.0829E−02 | −          | 1.1600E−02 | −          | 2.0546E−02 | −          | 2.1150E−02 | −          |
|     | 128 | 5.3614E−03 | 1.0142     | 5.6678E−03 | 1.0333     | 8.7825E−03 | 1.2262     | 9.0683E−03 | 1.2218     |
|     | 256 | 2.3803E−03 | 1.1715     | 2.4898E−03 | 1.1868     | 3.6215E−03 | 1.2780     | 3.7386E−03 | 1.2783     |
|     | 512 | 8.0956E−04 | 1.5559     | 8.4157E−04 | 1.5649     | 1.1868E−03 | 1.6095     | 1.2249E−03 | 1.6098     |
method). Some other notations, which will appear in later, are given:

$$\text{Err}(\tau, h) = \max_{0 \leq j \leq M} \| \xi^j \|_\infty, \quad \text{Order } 1 = \log_2 \frac{\text{Err}(2\tau, h)}{\text{Err}(\tau, h)}, \quad \text{Order } 2 = \log_2 \frac{\text{Err}(\tau, 2h)}{\text{Err}(\tau, h)}.$$ 

“Iter1” represents the number of iterations required by Algorithm 1. “Iter2” denotes the average number of iterations required by the PBiCGSTAB method (or the BiCGSTAB method) in Algorithm 1, i.e.,

$$\text{Iter2} = \sum_{m=1}^{\text{Iter1}} \frac{\text{Iter2}(m)}{\text{Iter1}},$$

where Iter2(m) is the number of iterations required by such the method in the mth iterative step of Algorithm 1. “Time” denotes the total CPU time in seconds for solving the system (2.5). “‡” means the maximum iterative step is reached but not convergence, and “†” means out of memory.

All experiments were performed on a Windows 10 (64 bit) PC-Intel(R) Core(TM) i7-8700k CPU 3.70 GHz, 16 GB of RAM using MATLAB R2016a.

Example 1 Consider Eq. (1.1) with \( T = 1 \), the initial value \( u(x, 0) = (1 + x)(1 - x) \) \( (x \in [-1, 1]) \), the nonlinear source term \( f(u(x, t), x, t) = u(x, t) - u^3(x, t) \) and the continuous coefficients \( d_+(x) = 1.5 \exp(-x) \) and \( d_-(x) = \exp(x) \). Obviously, it is hard to obtain the exact solution of Eq. (1.1). Thus, the numerical solution computed from the finer mesh \( (M = N = 1024) \) is treated as the exact solution.

Tables 1 and 2 show that the convergence orders of the NL-IES (2.3) and L-IES (2.4) for different \( \alpha \) and \( \lambda \) can indeed reach 1 in both time and space. Figure 1 is plotted to show the numerical solutions with different \( \lambda \), where \( \alpha = 1.5 \) and \( M = N = 129 \). It can be seen that the diffusion process becomes more gentle as \( \lambda \) increases, refer to [39] for a related discussion. In Table 3, the CPU time and number of iterations of the methods BS, \( \mathcal{I} \) and \( \mathcal{P} \) are reported. The method \( \mathcal{I} \) in many cases needs more than 1000 iterative steps to obtain the solutions of \( J^k z = -f(u^{(k)}) \), which implies that the Jacobian matrices are very ill-conditioned. For the method \( \mathcal{P} \), the Iter2 is greatly reduced compared with the method \( \mathcal{I} \). This means that our preconditioner \( P_\ell \) is efficient for solving the Jacobian equations in Algorithm 1, but the Iter2 grows slightly fast in several cases such as \( (\alpha, \lambda) = (1.9, 0) \). Moreover, for fixed \( \alpha \) and \( N \), the Iter2 of the method \( \mathcal{P} \) in Table 3 becomes smaller as \( \lambda \) increases. The reason may be revealed in Theorem 4.2. On the other hand, as seen from Table 3, the total CPU time of the

Fig. 1 The numerical solutions with \( \alpha = 1.5 \) for Example 1
### Table 3 Results of different methods when $M = N$ for Example 1

| $(\alpha, \lambda)$ | $N$ | BS | $I$ | $P$ |
|---------------------|-----|-----|-----|-----|
|                     |     | Iter1 | Time | (Iter1, Iter2) | Time | (Iter1, Iter2) | Time |
| (1.1, 0)            | 129 | 5.0   | 0.650 | (5.0, 192.8) | 5.099 | (5.0, 3.2) | 0.150 |
|                     | 257 | 5.0   | 4.022 | (5.0, 427.2) | 41.457 | (5.0, 22.8) | 2.167 |
|                     | 513 | 5.0   | 32.972 | (5.0, 733.8) | 270.725 | (5.0, 7.6) | 3.492 |
|                     | 1025 | †   | †   | ‡   | ‡   | (5.0, 12.0) | 18.897 |
| (1.5, 0)            | 129 | 4.0   | 0.439 | (4.0, 499.8) | 9.910 | (4.0, 11.3) | 0.335 |
|                     | 257 | 4.0   | 3.293 | (5.0, 701.0) | 66.740 | (4.0, 21.3) | 2.059 |
|                     | 513 | 4.0   | 27.968 | (7.0, 975.7) | 500.910 | (4.0, 58.3) | 19.980 |
|                     | 1025 | †   | †   | ‡   | ‡   | (4.0, 150.3) | 181.646 |
| (1.9, 0)            | 129 | 4.0   | 0.445 | (5.0, 745.8) | 19.111 | (4.0, 8.8) | 0.290 |
|                     | 257 | 4.0   | 3.312 | ‡   | ‡   | (4.0, 21.3) | 2.059 |
|                     | 513 | 4.0   | 28.241 | (7.0, 975.7) | 500.910 | (4.0, 58.3) | 19.980 |
|                     | 1025 | †   | †   | ‡   | ‡   | (4.0, 198.5) | 239.630 |
| (1.1, 5)            | 129 | 5.0   | 0.658 | (5.0, 253.8) | 7.205 | (5.0, 3.4) | 0.143 |
|                     | 257 | 5.0   | 3.985 | (5.0, 540.4) | 52.199 | (5.0, 4.2) | 0.558 |
|                     | 513 | 5.0   | 36.070 | ‡   | ‡   | (5.0, 7.6) | 3.518 |
|                     | 1025 | †   | †   | ‡   | ‡   | (5.0, 121.0) | 183.051 |
| (1.5, 5)            | 129 | 5.0   | 0.531 | (5.0, 510.6) | 13.891 | (5.0, 5.8) | 0.225 |
|                     | 257 | 5.0   | 3.983 | (9.0, 579.8) | 99.771 | (5.0, 14.6) | 1.792 |
|                     | 513 | 5.0   | 36.184 | (13.0, 752.5) | 715.394 | (5.0, 41.6) | 17.941 |
|                     | 1025 | †   | †   | ‡   | ‡   | (4.0, 180.0) | 17.517 |
| (1.9, 5)            | 129 | 4.0   | 0.448 | (4.0, 894.0) | 18.977 | (4.0, 5.8) | 0.140 |
|                     | 257 | 4.0   | 3.351 | ‡   | ‡   | (4.0, 15.5) | 1.509 |
|                     | 513 | 4.0   | 30.050 | ‡   | ‡   | (4.0, 51.3) | 17.517 |
|                     | 1025 | †   | †   | ‡   | ‡   | (4.0, 180.0) | 17.517 |
| (1.1, 10)           | 129 | 5.0   | 0.527 | (5.0, 282.0) | 7.527 | (5.0, 3.6) | 0.152 |
|                     | 257 | 5.0   | 4.038 | (5.0, 594.0) | 55.898 | (5.0, 3.8) | 0.527 |
|                     | 513 | 5.0   | 36.066 | (14.0, 381.5) | 391.810 | (5.0, 7.4) | 3.416 |
|                     | 1025 | †   | †   | ‡   | ‡   | (5.0, 14.4) | 22.481 |
| (1.5, 10)           | 129 | 5.0   | 0.541 | (5.0, 521.0) | 14.116 | (5.0, 3.4) | 0.144 |
|                     | 257 | 5.0   | 3.992 | (5.0, 858.2) | 81.628 | (5.0, 10.6) | 1.283 |
|                     | 513 | 5.0   | 36.087 | ‡   | ‡   | (5.0, 33.2) | 14.337 |
|                     | 1025 | †   | †   | ‡   | ‡   | (5.0, 110.2) | 166.950 |
| (1.9, 10)           | 129 | 4.0   | 0.449 | (5.0, 873.0) | 23.772 | (4.0, 4.3) | 0.140 |
|                     | 257 | 4.0   | 3.358 | ‡   | ‡   | (4.0, 13.5) | 1.289 |
|                     | 513 | 4.0   | 30.038 | ‡   | ‡   | (4.0, 45.8) | 15.640 |
|                     | 1025 | †   | †   | ‡   | ‡   | (4.0, 181.8) | 220.122 |

Method $P$ is the smallest one among them. The eigenvalues of the initial Jacobian matrix $J^0$ and its preconditioned matrix $P_\ell^{-1}J^0$ are drawn in Fig. 2. As can be seen, the eigenvalues of $P_\ell^{-1}J^0$ are clustered around 1.
Fig. 2 Spectra of $J_0$ and $P^{-1}_J J_0$, when $\alpha = 1.5$, $M = N = 65$ in Example 1. Top row: $\lambda = 0$; Bottom row: $\lambda = 5$

Example 2 In this example, we consider Eq. (1.1) with $T = 1$, the initial value $u(x, 0) = \frac{4 \exp(10x)}{(\exp(10x)+1)\pi} (x \in [-1, 1])$, the nonlinear source term $f(u(x, t), x, t) = -u(x, t) (1 - u(x, t))$ and the discontinuous coefficients

$$d_+(x) = \begin{cases} 1.5 \exp(-x), & -1 \leq x < 0, \\ 2 \operatorname{sech}(x), & 0 \leq x \leq 1 \end{cases}$$

$$d_-(x) = \begin{cases} \exp(x), & -1 \leq x < 0, \\ 0.1 + \operatorname{sech}(-x), & 0 \leq x \leq 1 \end{cases}$$

Similar to Example 1, we regard the numerical solution on the finer mesh ($M = N = 1024$) as our exact solution.

It can be seen from Tables 4 and 5 that the convergence orders of the two proposed implicit schemes are 1 in both time and space for the discontinuous coefficients. Figure 3 presents the numerical solutions ($\alpha = 1.1$ and $M = N = 129$) with different $\lambda$. From this figure, for a fixed $\alpha$, the diffusion becomes more gentle as $\lambda$ increases, see [39] for a related discussion. The performance of the method $P$ shown in Table 6 is the best one among them in aspects of CPU time and the number of iterations. The Iter1 of the methods BS, $T$ and $P$ is small, which indicates that the initial vector $u^{(0)}$ is a good enough initial value. As for the Iter2, the method $P$ requires less iterative steps than the method $T$ under the same termination condition. This illustrates that our proposed preconditioner $P$ is efficient and can accelerate solving the Jacobian equations in Algorithm 1. Furthermore, Fig. 4 displays the eigenvalues of $J_0$ and $P^{-1}_J J_0$.

Example 3 Since it is hard to theoretically analyze the regularity of the solution of Eq. (1.1), an example is provided here to investigate it. Considering (1.1) with $T = 1$, $u(x, 0) = x^{\frac{5}{2}} (1 - x)^{\frac{5}{2}} (0 \leq x \leq 1)$, $d_+(x) = \cos(\frac{\pi}{24}(x + 4))$, $d_-(x) = \sin(\frac{\pi}{24}(x + 4))$ and $f(u(x, t), x, t) = 2 \sin(u(x, t))$. Notice that no exact solution is available in such the case.

The asymptotic behavior of the solution near $x = 0$ and $x = 1$ is investigated. Fixing $M = N = 100$, The log-log plot Fig. 5 show the behaviors of $\delta_x u_j^t$ at three temporal points with different $\alpha$ and $\lambda$. After observation, it suggests that the solution possesses weak singularity like $u_x = O(x^{\alpha-2})$ near the boundary values. The regularity of the solutions...
Table 4  The maximum norm errors and convergence orders for Example 2 with $h = 2^{-10}$

| $\alpha$ | $M$ | $\lambda = 0$ |          | $\lambda = 1$ |          |
|----------|-----|---------------|----------|---------------|----------|
|          |     | L-IES (2.4)   | Order 1 | L-IES (2.4)   | Order 1 |
|          |     | $\text{Err}(\tau, h)$ |          | $\text{Err}(\tau, h)$ |          |
| 1.1      | 64  | 3.2431E−02    | –        | 3.3730E−02    | –        |
|          | 128 | 1.6340E−02    | 0.9890  | 1.6918E−02    | 0.9955  |
|          | 256 | 7.3028E−03    | 1.1619  | 7.5422E−03    | 1.1655  |
|          | 512 | 2.4886E−03    | 1.5531  | 2.5666E−03    | 1.5551  |
| 1.5      | 64  | 5.2358E−02    | –        | 5.0868E−02    | –        |
|          | 128 | 2.7642E−02    | 0.9215  | 2.6909E−02    | 0.9187  |
|          | 256 | 1.2690E−02    | 1.1232  | 1.2368E−02    | 1.1215  |
|          | 512 | 4.4268E−03    | 1.5194  | 4.3191E−03    | 1.5178  |
| 1.9      | 64  | 8.5313E−02    | –        | 8.3849E−02    | –        |
|          | 128 | 5.8082E−02    | 0.5547  | 5.7382E−02    | 0.5472  |
|          | 256 | 2.9567E−02    | 0.9741  | 2.9299E−02    | 0.9697  |
|          | 512 | 1.1116E−02    | 1.4114  | 1.1021E−02    | 1.4106  |
| 1.99     | 64  | 8.7279E−02    | –        | 8.5852E−02    | –        |
|          | 128 | 6.2942E−02    | 0.4716  | 6.2243E−02    | 0.4639  |
|          | 256 | 3.4261E−02    | 0.8775  | 3.3986E−02    | 0.8730  |
|          | 512 | 1.2865E−02    | 1.4131  | 1.2768E−02    | 1.4124  |
| $\alpha$ | $M$ | $\lambda = 5$ |        |        | $\lambda = 10$ |        |        |
|---------|-----|----------------|--------|--------|----------------|--------|--------|
|         |     | L-IES (2.4)    | Order 1 | NL-IES (2.3) | Order 1 | L-IES (2.4) | Order 1 | NL-IES (2.3) | Order 1 |
| 1.1     | 64  | 4.316E-03      | –       | 2.345E-03 | –       | 3.410E-03    | –       | 1.352E-03    | –       |
|         | 128 | 2.025E-03      | 1.0916  | 1.103E-03 | 1.0833 | 1.596E-03    | 1.0953  | 6.339E-03    | 1.0931  |
|         | 256 | 8.703E-04      | 1.2185  | 4.746E-04 | 1.2166 | 6.850E-04    | 1.2202  | 2.723E-04    | 1.2191  |
|         | 512 | 2.905E-04      | 1.5830  | 1.585E-04 | 1.5820 | 2.285E-04    | 1.5839  | 9.090E-05    | 1.5829  |
| 1.5     | 64  | 3.169E-02      | –       | 3.012E-02 | –       | 2.622E-02    | –       | 2.469E-02    | –       |
|         | 128 | 1.657E-02      | 0.9350  | 1.584E-02 | 0.9264 | 1.335E-02    | 0.9737  | 1.266E-02    | 0.9641  |
|         | 256 | 7.560E-03      | 1.1327  | 7.251E-03 | 1.1281 | 5.988E-03    | 1.1569  | 5.698E-03    | 1.1519  |
|         | 512 | 2.604E-03      | 1.5374  | 2.502E-03 | 1.5351 | 2.052E-03    | 1.5448  | 1.954E-03    | 1.5440  |
| 1.9     | 64  | 8.049E-02      | –       | 7.901E-02 | –       | 7.878E-02    | –       | 7.729E-02    | –       |
|         | 128 | 5.363E-02      | 0.5858  | 5.294E-02 | 0.5776 | 5.183E-02    | 0.6039  | 5.115E-02    | 0.5956  |
|         | 256 | 2.683E-02      | 0.9989  | 2.658E-02 | 0.9942 | 2.564E-02    | 1.0154  | 2.539E-02    | 1.0105  |
|         | 512 | 1.008E-02      | 1.4123  | 9.991E-03 | 1.4116 | 9.630E-03    | 1.4130  | 9.540E-03    | 1.4122  |
| 1.99    | 64  | 8.688E-02      | –       | 8.545E-02 | –       | 8.679E-02    | –       | 8.536E-02    | –       |
|         | 128 | 6.255E-02      | 0.4740  | 6.185E-02 | 0.4663 | 6.249E-02    | 0.4749  | 6.175E-02    | 0.4671  |
|         | 256 | 3.400E-02      | 0.8792  | 3.373E-02 | 0.8747 | 3.393E-02    | 0.8800  | 3.365E-02    | 0.8755  |
|         | 512 | 1.276E-02      | 1.4132  | 1.267E-02 | 1.4124 | 1.274E-02    | 1.4132  | 1.264E-02    | 1.4125  |
Table 5 The maximum norm errors and convergence orders for Example 2 with $\tau = h$

| $\alpha$ | $N$  | $\lambda = 0$ | $\lambda = 1$ |
|---------|------|----------------|----------------|
|         |      | L-IES (2.4)     | NL-IES (2.3)    | L-IES (2.4)     | NL-IES (2.3)    |
|         |      | $\text{Err}(\tau, h)$ | $\text{Order}^2$ | $\text{Err}(\tau, h)$ | $\text{Order}^2$ | $\text{Err}(\tau, h)$ | $\text{Order}^2$ | $\text{Err}(\tau, h)$ | $\text{Order}^2$ |
| 1.1     | 64   | 1.0317E−01      | –               | 1.3346E−01      | –               | 1.3487E−01      | –               |
|         | 128  | 6.8449E−02      | 0.5919          | 6.8706E−02      | 0.6053          | 7.7164E−02      | 0.7904          | 7.7912E−02      | 0.7917          |
|         | 256  | 4.5280E−02      | 0.5962          | 4.5395E−02      | 0.5979          | 3.8083E−02      | 1.0188          | 3.8428E−02      | 1.0197          |
|         | 512  | 2.6552E−02      | 0.7701          | 2.6588E−02      | 0.7718          | 1.3841E−02      | 1.4602          | 1.3963E−02      | 1.4605          |
| 1.5     | 64   | 4.5283E−02      | –               | 2.8941E−02      | –               | 2.7535E−02      | –               |
|         | 128  | 2.3622E−02      | 0.9388          | 2.2923E−02      | 0.9357          | 1.4966E−02      | 0.9514          | 1.4277E−02      | 0.9476          |
|         | 256  | 1.0920E−02      | 1.1132          | 1.0622E−02      | 1.1097          | 7.0537E−03      | 1.0852          | 6.7675E−03      | 1.0770          |
|         | 512  | 3.8072E−03      | 1.5202          | 3.7073E−03      | 1.5186          | 2.4552E−03      | 1.5225          | 2.3625E−03      | 1.5183          |
| 1.9     | 64   | 8.2087E−02      | –               | 8.0640E−02      | –               | 7.8230E−02      | –               | 7.6775E−02      | –               |
|         | 128  | 5.6383E−02      | 0.5419          | 5.5686E−02      | 0.5342          | 5.3996E−02      | 0.5349          | 5.3301E−02      | 0.5265          |
|         | 256  | 2.8883E−02      | 0.9650          | 2.8616E−02      | 0.9605          | 2.7681E−02      | 0.9640          | 2.7417E−02      | 0.9591          |
|         | 512  | 1.0852E−02      | 1.4123          | 1.0757E−02      | 1.4115          | 1.0404E−02      | 1.4118          | 1.0310E−02      | 1.4110          |
| 1.99    | 64   | 8.4400E−02      | –               | 8.2988E−02      | –               | 8.2930E−02      | –               | 8.1518E−02      | –               |
|         | 128  | 6.1409E−02      | 0.4588          | 6.0712E−02      | 0.4509          | 6.0689E−02      | 0.4505          | 5.9992E−02      | 0.4423          |
|         | 256  | 3.3581E−02      | 0.8708          | 3.3306E−02      | 0.8662          | 3.3280E−02      | 0.8668          | 3.3005E−02      | 0.8621          |
|         | 512  | 1.2614E−02      | 1.4126          | 1.2517E−02      | 1.4119          | 1.2502E−02      | 1.4125          | 1.2405E−02      | 1.4118          |
Table 5 continued

| $\alpha$ | $M$ | $\lambda = 5$ | $\lambda = 10$ |
|----------|-----|----------------|----------------|
|          |     | L-IES (2.4) | NL-IES (2.3) | L-IES (2.4) | NL-IES (2.3) |
|          |     | $\ err(\tau, h) \ Order^2$ | $\ err(\tau, h) \ Order^2$ | $\ err(\tau, h) \ Order^2$ | $\ err(\tau, h) \ Order^2$ |
| 1.1      | 64  | 2.1802E−01  | 2.1913E−01 | 3.1633E−01 | 3.1681E−01 |
|          | 128 | 1.3098E−01  | 0.7361     | 1.3152E−01 | 0.7365     |
|          | 256 | 6.6996E−02  | 0.9662     | 6.7307E−02 | 0.9665     |
|          | 512 | 2.4696E−02  | 1.4239     | 2.5081E−02 | 1.4242     |
| 1.5      | 64  | 3.5242E−02  | –          | 3.4077E−02 | –          |
|          | 128 | 1.8483E−02  | 0.9311     | 1.7949E−02 | 0.9249     |
|          | 256 | 8.3827E−03  | 1.1407     | 8.1585E−03 | 1.1375     |
|          | 512 | 2.8736E−03  | 1.5446     | 2.8000E−03 | 1.5429     |
| 1.9      | 64  | 6.5789E−02  | –          | 6.4329E−02 | –          |
|          | 128 | 4.7243E−02  | 0.4777     | 4.6554E−02 | 0.4666     |
|          | 256 | 2.4505E−02  | 0.9470     | 2.4248E−02 | 0.9410     |
|          | 512 | 9.2250E−03  | 1.4095     | 9.1333E−03 | 1.4087     |
| 1.99     | 64  | 7.6184E−02  | –          | 7.4777E−02 | –          |
|          | 128 | 5.7807E−02  | 0.3982     | 5.7111E−02 | 0.3888     |
|          | 256 | 3.2186E−02  | 0.8448     | 3.1911E−02 | 0.8397     |
|          | 512 | 1.2106E−02  | 1.4107     | 1.2009E−02 | 1.4099     |
of (1.1) is an interesting topic. However, this requires a more invasive modification of the mathematical analysis and algorithm developments, and this will be considered in our future work. We guess that three possible ways to resolve the essentially weak singularity in Eq. (1.1) may by: 1) using non-uniform meshes such as refining spatial mesh near the boundary values and the non-uniform meshes proposed in [49]; 2) adding suitable correction terms to remedy the loss of accuracy [50]; 3) applying the extrapolation technique [51] to Eqs. (2.3) and (2.4). Then, the matrix $A$ in Sect. 2.2 may become a general dense matrix. Some low-rank matrix techniques, such as HSS- and $\mathcal{H}$-matrix [52–54], can be considered to approximate $A$ when designing a preconditioner for the resultant system (2.5).

6 Concluding Remarks

The nonlinear all-at-once system (2.5) arising from the NL-TFDE (1.1) is studied. Firstly, the two implicit schemes (i.e., NL-IES (2.3) and L-IES (2.4)) in Sect. 2 are obtained through applying the finite difference method. Then, the stabilities and first-order convergences of such schemes are analyzed in Sect. 3 under several suitable assumptions. Secondly, for solving all the time steps in Eq. (2.3) simultaneously, the nonlinear all-at-once system (2.5) is derived from it. Then, Newton’s method is employed to solve this system (2.5). Once the method is used to solve such the nonlinear system, the following two basic problems need to be addressed: 1. How to find a good initial value for Newton’s method? 2. How to fast solving the Jacobian equations in Newton’s method? As for the first problem, the value, which is constructed by interpolating the solution of L-IES (2.4) on the coarse grid, is chosen as the initial guess. For the second problem, the PBiCGSTAB method with the preconditioner $P_\ell$ is employed to accelerate solving the Jacobian equations, which is discussed in Sect. 4. On the one hand, numerical examples in Sect. 5 show that the convergence orders of two proposed schemes (both in continuous and discontinuous coefficients cases) can indeed reach 1 in both time and space. On the other hand, they also indicate that our preconditioning strategy is effective for solving (2.5) with continuous or discontinuous coefficients. However, the performance of $P_\ell$ are not satisfactory. The reason may be that the diagonal block matrix in the Jacobian matrix (can be rewritten as $A$ plus this diagonal block matrix), which is resulted from $-\tau \frac{\partial f(u)}{\partial u}$, is not considered when designing $P_\ell$ in this work. Thus, a preconditioner designed with considering such the diagonal block matrix may be more effective to solve Eq. (2.5). In the future work, we will study along with this direction and give some relative
Table 6 Results of different methods when $M = N$ for Example 2

| $(\alpha, \lambda)$ | $N$ | BS Iter1 Time | $T$ Iter1, Iter2 Time | $P$ Iter1, Iter2 Time |
|---------------------|-----|----------------|------------------------|----------------------|
| $(1.1, 0)$          | 129 | 5.0 0.539      | (5.0, 207.0) 6.609     | (5.0, 3.8) 0.167     |
|                     | 257 | 5.0 4.022      | (5.0, 436.2) 42.799    | (5.0, 6.0) 0.815     |
|                     | 513 | 5.0 37.223     | ‡                      | (5.0, 9.2) 4.360     |
|                     | 1025| ‡ ‡            | ‡                      | (5.0, 15.4) 24.702   |
| $(1.5, 0)$          | 129 | 4.0 0.471      | (4.0, 426.8) 10.178    | (4.0, 11.3) 0.331    |
|                     | 257 | 4.0 3.413      | ‡                      | (4.0, 24.3) 2.394    |
|                     | 513 | 4.0 30.196     | ‡                      | (4.0, 62.0) 22.520   |
|                     | 1025| ‡ ‡            | ‡                      | (4.0, 154.8) 190.537 |
| $(1.9, 0)$          | 129 | 4.0 0.462      | (4.0, 880.5) 22.810    | (4.0, 8.0) 0.309     |
|                     | 257 | 4.0 3.426      | ‡                      | (4.0, 21.8) 3.258    |
|                     | 513 | 4.0 30.240     | ‡                      | (4.0, 66.3) 31.259   |
|                     | 1025| ‡ ‡            | ‡                      | (4.0, 212.5) 362.669 |
| $(1.1, 5)$          | 129 | 5.0 0.553      | (5.0, 271.8) 8.562     | (5.0, 2.8) 0.122     |
|                     | 257 | 5.0 4.105      | (5.0, 584.2) 60.355    | (5.0, 4.2) 0.536     |
|                     | 513 | 5.0 36.435     | ‡                      | (5.0, 7.6) 3.632     |
|                     | 1025| ‡ ‡            | ‡                      | (5.0, 14.2) 22.479   |
| $(1.5, 5)$          | 129 | 4.0 0.470      | (4.0, 542.5) 13.903    | (4.0, 6.5) 0.189     |
|                     | 257 | 4.0 3.460      | ‡                      | (4.0, 17.5) 1.704    |
|                     | 513 | 4.0 30.523     | ‡                      | (5.0, 46.4) 20.737   |
|                     | 1025| ‡ ‡            | ‡                      | (5.0, 136.2) 206.973 |
| $(1.9, 5)$          | 129 | 4.0 0.452      | (4.0, 907.0) 24.141    | (4.0, 4.5) 0.167     |
|                     | 257 | 4.0 3.437      | ‡                      | (4.0, 15.5) 2.241    |
|                     | 513 | 4.0 30.595     | ‡                      | (4.0, 54.3) 25.433   |
|                     | 1025| ‡ ‡            | ‡                      | (4.0, 197.5) 338.909 |
| $(1.1, 10)$         | 129 | 5.0 0.568      | (5.0, 277.0) 8.515     | (5.0, 2.8) 0.109     |
|                     | 257 | 5.0 4.069      | (5.0, 598.8) 58.637    | (5.0, 3.6) 0.409     |
|                     | 513 | 5.0 36.612     | ‡                      | (5.0, 6.6) 3.083     |
|                     | 1025| ‡ ‡            | ‡                      | (6.0, 12.3) 23.298   |
| $(1.5, 10)$         | 129 | 5.0 0.549      | (5.0, 571.2) 17.305    | (5.0, 4.2) 0.158     |
|                     | 257 | 5.0 4.097      | ‡                      | (5.0, 12.2) 1.524    |
|                     | 513 | 5.0 36.738     | ‡                      | (5.0, 38.6) 17.222   |
|                     | 1025| ‡ ‡            | ‡                      | (5.0, 124.0) 191.098 |
| $(1.9, 10)$         | 129 | 4.0 0.459      | (5.0, 841.2) 26.614    | (4.0, 3.3) 0.126     |
|                     | 257 | 4.0 3.434      | ‡                      | (4.0, 11.5) 1.689    |
|                     | 513 | 4.0 30.590     | ‡                      | (4.0, 46.0) 21.659   |
|                     | 1025| ‡ ‡            | ‡                      | (4.0, 179.0) 307.717 |

Theoretical analysis. Furthermore, inspired by [29,30], it is interesting to design a parareal algorithm to solve (2.5) with a uniform/nonuniform temporal step size.
Fig. 4 Spectra of $J^0$ and $P_{\ell}^{-1}J^0$, when $\alpha = 1.9$, $M = N = 65$ in Example 2. Top row: $\lambda = 0$; Bottom row: $\lambda = 10$.

Fig. 5 The log-log plot of difference quotient $\delta_x u_j^i$ versus the space for Example 3 with different $\alpha$ and $\lambda$. 
Acknowledgements We would like to thank anonymous referees for their many valuable comments and suggestions for improving the presentation of this work. The authors would like to thank Prof. Hong-Lin Liao for giving some helpful discussions. This research is supported by the National Natural Science Foundation of China (Nos. 61772003, 61876203 and 11801463), the Applied Basic Research Project of Sichuan Province (No. 20YJC3482), the Fundamental Research Funds for the Central Universities (No. JBK1902028) and the Ministry of Education of Humanities and Social Science Layout Project (No. 19JYA790094).

References

1. Cartea, Á., del Castillo-Negrete, D.: Fluid limit of the continuous-time random walk with general Lévy jump distribution functions. Phys. Rev. E 76, 041105 (2007). https://doi.org/10.1103/PhysRevE.76.041105
2. Baeumer, B., Meerschaert, M.M.: Tempered stable Lévy motion and transient super-diffusion. J. Comput. Appl. Math. 233, 2438–2448 (2010)
3. Meerschaert, M.M., Sikorskii, A.: Stochastic Models for Fractional Calculus. De Gruyter, Berlin (2012)
4. Podlubny, I.: Fractional Differential Equations, vol. 198. Academic Press, San Diego (1998)
5. Chakrabarty, A., Meerschaert, M.M.: Tempered stable laws as random walk limits. Stat. Probab. Lett. 81, 899–907 (2011)
6. Zheng, M., Karniadakis, G.E.: Numerical methods for SPDEs with tempered stable processes. SIAM J. Sci. Comput. 37, A1197–A1217 (2015)
7. Carr, P., Geman, H., Madan, D.B., Yor, M.: The fine structure of asset returns: an empirical investigation. J. Bus. 75, 305–332 (2002)
8. Carr, P., Geman, H., Madan, D.B., Yor, M.: Stochastic volatility for Lévy processes. Math. Financ. 13, 345–382 (2003)
9. Wang, W., Chen, X., Ding, D., Lei, S.-L.: Circulant preconditioning technique for barrier options pricing under fractional diffusion models. Int. J. Comput. Math. 92, 2596–2614 (2015)
10. Zhang, H., Liu, F., Turner, I., Chen, S.: The numerical simulation of the tempered fractional Black–Scholes equation for European barrier option. Appl. Math. Model. 40, 5819–5834 (2016)
11. Meerschaert, M.M., Zhang, Y., Baeumer, B.: Tempered anomalous diffusion in heterogeneous systems. Geophys. Res. Lett. 35, L17403 (2008). https://doi.org/10.1029/2008GL034899
12. Metzler, R., Klafter, J.: The restaurant at the end of the random walk: recent developments in the description of anomalous transport by fractional dynamics. J. Phys. A Math. Theor. 37, R161 (2004). https://doi.org/10.1088/0305-4470/37/31/R01
13. Zhang, Y., Meerschaert, M.M.: Gaussian setting time for solute transport in fluvial systems. Water Resour. Res. 47, W08601 (2011). https://doi.org/10.1029/2010WR010102
14. Zhang, Y., Meerschaert, M.M., Packman, A.I.: Linking fluvial bed sediment transport across scales. Geophys. Res. Lett. 39, L20404 (2012). https://doi.org/10.1029/2012GL053476
15. Zhao, Y.-L., Zhu, P.-Y., Luo, W.-H.: A fast second-order implicit scheme for non-linear time-space fractional diffusion equation with time delay and drift term. Appl. Math. Comput. 336, 231–248 (2018)
16. Gu, X.-M., Huang, T.-Z., Jin, C.-C., Carpentieri, B., Alikhany, A.A.: Fast iterative method with a second-order implicit difference scheme for time-space fractional convection–diffusion equation. J. Sci. Comput. 72, 957–985 (2017)
17. Li, M., Gu, X.-M., Huang, C., Fei, M., Zhang, G.: A fast linearized conservative finite element method for the strongly coupled nonlinear fractional Schrödinger equations. J. Comput. Phys. 358, 256–282 (2018)
18. Cartea, A., del Castillo-Negrete, D.: Fractional diffusion models of option prices in markets with jumps. Physica A 374, 749–763 (2007)
19. Marom, O., Momoniat, E.: A comparison of numerical solutions of fractional diffusion models in finance. Nonlinear Anal. Real World Appl. 10, 3435–3442 (2009)
20. Li, C., Deng, W.: High order schemes for the tempered fractional diffusion equations. Adv. Comput. Math. 42, 543–572 (2016)
21. Chen, M., Deng, W.: High order algorithms for the fractional substantial diffusion equation with truncated Lévy flights. SIAM J. Sci. Comput. 37, A890–A917 (2015)
22. Ng, M.K.: Iterative Methods for Toeplitz Systems. Oxford University Press, New York (2004)
23. Chan, R., Jin, X.-Q.: An Introduction to Iterative Toeplitz Solvers. SIAM, Philadelphia (2007)
24. Lei, S.-L., Fan, D., Chen, X.: Fast solution algorithms for exponentially tempered fractional diffusion equations. Numer. Methods Part. Differ. Equ. 34, 1301–1323 (2018)
25. Qu, W., Lei, S.-L.: On CSCS-based iteration method for tempered fractional diffusion equations. Jpn. J. Ind. Appl. Math. 33, 583–597 (2016)
26. Gu, X.-M., Huang, T.-Z., Li, H.-B., Li, L., Luo, W.-H.: On $k$-step CSCS-based polynomial preconditioners for Toeplitz linear systems with application to fractional diffusion equations. Appl. Math. Lett. 42, 53–58 (2015)
27. Gu, X.-M., Huang, T.-Z., Zhao, X.-L., Li, H.-B., Li, L.: Strang-type preconditioners for solving fractional diffusion equations by boundary value methods. J. Comput. Appl. Math. 277, 73–86 (2015)
28. Huang, Y.-C., Lei, S.-L.: Fast solvers for finite difference scheme of two-dimensional time-space fractional differential equations. Numer. Algorithms (2019). https://doi.org/10.1007/s11075-019-00742-6
29. Gander, M.J., Halpern, L.: Time parallelization for nonlinear problems based on diagonalization. In: Lee, C.-O., Cai, X.-C., Keyes, D.E., Kim, H.H., Klawonn, A., Park, E.-J., Widlund, O.B. (eds.) Domain Decomposition Methods in Science and Engineering XXIII, pp. 163–170. Springer, Berlin (2017)
30. Wu, S.: Toward parallel coarse grid correction for the parareal algorithm. SIAM J. Sci. Comput. 40, A1446–A1472 (2018)
31. Gander, M.J.: 50 years of time parallel time integration. In: Carraro, T., Geiger, M., Körkel, S., Rannacher, R. (eds.) Multiple Shooting and Time Domain Decomposition Methods, pp. 69–114. Springer, Berlin (2015)
32. Banjai, L., Peterseim, D.: Parallel multistep methods for linear evolution problems. IMA J. Numer. Anal. 32, 1217–1240 (2012)
33. McDonald, E., Pestana, J., Wathen, A.: Preconditioning and iterative solution of all-at-once systems for evolutionary partial differential equations. SIAM J. Sci. Comput. 40, A1012–A1033 (2018)
34. Ke, R., Ng, M.K., Sun, H.-W.: A fast direct method for block triangular Toeplitz-like with tri-diagonal block systems from time-fractional partial differential equations. J. Comput. Phys. 303, 203–211 (2015)
35. Lu, X., Pang, H.-K., Sun, H.-W.: Fast approximate inversion of a block triangular Toeplitz matrix with applications to fractional sub-diffusion equations. Numer. Linear Algebra Appl. 22, 866–882 (2015)
36. Huang, Y.-C., Lei, S.-L.: A fast numerical method for block lower triangular Toeplitz with dense Toeplitz blocks system with applications to time-space fractional diffusion equations. Numer. Algorithms 76, 605–616 (2017)
37. Lu, X., Pang, H.-K., Sun, H.-W., Vong, S.-W.: Approximate inversion method for time-fractional sub-diffusion equations. Linear Algebra Appl. 450, e2132 (2018). https://doi.org/10.1002/nla.2132
38. Zhao, Y.-L., Zhu, P.-Y., Gu, X.-M., Zhao, X.-L., Cao, J.: A limited-memory block bi-diagonal Toeplitz preconditioner for block lower triangular Toeplitz system from time-space fractional diffusion equation. J. Comput. Math. 362, 99–115 (2019)
39. Sabzikar, F., Meerschaert, M.M., Chen, J.: Tempered fractional calculus. J. Comput. Phys. 293, 14–28 (2015)
40. Kelley, C.T.: Solving nonlinear equations with Newton’s method. SIAM, Philadelphia (2003)
41. van der Vorst, H.A.: Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. 13, 631–644 (1992)
42. Zhuang, P., Liu, F., Anh, V., Turner, I.: Numerical methods for the variable-order fractional advection–diffusion equation with a nonsmooth solution term. SIAM J. Numer. Anal. 47, 1760–1781 (2009)
43. Ervin, V.J., Heuer, N., Roop, J.P.: Regularity of the solution to 1-D fractional order diffusion equations. Math. Comput. 87, 2273–2294 (2018)
44. Varga, R.S.: Geršgorin and His Circles. Springer, Berlin (2004)
45. Lin, F.-R., Yang, S.-W., Jin, X.-Q.: Preconditioned iterative methods for fractional diffusion equation. J. Comput. Phys. 256, 109–117 (2014)
46. Xu, J.: A novel two-grid method for semilinear elliptic equations. SIAM J. Sci. Comput. 15, 231–237 (1994)
47. Xu, J.: Two-grid discretization techniques for linear and nonlinear PDEs. SIAM J. Numer. Anal. 33, 1759–1777 (1996)
48. Kim, D., Park, E.-J., Seo, B.: A unified framework for two-grid methods for a class of nonlinear problems. Calcolo 55, 45 (2018). https://doi.org/10.1007/s10092-018-0287-y
49. Zhao, L., Deng, W.: High order finite difference methods on non-uniform meshes for space fractional operators. Adv. Comput. Math. 42, 425–468 (2016)
50. Chen, X., Zeng, F., Karniadakis, G.E.: A tunable finite difference method for fractional differential equations with non-smooth solutions. Comput. Methods Appl. Mech. Eng. 318, 193–214 (2017)
51. Hao, Z., Cao, W.: An improved algorithm based on finite difference schemes for fractional boundary value problems with non-smooth solution. J. Sci. Comput. 73, 395–415 (2017)
52. Hackbusch, W.: Hierarchical Matrices: Algorithms and Analysis, Springer Series in Computational Mathematics, 49. Springer, Berlin (2015). https://doi.org/10.1007/978-3-662-47324-5
53. Liu, X., Xia, J., de Hoop, M.V.: Parallel randomized and matrix-free direct solvers for large structured dense linear systems. SIAM J. Sci. Comput. 38, S508–S538 (2016)
54. Massei, S., Mazza, M., Robol, L.: Fast solvers for two-dimensional fractional diffusion equations using rank structured matrices. SIAM J. Sci. Comput. 41, A2627–A2656 (2019)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.