Advances in Difference Equations
a SpringerOpen Journal

RESEARCH Open Access

A GPIU method for fractional diffusion equations

Hai-Long Shen[1*], Yu-Han Li[1] and Xin-Hui Shao[1]

*Correspondence:
hailong_shen@126.com
[1] Department of Mathematics, College of Sciences, Northeastern University, Shenyang, P.R. China

Abstract

The fractional diffusion equations can be discretized by applying the implicit finite difference scheme and the unconditionally stable shifted Grünwald formula. Hence, the generating linear system has a real Toeplitz structure when the two diffusion coefficients are non-negative constants. Through a similarity transformation, the Toeplitz linear system can be converted to a generalized saddle point problem. We use the generalization of a parameterized inexact Uzawa (GPIU) method to solve such a kind of saddle point problem and give a new algorithm based on the GPIU method. Numerical results show the effectiveness and accuracy for the new algorithm.

Keywords: Fractional diffusion equations; Generalized saddle point problem; Stability; Toeplitz linear system; The shifted Grünwald formula

1 Introduction

The fractional differential operator is suitable for describing the memory, genetic, mechanical and electrical properties of various materials. Compared with the classical integer differential operator, it can more concisely and accurately describe the biological, mechanical and physical processes with historical memory and spatial global correlation characteristics, such as abnormal diffusion of particles, quantization problem of non-local field theory, the fractional capacitance theory, universal voltage shunt, chaotic circuit analysis, physical semiconductors field, dispersion in porous media, physical and engineering issues related to fractal dimensions, and non-Newtonian fluid mechanics.

The fractional diffusion equations can be abstracted from many practical problems, such as a random walk model describing the competition between sub-diffusion and super-diffusion Three different types of fractional diffusion equations can be derived according to the difference in particle waiting time and jumping steps: when the mean of the waiting time of each step is infinite and the squared mean of the jump steps is limited, the random walk model describes the abnormal sub-diffusion phenomenon, and the time fractional diffusion equation is derived accordingly; when the mean of the waiting time is limited and the squared mean of the jump steps is infinite, the random walk model describes the super-diffusion phenomenon, and the spatial fractional-order diffusion equation is derived accordingly; when the mean of the waiting time and the squared mean of the jump steps are both infinite, the random walk model describes the phenomenon of competition

Springer

between sub-diffusion and super-diffusion, and the fractional diffusion equation in time and space is derived accordingly; using the fractional convection-diffusion equations to simulate Levy's motion is very effective, it can more accurately simulate the solute motion process with long tail behavior; when studying Eulerian estimation of solute transport in porous media, the fractional-order Fick law can be used to replace the traditional Fick law to obtain the fractional-order convection-diffusion equations; in the application of non-Newtonian fluid mechanics, to solve the analytical solution of a material model problem between the Hook solid model and Newtonian fluid model, a special function related to fractional calculus needs to be applied; in the early days, the problem of dynamic response to broadband crust induced by earthquakes that was hardly affected by velocity can be described by a fractional linear rheological solid model. It is precisely because of the advantages of fractional derivatives and the wide application of fractional diffusion equations (FDEs) (1) in groundwater contaminant transport [1, 2], turbulent flow [3], image processing [4], finance [5], physics [6] and other fields, that people have drawn extensive attention to its research.

Considerable numerical methods have been developed as the major ways for solving FDEs [7–12]. However, some salient features of fractional differential operators lead to the unconditional instability [13]. In addition, most numerical methods for finite difference equations (FDEs) can generate a full coefficient matrix, which requires the computational cost of $O(N^3)$ and the storage of $O(N^2)$, where N is the number of grid points [14]. In contrast, the second-order diffusion equations which usually contribute to sparse coefficient matrices can be solved efficiently by fast iteration methods with computational complexity $O(N)$.

To guarantee the stability, Meerschaert and Tadjeran [13] proposed a shifted Grünwald discretization to approximate fractional diffusion equations and it has been proved that it is actually unconditionally stable. Then, based on the Meerschaert–Tadjeran method, a special structure of the full coefficient matrices was presented in [14]. The method maintained a Toeplitz-like structure and reduced the storage requirement from $O(N^2)$ to $O(N)$. Furthermore, the matrix-vector multiplication for the Toeplitz matrix can be calculated by the fast Fourier transform (FFT) with $O(N \log N)$ operations [15]. As a result, fast numerical methods for solving fractional equations with the shifted Grünwald formula have been developed, including the conjugate gradient normal residual (CGNR) methods [16, 17], the preconditioned CGNR (PCGNR) methods which employ some circulant preconditioners [17–19] and the multigrid method [20]. If the diffusion coefficients are constant, the full coefficient matrix originating from the Meerschaert–Tadjeran method will be non-Hermitian positive definite. Therefore, the Hermitian and skew-Hermitian splitting (HSS) iteration and the corresponding preconditioned HSS (PHSS) iteration [21] have also been developed to solve fractional equations.

Since the full coefficient matrix is also a real Toeplitz matrix, the FDE can be transformed into a generalized saddle point problem based on some properties of the real Toeplitz matrix [22]. Therefore, the generalized parameterized inexact Uzawa (GPIU) method [23, 24] is suitable for solving this new saddle point problem. However, it shows some differences from the traditional GPIU method. This method should store three $N/2$ order matrices. Although these matrices have symmetric structures (which will require less storage space), the complexity is less than $O(N^2)$. In the meantime, the complexity of calculation is also

$O(N^2)$ since the method needs to compute the multiplication of several $N/2$ order matrices.

The framework of this paper is as follows. In Sect. 2, we introduce the background of the discretization for the FDEs. Then the real Toeplitz linear system is converted to a generalized saddle point problem by similarity transformation in Sect. 3. In Sect. 4, the GPIU method is considered to solve this kind of generalized saddle point problem and its convergence is analyzed. Besides, a GPIU algorithm is presented. In Sect. 5, numerical examples are performed to illustrate the effectiveness and power of our new method. Finally, conclusions are given in Sect. 6.

2 The depiction of FDEs and finite difference discretization

Firstly, an initial-boundary value problem of the FDEs is given by

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} = d_+(x,t)\frac{\partial^\alpha u(x,t)}{\partial_+ x^\alpha} + d_-(x,t)\frac{\partial^\alpha u(x,t)}{\partial_- x^\alpha} + f(x,t), \\ x \in (x_L, x_R), \quad t \in (0,T], \\ u(x_L,t) = u(x_R,t) = 0, \quad 0 \le t \le T, \\ u(x,0) = u_0(x), \quad x \in [x_L, x_R], \end{cases} \tag{1}$$

where $\alpha \in (1,2)$ is the order of fractional derivatives, $f(x,t)$ is the source term and diffusion coefficient functions $d_\pm \geq 0$ with $d_+ + d_- \neq 0$. The left-sided and the right-sided fractional derivatives $\frac{\partial^\alpha u(x,t)}{\partial_+ x^\alpha}$ and $\frac{\partial^\alpha u(x,t)}{\partial_- x^\alpha}$ of (1) have the same definition as the Grünwald form [25]

$$\frac{\partial^\alpha u(x,t)}{\partial_+ x^\alpha} = \lim_{\Delta x \to 0^+} \frac{1}{\Delta x^\alpha} \sum_{k=0}^{\lfloor (x-x_L)/\Delta x \rfloor} g_k^{(\alpha)} u(x - k\Delta x, t),$$

$$\frac{\partial^\alpha u(x,t)}{\partial_- x^\alpha} = \lim_{\Delta x \to 0^-} \frac{1}{\Delta x^\alpha} \sum_{k=0}^{\lfloor (x_R-x)/\Delta x \rfloor} g_k^{(\alpha)} u(x + k\Delta x, t),$$

where $\lfloor x \rfloor$ denotes the floor of x and the Grünwald weights $g_k^{(\alpha)}$ are the alternating fractional binomial coefficient given as

$$\begin{cases} g_0^{(\alpha)} = 1, \\ g_k^{(\alpha)} = (-1)^k \binom{\alpha}{k}, \quad k \in N_+, \end{cases} \tag{2}$$

which can be evaluated by the following recursive form:

$$g_{k+1}^{(\alpha)} = \left(1 - \frac{\alpha+1}{k+1}\right) g_k^{(\alpha)}, \quad k \in N_+.$$

Besides, the coefficients $g_k^{(\alpha)}$ satisfy the following properties [13, 14, 17, 21] when $1 < \alpha < 2$:

$$\begin{cases} g_0^{(\alpha)} = 1, \quad g_1^{(\alpha)} = -\alpha < 0, \quad 1 \geq g_2^{(\alpha)} \geq g_3^{(\alpha)} \geq \cdots \geq 0, \\ \sum_{k=0}^\infty g_k^{(\alpha)} = 0, \quad \sum_{k=0}^m g_k^{(\alpha)} < 0, \quad (m \geq 1). \end{cases}$$

Let N and M be positive integers, then $\Delta x = \frac{x_R - x_L}{N+1}$ and $\Delta t = \frac{T}{M}$ are the sizes of spatial grids and time steps, respectively. Additionally, the spatial and temporal partition are defined as $x_i = x_L + i\Delta x$ for $i = 0, 1, \dots, N+1$ and $t_m = m\Delta t$ for $m = 0, 1, \dots, M$. We also let $u_i^{(m)} = u(x_i, t_m)$, $d_{\pm,i}^{(m)} = d_\pm(x_i, t_m)$ and $f_i^{(m)} = f(x_i, t_m)$.

Consider the shifted Grünwald approximations [13, 17, 21]:

$$\begin{aligned}
\frac{\partial^\alpha u(x_i, t_m)}{\partial_+ x^\alpha} &= \frac{1}{\Delta x^\alpha} \sum_{k=0}^{i+1} g_k^{(\alpha)} u_{i-k+1}^{(m)} + O(\Delta x), \\
\frac{\partial^\alpha u(x_i, t_m)}{\partial_- x^\alpha} &= \frac{1}{\Delta x^\alpha} \sum_{k=0}^{N-i+2} g_k^{(\alpha)} u_{i+k-1}^{(m)} + O(\Delta x),
\end{aligned}$$

where $g_k^{(\alpha)}$ is defined in (2) and the corresponding implicit finite difference scheme in (1) can be modified as follows:

$$\frac{u_i^{(m)} - u_i^{(m-1)}}{\Delta t} = \frac{d_{+,i}^{(m)}}{\Delta x^\alpha} \sum_{k=0}^{i+1} g_k^{(\alpha)} u_{i-k+1}^{(m)} + \frac{d_{-,i}^{(m)}}{\Delta x^\alpha} \sum_{k=0}^{N-i+2} g_k^{(\alpha)} u_{i+k-1}^{(m)} + f_i^{(m)}. \tag{3}$$

Let $u^{(m)} = [u_1^{(m)}, u_2^{(m)}, \dots, u_N^{(m)}]^T$, $f^{(m)} = [f_1^{(m)}, f_2^{(m)}, \dots, f_N^{(m)}]^T$ and $I \in R^{N\times N}$ be an identity matrix. Then the scheme (3) can construct the following equations:

$$\left(\frac{\Delta x^\alpha}{\Delta t} I + A^{(m)}\right) u^{(m)} = \frac{\Delta x^\alpha}{\Delta t} u^{(m-1)} + \Delta x^\alpha f^{(m)} \tag{4}$$

and

$$A^{(m)} = D_+^{(m)} G_\alpha + D_-^{(m)} G_\alpha^T, \tag{5}$$

where $D_\pm^{(m)} = \operatorname{diag}(d_{\pm,1}^{(m)}, \dots, d_{\pm,N}^{(m)})$ and

$$G_\alpha = -\begin{bmatrix}
g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \cdots & 0 & 0 \\
g_2^{(\alpha)} & g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \cdots & 0 \\
\vdots & g_2^{(\alpha)} & g_1^{(\alpha)} & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & 0 \\
g_{N-1}^{(\alpha)} & \ddots & \ddots & \ddots & \ddots & g_0^{(\alpha)} \\
g_N^{(\alpha)} & g_{N-1}^{(\alpha)} & \cdots & \cdots & g_2^{(\alpha)} & g_1^{(\alpha)}
\end{bmatrix}_{N\times N}. \tag{6}$$

It is obvious that G_α is a Toeplitz matrix [15] and the coefficient matrix $A^{(m)}$ is a full Toeplitz-like matrix, which is non-symmetric when $d_+ \neq d_-$ and the matrix-vector multiplication for the $A^{(m)}$ can be obtained in $O(N \log N)$ operations by the fast Fourier transform (FFT).

We can define $\nu_{N,M} = \frac{\Delta x^\alpha}{\Delta t} = (x_R - x_L)^\alpha T^{-1} \frac{M}{(N+1)^\alpha}$, related to the numbers of time steps and grid points. The linear system (4) is represented as follows:

$$M^{(m)} u^{(m)} = b^{(m-1)}, \tag{7}$$

where

$$M^{(m)} = \frac{\Delta x^{\alpha}}{\Delta t} I + A^{(m)} = \nu_{N,M} I + D_{+}^{(m)} G_{\alpha} + D_{-}^{(m)} G_{\alpha}^{T} \tag{8}$$

and

$$b^{(m-1)} = \nu_{N,M} \bigl(u^{(m-1)} + \Delta t f^{(m)} \bigr).$$

As the coefficient matrix $M^{(m)}$ in (8) is a strictly diagonally dominant M-matrix [14], we know the matrix $\frac{M^{(m)}+M^{(m)T}}{2}$ is also a non-singular M-matrix where $M^{(m)T}$ is used to denote the transposition of the matrix $M^{(m)}$. For the reason that every non-singular symmetric M-matrix is positive definite, $\frac{M^{(m)}+M^{(m)T}}{2}$ is a symmetric positive definite matrix. Therefore, $M^{(m)}$ is also positive definite (see [21, 26]).

3 The transformation for the Toeplitz matrix $M^{(m)}$

As mentioned in Sect. 2, the coefficient matrix $M^{(m)}$ is a real positive definite Toeplitz-like matrix. We assume that the diffusion coefficient functions are two non-negative constants, i.e., $d_{+,i}^{(m)} = d_{+} \geq 0$, $d_{-,i}^{(m)} = d_{-} \geq 0$, but $d_{+} + d_{-} \neq 0$. Then $M^{(m)}$ becomes a real non-symmetric positive definite Toeplitz matrix. In addition, M is appropriately chosen according to N and satisfies the limit of $\nu_{N,M}$ away from 0.

Then we transform the Toeplitz linear system (7) into a generalized saddle point problem by constructing an orthogonal matrix to deal with $M^{(m)}$. The first row and column of the real $N \times N$ Toeplitz matrix $M^{(m)}$ can be indicated as

$$\bigl[\nu_{N,M} + d_{+} g_{1}^{(\alpha)} + d_{-} g_{1}^{(\alpha)}, d_{+} g_{0}^{(\alpha)} + d_{-} g_{2}^{(\alpha)}, d_{-} g_{3}^{(\alpha)}, \ldots, d_{-} g_{N}^{(\alpha)} \bigr]$$

and

$$\bigl[\nu_{N,M} + d_{+} g_{1}^{(\alpha)} + d_{-} g_{1}^{(\alpha)}, d_{+} g_{2}^{(\alpha)} + d_{-} g_{0}^{(\alpha)}, d_{+} g_{3}^{(\alpha)}, \ldots, d_{+} g_{N}^{(\alpha)} \bigr]^{T},$$

respectively.

We use the Hermitian and skew-Hermitian splitting (HSS) method to split $M^{(m)}$ and get

$$M^{(m)} = H^{(m)} + S^{(m)}, \tag{9}$$

where $H^{(m)} = \frac{M^{(m)}+M^{(m)T}}{2}$ is symmetric positive definite and $S^{(m)} = \frac{M^{(m)}-M^{(m)T}}{2}$ is skew-symmetric. Interestingly, $H^{(m)}$ is a centrosymmetric Toeplitz matrix and $S^{(m)}$ is a skew-centrosymmetric Toeplitz matrix. Denote $J_N = [e_N, e_{N-1}, \ldots, e_1]$, where $e_i \in R^N$ is the column vector with the ith row being 1, i.e., J_N is a permutation matrix where 1 is on the counter-diagonal and 0 is elsewhere. Thus the matrices $H^{(m)}$ and $S^{(m)}$ are able to be replaced by

$$H^{(m)} = J_N H^{(m)} J_N \quad \text{and} \quad S^{(m)} = -J_N S^{(m)} J_N,$$

respectively.

Considering that N is even or odd, we discuss the structures of $H^{(m)}$ and $S^{(m)}$ in detail.

Case I: N is even, let $N = 2m$. Then $H^{(m)}$ can be rewritten as

$$H^{(m)} = \begin{bmatrix} E & J_m F J_m \\ F & J_m E J_m \end{bmatrix},$$

where $E \in R^{m\times m}$ is a symmetric Toeplitz matrix whose first column can be given as

$$\left[v_{N,M} + (d_+ + d_-) g_1^{(\alpha)}, \frac{d_+ + d_-}{2}\bigl(g_0^{(\alpha)} + g_2^{(\alpha)}\bigr), \frac{d_+ + d_-}{2} g_3^{(\alpha)}, \dots, \frac{d_+ + d_-}{2} g_m^{(\alpha)} \right]^T,$$

and $F \in R^{m\times m}$ is a Toeplitz matrix whose first row and column are as follows:

$$\left[\frac{d_+ + d_-}{2} g_{m+1}^{(\alpha)}, \frac{d_+ + d_-}{2} g_m^{(\alpha)}, \frac{d_+ + d_-}{2} g_{m-1}^{(\alpha)}, \dots, \frac{d_+ + d_-}{2}\bigl(g_0^{(\alpha)} + g_2^{(\alpha)}\bigr) \right],$$
$$\left[\frac{d_+ + d_-}{2} g_{m+1}^{(\alpha)}, \frac{d_+ + d_-}{2} g_{m+2}^{(\alpha)}, \frac{d_+ + d_-}{2} g_{m+3}^{(\alpha)}, \dots, \frac{d_+ + d_-}{2} g_N^{(\alpha)} \right]^T,$$

respectively.

Analogously, $S^{(m)}$ can be rewritten as

$$S^{(m)} = \begin{bmatrix} G & -J_m L J_m \\ L & -J_m G J_m \end{bmatrix},$$

where $G \in R^{m\times m}$ is a skew-symmetric Toeplitz matrix whose first column can be given as

$$\left[0, \frac{d_+ - d_-}{2}\bigl(g_2^{(\alpha)} - g_0^{(\alpha)}\bigr), \frac{d_+ - d_-}{2} g_3^{(\alpha)}, \dots, \frac{d_+ - d_-}{2} g_m^{(\alpha)} \right]^T,$$

and $L \in R^{m\times m}$ is a Toeplitz matrix whose first row and column are as follows:

$$\left[\frac{d_+ - d_-}{2} g_{m+1}^{(\alpha)}, \frac{d_+ - d_-}{2} g_m^{(\alpha)}, \frac{d_+ - d_-}{2} g_{m-1}^{(\alpha)}, \dots, \frac{d_+ - d_-}{2}\bigl(g_2^{(\alpha)} - g_0^{(\alpha)}\bigr) \right],$$
$$\left[\frac{d_+ - d_-}{2} g_{m+1}^{(\alpha)}, \frac{d_+ - d_-}{2} g_{m+2}^{(\alpha)}, \frac{d_+ - d_-}{2} g_{m+3}^{(\alpha)}, \dots, \frac{d_+ - d_-}{2} g_N^{(\alpha)} \right]^T,$$

respectively.

We give an orthogonal matrix P defined as

$$P = \frac{\sqrt{2}}{2} \begin{bmatrix} I_m & I_m \\ J_m & -J_m \end{bmatrix},$$

then multiply the left and right sides of (9) by P^T from the left and P from the right, respectively, we have the new forms of $\tilde{H}^{(m)}$ and $\tilde{S}^{(m)}$:

$$\tilde{H}^m = P^T H^{(m)} P = \begin{bmatrix} E + J_m F & 0 \\ 0 & E - J_m F \end{bmatrix}$$

and

$$\tilde{S}^{(m)} = P^T S^{(m)} P = \begin{bmatrix} 0 & G + J_m L \\ G - J_m L & 0 \end{bmatrix}.$$

Hence, let $\tilde{M}^{(m)} = P^T M^{(m)} P$, $\tilde{u}^{(m)} = P^T u^{(m)}$, and $\tilde{b}^{(m-1)} = P^T b^{(m-1)}$, we can rewrite Eq. (7) as the following linear system:

$$\tilde{M}^{(m)} \tilde{u}^{(m)} = \tilde{b}^{(m-1)} \tag{10}$$

and we have

$$\tilde{M}^{(m)} = P^T M^{(m)} P = P^T H^{(m)} P + P^T S^{(m)} P = \tilde{H}^{(m)} + \tilde{S}^{(m)} = \begin{bmatrix} E + J_m F & G + J_m L \\ G - J_m L & E - J_m F \end{bmatrix}.$$

For simplicity, let $B = E + J_m F$, $C = E - J_m F$ and $W = -G + J_m L$, then we can find that

$$W^T = (-G + J_m L)^T = -G^T + L^T J_m = G + J_m L.$$

The coefficient matrix $\tilde{M}^{(m)}$ can be expressed as

$$\tilde{M}^{(m)} = \begin{bmatrix} B & W^T \\ -W & C \end{bmatrix}, \tag{11}$$

and $\tilde{M}^{(m)}$ is real non-symmetric positive definite because $M^{(m)}$ is real positive definite.

Moreover, B and C are both symmetric positive definite matrices because $\tilde{H}^{(m)}$ is a symmetric positive definite matrix. So the linear system (7) is transformed into a generalized saddle point problem under the assumption of case I.

Case II: N is an odd number, let $N = 2m + 1$. Then the matrix $H^{(m)}$ can be written as

$$H^{(m)} = \begin{bmatrix} E & J_m z_1 & J_m F J_m \\ z_1^T J_m & \nu_{N,M} + (d_+ + d_-) g_1^{(\alpha)} & z_1^T \\ F & z_1 & J_m E J_m \end{bmatrix},$$

where E, F and J_m are similar to the above, and the column vector z_1 is expressed as follows:

$$\left[\frac{d_+ + d_-}{2} g_{m+1}^{(\alpha)}, \frac{d_+ + d_-}{2} g_m^{(\alpha)}, \dots, \frac{d_+ + d_-}{2} \big(g_0^{(\alpha)} + g_2^{(\alpha)} \big) \right]^T.$$

In addition, $S^{(m)}$ can be expressed as

$$S^{(m)} = \begin{bmatrix} G & J_m z_2 & -J_m L J_m \\ -z_2^T J_m & 0 & z_2^T \\ L & -z_2 & -J_m G J_m \end{bmatrix},$$

where G and L are similar to the above, and the column vector z_2 is expressed as follows:

$$\left[\frac{d_- - d_+}{2} g_{m+1}^{(\alpha)}, \frac{d_- - d_+}{2} g_m^{(\alpha)}, \dots, \frac{d_- - d_+}{2} \big(g_2^{(\alpha)} - g_0^{(\alpha)} \big) \right]^T.$$

Then the orthogonal matrix is defined as

$$P = \frac{\sqrt{2}}{2}\begin{bmatrix} I_m & 0 & I_m \\ 0 & \sqrt{2} & 0 \\ J_m & 0 & -J_m \end{bmatrix}.$$

Multiplying the left and right sides of (9) by P^T from the left and P from the right, respectively, we have the new forms of $\tilde{H}^{(m)}$ and $\tilde{S}^{(m)}$ as

$$\tilde{H}^{(m)} = P^T H^{(m)} P = \begin{bmatrix} E + J_m F & \sqrt{2}J_m z_1 & 0 \\ \sqrt{2}z_1^T J_m & \nu_{N,M} + (d_+ + d_-)g_1^{(\alpha)} & 0 \\ 0 & 0 & E - J_m F \end{bmatrix}$$

and

$$\tilde{S}^{(m)} = P^T S^{(m)} P = \begin{bmatrix} 0 & 0 & G + J_m L \\ 0 & 0 & -\sqrt{2}z_2^T J_m \\ G - J_m L & \sqrt{2}J_m z_2 & 0 \end{bmatrix}.$$

The Toeplitz linear system (7) can also be constructed as a generalized saddle point problem which has a form similar to (10), where

$$B = \begin{bmatrix} E + J_m F & \sqrt{2}J_m z_1 \\ \sqrt{2}z_1^T J_m & \nu_{N,M} + (d_+ + d_-)g_1^{(\alpha)} \end{bmatrix}, \qquad C = E - J_m F, \qquad W = \begin{bmatrix} G + J_m L \\ -\sqrt{2}z_2^T J_m \end{bmatrix}^T.$$

All in all, no matter what positive integer N we choose, the scheme for the FDEs is able to be transformed into a generalized saddle point problem.

4 A GPIU method on the transformed saddle point problem

By splitting the new coefficient matrix $\tilde{M}^{(m)}$ in the linear system (10), we get

$$\tilde{M}^{(m)} = X^{(m)} - Y^{(m)}, \tag{12}$$

where

$$X^{(m)} = \begin{bmatrix} Q_1 + B & 0 \\ -W + Q_3 & Q_2 \end{bmatrix} \quad \text{and} \quad Y^{(m)} = \begin{bmatrix} Q_1 & -W^T \\ Q_3 & Q_2 - C \end{bmatrix},$$

here the definitions of B, C and W are given in the above section, $Q_1 \in R^{\lceil N/2 \rceil}$ is a symmetric positive definite matrix, $Q_2 \in R^{\lfloor N/2 \rfloor}$ is a symmetric positive definite matrix, and $Q_3 \in R^{\lfloor N/2 \rfloor \times \lceil N/2 \rceil}$ is arbitrary, where $\lceil N/2 \rceil$ and $\lfloor N/2 \rfloor$ represents the ceil and the floor of $N/2$, respectively.

Considering the GPIU method [23, 24], we choose $Q_2 = C/\delta$ where $\delta > 0$, so $Y^{(m)}$ can be rewritten as

$$Y^{(m)} = \begin{bmatrix} Q_1 & -W^T \\ Q_3 & (1-\delta)Q_2 \end{bmatrix}.$$

Thus, the linear system (10) is redefined by the splitting form of (12) as

$$\begin{bmatrix} Q_1 + B & 0 \\ -W + Q_3 & Q_2 \end{bmatrix} \tilde{u}_{n+1}^{(m)} = \begin{bmatrix} Q_1 & -W^T \\ Q_3 & (1-\delta)Q_2 \end{bmatrix} \tilde{u}_n^{(m)} + \tilde{b}^{(m-1)}. \tag{13}$$

The form of the iteration matrix corresponding to (13) is as follows:

$$\boldsymbol{W} = \begin{bmatrix} Q_1 + B & 0 \\ -W + Q_3 & Q_2 \end{bmatrix}^{-1} \begin{bmatrix} Q_1 & -W^T \\ Q_3 & (1-\delta)Q_2 \end{bmatrix}. \tag{14}$$

Denote by $\rho(\boldsymbol{W})$ the spectral radius of the iteration matrix $\boldsymbol{W}$, so Eq. (13) converges if and only if $\rho(\boldsymbol{W}) < 1$.

Let λ be an eigenvalue of $\boldsymbol{W}$ and $[\tilde{u}_x^H, \tilde{u}_y^H]^H$ be the corresponding eigenvector with $\tilde{u}_x \in C^{\lceil N/2 \rceil}$ and $\tilde{u}_y \in C^{\lfloor N/2 \rfloor}$, then we obtain

$$\boldsymbol{W} \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} = \lambda \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix},$$

Or, equivalently,

$$\begin{cases} Q_1 \tilde{u}_x - W^T \tilde{u}_y = \lambda (Q_1 + B) \tilde{u}_x, \\ Q_3 \tilde{u}_x + (1-\delta) Q_2 \tilde{u}_y = \lambda (-W + Q_3) \tilde{u}_x + \lambda Q_2 \tilde{u}_y. \end{cases} \tag{15}$$

Aiming to obtain the convergence condition, the lemmas and theorem as follows are presented.

Lemma 4.1 *Suppose that B, C are symmetric positive definite, Q_1 is symmetric positive semidefinite, $Q_2 = C/\delta$ is symmetric positive definite where $\delta > 0$, and Q_3 is arbitrary. If λ is an eigenvalue of $\boldsymbol{W}$ defined by (14) and $[\tilde{u}_x^H, \tilde{u}_y^H]^H$ is the corresponding eigenvector, where $\tilde{u}_x \in C^{\lceil N/2 \rceil}$ and $\tilde{u}_y \in C^{\lfloor N/2 \rfloor}$, then*

(a) $\lambda \neq 1$,

(b) *when W is full of row rank, $\tilde{u}_x$ is a non-zero vector,*

(c) *when W is not full of row rank*

(i) *if $\lambda \neq 1 - \delta$, $\tilde{u}_x$ is a non-zero vector;*

(ii) *if $\lambda = 1 - \delta$, $\tilde{u}_x$ can be a zero vector, and $\lambda = 1 - \delta$ is at least $\lfloor N/2 \rfloor - r(W)$ eigenvalues of $\boldsymbol{W}$; where $r(W)$ represents the rank of matrix W and $r(W) = r(W^T)$.*

Proof (a) If $\lambda = 1$, then we have

$$\begin{cases} B\tilde{u}_x + W^T \tilde{u}_y = 0, \\ -W\tilde{u}_x + \delta Q_2 \tilde{u}_y = 0. \end{cases}$$

We can rewrite the system as

$$\begin{bmatrix} B & W^T \\ -W & C \end{bmatrix} \begin{bmatrix} \tilde{u}_x \\ \tilde{u}_y \end{bmatrix} = 0.$$

Evidently, the coefficient matrix is non-singular, so $[\tilde{u}_x^H, \tilde{u}_y^H]^H = 0$, which is contradictory, so $\lambda \neq 1$.

(b) If $\tilde{u}_x = 0$, then according to (15) we have

$$\begin{cases} W^T \tilde{u}_y = 0, \\ [\lambda - (1-\delta)]Q_2 \tilde{u}_y = 0. \end{cases} \tag{16}$$

When W is full of row rank, according to the first equation of (16), we have $\tilde{u}_y = 0$. This is a contradiction.

(c) When W is not full of row rank, let $\tilde{u}_x = 0$.

(i) If $\lambda \neq 1 - \delta$, according to the second equation of (16) and Q_2 is symmetric positive definite, we obtain $\tilde{u}_y = 0$, which is a contradiction.

(ii) If $\lambda = 1 - \delta$, the fundamental set of solution can be obtained by the first equation of (16) on $\tilde{u}_y$ which consists of $\lfloor N/2 \rfloor - r(W)$ non-zero vector solutions, i.e., $\lambda = 1 - \delta$ is at least $\lfloor N/2 \rfloor - r(W)$ eigenvalues, where $r(W)$ represents the rank of matrix W and $r(W) = r(W^T)$. □

Lemma 4.2 *Assume that B, C are symmetric positive definite, Q_1 is symmetric positive semidefinite, $Q_2 = C/\delta$ is symmetric positive definite where $\delta > 0$, and Q_3 is arbitrary. Let λ be an eigenvalue of $\boldsymbol{W}$ and $[\tilde{u}_x^H, \tilde{u}_y^H]^H$ be the corresponding eigenvector with $\tilde{u}_x \in C^{\lceil N/2 \rceil}$ and $\tilde{u}_y \in C^{\lfloor N/2 \rfloor}$. If $\tilde{u}_y = 0$,*

(a) *when W is full of row rank, then $0 \leq \lambda < 1$,*

(b) *when W is not full of row rank, then $|\lambda| < 1$ if and only if $0 < \delta < 2$.*

Proof (a) If $\tilde{u}_y = 0$, from (15), we obtain

$$\begin{cases} Q_1 \tilde{u}_x - \lambda(Q_1 + B)\tilde{u}_x = 0, \\ Q_3 \tilde{u}_x - \lambda(-W + Q_3)\tilde{u}_x = 0. \end{cases} \tag{17}$$

When W is full of row rank, since $\lambda \neq 1$ and $\tilde{u}_x \neq 0$, from Lemma 4.1, we denote

$$\alpha = \frac{\tilde{u}_x^H Q_1 \tilde{u}_x}{\tilde{u}_x^H \tilde{u}_x} \geq 0, \qquad \beta = \frac{\tilde{u}_x^H B \tilde{u}_x}{\tilde{u}_x^H \tilde{u}_x} > 0.$$

Through multiplying the both sides of the first equation in (17) by $\frac{\tilde{u}_x^H}{\tilde{u}_x^H \tilde{u}_x}$ from the left, we get

$$0 \leq \lambda = \frac{\alpha}{\alpha + \beta} < 1.$$

(b) When W is not full of row rank, if $\lambda \neq 1 - \delta$, then $\tilde{u}_x \neq 0$ from Lemma 4.1. As $\lambda \neq 1$, in the same way as with (a), we get $0 \leq \lambda < 1$.

If $\lambda = 1 - \delta$, we rewrite the system (17) as

$$\begin{cases} [\delta Q_1 - (1-\delta)B]\tilde{u}_x = 0, \\ [\delta Q_3 + (1-\delta)W]\tilde{u}_x = 0. \end{cases}$$

Thus, $\tilde{u}_x \in \text{null}[\delta Q_1 - (1-\delta)B]$ and $\tilde{u}_x \in \text{null}[\delta Q_3 + (1-\delta)W]$, where null$[x]$ represents the null space of the corresponding matrix. For guaranteeing that $|\lambda| < 1$, we have $|1-\delta| < 1$, i.e., $0 < \delta < 2$.

Moreover, we get $|\lambda| < 1$ if and only if $0 < \delta < 2$. □

Lemma 4.3 ([23, 27]) *If ϕ and φ are real numbers, the sufficient and necessary condition for the roots of the real quadratic equation $x^2 + \varphi x + \phi = 0$ to satisfy $|x| < 1$ is that $|\phi| < 1$ and $|\varphi| < 1 + \phi$.*

In order to ensure the convergence of the iteration method (13), the following theorems are used to give the necessary and sufficient conditions.

Theorem 4.1 *Suppose that B, C are symmetric positive definite, Q_1 is symmetric positive semidefinite, $Q_2 = C/\delta$ is symmetric positive definite where $\delta > 0$, and Q_3 is such that $W^T Q_2^{-1} Q_3$ is symmetric. Then the iteration method (13) is convergent if and only if*

$$0 < \delta < 2 \quad \textit{and} \quad (\delta - 2)\alpha + \frac{\delta - 2}{2}\beta + \frac{1}{2}\gamma < \tau < \delta\alpha + \beta,$$

where

$$\alpha = \frac{\tilde{u}_x^H Q_1 \tilde{u}_x}{\tilde{u}_x^H \tilde{u}_x} \geq 0, \qquad \beta = \frac{\tilde{u}_x^H B \tilde{u}_x}{\tilde{u}_x^H \tilde{u}_x} > 0,$$
$$\gamma = \frac{\tilde{u}_x^H W^T Q_2^{-1} W \tilde{u}_x}{\tilde{u}_x^H \tilde{u}_x} \geq 0, \qquad \tau = \frac{\tilde{u}_x^H W^T Q_2^{-1} Q_3 \tilde{u}_x}{\tilde{u}_x^H \tilde{u}_x},$$

and $[\tilde{u}_x^H, \tilde{u}_y^H]^H$ be a corresponding eigenvector where $\tilde{u}_x \in C^{\lceil N/2 \rceil}$ and $\tilde{u}_y \in C^{\lfloor N/2 \rfloor}$.

Proof Let λ be an eigenvalue of the iteration matrix $\boldsymbol{W}$ and $[\tilde{u}_x^H, \tilde{u}_y^H]^H$ be its corresponding eigenvector with $\tilde{u}_x \in C^{\lceil N/2 \rceil}$ and $\tilde{u}_y \in C^{\lfloor N/2 \rfloor}$. By Lemma 4.1, we know $\lambda \neq 1$.

If $\lambda = 1 - \delta$, then (15) results in

$$\begin{cases} [\delta Q_1 - (1-\delta)B]\tilde{u}_x = W^T \tilde{u}_y, \\ [(1-\delta)W + \delta Q_3]\tilde{u}_x = 0. \end{cases} \tag{18}$$

Equivalently, $\tilde{u}_x \in \text{null}[(1-\delta)W + \delta Q_3]$, $\tilde{u}_y = (WW^T)^{-1}[\delta W Q_1 - (1-\delta)WB]\tilde{u}_x$ when W is full of row rank, where null$[\cdot]$ represents the null space of the corresponding matrix. Then $\lambda = 1 - \delta$ is an eigenvalue of the iteration matrix $\boldsymbol{W}$.

When W is not full of row rank, $\tilde{u}_x \in \text{null}[(1-\delta)W + \delta Q_3]$ and there exist $\lfloor N/2 \rfloor - r(W)$ non-zero eigenvectors of $\tilde{u}_y$ corresponding to $\lambda = 1 - \delta$ which are at least $\lfloor N/2 \rfloor - r(W)$ eigenvalues of the iteration matrix $\boldsymbol{W}$. Particularly, when $\tilde{u}_x$ is the zero vector, it has been introduced in Lemma 4.1.

If $\lambda \neq 1 - \delta$, we have $\tilde{u}_x \neq 0$ from the Lemma 4.1, and (15) can be rewritten as

$$\begin{cases} Q_1 \tilde{u}_x - \lambda(Q_1 + B)\tilde{u}_x = W^T \tilde{u}_y, \\ \tilde{u}_y = \frac{1}{(1-\delta)-\lambda} Q_2^{-1}[\lambda(-W + Q_3) - Q_3]\tilde{u}_x. \end{cases} \tag{19}$$

Eliminating $\tilde{u}_y$ in (19), we obtain

$$\big[(1-\delta)-\lambda\big]\big[Q_1-\lambda(Q_1+B)\big]\tilde{u}_x = W^T Q_2^{-1}\big[\lambda(-W+Q_3)-Q_3\big]\tilde{u}_x. \tag{20}$$

Then multiplying the left and right sides of (20) by $\frac{\tilde{u}_x^H}{\tilde{u}_x^H\tilde{u}_x}$ from the left, we obtain

$$\big[(1-\delta)-\lambda\big]\big[\alpha-\lambda(\alpha+\beta)\big] = \lambda(-\gamma+\tau)-\tau, \tag{21}$$

where

$$\alpha = \frac{\tilde{u}_x^H Q_1\tilde{u}_x}{\tilde{u}_x^H\tilde{u}_x} \geq 0, \qquad \beta = \frac{\tilde{u}_x^H B\tilde{u}_x}{\tilde{u}_x^H\tilde{u}_x} > 0,$$
$$\gamma = \frac{\tilde{u}_x^H W^T Q_2^{-1} W\tilde{u}_x}{\tilde{u}_x^H\tilde{u}_x} \geq 0, \qquad \tau = \frac{\tilde{u}_x^H W^T Q_2^{-1} Q_3\tilde{u}_x}{\tilde{u}_x^H\tilde{u}_x}.$$

Then (21) can be modified as

$$\lambda^2 - \frac{(2-\delta)\alpha+(1-\delta)\beta-\gamma+\tau}{\alpha+\beta}\lambda + \frac{(1-\delta)\alpha+\tau}{\alpha+\beta} = 0. \tag{22}$$

When $\gamma = 0$, we can get $W\tilde{u}_x = 0$ and multiply the left and right sides of the first equation in (19) by $\tilde{u}_x$ from the left, we have

$$0 \leq \lambda = \frac{\alpha}{\alpha+\beta} < 1.$$

When $W\tilde{u}_x \neq 0$, we obtain $\gamma > 0$. In accordance with Lemma 4.3, the roots of the quadratic Eq. (22) satisfy $|\lambda| < 1$ if and only if

$$\begin{cases} |1-\delta| < 1, \\ |(1-\delta)\alpha+\tau| < \alpha+\beta, \\ |(2-\delta)\alpha+(1-\delta)\beta-\gamma+\tau| < (2-\delta)\alpha+\beta+\tau. \end{cases} \tag{23}$$

Solving the inequalities of (23), we have

$$\begin{cases} 0 < \delta < 2, \\ (\delta-2)\alpha + \frac{\delta-2}{2}\beta + \frac{1}{2}\gamma < \tau < \delta\alpha+\beta. \end{cases} \tag{24}$$

In addition, we consider the case (b) from Lemma 4.2 when $\tilde{u}_y = 0$, it actually satisfies the inequalities (24).

Therefore, we can get (24) where we have considered that $\alpha \geq 0$ and $\beta > 0$. □

Corollary 4.1 *Under the assumptions of Theorem 4.1, convergence of the iteration method (13) is satisfied if and only if* $2(2-\delta)Q_1 + (2-\delta)B + 2W^T Q_2^{-1}Q_3 - W^T Q_2^{-1} W$ *and* $\delta Q_1 + B - W^T Q_2^{-1} Q_3$ *are positive definite where* $0 < \delta < 2$.

By choosing different matrices Q_1, Q_2 and Q_3 in (12), we get considerable iteration algorithms from the GPIU method for solving the generalized saddle point problem (11).

Here, let $\tilde{u}_{n+1}^{(m)} = \left[\begin{smallmatrix}\tilde{u}_{x+1}^{(m)}\\ \tilde{u}_{y+1}^{(m)}\end{smallmatrix}\right]$, $\tilde{u}_{n}^{(m)} = \left[\begin{smallmatrix}\tilde{u}_{x}^{(m)}\\ \tilde{u}_{y}^{(m)}\end{smallmatrix}\right]$ and $\tilde{b}^{(m-1)} = \left[\begin{smallmatrix}\tilde{b}_{1}^{(m-1)}\\ \tilde{b}_{2}^{(m-1)}\end{smallmatrix}\right]$, where $\tilde{u}_{x+1}^{(m)}, \tilde{u}_{x}^{(m)}, \tilde{b}_{1}^{(m-1)} \in C^{\lceil N/2\rceil}$, and $\tilde{u}_{y+1}^{(m)}, \tilde{u}_{y}^{(m)}, \tilde{b}_{2}^{(m-1)} \in C^{\lfloor N/2\rfloor}$. Then an efficient GPIU algorithm can be presented.

The GPIU Algorithm If $Q_1 = \frac{1}{\omega} W^T Q_2^{-1} W\omega > 0$, $Q_2 = \frac{1}{\delta} C0 < \delta < 2$, $Q_3 = sW$, the GPIU method gives the following iteration scheme:

$$\begin{cases} \tilde{u}_{x+1}^{(m)} = \tilde{u}_{x}^{(m)} + (\frac{1}{\omega} W^T Q_2^{-1} W + B)^{-1}(\tilde{b}_{1}^{(m-1)} - B\tilde{u}_{x}^{(m)} - W^T \tilde{u}_{y}^{(m)}), \\ \tilde{u}_{y+1}^{(m)} = \tilde{u}_{y}^{(m)} + \delta C^{-1}[(1-s)W\tilde{u}_{x+1}^{(m)} + sW\tilde{u}_{x}^{(m)} - C\tilde{u}_{y}^{(m)} + \tilde{b}_{2}^{(m-1)}] \end{cases}$$

where the matrices B, C and W rely on N defined in Sect. 4.

5 Numerical examples

The FDEs problem (1) has been solved by pre-multiplying the coefficient matrix with P^T and post-multiplying it with P which is given in Sect. 3, firstly. Then the FDEs problem is changed into a generalized saddle point problem, so in Sect. 4 an algorithm is proposed based on the GPIU method. In this section, we discuss the GPIU algorithm with different parameters and comparison between the GPIU algorithm and the CGNR method (3). The initial value is defined as the zero vector at each time step, and each iteration process terminates when the current residual satisfies $\|r_k\|_2/\|r_0\|_2 < 10^{-7}$, where r_k is the residual vector of the linear system after k iterations. Then we collect data in the table as follows and define "N" as the number of spatial grid points, "M" as the number of time steps, "Error" as the difference between the exact solution and the approximate solution under the infinity norm, "CPU(s)" as the total CPU(s) time (seconds) to solve the whole FDEs problem, and "Iter" as the average number of iterations needed to solve the FDEs problem, i.e.,

$$\text{Iter} = \frac{1}{M}\sum_{m=1}^{M} \text{Iter}(m),$$

where Iter(m) represents the number of iterations needed to solve (7). Moreover, we obtain the true solution approximately by the program command inv($\tilde{M}^{(m)}$)$\tilde{b}^{(m-1)}$. All numerical experiments are executed in MATLAB (2016a) of a Windows 8.1 system on a machine with Intel (R) Core (TM) (i5-5257U CPU @ 2.70 GHz 2700 MHz).

Example 1 ([17]) Consider the initial-boundary value problem (1) on the spatial domain $[x_L, x_R] = [0, 2]$ and time interval $[0, T] = [0, 1]$ with the diffusion coefficients $d_+ = 0.6$, $d_- = 0.5$, and

$$u(x,0) = \exp\left(-\frac{(x-x_c)^2}{2\sigma^2}\right), \qquad x_c = 1.2, \qquad \sigma = 0.08, \qquad f(x,t) \equiv 0.$$

For satisfying $v_{N,M}$ is bounded away from 0, let $\Delta t \approx 2\Delta x^{\alpha}$. From Table 1, we show the advantage of efficiency according to the CPU time, Iter and Error. When N and M become large, the CPU time and Error will be slow and show inaccuracy, respectively. But the average number of iterations remains stabilized.

Table 2 demonstrates that not only the Iter but also CPU(s) by the CGNR method with optimal parameters are much higher than the GPIU method. The iteration steps and CPU

Table 1 The GPIU algorithm for solving Example 1 with different choices of parameters for $\alpha = 1.5$

N	M	(ω, δ, s)	CPU (s)	Iter	Error
2^6	93	(0.8, 0.8, 0.2)	0.0541	9.85	2.27e–07
		(0.8, 0.9, 0.2)	0.0465	6.97	2.16e–07
		(0.9, 0.9, 0.1)	0.0672	6.98	2.14e–07
		(1, 0.99, 0.01)	0.0432	4	2.72e–08
		(1, 1.1, 0.1)	0.0498	6.98	2.16e–07
2^7	259	(0.8, 0.8, 0.2)	0.4028	9.73	5.20e–07
		(0.8, 0.9, 0.2)	0.3590	6.95	5.02e–07
		(0.9, 0.9, 0.1)	0.3599	6.95	5.02e–07
		(1, 0.99, 0.01)	0.3286	3.98	1.73e–07
		(1, 1.1, 0.1)	0.4187	6.95	5.03e–07
2^8	728	(0.8, 0.8, 0.2)	3.7215	9.66	1.23e–06
		(0.8, 0.9, 0.2)	3.1373	6.93	1.18e–06
		(0.9, 0.9, 0.1)	3.2618	6.93	1.18e–06
		(1, 0.99, 0.01)	3.4343	3.93	5.68e–07
		(1, 1.1, 0.1)	3.4742	6.93	1.18e–06
2^9	2054	(0.8, 0.8, 0.2)	57.9257	9.61	2.96e–06
		(0.8, 0.9, 0.2)	56.3505	6.92	2.78e–06
		(0.9, 0.9, 0.1)	55.5128	6.92	2.78e–06
		(1, 0.99, 0.01)	53.3056	3.92	1.27e–06
		(1, 1.1, 0.1)	56.5414	6.91	2.79e–06

Table 2 Comparing results to solve Example 1 by the GPIU algorithm with $\omega = 1$, $\delta = 0.99$ and $s = 0.01$, and the CGNR method for $\alpha = 1.2$, 1.5 and 1.8

α	N	M	GPIU			CGNR			HSS		
			CPU (s)	Iter	Error	CPU (s)	Iter	Error	CPU (s)	Iter	Error
1.2	2^6	33	0.0266	4	3.63e–08	0.1506	49	1.12e–07	0.1199	18.36	2.88e–07
	2^7	74	0.0939	3.96	1.64e–07	0.5807	53	1.32e–06	0.5460	18.92	7.19e–07
	2^8	170	0.8145	3.86	5.18e–07	2.5166	54	1.19e–06	3.9674	19	1.33e–06
	2^9	389	10.6105	3.8123	1.12e–06	41.0703	54	1.38e–06	62.5631	19	3.37e–06
1.5	2^6	93	0.0481	4	2.72e–08	0.4913	58	1.06e–06	0.2374	20.79	3.27e–07
	2^7	259	0.3301	3.98	1.73e–07	2.0613	62	1.35e–06	1.7875	20.95	8.05e–07
	2^8	728	3.3763	3.93	5.68e–07	13.5068	65	1.09e–06	19.7542	21	1.59e–06
	2^9	2054	52.4298	3.92	1.27e–07	250.4274	65	1.19e–06	337.2273	21	4.19e–06
1.8	2^6	263	0.0994	4	2.42e–08	1.3840	67	1.17e–07	0.5653	22.95	3.74e–07
	2^7	904	1.1810	3.96	2.88e–07	8.3969	73	1.30e–06	5.8788	22.99	9.07e–07
	2^8	3126	12.2058	3.89	7.06e–07	60.9816	77	1.40e–06	72.6848	23	2.69e–06

time of the new method is obviously less than HSS. It also indicates that the Error is obviously improved for the FDEs over CGNR and HSS.

Figure 1 shows the convergence rate of GPIU and HSS. In order to get the least number of steps which can makes the error decrease to the ε times initial error, we normally define $-\ln(\rho(M))$ as the asymptotic convergence rate, where M is the iteration matrix of the iteration method for solving the equations. The minimum number of steps of the iteration is $k \geq \frac{-\ln\varepsilon}{-\ln(\rho(M))}$. In Fig. 1, the X-axis stands for the scale of the matrix, and the Y-axis stands for the theoretical minimum steps for iteration. It is shown that, for GPIU, the minimum number of steps increases slowly and almost tends to a stable value as N increases. But for HSS, the minimum number of steps increases sharply.

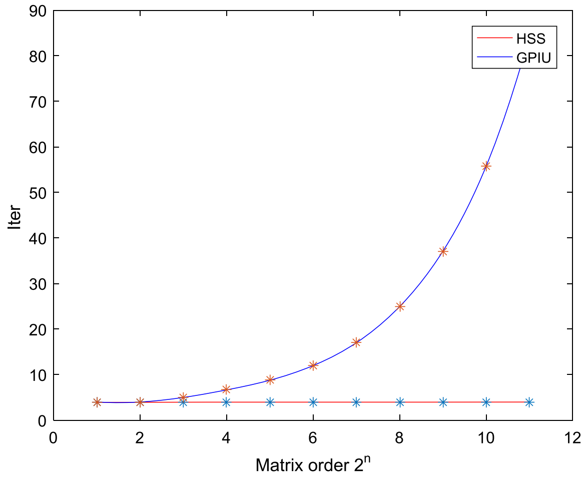

Figure 1 The convergence rate of GPIU and HSS

Table 3 Comparing results to solve Example 2 by the GPIU algorithm with $\omega = 1$, $\delta = 0.99$ and $s = 0.01$, and the CGNR method for $\alpha = 1.2$, 1.5 and 1.8

α	N	M	GPUI			CGNR			HSS		
			CPU (s)	Iter	Error	CPU (s)	Iter	Error	CPU (s)	Iter	Error
1.2	2^6	33	0.0419	5.5152	3.55e–06	0.2310	65	1.71e–05	0.0815	16	1.05e–05
	2^7	74	0.1142	5.04	4.19e–06	0.7962	77	2.58e–05	0.5448	22	3.36e–05
	2^8	170	0.8816	5.01	3.35e–06	3.4947	81	2.46e–05	4.1474	22	7.78e–05
	2^9	389	9.7131	4.12	1.85e–05	60.4473	82	2.51e–04	63.5913	22	1.77e–04
1.5	2^6	93	0.0561	4.98	6.10e–07	0.5494	70	2.49e–05	0.2148	24	1.83e–05
	2^7	259	0.3715	4.04	5.30e–06	2.8837	84	1.98e–04	1.6499	24	5.11e–05
	2^8	728	3.1912	4	3.88e–06	16.7189	88	2.32e–04	18.1177	24	1.28e–04
	2^9	2054	53.8474	4.00	6.35e–06	312.7302	89	2.35e–04	342.1307	24	3.09e–04
1.8	2^6	263	0.1348	4.00	5.37e–07	1.6771	78	1.51e–05	0.6147	27	2.49e–05
	2^7	904	1.1877	3.99	3.17e–06	11.4262	98	2.33e–05	5.4371	27	6.98e–05
	2^8	3126	13.6965	3.98	1.43e–05	81.1444	103	2.46e–04	73.2352	27	1.88e–04

Example 2 ([21]) The spatial domain $[x_L, x_R] = [0,1]$ and the time interval $[0,T] = [0,1]$ with $d_+ = 0.6$, $d_- = 0.2$, initial condition $u(x,0) = 4x^2(2-x)^2$, and the source term $f(x,t) = -32e^{-t}[x^2 + (2-x)^2 + 0.125x^2(2-x)^2 - 2.5(x^3 + (2-x)^3) + \frac{25}{22}(x^4 + (2-x)^4)]$.

Table 3 illustrates the numerical results of Example 2. Here we specially select the source term $f(x,t)$ as non-zero and nonlinear. Obviously, the Iter for the CGNR method is larger and the Iter of the new method is smaller than HSS. Moreover, the CPU time and the error of the GPIU method are much smaller than the CGNR method and HSS method. It indicates that the GPIU algorithm is still highly efficient and accurate for the case of nonlinear source term, so the GPIU algorithm can be extendable to the case of nonlinear source term.

Note. The memory usage for all tests in Tables 1, 2 and 3 is $O(n)$, where n denotes the order of the coefficient matrix of equations [7].

6 Concluding remarks

In this paper, a GPIU method is put forward to solve the generalized saddle point problem generated by the fractional diffusion equations with constant coefficients by premultiplying the orthogonal matrix we constructed and post-multiplying the transposition of it, respectively. Then the convergence condition of the GPIU method is analyzed and we give a new GPIU algorithm.

The numerical results show that the proposed method is more effective than the CGNR method and HSS, and its advantages become more obvious as N increases. Compared with the CGNR method and the HSS method, this GPIU method can achieve a faster convergence rate in practice and theory, and can obtain a more accurate convergence solution in a shorter time. Unlike the CGNR method and the HSS method, with the increase of the matrix order N, the number of iteration steps of the GPIU method grows slowly, it is almost stable near an extremely low value, and the error is significantly improved. The better convergence results of GPIU are due to the fact that the storage of the new algorithm is greatly reduced to $O(n)$, to be compared with the CGNR and HSS method, where n denotes the order of the coefficient matrix of equations.

For further consideration, more effective stationary methods like the GPIU method will be promoted to solve the FDE problems with constant or even variable diffusion coefficients.

Acknowledgements
The authors would like to thank the editors and reviewers for their reading.

Funding
This work was funded by the Natural Science Foundation of Liaoning Province (CN) (No. 20170540323); Central University Basic Scientific Research Business Expenses Special Funds (N2005013).

Availability of data and materials
The data are available from the corresponding author upon request.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
SHL designed, analyzed and wrote the paper. SXH guided the full text. LYH analyzed the data. All authors read and approved the final manuscript.

Authors' information
Hai-Long Shen is a Lecturer at Northeastern University, Shenyang, China. Xin-Hui Shao is a Professor and Master advisor at Northeastern University. Yu-Han Li is studying for a Master's degree at Northeastern University.

Publisher's Note

Received: 16 March 2020 Accepted: 27 May 2020 Published online: 04 August 2020

References

1. Benson, D., Wheatcraft, S.W., Meerschaert, M.M.: Application of a fractional advection-dispersion equation. Water Resour. Res. **36**, 1403–1413 (2000)
2. Benson, D., Wheatcraft, S.W., Meerschaert, M.M.: The fractional-order governing equation of Lévy motion. Water Resour. Res. **36**, 1413–1423 (2000)
3. Carreras, B.A., Lynch, V.E., Zaslavsky, G.M.: Anomalous diffusion and exit time distribution of particle tracers in plasma turbulence models. Phys. Plasmas **8**, 5096–5103 (2001)
4. Bai, J., Feng, X.: Fractional-order anisotropic diffusion for image denoising. IEEE Trans. Image Process. **16**, 2492–2502 (2007)
5. Raberto, M., Scalas, E., Mainardi, F.: Waiting-times and returns in high-frequency financial data: an empirical study. Physica A **314**, 749–755 (2002)
6. Sokolov, I.M., Klafter, J., Blumen, A.: Fractional kinetics. Phys. Today **11**, 28–53 (2002)

7. Shen, S., Liu, F., Anh, V., Tuner, I.: The fundamental solution and numerical solution of the Riesz fractional advection-dispersion equation. IMA J. Appl. Math. **73**, 850–872 (2008)
8. Gao, G.H., Sun, Z.Z.: A compact difference scheme for the fractional sub-diffusion equations. J. Comput. Phys. **230**, 586–595 (2011)
9. Liu, F., Zhuang, P., Burrage, K.: Numerical methods and analysis for a class of fractional advection-dispersion models. Comput. Math. Appl. **64**, 2990–3007 (2012)
10. Beumer, B., Kovács, M., Meerschaert, M.M.: Numerical solutions for fractional reaction-diffusion equations. Comput. Math. Appl. **55**, 2212–2226 (2008)
11. Deng, W.: Finite element method for the space and time fractional Fokker–Planck equation. SIAM J. Numer. Anal. **47**, 204–226 (2008)
12. Tadjeran, C., Meerschaert, M.M., Scheffler, H.P.: A second-order accurate numerical approximation for the fractional diffusion equation. J. Comput. Phys. **213**, 205–213 (2006)
13. Meerschaert, M.M., Tadjeran, C.: Finite difference approximations for two-sided space-fractional partial differential equations. Appl. Numer. Math. **56**, 80–90 (2006)
14. Wang, H., Wang, K., Sircar, T.: A direct $O(N\log^2 N)$ finite difference method for fractional diffusion equations. J. Comput. Phys. **229**, 8095–8104 (2010)
15. Chan, R., Ng, M.: Conjugate gradient methods for Toeplitz systems. SIAM Rev. **38**, 427–482 (1996)
16. Wang, K., Wang, H.: A fast characteristic finite difference method for fractional advection-diffusion equations. Adv. Water Resour. **34**, 810–816 (2011)
17. Lei, S.L., Sun, H.W.: A circulant preconditioner for fractional diffusion equations. J. Comput. Phys. **242**, 715–725 (2013)
18. Chan, R., Strang, G.: Toeplitz equations by conjugate gradients with circulant preconditioner. SIAM J. Sci. Stat. Comput. **10**, 104–119 (1989)
19. Chan, T.: An optimal circulant preconditioner for Toeplitz systems. SIAM J. Sci. Stat. Comput. **9**, 766–771 (1988)
20. Pang, H., Sun, H.: Multigrid method for fractional diffusion equations. J. Comput. Phys. **231**, 693–703 (2012)
21. Bai, Y.Q., Huang, T.Z., Gu, X.M.: Circulant preconditioned iterations for fractional diffusion equations based on Hermitian and skew-Hermitian splittings. Appl. Math. Lett. **48**, 14–22 (2015)
22. Chen, F., Jiang, Y.L.: On HSS and AHSS iteration methods for nonsymmetric positive definite Toeplitz systems. J. Comput. Appl. Math. **234**, 2432–2440 (2010)
23. Zhou, Y.Y., Zhang, G.F.: A generalization of parameterized inexact Uzawa method for generalized saddle point problems. Appl. Math. Comput. **215**, 599–607 (2009)
24. Bai, Z.Z., Wang, Z.Q.: On parameterized inexact Uzawa methods for generalized saddle point problems. Linear Algebra Appl. **428**, 2900–2932 (2008)
25. Podlubny, I.: Fractional Differential Equations. Academic Press, New York (1999)
26. Quarteroni, A., Sacco, R., Saleri, F.: Numerical Mathematics, 2nd edn. Texts in Appl. Math., vol. 37. Springer, Berlin (2007)
27. Young, D.M.: Iterative Solution for Large Linear Systems. Academic Press, New York (1971)