A PRECONDITIONED FAST HERMITE FINITE ELEMENT METHOD FOR SPACE-FRACTIONAL DIFFUSION EQUATIONS

MENG ZHAO AND AIJIE CHENG∗

School of Mathematics
Shandong University
Jinan, Shandong 250100, China

HONG WANG

Department of Mathematics
University of South Carolina
Columbia, SC 29208, USA

(Communicated by Peter E. Kloeden)

Abstract. We develop a fast Hermite finite element method for a one-dimensional space-fractional diffusion equation, by proving that the stiffness matrix of the method can be expressed as a Toeplitz block matrix. Then a block circulant preconditioner is presented. Numerical results are presented to show the utility of the fast method.

1. Introduction. Fractional partial differential equations (FPDEs) provide a competitive approach for the description of transport dynamics in complex systems which are governed by anomalous diffusion and non-exponential relaxation patterns [25, 29]. In the last decade, many numerical methods have been developed for the numerical solution of FPDEs, including finite difference methods, finite element methods and spectral methods [8, 11, 21, 20, 24, 31, 38]. In particular, Ervin and Roop developed and analyzed Galerkin finite element methods for space-fractional PDEs [11].

Due to the nonlocal nature of fractional differential operators, numerical methods for space-fractional PDEs usually generate dense or full stiffness matrices, for which traditional solvers require $O(m^3)$ of operations step and $O(m^2)$ of memory for a problem with $m$ unknowns. This is deemed computationally very expensive to solve. Fast numerical methods were previous developed for finite difference methods and finite volume methods [35, 36, 37], by proving that the stiffness matrix is Toeplitz like. This would reduce the memory from $O(m^2)$ to $O(m)$ and computational cost from $O(m^3)$ to $O(m \log m)$.

In this paper we develop a fast cubic Hermite element method for space-fractional diffusion equation by proving that the stiffness matrix can be expressed as a block Toeplitz matrix. Due to the stiffness matrix are ill-conditioned, the condition number of the stiffness matrix is huge with the increasing number of grids, even makes
the iterative method divergent. So we present a block circulant preconditioner to accelerate the Krylov subspace iterative method. The rest of the paper is organized as follows. In §2, we present the model problem. In §3 we assemble the stiffness matrix of the Hermite cubic finite element method. In §4 we prove that the stiffness matrix is a block Toeplitz matrix, which yields an $O(m)$ memory requirement of the stiffness matrix and a lossless $O(m \log m)$ matrix-vector multiplication. In §5 we present a block circulant preconditioner to accelerate the fast Krylov subspace iterative method. In §6 we present numerical experiments to investigate the performance of the fast method.

2. Problem formulation and preliminaries. We consider the homogeneous Dirichlet boundary-value problem of the one-dimensional steady-state fractional diffusion equation

$$-D(K(\gamma_0 D_x^{-\beta} + (1-\gamma)_x D_1^{-\beta})Du) = f(x), \quad 0 < x < 1,$$

$$u(0) = 0, \quad u(1) = 0. \quad (1)$$

Here $Du(x) = u'$ is the first-order differential operator, $2 - \beta$ with $0 < \beta < 1$ represents the order of anomalous diffusion, $K$ is the diffusivity coefficient, $0 \leq \gamma \leq 1$ indicates the relative weight of forward versus backward transition problem, $f(x)$ is the source and sink term, and $u_l, u_r$ are the prescribed Dirichlet boundary data. The left and right fractional integrals of order $\beta$ are defined by [27]

$$0 D_x^{-\beta} u(x) := \frac{1}{\Gamma(\beta)} \int_0^x \frac{u(s)}{(x-s)^{1-\beta}} ds,$$

$$D_1^{-\beta} u(x) := \frac{1}{\Gamma(\beta)} \int_x^1 \frac{u(s)}{(s-x)^{1-\beta}} ds, \quad (2)$$

where $\Gamma(\cdot)$ is the Gamma function. The left and right fractional derivatives of Caputo form of order $\beta$ are defined by [27]

$$C^\alpha_0 D_x^\beta u(x) := 0 D_x^{(1-\beta)} Du(x) = \frac{1}{\Gamma(1-\beta)} \int_0^x \frac{u'(s)}{(x-s)^{\beta}} ds,$$

$$C^\alpha_x D_1^\beta u(x) := -x D_1^{(1-\beta)} Du(x) = -\frac{1}{\Gamma(1-\beta)} \int_x^1 \frac{u'(s)}{(s-x)^{\beta}} ds. \quad (3)$$

Let $H^\mu_0(0, 1)$ with $\mu > 1/2$ be the fractional Sobolev space of order $\mu$ with trace 0 at the boundary $x = 0$ and 1. [11]

Multiplying (1) by any test function $v \in H^1_0(0, 1)$ gives the following weak formulation,

$$\int_0^1 K(\gamma_0 D_x^{-\beta} + (1-\gamma)_x D_1^{-\beta})Du(x)Dv(x)dx = \int_0^1 f(x)v(x)dx. \quad (4)$$

Then the Galerkin formulation is followed: for given $f \in H^{-(1-\beta/2)}(0, 1)$, find $u \in H^{1-\beta/2}_0(0, 1)$ such that

$$B(u, v) = \langle f, v \rangle, \quad \forall v \in H^{1-\beta/2}_0(0, 1), \quad (5)$$

where $\langle \cdot, \cdot \rangle$ is the duality pairing between $H^{-\mu}(0, 1)$ and $H^\mu_0(0, 1)$ and for the two-sided problem the associated bilinear form $B$ is defined: $H^{1-\beta/2}(0, 1) \times H^{1-\beta/2}(0, 1)$
\( B(u, v) \) such that
\[
B(u, v) = \gamma K(0_D^\beta D_u, D_v) + (1 - \gamma) K(\beta D_1^\beta D_u, D_v)
\]
\[
= -\gamma K(\beta D_1^\beta D_u, D_v)_{L^2(0, 1)}
\]
\[
-(1 - \gamma) K(\beta D_1^\beta D_u, D_v)_{L^2(0, 1)},
\]
\( \forall u, v \in H_0^1(0, 1) \).

We recall some results for fractional elliptic differential equations with constant coefficients. The following theorem was proved in [11].

**Theorem 2.1.** For \( 0 < \beta < 1 \) and \( K \in \mathbb{R}^+, \) the bilinear form \( B(\cdot, \cdot) \) is continuous and coercive on \( H_0^1(0, 1) \) \( \times H_0^1(0, 1) \). Hence, the Galerkin weak formulation [6] has a unique solution \( u \in H_0^1(0, 1) \) with the stability estimate
\[
\|u\|_{H^1(0, 1)} \leq C \|f\|_{H^{-1}(0, 1)}.
\]

A Galerkin finite element method: Seek \( u_h \in V_h \) such that
\[
B(u_h, v_h) = \langle f, v_h \rangle \quad \forall v_h \in V_h.
\]

**Theorem 2.2.** Assume that the weak solution \( u \) to problem [6] belongs to \( H^\alpha(0, 1) \) \( \cap H_0^1(0, 1) \) with \( 1 - \beta/2 \leq \alpha \leq 4 \). Let \( u_h \in V_h \) (refer to the definition at the end of \( \S 3 \)) be its finite element approximation. Then an optimal-order error estimate in the energy norm holds for \( 1 - \beta/2 \leq \alpha \leq 4 \)
\[
\|u_h - u\|_{H^\alpha(0, 1)} \leq C h^{\alpha-1+\beta/2} \|u\|_{H^\alpha(0, 1)}.
\]

3. **Hermite finite element method.** Let \( \Omega_h \) denote a uniform partition of \( \Omega = [0, 1] \), which is given by
\[
\Omega_h := \{0 = x_0 < x_1 < \cdots < x_{m-1} < x_m = 1\},
\]
where \( m \) is a positive integer. Let \( h = 1/m = x_i - x_{i-1}, \ I_i = (x_{i-1}, x_i), \ i = 1, 2, \cdots, m. \)

We define the finite dimensional subspace \( V_h \subset H_0^\mu(\Omega) \) on the mesh \( \Omega_h \), which can be expressed by
\[
V_h = \{ v : v \in H_0^\mu(\Omega) \cap C(\Omega), \ v|_{I_i} \in P_3(I_i) \},
\]
where \( P_3(I_i) \) is the space of cubic polynomials defined on \( I_i \).

The nodal base functions \( \varphi_0^{(i)}, \varphi_1^{(i)}, \cdots, \varphi_m^{(i)} \) and \( \varphi_0^{(i)}, \varphi_1^{(i)}, \cdots, \varphi_m^{(i)} \) of \( V_h \) can be expressed in the form
\[
\varphi_i^{(0)}(x) = \begin{cases} 
(1 - x_i - x/\!\!\!h)^2(2(x_i - x)/\!\!\!h) + 1, & x \in [x_{i-1}, x_i], \\
(1 - x_i - x/\!\!\!h)^2(2(x - x_i)/\!\!\!h) + 1, & x \in [x_i, x_{i+1}], \\
0, & \text{elsewhere}, \end{cases}
\]
\[
\varphi_i^{(1)}(x) = \begin{cases} 
(x - x_i)(x_i - x/h - 1)^2, & x \in [x_{i-1}, x_i], \\
(x - x_i)(x_i - x/h - 1)^2, & x \in [x_i, x_{i+1}], \\
0, & \text{elsewhere}, \end{cases}
\]
for \( i = 1, 2 \cdots, m - 1 \), and

\[
\varphi_0^{(0)}(x) = \begin{cases} (1 - \frac{x-x_1}{h})^2\left(\frac{2(x-x_1)}{h} + 1\right), & x \in [x_0, x_1], \\ 0, & \text{elsewhere}, \end{cases}
\]

\[
\varphi_m^{(0)}(x) = \begin{cases} (1 - \frac{x_{m-1}-x}{h})^2\left(\frac{2(x_{m-1}-x)}{h} + 1\right), & x \in [x_{m-1}, x_m], \\ 0, & \text{elsewhere}, \end{cases}
\]

\[
\varphi_0^{(1)}(x) = \begin{cases} (x-x_1)(\frac{x-x_1}{h} - 1)^2, & x \in [x_0, x_1], \\ 0, & \text{elsewhere}, \end{cases}
\]

\[
\varphi_m^{(1)}(x) = \begin{cases} (x-x_{m-1})(\frac{x_{m-1}-x}{h} - 1)^2, & x \in [x_{m-1}, x_m], \\ 0, & \text{elsewhere}, \end{cases}
\]

(14)

(15)

Let \( u_h = \sum_{i=0}^{m} (u_i \varphi_i^{(0)}(x) + u'_i \varphi_i^{(1)}(x)) \) denote the approximation of \( u \) in finite element space \( V_h \). Choosing the test function \( v \) to be \( \varphi_i^{(0)}(x), \varphi_i^{(1)}(x), \varphi_i^{(0)}(x), \varphi_i^{(1)}(x), \cdots, \varphi_{m-1}^{(0)}(x), \varphi_{m-1}^{(1)}(x) \) in sequence, we obtain

\[
\begin{align*}
\int_0^1 K_0 D_x^\beta \sum_{i=0}^{m} \left( u_i \frac{d\varphi_i^{(0)}(x)}{dx} + u'_i \frac{d\varphi_i^{(1)}(x)}{dx} \right) \frac{d\varphi_j^{(s)}(x)}{dx} dx \\
+ \int_0^1 K(1-\gamma) D_x^\beta \sum_{i=0}^{m} \left( u_i \frac{d\varphi_i^{(0)}(x)}{dx} + u'_i \frac{d\varphi_i^{(1)}(x)}{dx} \right) \frac{d\varphi_j^{(s)}(x)}{dx} dx \\
= \int_0^1 f(x) \frac{d\varphi_j^{(s)}(x)}{dx} dx, & j = 1, 2, \cdots, m - 1; s = 0, 1.
\end{align*}
\]

(16)

Thus, the Hermite \( P_3 \) finite element scheme for the homogeneous Dirichlet boundary value problem can be written in a matrix form

\[ Gu = f, \]

(17)

where \( u \) be an \( 2(m - 1) \)-dimensional vector given by

\[ u := [u_1, u'_1, u_2, u'_2, \cdots, u_{m-1}, u'_{m-1}]^T, \]

(18)

the \( 2(m - 1) \)-dimensional vector \( f \) can be expressed in a similar fashion, and \( G \) is an \( 2(m - 1) \times 2(m - 1) \) stiffness matrix.

We go through some tedious algebraic manipulations to evaluate all the entries of the stiffness matrix \( G \) and present them below.

Denote the matrix \( G = [g_{i,j}]_{i,j=1}^{2m-2} \). The entries of \( G \) are given by

\[
g_{2i-1,2i-1} = -36h^{\beta-1} \left[ \frac{1}{\Gamma(\beta+4)} 2^{\beta+3} - \frac{4}{\Gamma(\beta+5)} (2^{\beta+4} - 2) \right], \quad 1 \leq i \leq m - 1. \]

(19)

\[
g_{2i,2i} = h^{\beta+1} \left[ \frac{4}{\Gamma(\beta+4)} (2^{\beta+3} + 8) - \frac{24}{\Gamma(\beta+5)} (2^{\beta+4} + 4) \right], \quad 1 \leq i \leq m - 1.
\]

(20)
The lower triangular entries of the $G$ are given by

\begin{align}
\frac{g_{2i+1,2i}}{g_{2i-1,2i}} &= (2\gamma - 1)6h^\beta \left[ \frac{1}{\Gamma(\beta + 4)} (2^{\beta+4} - 8) - \frac{1}{\Gamma(\beta + 5)} (10 \cdot 2^{\beta+4} 8) \\
+ \frac{1}{\Gamma(\beta + 6)} (12 \cdot 2^{\beta+5} - 24) \right], 
1 \leq i \leq m - 1.
\end{align}

\begin{align}
g_{2i+1,2i} &= -36\gamma h^\beta \left[ \frac{1}{\Gamma(\beta + 4)} (3^{\beta+3} - 2) - \frac{4}{\Gamma(\beta + 5)} (3^{\beta+4} - 2^{\beta+5}) \\
+ \frac{4}{\Gamma(\beta + 6)} (3^{\beta+5} - 2^{\beta+7} + 6) \right] - 36(1 - \gamma)h^\beta \left[ \frac{1}{\Gamma(\beta + 4)} \\
\frac{4}{\Gamma(\beta + 5)} + \frac{4}{\Gamma(\beta + 6)} \right], 
1 \leq i \leq m - 2.
\end{align}

\begin{align}
\frac{g_{2i+2,2i}}{g_{2i+1,2i}} &= 6\gamma h^\beta \left[ \frac{1}{\Gamma(\beta + 4)} (2 \cdot 3^{\beta+3} - 8 \cdot 2^{\beta+3}) - \frac{1}{\Gamma(\beta + 5)} (10 \cdot 3^{\beta+4} \\
+ 8 \cdot 2^{\beta+4} - 36) + \frac{1}{\Gamma(\beta + 6)} (12 \cdot 3^{\beta+5} - 24 \cdot 2^{\beta+5}) \right] + \\
6(1 - \gamma)h^\beta \left[ \frac{2}{\Gamma(\beta + 4)} - \frac{10}{\Gamma(\beta + 5)} + \frac{12}{\Gamma(\beta + 6)} \right], 
1 \leq i \leq m - 2.
\end{align}

\begin{align}
g_{2i+2,2i} &= \gamma h^{\beta+1} \left[ \frac{4}{\Gamma(\beta + 4)} (3^{\beta+3} + 8 \cdot 2^{\beta+3} + 18) - \frac{24}{\Gamma(\beta + 5)} (3^{\beta+4} + 4 \cdot 2^{\beta+4} + \frac{36}{\Gamma(\beta + 6)} (3^{\beta+5} - 2) \right] - (1 - \gamma)h^{\beta+1} \\
&\left[ \frac{4}{\Gamma(\beta + 4)} + \frac{24}{\Gamma(\beta + 5)} - \frac{36}{\Gamma(\beta + 6)} \right], 
1 \leq i \leq m - 2.
\end{align}

and

\begin{align}
\frac{g_{2i+2,2i+1}}{g_{2i+1,2i+1}} &= -36\gamma h^{\beta-1} \left[ \frac{1}{\Gamma(\beta + 4)} ((i - j + 2)^{\beta+3} - 2(i - j)^{\beta+3} \\
+ (i - j - 2)^{\beta+3} - \frac{4}{\Gamma(\beta + 5)} ((i - j + 1)^{\beta+4} + 2(i - j - 2)^{\beta+4} \\
+ 2(i - j - 1)^{\beta+4} - (i - j - 2)^{\beta+4} + \frac{4}{\Gamma(\beta + 6)} ((i - j + 2)^{\beta+5} \\
- 4(i - j + 1)^{\beta+5} + 6(i - j)^{\beta+5} - 4(i - j - 1)^{\beta+5} \\
+ (i - j - 2)^{\beta+5}) \right], 
3 \leq i \leq m - 1, 1 \leq j \leq i - 2.
\end{align}
Similarly, the upper triangular entries of the $G$ are given by

$$g_{2i-1,2j} = 6\gamma h^2 \left[ \frac{1}{\Gamma(\beta + 4)} (2(i-j+2)^{\beta+3} + 8(i-j+1)^{\beta+3} - 8(i-j-1)^{\beta+3} - 2(i-j-2)^{\beta+3}) - \frac{1}{\Gamma(\beta + 5)} (10(i-j+2)^{\beta+4} + 8(i-j+1)^{\beta+4} - 36(i-j)^{\beta+4} + 8(i-j-1)^{\beta+4} + 10(i-j-2)^{\beta+4}) + \frac{1}{\Gamma(\beta + 6)} (12(i-j+2)^{\beta+5} - 24(i-j+1)^{\beta+5} + 24(i-j-1)^{\beta+5}) - 12(i-j-2)^{\beta+5}) \right], \ 3 \leq i \leq m-1, 1 \leq j \leq i-2,$$

$$g_{2i,2j} = \gamma h^{\beta+1} \left[ \frac{4}{\Gamma(\beta + 4)} ((i-j+2)^{\beta+3} + 8(i-j+1)^{\beta+3} + 18(i-j)^{\beta+3} + 8(i-j-1)^{\beta+3} + (i-j-2)^{\beta+3}) - \frac{24}{\Gamma(\beta + 5)} ((i-j+2)^{\beta+4} + 4(i-j+1)^{\beta+4} - 4(i-j-1)^{\beta+4} - (i-j-2)^{\beta+4}) + \frac{36}{\Gamma(\beta + 6)} ((i-j+2)^{\beta+5} - 2(i-j)^{\beta+5} + (i-j-2)^{\beta+5}) \right], \ 3 \leq i \leq m-1, 1 \leq j \leq i-2.$$

$$g_{2i,2j-1} = -g_{2i-1,2j}, \ 3 \leq i \leq m-1, 1 \leq j \leq i-2. \quad (30)$$

Similarly, the upper triangular entries of the $G$ are given by

$$g_{2i-1,2i+1} = -36(1-\gamma) h^{\beta-1} \left[ \frac{1}{\Gamma(\beta + 4)} (3^{\beta+3} - 2) - \frac{1}{\Gamma(\beta + 5)} (3^{\beta+4} - 2^{\beta+3}) + \frac{4}{\Gamma(\beta + 6)} (3^{\beta+5} - 3^{\beta+7} + 6) \right] - 36\gamma h^{\beta-1} \left[ \frac{1}{\Gamma(\beta + 4)} \right], \ 1 \leq i \leq m-2,$$

$$g_{2i-1,2i+2} = 6(1-\gamma) h^{\beta} \left[ \frac{1}{\Gamma(\beta + 4)} (3^{\beta+3} - 2 \cdot 2^{\beta+3}) - \frac{1}{\Gamma(\beta + 5)} (10 \cdot 3^{\beta+4} + 8 \cdot 2^{\beta+4} - 36) + \frac{1}{\Gamma(\beta + 6)} (12 \cdot 3^{\beta+5} - 24 \cdot 2^{\beta+5}) \right] + 6\gamma h^{\beta} \left[ \frac{2}{\Gamma(\beta + 4)} - \frac{10}{\Gamma(\beta + 5)} + \frac{12}{\Gamma(\beta + 6)} \right], \ 1 \leq i \leq m-2,$$

$$g_{2i,2i+2} = (1-\gamma) h^{\beta+1} \left[ \frac{4}{\Gamma(\beta + 4)} (3^{\beta+3} + 8 \cdot 2^{\beta+3} + 18) - \frac{24}{\Gamma(\beta + 5)} (3^{\beta+4} + 4 \cdot 2^{\beta+4}) + \frac{36}{\Gamma(\beta + 6)} (3^{\beta+5} - 2) \right] - \gamma h^{\beta+1} \left[ \frac{4}{\Gamma(\beta + 4)} + \frac{24}{\Gamma(\beta + 5)} - \frac{36}{\Gamma(\beta + 6)} \right], \ 1 \leq i \leq m-2,$$

$$g_{2i,2i+1} = -g_{2i-1,2i+2}, \ 1 \leq i \leq m-2. \quad (34)$$
and

\[ g_{2j-1,2i-1} = -36(1 - \gamma)h^{\beta-1} \left[ \frac{1}{\Gamma(\beta + 4)} (i - j + 2)^{\beta+3} - 2(i - j)^{\beta+3} + (i - j - 2)^{\beta+3} - \frac{4}{\Gamma(\beta + 5)} (i - j + 2)^{\beta+4} - 2(i - j + 1)^{\beta+4} + 2(i - j - 1)^{\beta+4} - (i - j - 2)^{\beta+4} + \frac{4}{\Gamma(\beta + 6)} (i - j + 2)^{\beta+5} \right. \]

\[ \left. - 4(i - j + 1)^{\beta+5} + 6(i - j)^{\beta+5} - 4(i - j - 1)^{\beta+5} + (i - j - 2)^{\beta+5} \right], \quad 3 \leq i \leq m - 1, 1 \leq j \leq i - 2. \]

\[ g_{2j,2i} = \quad (1 - \gamma)h^{\beta+1} \left[ \frac{4}{\Gamma(\beta + 4)} (i - j + 2)^{\beta+3} + 8(i - j + 1)^{\beta+3} + 18(i - j)^{\beta+3} + 8(i - j - 1)^{\beta+3} + (i - j - 2)^{\beta+3} - \frac{24}{\Gamma(\beta + 5)} (i - j + 2)^{\beta+4} + 4(i - j + 1)^{\beta+4} - 4(i - j - 1)^{\beta+4} - (i - j - 2)^{\beta+4} + \frac{36}{\Gamma(\beta + 6)} (i - j + 2)^{\beta+5} - 2(i - j)^{\beta+5} + (i - j - 2)^{\beta+5} \right], \quad 3 \leq i \leq m - 1, 1 \leq j \leq i - 2. \]

\[ g_{2j,2i-1} = -g_{2j-1,2i}, \quad 3 \leq i \leq m - 1, 1 \leq j \leq i - 2. \]

We have evaluated all the entries of the stiffness matrix $G$.

4. A fast and efficient method. From above entries of the stiffness matrix $G$, we note that the matrix $G$ be a full coefficient matrix, and it does not possess no any regular pattern for matrix structure. A direct solver for (17) (for example Gaussian elimination) requires $O(m^3)$ of computational work and $O(m^2)$ of storage memory, which represents an extremely high computational cost. What's more, the evaluation of the matrix-vector multiplication generally also costs $O(m^2)$ operations per iteration, for any Krylov subspace iterative method.

In this section, we present a fast and efficient solution, which significantly reduces the computational cost to $O(m \log m)$ per iteration and the storage requirement to $O(m)$. 
Theorem 4.1. The stiffness matrix $G$ can be expressed as a $2 \times 2$ block Toeplitz matrix.

Proof. Let the vector $v$ denote the reindexing of the vector $u$ that corresponds to the labeling given in \[18\]

$$
v := \begin{bmatrix} u_1, u_2, \cdots, u_{m-1}, u'_1, u'_2, \cdots, u'_{m-1} \end{bmatrix}^T.
$$

(39)

Then we have

$$
v = Pu,
$$

(40)

where $P$ represents the permutation matrix that maps $u$ to $v$. The vector $g$ has the reindexing of the vector $f$ that given in \[18\] by above same fashion.

Applied the permutation matrix $P$ to the matrix $G$ that given in \[17\], $PG$ gives the matrix $G$ with rows interchanged according to the permutation vector $u$, and $GP^T$ gives the matrix $G$ with columns interchanged according to the given permutation vector. Then we have

$$
A = PG^TP.
$$

(41)

Combining (40) and (41) to obtain a new linear system

$$
Av = g,
$$

(42)

the matrix $A$ can be decomposed as

$$
A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix},
$$

(43)

where the sub-matrices $A_{1,1}, A_{1,2}, A_{2,1}, A_{2,2}$ represent the stiffness matrices corresponding to the bilinear form $B(\varphi_0^{(0)}, \varphi_0^{(0)}), B(\varphi^{(1)}_i, \varphi_0^{(0)}), B(\varphi_0^{(0)}, \varphi^{(1)}_j)$ and $B(\varphi^{(1)}_i, \varphi^{(1)}_j)$, $i, j = 1, 2, \cdots, m - 1$, respectively. All of the four sub-matrices are of order $m - 1$.

We define the sub-matrix $A_{1,1} = [a_{i,j}^{(1,1)}]_{i,j=1}^{m-1}$. Comparing to the corresponding entries of the matrix $G$, we can obtain

$$
\begin{align*}
\alpha_{i,i}^{(1,1)} &= g_{2i-1,2i-1}, \quad 1 \leq i \leq m - 1, \\
\alpha_{i+1,i}^{(1,1)} &= g_{2i+1,2i-1}, \quad 1 \leq i \leq m - 2, \\
\alpha_{i,i+1}^{(1,1)} &= g_{2i-1,2i+1}, \quad 1 \leq i \leq m - 2, \\
\alpha_{i,i}^{(1,1)} &= g_{2i-1,2i-1}, \quad 3 \leq i \leq m - 1, 1 \leq j \leq i - 2, \\
\alpha_{j,i}^{(1,1)} &= g_{2j-1,2i-1}, \quad 3 \leq i \leq m - 1, 1 \leq j \leq i - 2.
\end{align*}
$$

(44)

We can find that the entries of each descending diagonal of $A_{1,1}$ are equivalent from observation. So, $A_{1,1}$ is an $(m - 1)$-order Toeplitz matrix.

Similarly, we define the sub-matrix $A_{1,2} = [a_{i,j}^{(1,2)}]_{i,j=1}^{m-1}$, $A_{2,1} = [a_{i,j}^{(2,1)}]_{i,j=1}^{m-1}$, $A_{2,2} = [a_{i,j}^{(2,2)}]_{i,j=1}^{m-1}$. Comparing to the corresponding entries of the matrix $G$, we can obtain
In order to store the matrix \( A \), we need only store the matrix-vector multiplication \( Aw \), which needs total \( 8m - 12 \) memory units.

**Theorem 4.2.** The new stiffness matrix \( A \) can be stored in \( 8m - 12 \) memory units.

**Proof.** Based on Theorem 4.1, the coefficient matrix \( A \) is composed of four Toeplitz matrices. In order to store the matrix \( A \) we need only store

\[
\begin{bmatrix}
    a^{(1,1)}_{1,1}, a^{(1,1)}_{2,1}, \ldots, a^{(1,1)}_{m-1,1}, a^{(1,1)}_{1,2}, a^{(1,1)}_{1,3}, \ldots, a^{(1,1)}_{1,m-1} \\
    a^{(1,2)}_{1,1}, a^{(1,2)}_{2,1}, \ldots, a^{(1,2)}_{m-1,1}, a^{(1,2)}_{1,2}, a^{(1,2)}_{1,3}, \ldots, a^{(1,2)}_{1,m-1} \\
    a^{(2,1)}_{1,1}, a^{(2,1)}_{2,1}, \ldots, a^{(2,1)}_{m-1,1}, a^{(2,1)}_{1,2}, a^{(2,1)}_{1,3}, \ldots, a^{(2,1)}_{1,m-1} \\
    a^{(2,2)}_{1,1}, a^{(2,2)}_{2,1}, \ldots, a^{(2,2)}_{m-1,1}, a^{(2,2)}_{1,2}, a^{(2,2)}_{1,3}, \ldots, a^{(2,2)}_{1,m-1}
\end{bmatrix}^T,
\]

which needs total \( 8m - 12 \) memory units.

**Theorem 4.3.** For any vector \( w \in \mathbb{R}^{2m-2} \), the matrix-vector multiplication \( Aw \) can be carried out in \( O(m \log m) \) operations.
Proof. We divide \( w \) into two parts in block form as follows:

\[
\begin{align*}
w &= \begin{bmatrix} v_1^T, v_2^T \end{bmatrix}^T, \\
v_1 &= \begin{bmatrix} w_1, w_2, \cdots, w_{m-1} \end{bmatrix}^T, \\
v_2 &= \begin{bmatrix} w_m, w_{m+1}, \cdots, w_{2m-2} \end{bmatrix}^T.
\end{align*}
\] (49)

Then the matrix-vector multiplication \( Aw \) is of the form

\[
Aw = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} A_{1,1}v_1 + A_{1,2}v_2 \\ A_{2,1}v_1 + A_{2,2}v_2 \end{bmatrix}.
\] (50)

We only need to show that, for a Toeplitz matrix \( T_{m-1} \) with the following form

\[
T_{m-1} = \begin{pmatrix}
t_0 & t_{-1} & \cdots & t_{3-m} & t_{2-m} \\
t_1 & t_0 & t_{-1} & \cdots & t_{3-m} \\
\vdots & t_1 & t_0 & \ddots & \vdots \\
t_{m-3} & \vdots & \ddots & \ddots & t_{-1} \\
t_{m-2} & t_{m-3} & \cdots & t_1 & t_0 \\
\end{pmatrix},
\] (51)

the matrix-vector multiplication \( T_{m-1}v \) can be computed by means of fast Fourier transform (FFTs). First we embedded \( T_{m-1} \) into a \((2m-2) \times (2m-2)\) circulant matrix \( C_{2m-2} \) such that

\[
C_{2m-2} = \begin{bmatrix} T_{m-1} & B_{m-1} \\ B_{m-1} & T_{m-1} \end{bmatrix},
\]

\[
B_{m-1} = \begin{bmatrix}
t_{2-m} & \cdots & t_{-2} & t_{-1} \\
t_{m-2} & 0 & t_{2-m} & \ddots \\
\vdots & t_{m-2} & 0 & \ddots \\
t_2 & \ddots & \ddots & \ddots & t_{2-m} \\
t_1 & t_2 & \cdots & t_{2-m} & 0 \\
\end{bmatrix}.
\] (52)

More precisely, the matrix-vector multiplication can be computed by:

\[
C_{2m-2} \begin{bmatrix} v \\ 0 \end{bmatrix} = \begin{bmatrix} T_{m-1} & B_{m-1} \\ B_{m-1} & T_{m-1} \end{bmatrix} \begin{bmatrix} v \\ 0 \end{bmatrix} = \begin{bmatrix} T_{m-1}v \\ B_{m-1}v \end{bmatrix}.
\] (53)

Furthermore, it is well-known that circulant matrices can be diagonalized by the discrete Fourier transform matrix \( F_{2m-2} \) [7][13]

\[
C_{2m-2} = F_{2m-2}^{-1} \text{diag}(F_{2m-2}c_{2m-2}) F_{2m-2}.
\] (54)

where \( c_{2m-2} \) is the first column vector of \( C_{2m-2} \). Therefore, for any vector \( v \in \mathbb{R}^{m-1} \), the matrix-vector multiplication \( T_{m-1}v \) can be carried out in \( O(m \log m) \)
Table 1. The condition number of stiffness matrix $A$ with $K = 1, \gamma = 0.5, \beta = 0.5$.

| $h$  | $2^{-8}$     | $2^{-9}$       | $2^{-10}$      |
|------|--------------|----------------|----------------|
| Cond($A$) | 2.329917 $\times 10^6$ | 9.320396 $\times 10^6$ | 3.728232 $\times 10^7$ |

operations. So, the matrix-vector multiplication $A_{1,1}v_1$, $A_{1,2}v_2$, $A_{2,1}v_1$ and $A_{2,2}v_2$ can be evaluated in $O(m \log m)$ operations. Consequently, $Aw$ can be evaluated in $O(m \log m)$ operations.

5. A block preconditioned fast Krylov subspace method. In Theorem 4.3 we have shown that the fast matrix-vector multiplication reduces the computational cost of a Krylov subspace method from $O(m^2)$ to $O(m \log m)$ per iteration. By numerical experiments from Table 1, we find the condition number of matrix $A$ becomes larger and larger with the increasing number of grids. Thus the number of iterations becomes predictively more and more and it gives rise to nonlinear increasing of the overall computational cost. In this section we present an efficient preconditioner for the fast method.

For systems with Toeplitz blocks, we will apply two classes of preconditioners: Strang’s circulant preconditioner and Chan’s circulant preconditioner, which are outlined below. In [6] the authors study block preconditioners for solving separable block systems with Toeplitz blocks.

For a Toeplitz matrix of order $n$, which is of the form (51), the Strang preconditioner $S = S(T_n)$ is defined to be a circulant matrix obtained by copying the central diagonals of $T_n$ and reflecting them around to complete the circulant requirement. The diagonals $s_j$ of $S = [s_{k-j}]_{0 \leq k,l \leq n}$ are given by

\[
s_j = \begin{cases}
  t_j, & 0 < j \leq \lfloor n/2 \rfloor, \\
  t_{j-n}, & \lfloor n/2 \rfloor < j \leq n - 1, \\
  s_{n+j}, & 0 < -j \leq n - 1.
\end{cases}
\]

(55)

It was shown [5] that

\[
\|T_n - S\|_p = \min_{C_n} \|T_n - C_n\|_p, \quad p = 1, \infty.
\]

(56)

Chan’s preconditioner $C_F = C_F(T_n)$ is defined to be the minimizer of $\|T_n - C_n\|_F$ over all circulant matrices $C_n$. The diagonals $c_j$ of $C_F(T_n)$ are given by

\[
c_j = \begin{cases}
  \frac{(n-j)t_j + jt_{j-n}}{n}, & 0 \leq j \leq n - 1, \\
  c_{n+j}, & 0 < -j \leq n - 1.
\end{cases}
\]

(57)

We note that the stiffness matrix $A$, which is of the form (43), is a $2 \times 2$ block system with Toeplitz blocks of order $m - 1$. To motivate the development of an efficient preconditioner based on Strang’s preconditioner and T.Chen’s preconditioner, we construct a $2 \times 2$ block matrix with circulant block preconditioner $M$ for $A$ as follows

\[
M := \begin{bmatrix}
  M_{1,1} & M_{1,2} \\
  M_{2,1} & M_{2,2}
\end{bmatrix},
\]

(58)
where $M_{1,1}$, $M_{1,2}$, $M_{2,1}$ and $M_{2,2}$ are constructed as $(m - 1) \times (m - 1)$ circulant preconditioners for $A_{1,1}$, $A_{1,2}$, $A_{2,1}$ and $A_{2,2}$, respectively.

We will use a preconditioned fast conjugate gradient squared method \cite{28} to solve the system (42). In this Krylov subspace algorithm, the matrix-vector multiplications $Aw$ can be performed in $O(m \log m)$ operations by Theorem 4.3. Then, we will analyze the computational cost of the inverse of the preconditioner $M$.

**Theorem 5.1.** For any vector $w \in \mathbb{R}^{2m-2}$, the matrix-vector multiplication $M^{-1}w$ can be carried out in $O(m \log m)$ operations.

**Proof.** Since $M_{1,1}$, $M_{1,2}$, $M_{2,1}$ and $M_{2,2}$ are circulant matrices of order $(m - 1)$, by (54) the inverse of the preconditioner $M$ can been written as

\[
M = \begin{bmatrix}
M_{1,1} & M_{1,2} \\
M_{2,1} & M_{2,2}
\end{bmatrix}
= \begin{bmatrix}
F_{m-1}^{-1}A_{11}F_{m-1}^{-1} & F_{m-1}^{-1}A_{12}F_{m-1}^{-1} \\
F_{m-1}^{-1}A_{21}F_{m-1}^{-1} & F_{m-1}^{-1}A_{22}F_{m-1}^{-1}
\end{bmatrix}
= \begin{bmatrix}
F_{m-1}^{-1} & 0 \\
0 & F_{m-1}^{-1}
\end{bmatrix}
\begin{bmatrix}
\Lambda_{11} & \Lambda_{12} \\
\Lambda_{21} & \Lambda_{22}
\end{bmatrix}
\begin{bmatrix}
F_{m-1}^{-1} & 0 \\
0 & F_{m-1}^{-1}
\end{bmatrix}
= (F_{m-1}^{-1} \otimes I_2)
\begin{bmatrix}
\Lambda_{11} & \Lambda_{12} \\
\Lambda_{21} & \Lambda_{22}
\end{bmatrix}
(F_{m-1}^{-1} \otimes I_2).
\]

(59)

Here $\Lambda_{11}$, $\Lambda_{12}$, $\Lambda_{21}$ and $\Lambda_{22}$ are diagonal matrices of order $(m - 1)$, $\Lambda_{ij} = F_{m-1}^{-1}c_{m-1}$, where $c_{m-1}$ is the first column vector of $M_{i,j}$, $i,j = 1, 2$.

To prove the Theorem 5.1 we define the following matrix

\[
\Lambda = \begin{bmatrix}
\Lambda_{11} & \Lambda_{12} \\
\Lambda_{21} & \Lambda_{22}
\end{bmatrix}
= \begin{bmatrix}
\lambda_{11} & \ldots & \lambda_{1m} \\
& \ddots & \ddots & \ddots \\
& & \lambda_{11} & \ldots & \lambda_{1m} \\
\lambda_{21} & \ldots & \ldots & \lambda_{21} & \ldots & \lambda_{22} & \ldots & \ldots & \lambda_{2m}
\end{bmatrix}
\]

(60)

We apply the permutation matrix $P$ that given in (40) to interchange rows and columns of the matrix $\Lambda$, and obtain a block-diagonal matrix $\Lambda_D$ of order $m - 1$

\[
\Lambda_D = P^T \Lambda P.
\]

(61)
According to the (61), the inverse of the block-diagonal matrix $\Lambda_D$ has the following formulation

$$\Lambda_D^{-1} = \begin{bmatrix}
\begin{pmatrix}
\lambda_{11}^1 & \lambda_{12}^1 \\
\lambda_{21}^1 & \lambda_{22}^1
\end{pmatrix} & \\
\begin{pmatrix}
\lambda_{11}^2 & \lambda_{12}^2 \\
\lambda_{21}^2 & \lambda_{22}^2
\end{pmatrix} & \\
& \ddots \\
\begin{pmatrix}
\lambda_{11}^{m-1} & \lambda_{12}^{m-1} \\
\lambda_{21}^{m-1} & \lambda_{22}^{m-1}
\end{pmatrix}
\end{bmatrix}^{-1},$$

(62)

The inverse of the $2 \times 2$ matrix $S_i$ can be computed as

$$S_i^{-1} = \begin{pmatrix}
\lambda_{11}^i & \lambda_{12}^i \\
\lambda_{21}^i & \lambda_{22}^i
\end{pmatrix}^{-1} = \frac{1}{\lambda_{11}^i \lambda_{22}^i - \lambda_{12}^i \lambda_{21}^i} \begin{pmatrix}
\lambda_{22}^i & -\lambda_{12}^i \\
-\lambda_{21}^i & \lambda_{22}^i
\end{pmatrix}.$$  

(63)

So the matrix-vector multiplication $\Lambda^{-1}w$ can be carried out in $O(m)$ operations for any vector $w \in \mathbb{R}^{2m-2}$. Note that

$$M^{-1} = (F_{m-1}^{-1} \otimes I_2)\Lambda^{-1}(F_{m-1} \otimes I_2).$$  

(64)

Consequently, $M^{-1}w$ can be evaluated in $O(m \log m)$ operations.

6. **Numerical experiments.** we consider the numerical simulation of the Dirichlet boundary-value problem of a two-sided fractional differential equation [1] with $K = 1$, $\gamma = 0.5$, $u_r = 0$, $u_l = 0$, and the source term given by

$$f(x) = -64 \left[ \frac{6}{\Gamma(\beta + 2)}(ax^{\beta+1} + (1-x)^{\beta+1}) - \frac{12}{\Gamma(\beta + 3)}(ax^{\beta+2} + (1-x)^{\beta+2}) + \frac{60}{\Gamma(\beta + 4)}(ax^{\beta+3} + (1-x)^{\beta+3}) - \frac{120}{\Gamma(\beta + 5)}(ax^{\beta+4} + (1-x)^{\beta+4}) \right].$$

The true solution of the above problem is given by

$$u(x) = 64x^3(1-x)^3, \quad x \in [0, 1].$$

In the numerical experiments, we investigate the performance of the cubic Hermite finite element method [16] to solve the two-sided fractional elliptic differential problem on $\Omega_h$. We use the numerical solutions by Gaussian elimination as a benchmark. We then solve the linear system by the conjugate gradient squared (CGS) methods, by the fast conjugate gradient squared (FCGS) method in which the matrix-vector multiplication of $Aw$ is carried out via Theorem 4.3 by the preconditioned fast conjugate gradient squared method with Chan’ preconditioner (CFCGS) and the preconditioned fast conjugate gradient method with the Strang’ preconditioner (SFCGS) presented in Section 5.
Table 2. The $L^2$ error and $H^{1-\frac{\beta}{2}}$ error of the cubic Hermite element method with $\beta = 0.1, 0.5, 0.9$.

| $\beta$ | $h$ | DOF | $\|u_h - u\|_{L^2}$ | order | $\|u_h - u\|_{H^{1-\frac{\beta}{2}}}$ | order |
|---------|-----|-----|-----------------|-------|-----------------|-------|
| 0.5     | $2^{-3}$ | 16  | $4.872892 \times 10^{-4}$ |       | $2.512934 \times 10^{-3}$ |       |
|         | $2^{-4}$ | 32  | $3.573631 \times 10^{-5}$ | 3.7963 | $2.576521 \times 10^{-4}$ | 3.2858 |
|         | $2^{-5}$ | 64  | $2.236217 \times 10^{-6}$ | 3.9982 | $2.481970 \times 10^{-5}$ | 3.3758 |
|         | $2^{-6}$ | 128 | $1.342295 \times 10^{-7}$ | 4.0582 | $2.249708 \times 10^{-6}$ | 3.4636 |
|         | $2^{-7}$ | 256 | $7.909661 \times 10^{-9}$ | 4.0849 | $2.033970 \times 10^{-7}$ | 3.4673 |
| 0.1     | $2^{-3}$ | 16  | $7.851432 \times 10^{-4}$ |       | $8.011606 \times 10^{-3}$ |       |
|         | $2^{-4}$ | 32  | $6.192597 \times 10^{-5}$ | 3.6643 | $1.304843 \times 10^{-3}$ | 2.6352 |
|         | $2^{-5}$ | 64  | $4.208159 \times 10^{-6}$ | 3.8792 | $1.805192 \times 10^{-4}$ | 2.8536 |
|         | $2^{-6}$ | 128 | $2.713900 \times 10^{-7}$ | 3.9547 | $2.295487 \times 10^{-5}$ | 2.9752 |
|         | $2^{-7}$ | 256 | $1.794745 \times 10^{-8}$ | 3.9185 | $2.080895 \times 10^{-6}$ | 3.0311 |
| 0.9     | $2^{-3}$ | 16  | $3.622149 \times 10^{-4}$ |       | $4.846137 \times 10^{-4}$ |       |
|         | $2^{-4}$ | 32  | $2.767086 \times 10^{-5}$ | 3.7104 | $4.110739 \times 10^{-5}$ | 3.5593 |
|         | $2^{-5}$ | 64  | $1.826287 \times 10^{-6}$ | 3.9213 | $3.012015 \times 10^{-6}$ | 3.7705 |
|         | $2^{-6}$ | 128 | $1.180438 \times 10^{-7}$ | 3.9515 | $2.132241 \times 10^{-7}$ | 3.8202 |
|         | $2^{-7}$ | 256 | $7.376961 \times 10^{-9}$ | 4.0001 | $1.415327 \times 10^{-8}$ | 3.9131 |

In the numerical experiment we compute the errors of $\|u_h - u\|_{H^{1-\frac{\beta}{2}}}$ in the energy norms

$$\|u_h - u\|_{H^{1-\frac{\beta}{2}}} \leq C\|u_h - u\|_{H^0}^\beta \|u_h - u\|_{H^1}^{1-\beta}.$$  

From table 2, we observe that the cubic hermite finite element methods have a convergence order of 4 in the $L^2$ norm and a convergence order of $3 + \beta$ in the fractional energy norm. From table 3 and 4 we present the numerical results and reach the following observations: (i) The CGS method consumes much more CPU time than Gaussian elimination. The reason is the number of iterations exceeds the number of grid points $m$, that means the computational cost has exceeded $O(m^3)$ operations. (ii) The FCGS method requires only $O(m \log m)$ operations per iteration and significantly reduces the over CPU time. But the number of iterations has not changed, when the number of grid points is huge, iterative methods will diverge. (iii) Both the CFCSGS and the SFCGS methods significantly reduce the number of iterations in the FCGS method, consequently, these preconditioned FCGS methods further significantly reduce the CPU time, due to the significant reduction of the number of iterations.

**Acknowledgments.** This work was supported in part by the OSD/ARO MURI Grant W911NF-15-1-0562, by the National Natural Science Foundation of China under Grants 11471194, 11571115 and 91630207, by the National Science Foundation under Grant DMS-1620194, by the National Science and technology major projects of China under Grants 2011ZX05052 and 2011ZX05011-004, by Natural Science Foundation of Shandong Province of China under Grant ZR2011AM015, by Taishan Scholars Program of Shandong Province of China, and by the China Scholarship Council (File No. 201606220127). The authors would like to express their sincere thanks to the referees for their very helpful comments and suggestions, which greatly improved the quality of this paper.
Table 3. Performance of Gauss, CGS methods with $\beta = 0.5$

| $h$  | CPU(s) | Itr. # | CPU(s) | Itr. # |
|------|--------|--------|--------|--------|
| $2^{-6}$ | 0.0916 |        | 0.2103 | 137    |
| $2^{-7}$ | 0.4865 |        | 1.5369 | 272    |
| $2^{-8}$ | 3.5858 |        | 9.8734 | 415    |
| $2^{-9}$ | 24.4774|        | 62.7682| 665    |
| $2^{-10}$| 186.9874|      | 391.5595| 899    |
| $2^{-11}$| 1485.1450|     | 2731.0751| 1676   |
| $2^{-12}$| out of memory | | out of memory | |

Table 4. Performance of FCGS, SFCGS and PFCGS methods with $\beta = 0.5$

| $h$  | FCGS | SFCGS | CFPGS |
|------|------|-------|-------|
|      | CPU(s) | Itr. # | CPU(s) | Itr. # | CPU(s) | Itr. # |
| $2^{-6}$ | 0.0832 | 137     | 0.0305 | 7     | 0.0341 | 9     |
| $2^{-7}$ | 0.1809 | 272     | 0.0358 | 7     | 0.0419 | 10    |
| $2^{-8}$ | 0.2890 | 415     | 0.0503 | 7     | 0.0445 | 11    |
| $2^{-9}$ | 0.9907 | 665     | 0.0680 | 8     | 0.0653 | 12    |
| $2^{-10}$| 2.4525 | 899     | 0.0932 | 8     | 0.1046 | 14    |
| $2^{-11}$| 16.0752| 1676    | 0.1536 | 9     | 0.2422 | 16    |
| $2^{-12}$| 26.4751| 6421    | 0.1914 | 9     | 0.5955 | 19    |
| $2^{-13}$| N/A   | > 30,000| 0.6319| 10    | 1.4107 | 22    |

REFERENCES

[1] T. Basu and H. Wang, A fast second-order finite difference method for space-fractional diffusion equations, Int. J. Numer. Anal. Mod., 9 (2012), 658–666.
[2] D. Benson, S. W. Wheatcraft and M. M. Meerschaert, The fractional-order governing equation of Lévy motion, Water Resour. Res., 36 (2000), 1413–1423.
[3] W. Bu, Y. Tang and J. Yang, Galerkin finite element method for two-dimensional Riesz space fractional diffusion equations J. Comput. Phys., 276 (2014), 26–38.
[4] R. H. Chan and M. K. Ng, Conjugate gradient methods for Toeplitz systems SIAM Review, 38 (1996), 427–482.
[5] R. H. Chan, Circulant preconditioners for Hermitian Toeplitz systems SIAM Journal on Matrix Analysis and Applications, 10 (1989), 542–550.
[6] R. H. Chan and X. Q. Jin, A family of block preconditioners for block systems SIAM journal on scientific and statistical computing, 13 (1992), 1218–1235.
[7] P. J. Davis, Circulant Matrices, Wiley-Intersciences, New York, 1979.
[8] W. Deng, Finite element method for the space and time fractional Fokker-Planck equation SIAM Journal on Numerical Analysis, 47 (2008), 204–226.
[9] K. Diethelm and A. D. Freed, An efficient algorithm for the evaluation of convolution integrals Computers and Mathematics with Applications 51 (2006), 51–72.
[10] N. Du and H. Wang, A fast finite element method for space-fractional dispersion equations on bounded domains in $\mathbb{R}^d$ SIAM J. Sci. Comput., 37 (2015), A1614–A1645.
[11] V. J. Ervin and J. P. Roop, Variational formulation for the stationary fractional advection dispersion equation Numer. Methods for Partial Differential Equations, 22 (2005), 558–576.
[12] V. J. Ervin and J. P. Roop, Variational Solution of Fractional Advection Dispersion Equations on Bounded Domains in $\mathbb{R}^d$, *Numer. Methods for Partial Differential Equations*, 23 (2007), 256–281.

[13] R. M. Gray, Toeplitz and circulant matrices: A review, *Communications and Information Theory*, 2 (2016), 155–239.

[14] J. Jia and H. Wang, A preconditioned fast finite volume scheme for a fractional differential equation discretized on a locally refined composite mesh, *J. Comput. Phys.*, 209 (2015), 842–862.

[15] Y. Jiang and X. Xu, Multigrid methods for space fractional partial differential equations, *J. Comput. Phys.*, 302 (2015), 374–392.

[16] S. L. Lei and H. W. Sun, A circulant preconditioner for fractional diffusion equations, *J. Comput. Phys.*, 242 (2013), 715–725.

[17] C. Li, Z. Zhao and Y. Q. Chen, Numerical approximation of nonlinear fractional differential equations with subdiffusion and superdiffusion, *Computers and Mathematics with Applications*, 62 (2011), 855–875.

[18] X. Li and C. Xu, The existence and uniqueness of the weak solution of the space-time fractional diffusion equation and a spectral method approximation, *Commun. Comput. Phys.*, 8 (2010), 1016–1051.

[19] F. R. Lin, S. W. Yang and X. Q. Jin, Preconditioned iterative methods for fractional diffusion equation, *J. Comput. Phys.*, 256 (2014), 109–117.

[20] Y. Lin and C. Xu, Finite difference/spectral approximations for the time-fractional diffusion equation, *J. Comput. Phys.*, 225 (2007), 1533–1552.

[21] F. Liu, V. Anh and I. Turner, Numerical solution of the space fractional Fokker-Planck equation, *J. Comput. Appl. Math.*, 166 (2014), 209–219.

[22] C. W. Lv and C. Xu, Improved error estimates of a finite difference/spectral method for time-fractional diffusion equations, *Int. J. Numer. Anal. Mod.*, 12 (2015), 384–400.

[23] M. M. Meerschaert, H. P. Scheffler and C. Tadjeran, Finite difference methods for two-dimensional fractional dispersion equation, *J. Comput. Phys.*, 211 (2006), 249–261.

[24] M. M. Meerschaert and C. Tadjeran, Finite difference approximations for fractional advection-dispersion flow equations, *J. Comput. Appl. Math.*, 172 (2004), 65–77.

[25] R. Metzler and J. Klafter, The random walk’s guide to anomalous diffusion: A fractional dynamics approach, *Phys. Rep.*, 339 (2000), 1–77.

[26] H. K. Pang and H. W. Sun, Multigrid method for fractional diffusion equations, *J. Comput. Phys.*, 231 (2012), 693–703.

[27] I. Podlubny, *Fractional Differential Equations*, Academic Press, New York, 1999.

[28] Y. Saad, *Iterative Methods for Sparse Linear Systems* 2nd ed., SIAM, Rhode Island, 2003.

[29] H. G. Sun, W. Chen and Y. Q. Chen, Fractional differential models for anomalous diffusion, *Physica A: Statistical Mechanics and its Applications*, 389 (2010), 2719–2724.

[30] H. Tian, H. Wang and W. Q. Wang, An efficient collocation method for a non-local diffusion model, *Int. J. Numer. Anal. Mod.*, 10 (2013), 815–825.

[31] P. Vabishchevich, Numerical solution of nonstationary problems for a convection and a space-fractional diffusion equation, *Int. J. Numer. Anal. Mod.*, 13 (2016), 296–309.

[32] H. Wang and D. P. Yang, Wellposedness of variable-coefficient conservative fractional elliptic differential equations, *SIAM Journal on Numerical Analysis*, 51 (2013), 1088–1107.

[33] H. Wang, D. P. Yang and S. F. Zhu, Accuracy of finite element methods for boundary-value problems of steady-state fractional diffusion equations, *Journal of Scientific Computing*, 70 (2017), 429–449.

[34] H. Wang and X. H. Zhang, A high-accuracy preserving spectral Galerkin method for the Dirichlet boundary-value problem of variable-coefficient conservative fractional diffusion equations, *J. Comput. Phys.*, 281 (2015), 67–81.

[35] H. Wang and T. Basu, A fast finite difference method for two-dimensional space-fractional diffusion equations, *SIAM J. Sci. Comput.*, 34 (2012), A2444–A2458.

[36] H. Wang and N. Du, A superfast-preconditioned iterative method for steady-state space-fractional diffusion equations, *J. Comput. Phys.*, 240 (2013), 49–57.
[37] H. Wang, K. Wang and T. Sircar, A direct $O(N \log^2 N)$ finite difference method for fractional diffusion equations, *J. Comput. Phys.*, **229** (2010), 8095–8104.

[38] L. L. Wei, Y. N. He and Y. Zhang, Numerical analysis of the fractional seventh-order KdV equation using an implicit fully discrete local discontinuous Galerkin method, *Int. J. Numer. Anal. Mod.*, **10** (2013), 430–444.

Received August 2016; revised April 2017.

E-mail address: mathzm@aliyun.com
E-mail address: aijie@sdu.edu.cn
E-mail address: hwang@math.sc.edu