East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.160418.190518

Vol. **9**, No. 1, pp. 28-44
February 2019

A Preconditioned Fast Finite Volume Method for Distributed-Order Diffusion Equation and Applications

Hongfei Fu[1,*], Huan Liu[2] and Xiangcheng Zheng[3]

[1]*College of Science, China University of Petroleum (East China), Qingdao, Shandong 266580, China.*
[2]*School of Mathematics, Shandong University, Jinan, Shandong 250100, China.*
[3]*Department of Mathematics, University of South Carolina, Columbia, South Carolina 29208, USA.*

Received 16 April 2018; Accepted (in revised version) 29 May 2018.

Abstract. A Crank-Nicolson finite volume scheme for the modeling of the Riesz space distributed-order diffusion equation is proposed. The corresponding linear system has a symmetric positive definite Toeplitz matrix. It can be efficiently stored in $\mathcal{O}(NK)$ memory. Moreover, for the finite volume scheme, a fast version of conjugate gradient (FCG) method is developed. Compared with the Gaussian elimination method, the computational complexity is reduced from $\mathcal{O}(MN^3+NK)$ to $\mathcal{O}(l_AMN\log N+NK)$, where l_A is the average number of iterations at a time level. Further reduction of the computational cost is achieved due to use of a circulant preconditioner. The preconditioned fast finite volume method is combined with the Levenberg-Marquardt method to identify the free parameters of a distribution function. Numerical experiments show the efficiency of the method.

AMS subject classifications: 35R11, 65F08, 65F10, 65M08, 65T50

Key words: Distributed-order diffusion equation, finite volume method, fast conjugate gradient method, circulant preconditioner, parameter identification.

1. Introduction

In the past few decades, fractional partial differential equations (PDEs) have been widely used to model complex physical phenomena with long-range time memory and spatial interactions [3, 7, 27, 29, 36]. Systematic introduction to fractional calculus and fractional differential equations can be found in Refs. [33, 35].

*Corresponding author. *Email addresses:* hongfeifu@upc.edu.cn (H. Fu), liuhmath@163.com (H. Liu), xz3@math.sc.edu (X. Zheng)

Unlike the PDEs of integer-order, analytical solutions of fractional PDEs are rarely available, so that numerical methods have to be employed — cf. Refs. [8,9,13,17,18,24–26,30, 31,39,44,46]. However, the nonlocal nature of fractional differential operators leads to dense stiffness matrices and/or long tails in the time direction. Thus, traditional approximation methods using numerical discretisation, have a high computational cost, especially in multidimensional situations. In 2010, Wang *et al.* [41] proposed a direct fast finite difference method for space-fractional diffusion equations, which retained the same accuracy as regular finite difference methods but required only $\mathcal{O}(N)$ memory storage with the computational cost $\mathcal{O}(N\log^2 N)$. After that, fast solution methods have been extended to various fractional PDEs, including space-fractional PDEs [19,21,34,42,43], time-fractional PDEs [20,45] and space-time-fractional PDEs [10,11,15].

Recently, Li *et al.* [23] considered a finite volume method for a distributed-order space-fractional model and proved the unconditional stability, convergence and the second order accuracy of the method, both in space and time. Here, we want to develop a preconditioned fast finite volume method for a distributed-order space-fractional model and apply it to an inverse problem to determine the free parameters of the corresponding distribution function. Starting with the investigation of the matrix structure of the method and its efficient storage, we then develop a preconditioned fast conjugate gradient (PFCG) method based on a circulant preconditioner and fast matrix-vector multiplication. Numerical experiments show a largely reduced CPU usage, hence the method is well suited to large-scale modeling and simulation. Let us recall that various application problems require the identification of free parameters in the corresponding mathematical models — e.g. given experimental data, determine a parameter by minimising the difference between the numerical output and experimental data. Such procedures are usually considered as inverse problems [6,12], and in this work we develop a PFCG-based optimisation algorithm, which is based on the Levenberg-Marquardt iterative method with the Armijo rule. It is numerically tested, including the situations when the observation data contaminated by random noise. The numerical tests show the efficiency and accuracy of the method proposed.

The rest of the paper is organised as follows. In Section 2, we consider the Riesz space distributed-order diffusion equation and describe the corresponding finite volume approximations. Section 3 discusses the structure of the finite volume scheme matrix and its efficient storage. In Section 4, we develop a PFCG iterative method for the finite volume scheme and test its efficiency. Section 5 is devoted to the identification of free parameters for a distributed-order diffusion equation with the distribution function (2.2). We note that numerical experiments show the strong performance of the method. Our concluding remarks are in Section 6.

2. A Diffusion Equation and Finite Volume Approximations

In this paper, we develop a preconditioned fast finite volume method for the Riesz space distributed-order diffusion equation

$$\begin{aligned}&\frac{\partial u(x,t)}{\partial t}-\int_1^2 P(\alpha)\frac{\partial^\alpha u(x,t)}{\partial|x|^\alpha}d\alpha=f(x,t),\quad (x,t)\in(0,1)\times(0,T],\\&u(x,0)=u_o(x),\quad x\in(0,1),\quad u(0,t)=u(1,t)=0,\quad t\in(0,T],\end{aligned}\tag{2.1}$$

where $f(x,t)$ is the source or sink term, $P(\alpha)$ a non-negative weight function satisfying the conditions

$$P(\alpha)\geq 0,\quad P(\alpha)\neq 0,\quad \alpha\in(1,2),\quad 0<\int_1^2 P(\alpha)d\alpha<\infty,$$

and $u_0(x)$ a given function. We note that, according to [38], the function $P(\alpha)$ can be chosen as

$$P(\alpha)=l^{\alpha-2}D\left(A_1\delta(\alpha-\widetilde{\alpha}_1)+A_2\delta(\alpha-\widetilde{\alpha}_2)\right),\tag{2.2}$$

where l and D are positive constants, $A_1>0$, $A_2>0$ and $0<\widetilde{\alpha}_1<\widetilde{\alpha}_2<2$. If $\widetilde{\alpha}_1\neq 1$ and $\widetilde{\alpha}_2\neq 1$, then $P(1)=0$. Throughout the paper we also assume that $P(1)=0$. Moreover, $\partial^\alpha u(x,t)/\partial|x|^\alpha$ refers to the Riesz fractional derivative [37], defined by

$$\frac{\partial^\alpha u(x,t)}{\partial|x|^\alpha}:=\begin{cases}-\dfrac{1}{2\cos(\pi\alpha/2)}\left(\dfrac{\partial^\alpha u(x,t)}{\partial_+x^\alpha}+\dfrac{\partial^\alpha u(x,t)}{\partial_-x^\alpha}\right), & 1<\alpha<2,\\ \dfrac{\partial^2 u(x,t)}{\partial x^2}, & \alpha=2,\end{cases}$$

where

$$\begin{aligned}\frac{\partial^\alpha u(x,t)}{\partial_+x^\alpha}&=\frac{1}{\Gamma(n-\alpha)}\frac{\partial^n}{\partial x^n}\int_0^x\frac{u(s,t)}{(x-s)^{\alpha-n+1}}ds,\\ \frac{\partial^\alpha u(x,t)}{\partial_-x^\alpha}&=\frac{(-1)^n}{\Gamma(n-\alpha)}\frac{\partial^n}{\partial x^n}\int_x^1\frac{u(s,t)}{(s-x)^{\alpha-n+1}}ds\end{aligned}$$

are the left and right Riemann-Liouville fractional derivatives [35] and $\Gamma(\cdot)$ is the Gamma function.

Now we discretise the distributed-order diffusion model (2.1), starting with the integral term. Let K be a positive integer and $\xi_k=1+k\rho$, $\rho=1/K$, $k=0,1,\cdots,K$ the corresponding uniform partition of the interval $[1,2]$. Setting $\alpha_k:=(\xi_{k-1}+\xi_k)/2$ and using the midpoint quadrature rule, we write

$$\int_1^2 P(\alpha)\frac{\partial^\alpha u(x,t)}{\partial|x|^\alpha}d\alpha=\sum_{k=1}^K P(\alpha_k)\frac{\partial^{\alpha_k}u(x,t)}{\partial|x|^{\alpha_k}}\rho+\mathcal{O}(\rho^2).\tag{2.3}$$

Analogously, for a positive integer M, we define the uniform partition $t_m=m\tau$, $\tau=T/M$, $m=0,1,\cdots,M$ of the interval $[0,T]$ and, taking into account the representation (2.3), we approximate the model (2.1) by the following semi-discrete Crank-Nicolson scheme

$$\begin{aligned}\frac{u(x,t_m)-u(x,t_{m-1})}{\tau}=&\frac{\rho}{2}\sum_{k=1}^K P(\alpha_k)\left(\frac{\partial^{\alpha_k}u(x,t_m)}{\partial|x|^{\alpha_k}}+\frac{\partial^{\alpha_k}u(x,t_{m-1})}{\partial|x|^{\alpha_k}}\right)\\&+\frac{1}{2}\left(f(x,t_m)+f(x,t_{m-1})\right)+\mathcal{O}(\rho^2+\tau^2),\end{aligned}\tag{2.4}$$

which has the second-order accuracy in time for sufficiently smooth functions $u(x,t)$.

Finally, we choose another positive integer N and consider the spatial partition $x_i = ih$, $h = 1/(N+1)$, $i = 0,1,\cdots,N+1$ of the interval $[0,1]$. Setting $x_{i-1/2} := (x_{i-1}+x_i)/2$, we integrate the Eq. (2.4) over the intervals $[x_{i-1/2}, x_{i+1/2}]$, $1 \le i \le N$ thus obtaining

$$\begin{aligned}
&\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_m)dx - \frac{\rho\tau}{2}\sum_{k=1}^{K} a_k \left(\frac{\partial^{\beta_k} u(x,t_m)}{\partial_+ x^{\beta_k}} - \frac{\partial^{\beta_k} u(x,t_m)}{\partial_- x^{\beta_k}} \right)\Big|_{x_{i-1/2}}^{x_{i+1/2}} \\
&= \int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_{m-1})dx + \frac{\rho\tau}{2}\sum_{k=1}^{K} a_k \left(\frac{\partial^{\beta_k} u(x,t_{m-1})}{\partial_+ x^{\beta_k}} - \frac{\partial^{\beta_k} u(x,t_{m-1})}{\partial_- x^{\beta_k}} \right)\Big|_{x_{i-1/2}}^{x_{i+1/2}} \\
&\quad + \frac{\tau}{2}\int_{x_{i-1/2}}^{x_{i+1/2}} (f(x,t_m) + f(x,t_{m-1}))\,dx + \mathcal{O}(\tau h(\rho^2+\tau^2))
\end{aligned} \tag{2.5}$$

with $\beta_k = \alpha_k - 1 \in (0,1)$ and $a_k = -P(\alpha_k)/(2\cos(\pi\alpha_k/2)) > 0$.

Let $\mathcal{S}_h(0,1)$ be the space of continuous functions, linear on each subinterval (x_{i-1}, x_i), vanishing at the points $x = 0$ and $x = 1$. Then the finite volume solution $u_h^m(x) := u_h(x,t_m) \in \mathcal{S}_h(0,1)$ for the model (2.1) can be written in the form

$$u_h^m(x) = \sum_{i=1}^{N} u_i^m \phi_i(x), \tag{2.6}$$

where ϕ_i, $1 \le i \le N$, are the nodal basis function defined by

$$\phi_i(x) = \begin{cases} \dfrac{x - x_{i-1}}{h}, & x \in [x_{i-1}, x_i], \\ \dfrac{x_{i+1} - x}{h}, & x \in [x_i, x_{i+1}], \\ 0, & \text{elsewhere}, \end{cases}$$

and u_i^m are the finite volume approximation of $u(x_i,t_m)$ for $i = 1,2,\cdots,N$ and $m = 1,2,\cdots,M$. In particular, $u_i^0 = u_o(x_i)$ for $i = 1,2,\cdots,N$.

Replacing $u(x,t_m)$ in (2.5) by $u_h^m(x)$ of (2.6) and omitting the truncation-error term, we arrive at the fully-discrete finite volume scheme of the model (2.1) — viz.

$$\begin{aligned}
&\sum_{j=1}^{N} u_j^m \int_{x_{i-1/2}}^{x_{i+1/2}} \phi_j(x)dx - \frac{\rho\tau}{2}\sum_{j=1}^{N} u_j^m \sum_{k=1}^{K} a_k \left(\frac{\partial^{\beta_k}\phi_j(x)}{\partial_+ x^{\beta_k}} - \frac{\partial^{\beta_k}\phi_j(x)}{\partial_- x^{\beta_k}} \right)\Big|_{x_{i-1/2}}^{x_{i+1/2}} \\
&= \sum_{j=1}^{N} u_j^{m-1} \int_{x_{i-1/2}}^{x_{i+1/2}} \phi_j(x)dx + \frac{\rho\tau}{2}\sum_{j=1}^{N} u_j^{m-1} \sum_{k=1}^{K} a_k \left(\frac{\partial^{\beta_k}\phi_j(x)}{\partial_+ x^{\beta_k}} - \frac{\partial^{\beta_k}\phi_j(x)}{\partial_- x^{\beta_k}} \right)\Big|_{x_{i-1/2}}^{x_{i+1/2}} \\
&\quad + \frac{\tau}{2}\int_{x_{i-1/2}}^{x_{i+1/2}} (f(x,t_m) + f(x,t_{m-1}))\,dx.
\end{aligned} \tag{2.7}$$

Consider now the vectors

$$\mathbf{u}^m := \left(u_1^m, u_2^m, \cdots, u_N^m\right)^\top, \quad \mathbf{F}^m := \left(F_1^m, F_2^m, \cdots, F_N^m\right)^\top, \quad m = 1,2,\cdots,M,$$

where

$$F_i^m = \frac{\tau}{2}\int_{x_{i-1/2}}^{x_{i+1/2}} (f(x,t_m)+f(x,t_{m-1}))\,dx, \quad 1 \le i \le N.$$

The finite volume scheme (2.7) can be written as the system of linear algebraic equations

$$(\mathbf{A}-\eta\mathbf{B})\,\mathbf{u}^m = (\mathbf{A}+\eta\mathbf{B})\,\mathbf{u}^{m-1}+\mathbf{F}^m, \quad \eta := \frac{\rho\tau}{2}, \tag{2.8}$$

where $\mathbf{A}=(\mathbf{A}_{i,j})_{i,j=1}^N$ and $\mathbf{B}=(\mathbf{B}_{i,j})_{i,j=1}^N$ are, respectively, mass and stiffness matrices with entries

$$\begin{aligned} \mathbf{A}_{i,j} &= \int_{x_{i-1/2}}^{x_{i+1/2}} \phi_j(x)dx, \\ \mathbf{B}_{i,j} &= \sum_{k=1}^{K} a_k \left(\frac{\partial^{\beta_k}\phi_j(x)}{\partial_+ x^{\beta_k}} - \frac{\partial^{\beta_k}\phi_j(x)}{\partial_- x^{\beta_k}} \right)\Big|_{x_{i-1/2}}^{x_{i+1/2}}. \end{aligned} \tag{2.9}$$

It was established in [23, Theorems 4,5] that the finite volume scheme (2.7) is unconditionally stable and converges as $\mathcal{O}(\rho^2+\tau^2+h^2)$. Although it resembles the scheme for the classical second-order diffusion equation, for nodal basis functions $\phi_i(x)$ the derivatives $\partial^{\beta_k}\phi_j(x)/\partial_\pm x^{\beta_k}$ are supported on the whole interval $[0,1]$, so that the corresponding stiffness matrix $\mathbf{B}$ in (2.9) is full. The inversion of such matrices generally requires $\mathcal{O}(N^3)$ operations and $\mathcal{O}(N^2)$ memory for the storage, which is computationally costly, especially in large-scale modeling and simulations.

3. Matrix Structure and Efficient Storage

To develop an efficient and faithful approximation method for the system (2.8), we have to investigate the structure of the matrices $\mathbf{A}$ and $\mathbf{B}$.

Direct calculations show that

$$\int_{x_{i-1/2}}^{x_{i+1/2}} \phi_j(x)dx = \begin{cases} h/8, & |j-i|=1, \\ 3h/4, & j=i, \\ 0, & \text{otherwise}, \end{cases} \tag{3.1}$$

and

$$\begin{aligned} \frac{\partial^{\beta_k}\phi_j(x)}{\partial_+ x^{\beta_k}}\Big|_{x=x_{i-1/2}} &= \chi_k \begin{cases} 0, & j>i, \\ s_{i-j}^{(\beta_k)}, & j\le i, \end{cases} \\ \frac{\partial^{\beta_k}\phi_j(x)}{\partial_+ x^{\beta_k}}\Big|_{x=x_{i+1/2}} &= \chi_k \begin{cases} 0, & j>i+1, \\ s_{i-j+1}^{(\beta_k)}, & j\le i+1, \end{cases} \\ \frac{\partial^{\beta_k}\phi_j(x)}{\partial_- x^{\beta_k}}\Big|_{x=x_{i-1/2}} &= \chi_k \begin{cases} s_{j-i+1}^{(\beta_k)}, & j\ge i-1, \\ 0, & j<i-1, \end{cases} \\ \frac{\partial^{\beta_k}\phi_j(x)}{\partial_- x^{\beta_k}}\Big|_{x=x_{i+1/2}} &= \chi_k \begin{cases} s_{j-i}^{(\beta_k)}, & j\ge i, \\ 0, & j<i, \end{cases} \end{aligned} \tag{3.2}$$

where $\chi_k := 1/h^{\beta_k}\Gamma(2-\beta_k)$ and

$$s_i^{(\beta_k)} = \begin{cases} \left(\frac{1}{2}\right)^{1-\beta_k}, & i=0, \\ \left(\frac{3}{2}\right)^{1-\beta_k} - 2\left(\frac{1}{2}\right)^{1-\beta_k}, & i=1, \\ \left(i+\frac{1}{2}\right)^{1-\beta_k} - 2\left(i-\frac{1}{2}\right)^{1-\beta_k} + \left(i-\frac{3}{2}\right)^{1-\beta_k}, & 2\le i\le N. \end{cases} \tag{3.3}$$

Using the representations (3.1)-(3.3), we conclude that **A** is a tridiagonal matrix — viz.

$$\mathbf{A} = \frac{h}{8}\text{tridiag}(1,6,1), \tag{3.4}$$

whereas **B** has the entries

$$\mathbf{B}_{i,j} = \begin{cases} \sum_{k=1}^{K} \hat{a}_k \left(s_{i-j+1}^{(\beta_k)} - s_{i-j}^{(\beta_k)}\right), & j < i-1, \\ \sum_{k=1}^{K} \hat{a}_k \left(s_2^{(\beta_k)} - s_1^{(\beta_k)} + s_0^{(\beta_k)}\right), & j = i-1, \\ 2\sum_{k=1}^{K} \hat{a}_k \left(s_1^{(\beta_k)} - s_0^{(\beta_k)}\right), & j = i, \\ \sum_{k=1}^{K} \hat{a}_k \left(s_2^{(\beta_k)} - s_1^{(\beta_k)} + s_0^{(\beta_k)}\right), & j = i+1, \\ \sum_{k=1}^{K} \hat{a}_k \left(s_{j-i+1}^{(\beta_k)} - s_{j-i}^{(\beta_k)}\right), & j > i+1 \end{cases} \tag{3.5}$$

and

$$\hat{a}_k = -\frac{P(\alpha_k)}{2h^{\beta_k}\Gamma(2-\beta_k)\cos(\pi\alpha_k/2)} > 0.$$

Moreover, using the notation

$$q_i = \begin{cases} 2\sum_{k=1}^{K} \hat{a}_k \left(s_1^{(\beta_k)} - s_0^{(\beta_k)}\right), & i=0, \\ \sum_{k=1}^{K} \hat{a}_k \left(s_2^{(\beta_k)} - s_1^{(\beta_k)} + s_0^{(\beta_k)}\right), & i=1, \\ \sum_{k=1}^{K} \hat{a}_k \left(s_{i+1}^{(\beta_k)} - s_i^{(\beta_k)}\right), & i\ge 2 \end{cases} \tag{3.6}$$

we write the stiffness matrix $\mathbf{B}$ of (3.5) in the form

$$\mathbf{B}=\begin{pmatrix} q_0 & q_1 & q_2 & \cdots & q_{N-2} & q_{N-1} \\ q_1 & q_0 & q_1 & \ddots & \ddots & q_{N-2} \\ q_2 & q_1 & q_0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & q_2 \\ q_{N-2} & \ddots & \ddots & \ddots & q_0 & q_1 \\ q_{N-1} & q_{N-2} & \cdots & \cdots & q_1 & q_0 \end{pmatrix}, \tag{3.7}$$

thus noting that this is a Toeplitz symmetric matrix.

The tridiagonal matrix $\mathbf{A}$ requires at most $3N-2=\mathcal{O}(N)$ memory to store and for Toeplitz matrix $\mathbf{B}$ we only have to store N entries q_i. In addition, the Eq. (3.6) implies that the terms $\hat{a}_k$, $1\le k\le K$ and $s_i^{(\beta_k)}$, $i=0,\cdots,N$ and $k=1,\cdots,K$ need, respectively, $\mathcal{O}(K)$ and $\mathcal{O}(NK)$ storage memory, whereas $\mathbf{u}^m$, $\mathbf{u}^{m-1}$, and $\mathbf{F}^m$ require $\mathcal{O}(N)$ memory. These considerations lead to the following conclusion.

Theorem 3.1. *The finite volume method* (2.7) *requires* $\mathcal{O}(NK)$ *storage memory.*

Let us now consider the matrix $\mathbf{A}-\eta\mathbf{B}$ of the system (2.8) in a more detail.

Theorem 3.2. *The matrix* $\mathbf{A}-\eta\mathbf{B}$ *of the Eq.* (2.8) *is symmetric and positive definite.*

Proof. It is clear that $\mathbf{B}$ is a symmetric matrix and $\mathbf{A}$ is symmetric and positive definite. We show that $\mathbf{B}$ negative definite. If $0<\beta_k<1$, then using [13, Lemma 1] we obtain that

$$\begin{aligned} &s_1^{(\beta_k)}-s_0^{(\beta_k)}<0, \quad s_2^{(\beta_k)}-s_1^{(\beta_k)}+s_0^{(\beta_k)}>0, \\ &s_i^{(\beta_k)}<0, \quad s_{i+1}^{(\beta_k)}-s_i^{(\beta_k)}>0, \quad \text{for } i\ge 2. \end{aligned} \tag{3.8}$$

Substituting (3.8) into (3.6) shows that

$$q_0<0, \quad q_1>0 \quad \text{and} \quad q_i>0, \quad \text{if} \quad i\ge 2,$$

and following the proof of [13, Theorem 1], we establish the strict diagonal dominance of the matrix $\mathbf{B}$. Noting the negativity of the diagonal elements of $\mathbf{B}$, we conclude that $\mathbf{B}$ is negative definite, which yields the positive definiteness of $\mathbf{A}-\eta\mathbf{B}$. □

4. A Preconditioned Fast Conjugate Gradient Method

We now introduce a PFCG method for the system (2.8). For this, we first recall the standard preconditioned conjugate gradient (PCG) method for (2.8) — cf. [2]. As was already mentioned, the coefficient matrix can be efficiently stored by using $\mathcal{O}(NK)$ memory. Therefore, to develop a fast version PCG solver, we need a fast matrix-vector multiplication procedure and an efficient preconditioner.

Algorithm 4.1 Preconditioned Conjugate Gradient Method

At each time level t_m, choose an initial vector $\mathbf{x}^{(0)} = \mathbf{u}^{m-1}$.
Compute $\mathbf{r}^{(0)} = 2\eta\mathbf{B}\mathbf{u}^{m-1} + \mathbf{F}^m$.
for $i = 1, 2, \cdots$ **do**
  Solve the system $\mathbf{M}\mathbf{z}^{(i-1)} = \mathbf{r}^{(i-1)}$, with a preconditioner $\mathbf{M}$.
  $\rho_{i-1} = \mathbf{r}^{(i-1)^\top}\mathbf{z}^{(i-1)}$;
  if $i = 1$ **then**
    set $\mathbf{p}^{(1)} = \mathbf{z}^{(0)}$;
  else
    $\beta_{i-1} = \rho_{i-1}/\rho_{i-2}, \quad \mathbf{p}^{(i)} = \mathbf{z}^{(i-1)} + \beta_{i-1}\mathbf{p}^{(i-1)}$;
  end if
  $\mathbf{q}^{(i)} = (\mathbf{A} - \eta\mathbf{B})\,\mathbf{p}^{(i)}, \quad \alpha_i = \rho_{i-1}/\mathbf{p}^{(i)^\top}\mathbf{q}^{(i)}$;
  $\mathbf{x}^{(i)} = \mathbf{x}^{(i-1)} + \alpha_i\mathbf{p}^{(i)}, \quad \mathbf{r}^{(i)} = \mathbf{r}^{(i-1)} - \alpha_i\mathbf{q}^{(i)}$;
  Check convergence; continue if necessary.
end for
$\mathbf{u}^m = \mathbf{x}^{(i)}$.

4.1. Fast matrix-vector multiplication

In the above PCG algorithm, the matrix-vector multiplication $(\mathbf{A} - \eta\mathbf{B})\,\mathbf{p}^{(i)}$ has to be evaluated at each time step that generally costs $\mathcal{O}(N^2)$ operations per iteration. All other computations are already performed with $\mathcal{O}(N)$ operations, which is optimal. Therefore, to reduce computational cost, one needs to incorporate an efficient matrix-vector multiplication $(\mathbf{A} - \eta\mathbf{B})\,\mathbf{v}$ for any N-dimensional vector $\mathbf{v}$.

Theorem 4.1. *The matrix-vector multiplication* $(\mathbf{A} - \eta\mathbf{B})\,\mathbf{v}$ *can be performed in* $\mathcal{O}(N \log N)$ *operations.*

Proof. Since $\mathbf{A} - \eta\mathbf{B}$ is a tridiagonal plus Toeplitz matrix, it follows from [16] that $(\mathbf{A} - \eta\mathbf{B})\,\mathbf{v}$ can be calculated in $\mathcal{O}(N \log N)$ operations. □

Remark 4.1. Following Theorem 4.1, we conclude that the total computational cost for the finite volume method (2.8) based on the fast version of CG solver is $\mathcal{O}(l_A MN \log N + NK)$, where l_A is the average number of iterations at each time level, and $\mathcal{O}(NK)$ comes from the evaluation of $\{q_i\}_{i=0}^N$ in (3.6).

4.2. Circulant preconditioner

The fast matrix-vector multiplication considered in this section allows to reduce computational complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ per iteration. However, in large-scale problems — i.e. when the spatial mesh size tends to zero, the condition numbers of the coefficient matrix $\mathbf{A} - \eta\mathbf{B}$ can be quite large, which could significantly increase the computational cost of the algorithm. Therefore, efficient preconditioners for finite volume scheme (2.8) play a vital role.

There are a variety of such preconditioners for Toeplitz systems and it is not possible to provide even a very brief review here. In particular, T. Chan circulant preconditioner [4,5] is an optimal circulant preconditioner minimising the norms $\|\mathbf{C}-\mathbf{T}\|_F$ on the set of $n\times n$ circulant matrices $\mathbf{C}$ for the Frobenius norm $\|\cdot\|_F$. For Toeplitz symmetric matrices

$$\mathbf{T}=\begin{pmatrix} t_0 & t_1 & \cdots & t_{n-2} & t_{n-1} \\ t_1 & t_0 & t_1 & & t_{n-2} \\ \vdots & t_1 & t_0 & \ddots & \vdots \\ t_{n-2} & & \ddots & \ddots & t_1 \\ t_{n-1} & t_{n-2} & \cdots & t_1 & t_0 \end{pmatrix}, \tag{4.1}$$

this circulant preconditioner is also symmetric, and its first column has the entries —cf. [4]:

$$c_j=\frac{(n-j)t_j+jt_{n-j}}{n},\quad j=0,1,\cdots,n-1. \tag{4.2}$$

Since both $\mathbf{A}$ and $\mathbf{B}$ are Toeplitz symmetric matrices, we can follow T. Chan approach and construct a circulant matrix for $\mathbf{Q}:=\mathbf{A}-\eta\mathbf{B}$. More precisely,

Step 1. Construct T. Chan circulant approximation $Circ_F(\mathbf{A})$ of the Toeplitz symmetric matrix $\mathbf{A}$, the first column of which is

$$c_j(\mathbf{A})=\frac{h}{8}\begin{cases}6, & j=0,\\ (N-1)/N, & j=1,\cdots,N-1.\end{cases}$$

Step 2. Construct T. Chan circulant approximation $Circ_F(\mathbf{B})$ of the Toeplitz symmetric matrix $\mathbf{B}$, the first column of which is

$$c_j(\mathbf{B})=((N-j)q_j+jq_{N-j})/N,\quad j=0,1,\cdots,N-1.$$

Step 3. Form the corresponding circulant preconditioner $\mathbf{C}_T(\mathbf{Q})$ for $\mathbf{Q}$ by setting

$$\mathbf{C}_T(\mathbf{Q}):=Circ_F(\mathbf{A})-\eta Circ_F(\mathbf{B}). \tag{4.3}$$

Remark 4.2. T. Chan circulant preconditioner $\mathbf{C}_T(\mathbf{Q})$ of (4.3) can be diagonalised by the discrete Fourier transform (DFT) matrix $\mathbf{F}_N$ [14] as

$$\mathbf{C}_T(\mathbf{Q})=\mathbf{F}_N^{-1}\mathrm{diag}(\mathbf{F}_N\mathbf{c})\mathbf{F}_N,$$

where $\mathbf{c}=(c_j)_{j=0}^{N-1}$ is the first column vector of $\mathbf{C}_T(\mathbf{Q})$ with $\mathbf{c}_j=c_j(\mathbf{A})-\eta c_j(\mathbf{B})$. Hence, it can be inverted in $\mathcal{O}(N\log N)$ operations. The PFCG method based on this preconditioner, requires only $\mathcal{O}(N\log N)$ operations to find the solution of the linear algebraic system (2.8).

4.3. The efficiency of the method

Here, we consider numerical examples aimed to study the performance of the finite volume scheme (2.8). It is solved by CG solver, by the fast version of CG without preconditioning (FCG solver), by PFCG solver and by Gaussian elimination (Gauss). All simulations are carried out in MATLAB R2010 environment on a Lenovo E470c laptop with Intel(R) Core(TM) i5-6200U processor of 8 GB RAM and 2.30GHz CPU. In all numerical experiments the corresponding iterative solver stops if

$$\frac{||\mathbf{r}^{(k)}||_2}{||\mathbf{r}^{(0)}||_2} \leq 10^{-12}.$$

Example 4.1. Let $0 < x < 1$ and $0 < t < 1$. Following [23], we consider the approximate solution of the distributed-order equation (2.1) with the forcing function $f(x,t) = e^t x^2(1-x)^2 - e^t[\omega(x) + \omega(1-x)]$ and the weight function $P(\alpha) = -2\Gamma(5-\alpha)\cos(\pi\alpha/2)$, where

$$\omega(x) = \frac{24}{\ln x}(x^3 - x^2) - \frac{12}{\ln x}\Big[(3x^2 - 2x) - \frac{x^2 - x}{\ln x}\Big] + \frac{2}{\ln x}\Big[6x - 2 - \frac{5x-3}{\ln x} + \frac{2x-2}{\ln^2 x}\Big].$$

Table 1: Computational efficiency of the finite volume scheme (2.8) solved by different solvers.

	N	Error	Cov.	CPU time	# of Itr.
Gauss	$2^8 = 256$	3.1982×10^{-6}	—	3m29s	—
	$2^9 = 512$	7.9896×10^{-7}	2.00	21m07s	—
	$2^{10} = 1024$	1.9661×10^{-7}	2.02	4h17m	—
	$2^{11} = 2048$	—	—	> 35h	—
CG	$2^8 = 256$	3.1982×10^{-6}	—	29.44s	110
	$2^9 = 512$	7.9896×10^{-7}	2.00	13m05s	209
	$2^{10} = 1024$	1.9661×10^{-7}	2.02	1h36m	400
	$2^{11} = 2048$	4.5918×10^{-8}	2.10	9h27m	773
FCG	$2^8 = 256$	3.1982×10^{-6}	—	18.63s	110
	$2^9 = 512$	7.9896×10^{-7}	2.00	49.94s	209
	$2^{10} = 1024$	1.9661×10^{-7}	2.02	2m44s	400
	$2^{11} = 2048$	4.5918×10^{-8}	2.10	9m32s	773
	$2^{12} = 4096$	8.6297×10^{-9}	2.41	1h02m	1485
	$2^{13} = 8192$	3.1873×10^{-9}	1.44	4h03m	2875
PFCG	$2^8 = 256$	3.1982×10^{-6}	—	2.78s	10
	$2^9 = 512$	7.9896×10^{-7}	2.00	4.06s	12
	$2^{10} = 1024$	1.9661×10^{-7}	2.02	7.31s	15
	$2^{11} = 2048$	4.5918×10^{-8}	2.10	11.70s	19
	$2^{12} = 4096$	8.6297×10^{-9}	2.41	42.46s	23
	$2^{13} = 8192$	3.1873×10^{-9}	1.44	1m48s	29

The exact solution of this problem is the function $u(x,t) = e^t x^2(1-x)^2$.

In all numerical tests, the parameters K and M are the same — viz. $K = M = 1000$. Table 1 shows the performance of various solvers with respect to error, convergence order, CPU time, and iteration number. Since these methods are developed without using any lossy compression, all solvers generate identical numerical solutions with the second-order convergence rate in space. It is also worth noting that, in comparison to conventional solvers such as Gauss and CG, the FCG solver has a significantly reduced CPU usage. It requires only $\mathcal{O}(NK)$ memory and has $\mathcal{O}(l_A MN \log N + NK)$ computational complexity overall. Thus for $N = 2^{10}$, the Gauss runs at least 4 hours, the CG one and half an hour, while FCG needs at most 3 minutes! However, if the number of unknowns grows, the coefficient matrix $\mathbf{A} - \eta\mathbf{B}$ of (2.8) becomes ill-conditioned, the number of iterations and computational cost increase even for FCG solver. For example, if N increases from 2^8 to 2^{13}, the number of iterations jumps from 110 to 2875 and CPU usage increases from 18 seconds to more than 4 hours! The PFCG solver, where the circulant preconditioner of Subsection 4.2 is used, reduces the computational cost further. Thus, for $N = 2^{13}$ the FCG solver needs 2875 iterations, the PFCG requires only 29 iterations. CPU time also decreases from more than 4 hours to less than 2 minutes! This example demonstrates the advantage of the introduced fast finite volume method for the Riesz space distributed-order diffusion equation (2.1).

5. Parameter Identification

In applications, various free parameters of mathematical models have to be identified — e.g. space-fractional orders in two-dimensional space-fractional diffusion models [6] and time-fractional order in time-fractional subdiffusion models [12]. Here, we consider the a priori unknown parameters $\widetilde{\alpha}_i$, $i = 1,2$ in the distribution function $P(\alpha)$ of the model (2.2) — cf. [38]. Therefore, given the source function $f(x,t)$, the initial value $u_0(x)$ of the distributed-order diffusion equation (2.1), and the observation data for the state variable $\mathbf{g}$ at the final time, we construct an optimisation model by minimising the difference between the numerical output and the observation data. After that, the Levenberg–Marquardt (L-M) iterative algorithm [32, 40] is used to identify the parameters $\widetilde{\alpha}_i$, $i = 1,2$.

5.1. PFCG-based L-M regularization method

Suppose that $1 < \widetilde{\alpha}_1 < \widetilde{\alpha}_2 < 2$. Set $\mathbf{p} := (\widetilde{\alpha}_1, \widetilde{\alpha}_2)^\top$ and introduce a nonlinear residual function

$$\mathbf{r}(\mathbf{p}) = (r_1(\mathbf{p}), r_2(\mathbf{p}), \cdots, r_N(\mathbf{p}))^\top \tag{5.1}$$

such that $r_i(\mathbf{p}) = u(x_i, T; \mathbf{p}) - g_i$. In order to determine parameters $\widetilde{\alpha}_i$, $i = 1,2$, we consider the parameter identification problem consisting in finding a vector $\mathbf{p}_{inv} := (\alpha_1^*, \alpha_2^*)^\top$, minimising the nonlinear least square problem

$$\mathbf{p}_{inv} = \arg \min_{\mathbf{p} \in (1,2)\times(1,2)} \mathscr{F}(\mathbf{p}) := \frac{1}{2} \sum_{i=1}^{N} [r_i(\mathbf{p})]^2, \tag{5.2}$$

where g_i is the i-th observed value of $\mathbf{g}$ at the point x_i.

The solution of the problem (5.2) can be established by an iterative algorithm such as Gauss-Newton method. According to [40], this algorithm can be represented in the form

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{d}_k, \quad \mathbf{d}_k = -(\mathbf{J}_k^\top \mathbf{J}_k)^{-1} \mathbf{J}_k^\top \mathbf{r}_k, \tag{5.3}$$

where the subscript k is the relevant iteration, $\mathbf{r} = \mathbf{r}(\mathbf{p})$ is defined in (5.1) and $\mathbf{J} = \mathbf{J}(\mathbf{p})$ is the Jacobian of $\mathbf{r}(\mathbf{p})$, so that

$$\mathbf{J} = \begin{pmatrix} \dfrac{\partial u(x_1,T;\mathbf{p})}{\partial \widetilde{\alpha}_1}, \dfrac{\partial u(x_2,T;\mathbf{p})}{\partial \widetilde{\alpha}_1}, \cdots, \dfrac{\partial u(x_N,T;\mathbf{p})}{\partial \widetilde{\alpha}_1} \\ \dfrac{\partial u(x_1,T;\mathbf{p})}{\partial \widetilde{\alpha}_2}, \dfrac{\partial u(x_2,T;\mathbf{p})}{\partial \widetilde{\alpha}_2}, \cdots, \dfrac{\partial u(x_N,T;\mathbf{p})}{\partial \widetilde{\alpha}_2} \end{pmatrix}^\top. \tag{5.4}$$

Remark 5.1. In practical computation, the exact values of partial derivatives in (5.4) cannot be derived. They have to be approximated. One possibility is to employ the first-order finite difference scheme

$$\frac{u(x_i,T;\mathbf{p}+\epsilon\mathbf{e}_j) - u(x_i,T;\mathbf{p})}{\epsilon}, \tag{5.5}$$

where ϵ is a small differential step size and $\mathbf{e}_1 = (1,0)^\top$, $\mathbf{e}_2 = (0,1)^\top$ are unit vectors. Such an approximation is used here.

The Gauss-Newton iteration method is well-defined, if the Jacobian $\mathbf{J}_k$ is a full column rank matrix. This condition may not be satisfied in certain cases. Therefore, the vector $\mathbf{d}_k$ can be not properly directed, and Levenberg [22] and Marquardt [28] proposed an approach to overcome the difficulty mentioned. They modified the Gauss-Newton method (5.3) as follows

$$\mathbf{p}_{k+1} = \mathbf{p}_k + \mathbf{d}_k, \quad \mathbf{d}_k = -(\mathbf{J}_k^\top \mathbf{J}_k + \alpha_k \mathbf{I})^{-1} \mathbf{J}_k^\top \mathbf{r}_k, \tag{5.6}$$

where α_k is a positive penalty parameter and $\mathbf{I}$ the 2×2 unit matrix. The reader is referred to [40] for more details concerning the L-M algorithm.

It is well-known that the solution of the optimisation problem (5.2) reduces to the inverse problem (5.4)-(5.6). In each iteration of the L-M method, we first have to apply the finite volume scheme (2.8) to the distributed-order diffusion equation (2.1) with special chosen parameters $\mathbf{p} = (\widetilde{\alpha}_1, \widetilde{\alpha}_2)^\top$ to derive an approximate solution at the final time. As was already mentioned, the computational cost of solving direct problem is very high and the invertibility problem could not be solved efficiently. Nevertheless, the L-M iterative method combined with the PFCG solver from Section 4 can be used to find required parameters of $P(\alpha)$.

The parameters identification procedure is summarised in Algorithm 5.1, where the Armijo rule [1] is used to guarantee that the objective function $\mathscr{F}$ has a sufficient descent direction. Other rules and related convergence results can be found in [40].

Algorithm 5.1 PFCG-based Parameter Identification Algorithm

Consider the observation data $\mathbf{g}$ and related information of model (2.1).
Chose an initial vector $\mathbf{p}_0$ and $\rho \in (0,1)$, $\sigma \in (0,1/2)$, $\alpha_0 > 0$ and a small ϵ.
for $k = 0, 1, \cdots, K_{max}$ **do**

Step 1. Solve the finite volume scheme (2.8) using the PFCG solver corresponding to $\mathbf{p}_k$ and $\mathbf{p}_k + \epsilon \mathbf{e}_j$ to obtain $u(\cdot, T; \mathbf{p}_k)$ and $u(\cdot, T; \mathbf{p}_k + \epsilon \mathbf{e}_j)$, respectively.

Step 2. Use (5.4)-(5.5) and (5.1) to compute $\mathbf{J}_k$ and $\mathbf{r}_k$, respectively, and then update the search direction $\mathbf{d}_k$ in (5.6).

Step 3. Determine the search step ρ^m by the Armijo rule:

$$\mathscr{F}(\mathbf{p}_k + \rho^m \mathbf{d}_k) \leq \mathscr{F}(\mathbf{p}_k) + \sigma \rho^m \mathbf{d}_k^\top \mathbf{J}_k^\top \mathbf{r}_k,$$

where m is the smallest nonnegative integer.

Step 4. If $|\rho^m d_k| \leq Tol$, then stop and let $\mathbf{p}_{inv} := \mathbf{p}_k$. Otherwise update

$$\mathbf{p}_{k+1} := \mathbf{p}_k + \rho^m \mathbf{d}_k, \quad \alpha_{k+1} := \alpha_k / 2$$

and go to **Step 1**.

end for

5.2. Numerical test

Example 5.1. Let us identify the parameters $\mathbf{p} = (\widetilde{\alpha}_1, \widetilde{\alpha}_2)^\top$ in the distribution function

$$P(\alpha) = l^{\alpha-2} \left(\delta(\alpha - \widetilde{\alpha}_1) + \delta(\alpha - \widetilde{\alpha}_2)\right).$$

We consider the domain $(x,t) \in [0,1] \times [0,1]$ and the problem (2.1) with the right-hand side $f(x,t) = 10$, the initial condition $u(x,0) = x(1-x)$ and the boundary conditions. We also set $\mathbf{p} = (1.3,\ 1.6)$, $l = 2$, $\rho = 0.9$, $\sigma = 0.25$, $\alpha_0 = 1$, $\epsilon = 10^{-3}$ and $Tol = 10^{-15}$.

We consider the numerical solution of (2.1) with $K = M = N = 32$ as observation data $\mathbf{g}$. The real data can be contaminated by noise, therefore we also include a random small perturbation of the observation data — viz.

$$g_i^\varepsilon := g_i(1 + \varepsilon\% \ \text{randn}(i)), \quad 1 \leq i \leq N,$$

where ε shows the noise level and "randn" is the random noise generated by standard normal distribution.

We will deal with the cases $\varepsilon = 0$, $\varepsilon = 0.1$, $\varepsilon = 1$ and in each case, three initial guesses $\mathbf{p}_0 = (1.2, 1.2)$, $\mathbf{p}_0 = (1.8, 1.8)$ and $\mathbf{p}_0 = (1.4, 1.5)$ are used. Set

$$e_{\mathbf{p}} := \|\mathbf{p} - \mathbf{p}_{inv}\|_\infty = \max\{|\widetilde{\alpha}_1 - \alpha_1^*|, |\widetilde{\alpha}_2 - \alpha_2^*|\}.$$

We solve the model numerically for $K = M = N = 32$, using the following approximation of the delta-function

$$\delta(\alpha - \widetilde{\alpha}_j) \approx \frac{1}{0.05\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\alpha - \widetilde{\alpha}_j}{0.05}\right)^2}, \quad j = 1, 2.$$

Table 2: Numerical observation of Example 5.1 with ε%-level noise-contaminated data.

ε	$\mathbf{p}_0$	$\mathbf{p}_{inv}$	$e_{\mathbf{p}}$	# of Itr.
0	$(1.2, 1.2)$	$(1.3000, 1.6000)$	4.8850×10^{-14}	24
	$(1.8, 1.8)$	$(1.3000, 1.6000)$	1.2212×10^{-14}	24
	$(1.4, 1.5)$	$(1.3000, 1.6000)$	4.4631×10^{-14}	15
0.1	$(1.2, 1.2)$	$(1.3019, 1.5990)$	1.8866×10^{-3}	26
	$(1.8, 1.8)$	$(1.3019, 1.5990)$	1.8866×10^{-3}	26
	$(1.4, 1.5)$	$(1.3019, 1.5990)$	1.8866×10^{-3}	15
1	$(1.2, 1.2)$	$(1.3049, 1.5988)$	4.9325×10^{-3}	17
	$(1.8, 1.8)$	$(1.3049, 1.5988)$	4.9325×10^{-3}	17
	$(1.4, 1.5)$	$(1.3049, 1.5988)$	4.9325×10^{-3}	10

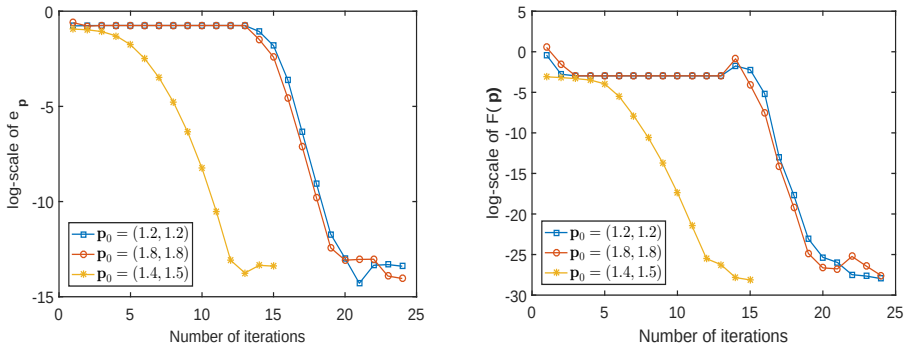

Figure 1: Uncontaminated case of Example 5.1.

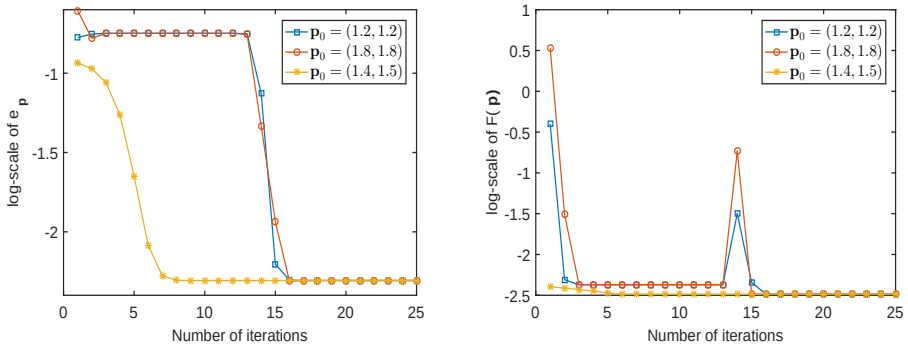

Figure 2: 1%-level contaminated case of Example 5.1.

Table 2 shows numerical results obtained via PFCG solver for the parameter estimation problem. For uncontaminated data and for data contaminated by 1%-level random noise, the maximum error $e_{\mathbf{p}}$ and objective function $\mathscr{F}(\mathbf{p})$ are presented in Figs. 1-2. Let us note that the PFCG-based L-M algorithm efficiently works at various contamination levels with different initial vectors. Compared with the uncontaminated case, the parameters

are approximated with relatively lower accuracy but the final results are still acceptable. Moreover, better initial guesses lead to a faster convergence of the parameter identification procedure, consistent with our expectations.

6. Conclusions

We proposed a Crank-Nicolson finite volume scheme for the modeling of the Riesz space distributed-order diffusion equation. It is shown that the corresponding linear system has a symmetric positive definite Toeplitz matrix. It can be efficiently stored in $\mathcal{O}(NK)$ memory. Moreover, based on fast matrix-vector multiplications and an efficient T. Chan preconditioner developed in Section 4, a preconditioned fast conjugate gradient solver is proposed for the finite volume scheme (2.8). Compared with the Gaussian elimination method, the computational complexity is reduced from $\mathcal{O}(MN^3+NK)$ to $\mathcal{O}(l_AMN\log N+NK)$, where l_A is the average number of iterations at a time level. Finally, the proposed preconditioned fast finite volume method combined with Levenberg-Marquardt method is applied to solve an parameter identification problem. Numerical experiments show the efficiency of the method.

Acknowledgements

This work was supported in part by the grants of National Natural Science Foundation of China (91630207, 11571115), the Shandong Provincial Natural Science Foundation (ZR2017MA006), by the Fundamental Research Funds for the Central Universities (18CX02044A), and by the National Science Foundation (DMS-1620194).

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 the paper.

References

[1] L. Armijo, *Minimization of functions having Lipschitz continuous partial derivatives*, Pacific J. Math. **16**, 1–3 (1966).

[2] R. Barrett, M. Berry, T.F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H. Van der Vorst, *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, SIAM (1994).

[3] D. Benson, S.W. Wheatcraft and M.M. Meerschaert, *The fractional-order governing equation of Lévy motion*, Water Resour. Res. **36**, 1413–1423 (2000).

[4] T.F. Chan, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. and Stat. Comput. **9**, 766–771 (1988).

[5] R.H. Chan and M.K. Ng, *Conjugate gradient methods for Toeplitz systems*, SIAM Review **38**, 427–482 (1996).

[6] S. Chen, F. Liu, X. Jiang, I. Turner and K. Burrage, *Fast finite difference approximation for identifying parameters in a two-dimensional space-fractional nonlocal model with variable diffusivity coefficients*, SIAM J. Numer. Anal. **54**, 606–624 (2016).

[7] D. del-Castillo-Negrete, B.A. Carreras and V.E. Lynch, *Fractional diffusion in plasma turbulence*, Phys. Plasmas **11**, 3854 (2004).
[8] R. Du, Z. Hao and Z. Sun, *Lubich second-order methods for distributed-order time-fractional differential equations with smooth solutions*, East. Asia. J. Appl. Math. **6**, 131–151 (2016).
[9] V.J. Ervin and J.P. Roop, *Variational formulation for the stationary fractional advection dispersion equation*, Numer. Methods Partial Differential Equations **22**, 558–576 (2006).
[10] H. Fu, M.K. Ng and H. Wang, *A divide-and-conquer fast finite difference method for space-time fractional partial differential equation*, Comput. Math. Appl. **73**, 1233–1242 (2017).
[11] H. Fu and H. Wang, *A preconditioned fast finite difference method for space-time fractional partial differential equations*, Fract. Calc. Appl. Anal. **20**, 88–116 (2017).
[12] H. Fu, H. Wang and Z. Wang, *POD/DEIM reduced-order modeling of time-fractional partial differential equations with applications in parameter identification*, J. Sci. Comput. **74**, 220–243 (2018).
[13] L. Feng, P. Zhuang, F. Liu and I. Turner, *Stability and convergence of a new finite volume method for a two-sided space-fractional diffusion equation*, Appl. Math. Comput. **257**, 52–65 (2015).
[14] R.M. Gray, *Toeplitz and Circulant Matrices: A Review*, Foundations and Trends in Communications and Information Theory, Vol 2, Issue 3, pp. 155–239 (2006).
[15] X. Gu, T. Huang, C. Ji, B. Carpentieri and A.A. Alikhanov, *Fast iterative method with a second-order implicit difference scheme for time-space fractional convection-diffusion equation*, J. Sci. Comput. **72**, 957–985 (2017).
[16] X. Gu, T. Huang, X. Zhao, W. Xu, H. Li and L. Li, *Circulant preconditioned iterative methods for peridynamic model simulation*, Appl. Math. Comput. **248**, 470–479 (2014).
[17] C. Ji and Z. Sun, *An unconditionally stable and high-order convergent difference scheme for stokes' first problem for a heated generalized second grade fluid with fractional derivative*, Numer. Math. Theor. Meth. Appl. **10**, 597–613 (2017).
[18] B. Jin, R. Lazarov, Y. Liu and Z. Zhou, *The Galerkin finite element method for a multi-term time-fractional diffusion equation*, J. Comput. Phys. **281**, 825–843 (2015).
[19] X. Jin, F. Lin and Z. Zhao, *Preconditioned iterative methods for two-dimensional space fractional diffusion equations*, Commun. Comput. Phys. **18**, 468–488 (2015).
[20] R. Ke, M.K. Ng and H. Sun, *A fast direct method for block triangular Toeplitz-like with tridiagonal block systems from time-fractional partial differential equations*, J. Comput. Phys. **303**, 203–211 (2015).
[21] S. Lei, X. Chen and X. Zhang, *Multilevel circulant preconditioner for high-dimensional fractional diffusion equations*, East. Asia. J. Appl. Math. **6**, 109–130 (2016).
[22] K. Levenberg, *A method for the solution of certain nonlinear problems in least squares*, Qart. Appl. Math. **2**, 164–166 (1944).
[23] J. Li, F. Liu, L. Feng and I. Turner, *A novel finite volume method for the Riesz space distributed-order diffusion equation*, Comput. Math. Appl. **74**, 772–783 (2017).
[24] H. Li, X. Wu and J. Zhang, *Numerical solution of the time-fractional sub-diffusion equation on an unbounded domain in two-dimensional space*, East. Asia. J. Appl. Math. **7**, 439–454 (2017).
[25] Y. Lin and C. Xu, *Finite difference/spectral approximations for the time-fractional diffusion equation*, J. Comput. Phys. **225**, 1533–1552 (2007).
[26] H. Ma and Y. Yang, *Jacobi spectral collocation method for the time variable-order fractional mobile-immobile advection-dispersion solute transport model*, East. Asia. J. Appl. Math. **6**, 337–352 (2016).
[27] R.L. Magin, *Fractional Calculus in Bioengineering*, Begell House Publishers (2006).
[28] D.W. Marquardt, *An algorithm for least-squares estimation of nonlinear inequalities*, SIAM J. Appl. Math. **11**, 431–441 (1963).

[29] R. Metler and J. Klafter, *The random walk's guide to anomalous diffusion: a fractional dynamics approach*, Phys. Reports **339**, 1–77 (2000).
[30] M.M. Meerschaert and C. Tadjeran, *Finite difference approximations for two-sided space-fractional partial differential equations*, Appl. Numer. Math. **56**, 80–90 (2006).
[31] K. Mustapha, *An implicit finite-difference time-stepping method for a sub-diffusion equation, with spatial discretization by finite elements*, IMA J. Numer. Anal. **31**, 719–739 (2011).
[32] J. Nocedal and S.J. Wright, *Numerical Optimization*, Springer (2006).
[33] K.B. Oldham and J. Spanier, *The Fractional Calculus*, Academic Press (1974).
[34] J. Pan, M.K. Ng and H. Wang, *Fast iterative solvers for linear systems arising from time-dependent space-fractional diffusion equations*, SIAM J. Sci. Comput. **38**, A2806–A2826 (2016).
[35] I. Podlubny, *Fractional Differential Equations*, Academic Press (1999).
[36] L. Sabatelli, S. Keating, J. Dudley and P. Richmond, *Waiting time distributions in financial markets*, Eur. Phys. J. B **27**, 273–275 (2002).
[37] A.I. Saichev and G.M. Zaslavsky, *Fractional kinetic equations: solutions and applications*, Chaos **7**, 753–764 (1997).
[38] I.M. Soklov, A.V. Chechkin and J. Klafter, *Distributed-order fractional kinetics*, Acta Phys. Polon. B **35**, 1323–1341 (2004).
[39] M. Stynes, E. O'Riordan, and J.L. Gracia, *Error analysis of a finite difference method on graded meshes for a time-fractional diffusion equation*, SIAM J. Numer. Anal. **55**, 1057–1079 (2017).
[40] W. Sun and Y. Yuan, *Optimization Theory and Methods: Nonlinear Programming*, Springer (2006).
[41] H. Wang, K. Wang and T. Sircar, *A direct $\mathcal{O}(N \log^2 N)$ finite difference method for fractional diffusion equations*, J. Comput. Phys. **229**, 8095–8104 (2010).
[42] H. Wang and T.S. Basu, *A fast finite difference method for two-dimensional space-fractional diffusion equations*, SIAM J. Sci. Comput. **34**, 2444–2458 (2012).
[43] S. Wu and T. Zhou, *Fast parareal iterations for fractional diffusion equations*, J. Comput. Phys. **329**, 210–226 (2017).
[44] S.B. Yuste and J. Quintana-Murillo, *A finite difference scheme with non-uniform timesteps for fractional diffusion equations*, Comput. Phys. Commun. **183**, 2594–2600 (2012).
[45] F. Zeng, Z. Zhang and G. Karniadakis, *Fast difference schemes for solving high-dimensional time-fractional subdiffusion equations*, J. Comput. Phys. **307**, 15–33 (2016).
[46] J. Zhou, D. Xu and H. Chen, *A weak Galerkin finite element method for multi-term time-fractional diffusion equations*, East. Asia. J. Appl. Math. **8**, 181–193 (2018).