J. Math. Computer Sci., 31 (2023), 353–366

Online: ISSN 2008-949X

Journal of Mathematics and Computer Science

Journal Homepage: www.isr-publications.com/jmcs

# Numerical solutions of the space-time fractional diffusion equation via a gradient-descent iterative procedure

Kanjanaporn Tansri, Adisorn Kittisopaporn, Pattrawut Chansangiam*

*Department of Mathematics, School of Science, King Mongkut's Institute of Technology Ladkrabang, Bangkok 10520, Thailand.*

## Abstract

A one-dimensional space-time fractional diffusion equation describes anomalous diffusion on fractals in one dimension. In this paper, this equation is discretized by finite difference schemes based on the Grünwald-Letnikov approximation for Riemann-Liouville and Caputo's fractional derivatives. It turns out that the discretized equations can be put into a compact form, i.e., a linear system with a block lower-triangular coefficient matrix. To solve the linear system, we formulate a matrix iterative algorithm based on gradient-descent technique. In particular, we work out for the space fractional diffusion equation. Theoretically, the proposed solver is always applicable with satisfactory convergence rate and error estimates. Simulations are presented numerically and graphically to illustrate the accuracy, the efficiency, and the performance of the algorithm, compared to other iterative procedures for linear systems.

**Keywords:** Fractional diffusion equation, Grünwald-Letnikov approximation, gradient descent, iterative method, matrix norms.

**2020 MSC:** 65F10, 65N22, 34A08, 15A60.

## 1. Introduction

In recent years, fractional calculus and fractional differential equations (FDEs) have gotten extensive attention. The differential equations with non-integer order in time and space can be better and more powerful to describe many certain phenomena in many diverse fields especially in physics and engineering, e.g., design of analogue circuits [28], electrical impedance [18], chaotic systems [16], fluid mechanics [20], and signal processing [1]. This paper focuses on a one-dimensional space-time fractional diffusion equation

$$ {}^{C}D_t^{\beta}u(x,t) - {}^{RL}D_x^{\alpha}u(x,t) = f(x,t), \tag{1.1}$$

on bounded domains $x \in [a,b]$ and $t \in [0,T]$. The fractional diffusion equation describes anomalous diffusion on fractals in one dimension. Physical considerations restrict the fractional orders to be $\alpha \in (1,2]$

*Corresponding author
Email address: `pattrawut.ch@kmitl.ac.th` (Pattrawut Chansangiam)

doi: 10.22436/jmcs.031.04.02

*Received: 2022-07-25 Revised: 2023-01-23 Accepted: 2023-04-19*

and $\beta \in (0,1]$. Let us denote the Gamma function by $\Gamma(\cdot)$. Recall that the operator ${}^{C}D_t^{\beta}$ is the Caputo's fractional derivative of order $\beta$ with respect to the time $t$, i.e.,

$$ {}^{C}D_t^{\beta}u(x,t) = \frac{1}{\Gamma(1-\beta)} \int_0^t \frac{\partial}{\partial \tau} u(x,\tau) \frac{d\tau}{(t-\tau)^{\beta}} \quad \text{for} \quad \beta \in (0,1), $$

and the case $\beta = 1$ is the first-order partial derivative with respect to the time $t$. The symbol ${}^{RL}D_x^{\alpha}$ denotes the Riemann-Liouville derivative of order $\alpha$ with respect to the spatial variable $x$. Indeed, the case $\alpha = 2$ is the second-order partial derivative with respect to $x$, and for another case $\alpha \in (1,2)$ we have

$$ {}^{RL}D_t^{\alpha}u(x,t) = \frac{1}{\Gamma(2-\alpha)} \frac{\partial^2}{\partial x^2} \int_0^x \frac{u(\xi,t)}{(x-\xi)^{\alpha-1}} \, d\xi. $$

The case $\beta = 1$, Eq. (1.1) is known as the space fractional diffusion equation. We assume that Eq. (1.1) is subjected to conditions

$$ u(x,t=0)=0, \quad u(x=a,t)=0, \quad \text{and} \quad u(x=b,t)=\tilde{b}(t), $$

where $\tilde{b}$ is a given function of $t$. In the recent decade, there have been many studies about applications and theory of the space-time fractional diffusion equation on bounded domains, e.g., [4, 17]. Theory of solutions for Eq. (1.1) and related equations were also investigated, e.g., existence and regularity of solution [26], and stochastic solution [24]. Posterior error analysis of solutions for Eq. (1.1) were studied in [5, 27]. As applications, this equation was used to modeling random walk models, e.g., [12–15]. Moreover, it arises in many anomalous diffusion processes observed in physics [3, 23] and cell biology [29, 33].

As most of the fractional differential equations have no exact solutions, the approximate solution becomes a good choice to seek for. Therefore, numerical methods for solving them have been studied for many years. Many authors proposed various finite difference or finite element methods to discretize Eq. (5.1); see e.g. [6–8, 11, 21, 31, 32]. The following Grünwald-Letnikov approximation is a finite difference schemes which is often used to discretize the fractional derivatives (see, e.g., [30]):

$$ {}^{RL}D_x^{\alpha}u(x,t) = \lim_{N_x \to \infty} \frac{1}{h_x^{\alpha}} \sum_{k=0}^{N_x} g_{\alpha,k} u(x-(k-1)h_x,t), \tag{1.2} $$

where the coefficients $g_{\alpha,k}$ are defined as

$$ g_{\alpha,k} = \frac{\Gamma(k-\alpha)}{\Gamma(-\alpha)\Gamma(k+1)}. \tag{1.3} $$

The above approximation can also be applied to the Caputo's fractional derivative via the difference formula between the two derivatives [25]:

$$ {}^{RL}D_t^{\alpha}u - {}^{C}D_t^{\alpha}u = \sum_{\nu=0}^{\lfloor \alpha \rfloor} r_{\nu}^{\alpha}(t)u^{(\nu)}(0), \quad \text{where} \quad r_{\nu}^{\alpha}(t) = \frac{t^{\nu-\alpha}}{\Gamma(\nu+1-\alpha)}. \tag{1.4} $$

Here, $\lfloor \cdot \rfloor$ denotes the floor function. In the case $\alpha \in (0,1]$, the difference is zero for the homogeneous initial value $u(x,0)=0$, see, e.g., [30]. Another interesting idea is to put the discretized equations into a compact linear system, and then formulate a gradient-descent iterative algorithm to solve the linear system; see, e.g., a treatment for the case of heat and Poisson equations in [19].

In this work, we discretize the space-time fractional diffusion equation (1.1) using the Grünwald-Letnikov approximation to both of the Caputo and the Riemann-Liouville derivatives (see Section 2). Next, we transform the discretized equations into a linear system with a block lower-triangular coefficient matrix. Then, we propose an iterative procedure to solve the linear system using gradients and the steepest descent technique (see Section 3). The structure of the coefficient matrix is utilized in computations of the proposed procedure. We make a convergence analysis of the proposed method which includes the proof of convergence, the convergence rate, and error estimates (see Section 4). As an interesting particular case, we consider the space fractional diffusion equation (see Section 5). Numerical experiments for (1.1) and the space fractional diffusion equation are provided (see Section 6) to show the accuracy and efficiency of the proposed algorithm. We also compare the performance of the proposed algorithm to well-known iterative schemes for linear systems, e.g., GI, LSI, SOR, and JOR algorithms.

## 2. The space-time fractional diffusion equation: discretization and linearization

Consider the space-time fractional diffusion equation (1.1) subjected to the the initial condition $u(x,0) = 0$ and the boundary conditions $u(a,t) = 0$ and $u(b,t) = \tilde{b}(t)$, where $\tilde{b}$ is a given function. We shall discretize this equation by partitioning the spatial domain $[a,b]$ and the time domain $[0,T]$ into $N_x$ and $N_t$ subintervals, respectively. The grid points for each $i \in \{0,1,\dots,N_x\}$ and $j \in \{0,1,\dots,N_t\}$ are $x_i = a + ih_x$ and $t_j = jh_t$, where

$$h_x = \frac{b-a}{N_x} \quad \text{and} \quad h_t = \frac{T}{N_t}. \tag{2.1}$$

For each $i,j$, let us denote the given value $f(x_i,t_j)$ by $f_{ij}$. From the initial and boundary conditions, we would like to find the unknown $u_{ij} := u(x_i,t_j)$ for each $i = 1,\dots,N_x-1$ and $j = 1,\dots,N_t$.

We approximate the Caputo and Riemann-Liouville derivatives appeared in Eq. (1.1) using the Grünwald-Letnikov formulas (1.2)-(1.3), and the difference formula (1.4). Thus, for each $i = 0,1,\dots,N_x-2$ and $j = 0,1,\dots,N_t-1$, we obtain

$$\frac{1}{h_t^{\beta}}\sum_{k=0}^{j+1} g_{\beta,k}u_{i,j-k+1} - \frac{1}{h_x^{\alpha}}\sum_{l=0}^{i+1} g_{\alpha,l}u_{i-l+1,j+1} = f_{i,j+1}. \tag{2.2}$$

By denoting

$$\bar{h} = \frac{h_x^{\alpha}}{g_{\alpha,0}h_t^{\beta}} \quad \text{and} \quad g_l = \frac{g_{\alpha,l}}{g_{\alpha,0}},$$

we can rewrite Eq. (2.2) as

$$\bar{h}\sum_{k=0}^{j+1} g_{\beta,k}u_{i,j-k+1} - \sum_{l=1}^{i+1} g_l u_{i-l+1,j+1} - u_{i+1,j+1} = h_t^{\beta}\bar{h}f_{i,j+1}. \tag{2.3}$$

For convenience, let us denote $\widehat{h} = -h_t^{\beta}\bar{h}$ and

$$G = g_1 I_{N_t} - \bar{h}\begin{bmatrix} g_{\beta,0} & 0 & \cdots & 0 \\ g_{\beta,1} & g_{\beta,0} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{\beta,N_t-1} & g_{\beta,N_t-2} & \cdots & g_{\beta,0} \end{bmatrix},$$

where $I_{N_t} \in \mathbb{R}^{N_t \times N_t}$ is identity matrix. We now form the following matrix and vectors:

$$T = \left[\begin{array}{c|c|c|c|c} I_{N_t} & 0 & 0 & \cdots & 0 \\ \hline G & I_{N_t} & 0 & \cdots & 0 \\ \hline g_2 I_{N_t} & G & I_{N_t} & \cdots & 0 \\ \hline \vdots & \ddots & \ddots & \ddots & \vdots \\ \hline g_{N_x-2}I_{N_t} & \cdots & g_2 I_{N_t} & G & I_{N_t} \end{array}\right], \quad \mathbf{u} = \begin{bmatrix} u_{11} \\ u_{12} \\ \vdots \\ u_{N_x-1,N_t} \end{bmatrix}, \quad \text{and} \quad \mathbf{f} = \widehat{h}\begin{bmatrix} f_{01} \\ f_{02} \\ \vdots \\ f_{N_x-2,N_t} \end{bmatrix}.$$

Hence, the discretized equation (2.3) can be transformed into a linear system

$$T\mathbf{u} = \mathbf{f} \tag{2.4}$$

consisting of $N := (N_x - 1)N_t$ equations and $N$ variables $u_{11}, \ldots, u_{N_x-1,N_t}$. Note that the coefficient $T$ is a block lower-triangular matrix with identity matrices in the main-diagonal blocks, so that $T$ is sparse and invertible (with determinant equal to 1). We demonstrate an example of the matrix $T$ in the case of $N_x = 4$ and $N_t = 2$ as follows:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ g_1 - \bar{h}g_{\beta,0} & 0 & 1 & 0 & 0 & 0 \\ -\bar{h}g_{\beta,1} & g_1 - \bar{h}g_{\beta,0} & 0 & 1 & 0 & 0 \\ g_2 & 0 & g_1 - \bar{h}g_{\beta,0} & 0 & 1 & 0 \\ 0 & g_2 & -\bar{h}g_{\beta,1} & g_1 - \bar{h}g_{\beta,0} & 0 & 1 \end{bmatrix}.$$

## 3. A gradient-descent iterative procedure to solve the arised linear system

In this section, we introduce an iterative algorithm for solving the linear system (2.4) based on the idea of gradient-descent. Since the coefficient $T$ is an invertible matrix, the exact solution vector $\mathbf{u}$ can be solved directly by $\mathbf{u}^* = T^{-1}\mathbf{f}$. We measure the errors for approximate solutions by the Euclidean vector norm $\|\cdot\|_F$. So, we define an error function $\Psi : \mathbb{R}^N \to \mathbb{R}$ by

$$\Psi(\mathbf{u}) := \|T\mathbf{u} - \mathbf{f}\|_F^2.$$

The form of the gradient-descent iterative procedure is written by the recursive equation as follows:

$$\mathbf{u}_{k+1} = \mathbf{u}_k - \tau_{k+1}\nabla\Psi(\mathbf{u}_k).$$

This equation generates a sequence of approximate solutions $\mathbf{u}_k$ starting from an arbitrary initial solution $\mathbf{u}_0$. From the gradient of $\Psi$, we can find an appropriate direction and the step size $\tau_{k+1}$, so that the sequence $\mathbf{u}_k$ could converge to $\mathbf{u}^*$.

To find the gradient of $\Psi$, recall the following lemma.

**Lemma 3.1** (e.g., [22]). *For compatible rectangular matrices* $A, B, X$, *the gradient formulas hold:*

$$\frac{d}{dX}\operatorname{tr}(AX) = \frac{d}{dX}\operatorname{tr}(XA) = A^T, \qquad \frac{d}{dX}\operatorname{tr}(AXX^TB) = (BA + A^TB^T)X.$$

It follows from Lemma 3.1 that

$$\begin{aligned} \nabla\Psi(\mathbf{u}) &= \frac{d}{d\mathbf{u}}(T\mathbf{u}-\mathbf{f})^T(T\mathbf{u}-\mathbf{f}) = \frac{d}{d\mathbf{u}}\operatorname{tr}((T\mathbf{u}-\mathbf{f})(T\mathbf{u}-\mathbf{f})^T) \\ &= \frac{d}{d\mathbf{u}}\operatorname{tr}(T\mathbf{u}\mathbf{u}^TT^T - \mathbf{f}\mathbf{u}^TT^T - T\mathbf{u}\mathbf{f}^T + \mathbf{f}\mathbf{f}^T) \\ &= \frac{d}{d\mathbf{u}}\operatorname{tr}(T\mathbf{u}\mathbf{u}^TT^T) - \frac{d}{d\mathbf{u}}\operatorname{tr}(\mathbf{u}\mathbf{f}^TT) - \frac{d}{d\mathbf{u}}\operatorname{tr}(\mathbf{f}^TT\mathbf{u}) + \frac{d}{d\mathbf{u}}\operatorname{tr}(\mathbf{f}\mathbf{f}^T) \\ &= (T^TT + T^TT)\mathbf{u} - T^T\mathbf{f} - T^T\mathbf{f} = 2T^T(T\mathbf{u}-\mathbf{f}). \end{aligned}$$

Hence, we have the iterative equation as follows:

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \tau_{k+1}T^T(\mathbf{f} - T\mathbf{u}_k). \tag{3.1}$$

In accordance with the gradient-descent, the step size $\tau_{k+1}$ is generated to minimize the error occurring at each iteration. We will examine an error at step $k+1$ by the following function

$$\psi_{k+1}(\tau) := \|T(\mathbf{u}_k + \tau T^T(\mathbf{f} - T\mathbf{u}_k)) - \mathbf{f}\|_F^2, \quad \tau \geqslant 0.$$

We compute the derivative of $\psi_{k+1}$ as follows:

$$\frac{d}{d\tau}\psi_{k+1}(\tau) = 2\tau\,\mathrm{tr}(TT^T(\mathbf{f}-T\mathbf{u}_k)(\mathbf{f}-T\mathbf{u}_k)^T TT^T) - 2\,\mathrm{tr}(TT^T(\mathbf{f}-T\mathbf{u}_k)(\mathbf{f}-T\mathbf{u}_k)^T).$$

We now check the second derivative

$$\frac{d^2}{d\tau^2}\psi_{k+1}(\tau) = 2\|TT^T(\mathbf{f}-T\mathbf{u}_k)\|_F^2,$$

which is positive. Thus, the optimal step size can be obtained by solving the equation $\psi'_{k+1}(\tau) = 0$ to get the minimizer

$$\tau_{k+1} = \frac{\mathrm{tr}(TT^T(\mathbf{f}-T\mathbf{u}_k)(\mathbf{f}-T\mathbf{u}_k)^T)}{\mathrm{tr}(TT^T(\mathbf{f}-T\mathbf{u}_k)(\mathbf{f}-T\mathbf{u}_k)^T TT^T)} = \frac{\|T^T(\mathbf{f}-T\mathbf{u}_k)\|_F^2}{\|TT^T(\mathbf{f}-T\mathbf{u}_k)\|_F^2}.$$

We can avoid duplicated multiplications for the term $\|T^T(\mathbf{f}-T\mathbf{u}_k)\|_F^2$ as follows. Let us write the matrix $V = T^T T = [v_{ij}]$ and the vector

$$\mathbf{y} = T^T\mathbf{f} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_{N_x-1} \end{bmatrix}, \quad \text{where} \quad \mathbf{y}_i = \begin{bmatrix} y_{(i-1)N_t+1} \\ y_{(i-1)N_t+2} \\ \vdots \\ y_{iN_t} \end{bmatrix} \quad \text{for} \quad i = 1,2,\dots,N_x-1.$$

Since $T$ is lower-triangular, we can describe $\mathbf{y}_i$ more precisely. Indeed, for each $i = 1,2,\dots,N_x-1$, denote $\mathbf{f}_i = [f_{i-1,1}\ f_{i-1,2}\ \dots f_{i-1,N_t}]^T$. It follows that

$$\mathbf{y}_i = \begin{cases} \mathbf{f}_i + G^T\mathbf{f}_{i+1} + \sum_{r=i+2}^{N_x-1} g_{r-1}\mathbf{f}_r, & \text{for } i = 1,2,\dots,N_x-3, \\ \mathbf{f}_i + G^T\mathbf{f}_{i+1}, & \text{for } i = N_x-2, \\ \mathbf{f}_i, & \text{for } i = N_x-1. \end{cases}$$

The approximate solution at k-th step is expressed as $\mathbf{u}_k = [u_1^{(k)}\ u_2^{(k)}\ \dots\ u_N^{(k)}]^T$. A direct computation reveals that

$$T^T(\mathbf{f}-T\mathbf{u}_k) = \mathbf{y} - V\mathbf{u}_k = \begin{bmatrix} y_1 - \sum_{i=1}^N v_{1i}u_i^{(k)} \\ \vdots \\ y_N - \sum_{i=1}^N v_{Ni}u_i^{(k)} \end{bmatrix},$$

so that

$$\|T^T(\mathbf{f}-T\mathbf{u}_k)\|_F^2 = \sum_{i=1}^N (y_i - \sum_{j=1}^N v_{ij}u_j^{(k)})^2.$$

Similarly, by denoting $\mathbf{z} = T\mathbf{y} = [z_1\ z_2\ \dots\ z_N]^T$ and $W = TV = [w_{ij}]$ we obtain

$$\|TT^T(\mathbf{f}-T\mathbf{u}_k)\|_F^2 = \sum_{i=1}^N (z_i - \sum_{j=1}^N w_{ij}u_j^{(k)})^2.$$

From the direction and the step size, we can derive the following iterative procedure.

**Algorithm 1:** A gradient-descent iterative algorithm for Eq. (2.4)

```
Input: $T$, $\mathbf{f}$, $\epsilon$, $\mathbf{u}_0$
Compute
$\mathbf{y}_i = \begin{cases} \mathbf{f}_i + G^T\mathbf{f}_{i+1} + \sum_{r=i+2}^{N_x-1} g_{r-1}\mathbf{f}_r, & \text{for } i = 1,2,\dots,N_x-3, \\ \mathbf{f}_i + G^T\mathbf{f}_{i+1}, & \text{for } i = N_x-2, \\ \mathbf{f}_i, & \text{for } i = N_x-1; \end{cases}$
$V = T^T T$;
$\mathbf{z} = T\mathbf{y}$;
$W = TV$;
Set $k := 0$;
while $\|T\mathbf{u}_k - \mathbf{f}\|_F \geqslant \epsilon$ do
    $\tau_{k+1} = \sum_{i=1}^N (y_i - \sum_{j=1}^N v_{ij}u_j^{(k)})^2 / \sum_{i=1}^N (z_i - \sum_{j=1}^N w_{ij}u_j^{(k)})^2$;
    $\mathbf{u}_{k+1} = \mathbf{u}_k + \tau_{k+1}(\mathbf{y} - V\mathbf{u}_k)$;
    update $k$;
end
```

*Remark* 3.2. Despite the relative error $\|T\mathbf{u}_k - \mathbf{f}\|_F$, we can use another stopping rule concerning the absolute error $\|\mathbf{u}_k - \mathbf{u}_{k-1}\|_F$.

## 4. Convergence analysis of the algorithm

In this section, we investigate the applicability, the convergence rate, and error estimates of the proposed algorithm.

### 4.1. *The applicability of the algorithm*

Now, we study the convergence properties of Algorithm 1. Let us recall the notion of a strongly convex function.

**Definition 4.1.** Let $\Phi : \mathbb{R}^n \to \mathbb{R}$ be such that the second-order derivative $\nabla^2\Phi(x)$ is a symmetric matrix. We say that $\Phi$ is a strongly convex function if there exist two nonnegative constants $r$ and $s$ such that the matrix ordering $rI \prec \nabla^2\Phi(x) \prec sI$ holds for all $x \in \mathbb{R}^n$. Here, the partial order $\prec$ means that the scalar inequalities $r\|w\|_F^2 \leqslant w^T\nabla^2 s(x)w \leqslant s\|w\|_F^2$ hold for any vector $w \in \mathbb{R}^n$.

The following lemma gives an important property of the strongly convex function.

**Lemma 4.2** (see, e.g., [2]). *Let $\Phi : \mathbb{R}^n \to \mathbb{R}$ be a strongly convex function according to Definition 4.1. Then, for any $u, v \in \mathbb{R}^n$, we have*

$$\Phi(v) \leqslant \Phi(u) + \nabla\Phi(u)^T(v-u) + \frac{s}{2}\|v-u\|_F^2, \tag{4.1}$$

$$\Phi(v) \geqslant \Phi(u) + \nabla\Phi(u)^T(v-u) + \frac{r}{2}\|v-u\|_F^2. \tag{4.2}$$

**Theorem 4.3.** *For any given initial vector $\mathbf{u}_0$, Algorithm 1 produces a sequence $\{\mathbf{u}_k\}$ of approximate solutions converging to a unique solution $\mathbf{u}^*$.*

*Proof.* Since the coefficient matrix $T$ is invertible, the linear system $T\mathbf{u} = \mathbf{f}$ has a unique solution $\mathbf{u}^*$. We will prove that $\mathbf{u}_k \to \mathbf{u}^*$ as $k \to \infty$. At some iteration number $k$, if the gradient $\nabla\Psi(\mathbf{u}_k) = 2T^T(T\mathbf{u}_k - f)$ equals to zero, then the invertibility of $T$ implies that $\mathbf{u}_k$ is the desire solution. Now, suppose that $\nabla\Psi(\mathbf{u}_k) \neq 0$ for all $k$. Note that the second-order derivative $\nabla^2\Psi(\mathbf{u}) = 2T^T T$ is always a positive semidefinite symmetric matrix. Since $T$ is invertible, the matrix $T^T T$ is positive definite. Denote by $\lambda$ and $\mu$ the maximum and the minimum eigenvalues of $T^TT$, respectively. It follows that $0 < \mu \leqslant \lambda$ and the following matrix inequalities hold:

$$2\mu I \prec \nabla^2\Psi(\mathbf{u}) \prec 2\lambda I,$$

meaning that $\Psi$ is a strongly convex function. Note that from the recursive equation (3.1), we have

$$\mathbf{u}_{k+1} - \mathbf{u}_k = \tau_{k+1}T^T(f - T\mathbf{u}_k).$$

Now, applying Lemma 4.2 to the function $\Psi$, we have from Eq. (4.1) that

$$\Psi(\mathbf{u}_{k+1}) \leqslant \Psi(\mathbf{u}_k) - 2\tau\|\nabla\Psi(\mathbf{u}_k)\|_F^2 + \frac{\lambda\tau^2}{4}\|\nabla\Psi(\mathbf{u}_k)\|_F^2. \tag{4.3}$$

Consider the right-hand side of (4.3) as a real-valued function $f$ in a variable $\tau \geqslant 0$, then the function $f$ has a minimizer given by $\tau = 4/\lambda$. It follows that

$$\Psi(\mathbf{u}_{k+1}) \leqslant \min_{\tau\geqslant 0} f(\tau) = \Psi(\mathbf{u}_k) - \frac{4}{\lambda}\|\nabla\Psi(\mathbf{u}_k)\|_F^2. \tag{4.4}$$

On the other hand, applying Eq. (4.2) to the function $\Psi$, we have

$$\Psi(\mathbf{u}_{k+1}) \geqslant \Psi(\mathbf{u}_k) - 2\tau\|\nabla\Psi(\mathbf{u}_k)\|_F^2 + \frac{\mu\tau^2}{4}\|\nabla\Psi(\mathbf{u}_k)\|_F^2. \tag{4.5}$$

Similarly, a minimizer of the right-hand side of (4.5) is given by $\tau = 4/\mu$. We continue this process to obtain

$$0 \geqslant \Psi(\mathbf{u}_k) - \frac{4}{\mu}\|\nabla\Psi(\mathbf{u}_k)\|_F^2$$

and thus $\|\nabla\Psi(\mathbf{u}_k)\|_F^2 \geqslant \mu\Psi(\mathbf{u}_k)/4$. Hence, by taking account of (4.4), we get

$$\Psi(\mathbf{u}_{k+1}) \leqslant (1 - \frac{\mu}{\lambda})\Psi(\mathbf{u}_k). \tag{4.6}$$

It follows inductively that for any natural number $k$,

$$\Psi(\mathbf{u}_k) \leqslant (1 - \frac{\mu}{\lambda})^k\Psi(\mathbf{u}_0). \tag{4.7}$$

Since $0 < 1 - (\mu/\lambda) < 1$, we conclude that $\Psi(\mathbf{u}_k) \to 0$, i.e., $\mathbf{u}_k \to \mathbf{u}^*$ as $k \to \infty$. □

### 4.2. *Theoretical performance of the algorithm*

We now discuss theoretical performance of Algorithm 1 through the asymptotic convergence rate and error estimates. Let us recall relevant matrix norms. The Euclidean vector norm is a special case of the Frobenius norm, which is defined for any rectangular matrix $A$ by $\|A\|_F := \sqrt{\operatorname{tr}(A^TA)}$. The spectral norm $\|\cdot\|_2$ of a square matrix is indeed the submultiplicative matrix norm compatible with the Euclidean vector norm in the sense that the scalar inequality

$$\|Ax\|_F \leqslant \|A\|_2\,\|x\|_F \tag{4.8}$$

holds for every square matrix $A$ and every vector $x$ of compatible sizes; see, e.g., [22].

We will use the letter $\kappa$ to denote the condition number of the coefficient $T$, which is defined by

$$\kappa = \sqrt{\frac{\lambda_{\max}(T^TT)}{\lambda_{\min}(T^TT)}} = \sqrt{\frac{\lambda}{\mu}}.$$

On account of the bounds (4.6) and (4.7), it follows that

$$\|T\mathbf{u}_k - \mathbf{f}\|_F \leqslant (1-\kappa^{-2})^{\frac{1}{2}} \|T\mathbf{u}_{k-1} - \mathbf{f}\|_F, \tag{4.9}$$
$$\|T\mathbf{u}_k - \mathbf{f}\|_F \leqslant (1-\kappa^{-2})^{\frac{k}{2}} \|T\mathbf{u}_0 - \mathbf{f}\|_F. \tag{4.10}$$

We state the above results as follows.

**Corollary 4.4.** *Let $\kappa$ be the condition number of $T$. Then the convergence rate of Algorithm 1 (with respect to the relative error $\|T\mathbf{u}_k - f\|_F$) is regulated by the factor $\sqrt{1-\kappa^{-2}}$. In addition, Eqs. (4.9) and (4.10) describe the relative-error estimates compared to the previous step and the initial step, respectively. To be more precise, the relative error decreases at every iteration by Eq. (4.4).*

**Corollary 4.5.** *Algorithm 1 has the asymptotic rate of convergence with respect to the absolute error governed by $\sqrt{1-\kappa^{-2}}$. In addition, the error estimate $\|\mathbf{u}_k - \mathbf{u}^*\|_F$ is described as follows:*

$$\|\mathbf{u}_k - \mathbf{u}^*\|_F \leqslant \kappa(1-\kappa^{-2})^{\frac{k}{2}} \|\mathbf{u}_0 - \mathbf{u}^*\|_F, \tag{4.11}$$
$$\|\mathbf{u}_k - \mathbf{u}^*\|_F \leqslant \sqrt{\kappa^2 - 1} \|\mathbf{u}_{k-1} - \mathbf{u}^*\|_F. \tag{4.12}$$

*Proof.* From the estimate (4.10), we can apply the property (4.8) to get

$$\begin{aligned}\|\mathbf{u}_k - \mathbf{u}^*\|_F = \|T^{-1}T\mathbf{u}_k - T^{-1}T\mathbf{u}^*\|_F &\leqslant \|T^{-1}\|_2 \|T\mathbf{u}_k - \mathbf{f}\|_F \\ &\leqslant \|T^{-1}\|_2 (1-\kappa^{-2})^{\frac{k}{2}} \|T\mathbf{u}_0 - \mathbf{f}\|_F \leqslant \|T^{-1}\|_2 \|T\|_2 (1-\kappa^{-2})^{\frac{k}{2}} \|\mathbf{u}_0 - \mathbf{u}^*\|_F.\end{aligned}$$

Since $T$ is invertible, we have (e.g., [22])

$$\kappa = \|T\|_2 \cdot \|T^{-1}\|_2.$$

Thus, the inequality (4.11) holds. Similarly, the desired inequality (4.12) comes from Eq. (4.9). □

From the results in this subsection, we summarize that the condition number of the coefficient matrix $T$ effects the performance of the algorithm. Indeed, the proposed algorithm converges fast to the desired solution if the condition number is close to 1.

## 5. A treatment for the space fractional diffusion equation

In this section, we investigate a space fractional diffusion equation

$$\frac{\partial u(x,t)}{\partial t} - {}^{RL}D_x^{\alpha} u(x,t) = f(x,t), \quad \alpha \in (1,2], x \in [a,b], t \in [0,T]. \tag{5.1}$$

Here, we impose the initial condition $u(x,0) = 0$, and the boundary conditions $u(a,t) = 0$, $u(b,t) = \tilde{b}$, where $\tilde{b}$ is a given function. Eq. (5.1) is a particular case of Eq. (1.1) for which the Caputo fractional order $\beta$ is equal to 1. We partition the spatial domain $[a,b]$ and the time domain $[0,T]$ as those in Section 2, so that we can compute an approximate solution at $(x_i, t_j)$ with $x_i = a + ih_x$ and $t_j = jh_t$, where $h_x$ and $h_t$ are defined as (2.1). The partial derivative $\partial u(x,t)/\partial t$ is approximated by the forward time difference method. The Riemann-Liouville derivative ${}^{RL}D_x^{\alpha} u(x,t)$ is approximated by the Grünwald-Letnikov approximation (1.2)-(1.3). Thus, for each $i \in \{0,1,\ldots,N_x-2\}$ and $j \in \{0,1,\ldots,N_t-1\}$, we have

$$\frac{u_{i,j+1} - u_{ij}}{h_t} - \frac{1}{h_x^{\alpha}} \sum_{k=0}^{i+1} g_{\alpha,k} u_{i-k+1,j+1} = f_{i,j+1}$$

or equivalently,

$$u_{i+1,j+1} = \frac{h_x^\alpha}{g_{\alpha,0} h_t}(u_{i,j+1} - u_{ij} - h_t f_{i,j+1}) - \frac{1}{g_{\alpha,0}} \sum_{k=1}^{i+1} g_{\alpha,k} u_{i-k+1,j+1}. \tag{5.2}$$

We can transform a system of $N := (N_x - 1)N_t$ linear equations (5.2) in $N$ variables $u_{11}, \dots, u_{N_x-1,N_t}$ into a linear system

$$T^* \mathbf{u} = \mathbf{f},$$

where $T^*$ is a block lower-triangular matrix and $\mathbf{u}, \mathbf{f}$ are vectors as follows:

$$\left[\begin{array}{c|c|c|c|c} I_{N_t} & 0 & 0 & \cdots & 0 \\ \hline \tilde{G} & I_{N_t} & 0 & \cdots & 0 \\ \hline g_2 I_{N_t} & \tilde{G} & I_{N_t} & \cdots & 0 \\ \hline \vdots & \ddots & \ddots & \ddots & \vdots \\ \hline g_{N_x-2} I_{N_t} & \cdots & g_2 I_{N_t} & \tilde{G} & I_{N_t} \end{array}\right] \begin{bmatrix} u_{11} \\ u_{12} \\ \vdots \\ u_{N_x-1,N_t} \end{bmatrix} = \hat{h} \begin{bmatrix} f_{01} \\ f_{02} \\ \vdots \\ f_{N_x-2,N_t} \end{bmatrix}.$$

Here, $\tilde{G}$ is a tridiagonal matrix of size $N_t \times N_t$ viewed as:

$$\tilde{G} = \begin{bmatrix} g_1 - \bar{h} & 0 & & 0 \\ \bar{h} & g_1 - \bar{h} & 0 & \\ & \ddots & \ddots & \ddots \\ 0 & & \bar{h} & g_1 - \bar{h} \end{bmatrix}.$$

Note that T is still sparse and invertible. Therefore, the space-fractional diffusion equation can be solved by Algorithm 1, where $T^*$ is used instead of T.

## 6. Numerical simulations

In this section, we carry out numerical simulations to perform the capability and effectiveness of the proposed algorithm. We compare Algorithm 1 (denoted by TauOpt) with the following well-known iterative methods for linear systems: gradient-based iterative (GI) algorithm [9], least-squares iterative (LSI) algorithm [10], Jacobi over relaxation (JOR) algorithm [35], and the classical SOR algorithm [34].

We implement all simulations using MATLAB R2020b on the same PC environment: Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz with RAM 8.00 GB. At the $k$-th iteration step, we concern the relative error $\|T\mathbf{u}_k - f\|_F$. We use tic and toc functions in MATLAB to measure the computational time (CT: in seconds) consumed for each simulation.

Recall that for each positive parameter $c$ and real parameter $d$, the $(c, d)$-Mittag-Leffler function is a special function expressed in the form of a convergent Taylor series:

$$\mathcal{E}_{c,d}(x) = \sum_{k=0}^{\infty} \frac{x^k}{\Gamma(ck + d)}.$$

Recall that the following derivative formulas hold (see, e.g., [25]):

$$ {}^C D_x^p x^\beta = \frac{\Gamma(\beta+1)}{\Gamma(\beta - p + 1)} x^{\beta - p}, \qquad 0 < p < 1, \beta < 1 \text{ and } x > 0, \tag{6.1}$$

$$ {}^{RL} D_x^p \cos(x) = x^{-p} \mathcal{E}_{2,1-p}(-x^2), \qquad p > 0 \text{ and } x > 0. \tag{6.2}$$

**Example 6.1.** Let us consider the space-time fractional diffusion equation

$$ {}^C D_t^{1/3} u(x,t) - {}^{RL} D_x^{3/2} u(x,t) = 1.1077 t^{2/3} \cos(x) - t x^{-1.5} \mathcal{E}_{2,-0.5}(-x^2)$$

on the spatial domain $[\pi/2, \pi]$ and the time domain $[0, 0.1]$. From (6.1) and (6.2), we can check that the exact solution of this equation is

$$u^*(x,t) = t\cos x.$$

Choosing the numbers of partitions $N_x = 20$ and $N_t = 20$, so that the coefficient matrix $T$ is of size $380 \times 380$. The comparison of approximate solutions and the exact solution are numerically shown in Table 1. We see that the 4-digits approximate solutions at the chosen values of the variables $x$ and $t$ are very close to the exact solution. Also, Figure 1 illustrates the three-dimensional plot of the exact and approximate solutions. Besides, we compare the proposed algorithm with GI, LSI, SOR, and JOR algorithms. To do this, we remove their stopping rules and execute the same number of iterations. The results of running each algorithm for 100 iterations are shown in Table 2 and Figure 2. The numerical and graphical results imply that the proposed algorithm performs well in both relative errors and computational time.

Table 1: The approximate and exact solutions for Example 6.1.

| | $u(x,t)$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $x = 0.55\pi$ | | $x = 0.65\pi$ | | $x = 0.75\pi$ | | $x = 0.85\pi$ | |
| $t$ | approx. | exact | approx. | exact | approx. | exact | approx. | exact |
| 0.005 | -0.0019 | -0.0020 | -0.0128 | -0.0131 | -0.0424 | -0.0434 | -0.2036 | -0.2049 |
| 0.010 | -0.0033 | -0.0033 | -0.0198 | -0.0196 | -0.0591 | -0.0583 | -0.2214 | -0.2197 |
| 0.015 | -0.0043 | -0.0043 | -0.0252 | -0.0252 | -0.0717 | -0.0715 | -0.2488 | -0.2492 |
| 0.020 | -0.0053 | -0.0053 | -0.0301 | -0.0301 | -0.0831 | -0.0831 | -0.2756 | -0.2757 |
| 0.025 | -0.0062 | -0.0062 | -0.0347 | -0.0347 | -0.0936 | -0.0937 | -0.2999 | -0.2999 |
| 0.030 | -0.0071 | -0.0071 | -0.0389 | -0.0389 | -0.1036 | -0.1036 | -0.3225 | -0.3224 |
| 0.035 | -0.0078 | -0.0078 | -0.0430 | -0.0430 | -0.1128 | -0.1129 | -0.3435 | -0.3435 |
| 0.040 | -0.0086 | -0.0086 | -0.0469 | -0.0469 | -0.1217 | -0.1218 | -0.3636 | -0.3635 |
| 0.045 | -0.0094 | -0.0094 | -0.0506 | -0.0506 | -0.1302 | -0.1303 | -0.3826 | -0.3826 |
| 0.050 | -0.0102 | -0.0101 | -0.0543 | -0.0543 | -0.1384 | -0.1385 | -0.4009 | -0.4009 |

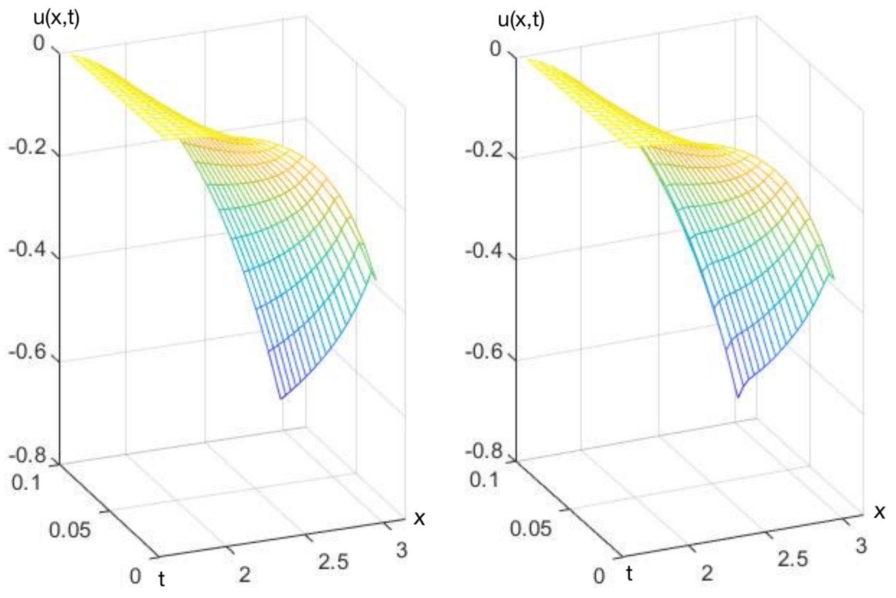

Figure 1: The 3D plot of the exact (left) and approximate (right) solutions for Example 6.1.

Table 2: Relative error and computational time for Example 6.1.

| Method | Relative error | CT |
|---|---|---|
| TauOpt | 0.0616 | 0.0231 |
| SOR | 0.0703 | 1.1119 |
| JOR | 0.0722 | 1.4135 |
| GI | 0.0697 | 0.0744 |
| LSI | 0.0692 | 1.1329 |

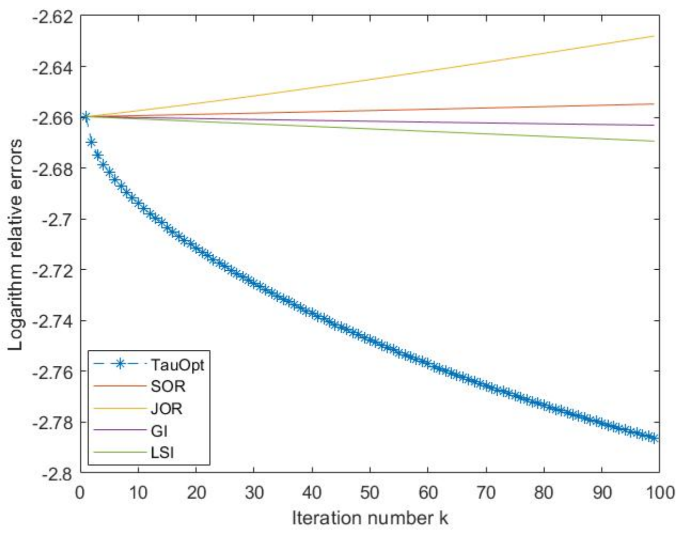

Figure 2: The relative-error plot for Example 6.1.

**Example 6.2.** Consider the space-fractional diffusion equation

$$\frac{\partial u(x,t)}{\partial t} - {}^{RL}D_x^{4/3}u(x,t) \; = \; 6t\cos x - 3t^2x^{-4/3}\mathcal{E}_{2,-1/3}(-x^2)$$

on the spatial domain $[\pi/2, \pi]$ and the time domain $[0, 0.1]$. From (6.2), we can check that the exact solution of this equation is given by

$$u^*(x,t) \; = \; 3t^2\cos x.$$

We choose the numbers of partitions of $x$ and $t$ by $N_x = 20$ and $N_t = 20$, respectively. Apparently, the size of the coefficient matrix $T$ is of $380 \times 380$. The results after running Algorithm 1 are illustrated numerically in Table 3 and graphically in Figure 3. Both of them imply that the approximate solutions are very close to the exact solution. Moreover, we collate the proposed algorithm with GI, LSI, SOR, and JOR algorithms. The results after executing 100 iterations are numerically shown in Table 4. Figure 4 displays the comparison of the logarithm relative errors between the mentioned algorithms. In particular, the relative error for the proposed algorithm is decreasing at every iteration, while the errors for other algorithms may not be decreased or decreased slowly. Both numerical and graphical illustrations indicate that the proposed algorithm outperforms other mentioned algorithms.

Table 3: The approximate and exact solutions for Example 6.2.

| | $u(x,t)$ | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $x = 0.55\pi$ | | $x = 0.65\pi$ | | $x = 0.75\pi$ | | $x = 0.85\pi$ | |
| t | approx. | exact | approx. | exact | approx. | exact | approx. | exact |
| 0.005 | -0.0000 | -0.0000 | -0.0001 | -0.0000 | -0.0001 | -0.0000 | -0.0001 | -0.0001 |
| 0.010 | -0.0000 | -0.0000 | -0.0001 | -0.0001 | -0.0003 | -0.0002 | -0.0004 | -0.0003 |
| 0.015 | -0.0000 | -0.0001 | -0.0004 | -0.0003 | -0.0006 | -0.0005 | -0.0008 | -0.0006 |
| 0.020 | -0.0001 | -0.0002 | -0.0006 | -0.0005 | -0.0011 | -0.0009 | -0.0013 | -0.0011 |
| 0.025 | -0.0002 | -0.0003 | -0.0010 | -0.0009 | -0.0016 | -0.0013 | -0.0019 | -0.0017 |
| 0.030 | -0.0003 | -0.0004 | -0.0014 | -0.0012 | -0.0022 | -0.0019 | -0.0027 | -0.0024 |
| 0.035 | -0.0005 | -0.0006 | -0.0019 | -0.0017 | -0.0029 | -0.0026 | -0.0036 | -0.0033 |
| 0.040 | -0.0006 | -0.0007 | -0.0024 | -0.0022 | -0.0038 | -0.0034 | -0.0047 | -0.0043 |
| 0.045 | -0.0008 | -0.0009 | -0.0031 | -0.0028 | -0.0048 | -0.0043 | -0.0058 | -0.0054 |
| 0.050 | -0.0011 | -0.0011 | -0.0038 | -0.0034 | -0.0058 | -0.0053 | -0.0071 | -0.0067 |

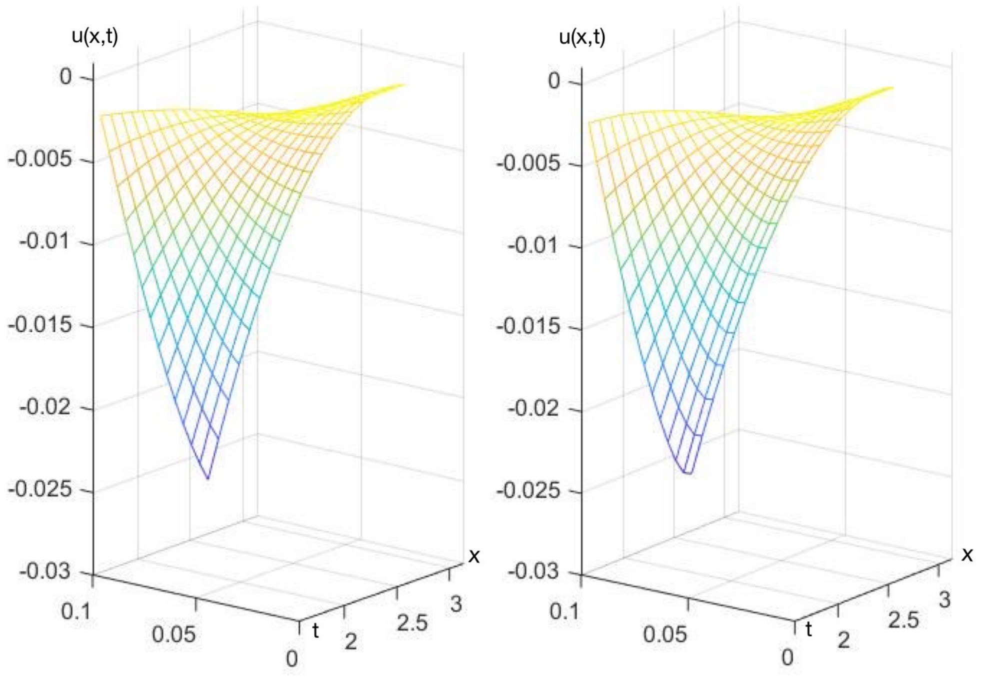

Figure 3: The 3D plot of the exact (left) and approximate (right) solutions for Example 6.2.

Table 4: Relative error and computational time for Example 6.2.

| Method | Relative error | CT |
|---|---|---|
| TauOpt | 0.0718 | 0.0197 |
| SOR | 0.1580 | 0.9343 |
| JOR | 0.1609 | 1.3269 |
| GI | 0.1501 | 0.0824 |
| LSI | 0.1438 | 1.3419 |

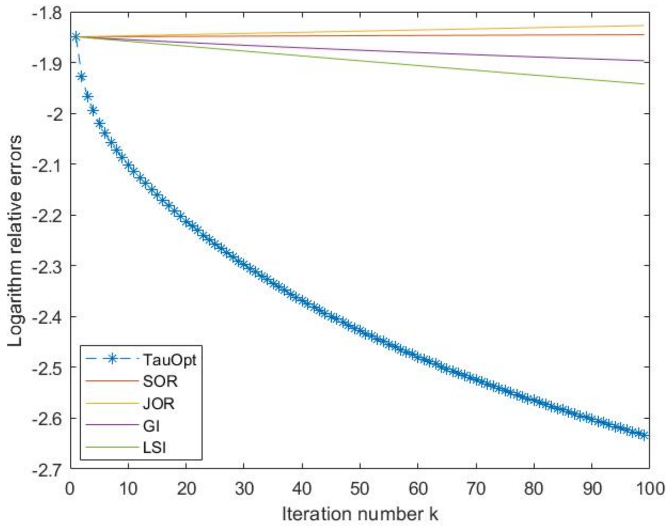

Figure 4: The logarithm relative-error plot for Example 6.2.

## 7. Conclusion

This paper develops a discretization of the space-time fractional diffusion equation in which the approximate equation is done by the finite difference scheme of Grünwald-Letnikov approximation. We transform the set of approximate equations into a single linear system with a sparse invertible coefficient matrix T. We then propose an iterative procedure based on the gradient-descent technique to solve the linear system. Convergence analysis insists that the proposed algorithm is always applicable with satisfactory convergence rate and error estimates. Moreover, we examine an interesting particular case, namely, the space fractional diffusion equation. Since T is sparse and the procedure avoid duplicated computations, the proposed algorithm consumes not so much time in each iteration. The numerical experiments verify that the proposed algorithm performs well in both computational time and relative errors, compared to well-known iterative methods for linear systems.

## Acknowledgment

This research project is supported by National Research Council of Thailand (NRCT): (N41A640234).

## References

[1] K. Assaleh, W. M. Ahmad, *Modeling of speech signals using fractional calculus*, 2007 9th International Symposium on Signal Processing and Its Applications, (2007), 1–4. 1

[2] S. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, (2004). 4.2

[3] A. V. Chechkin, R. Gorenflo, I. M. Sokolov, *Fractional diffusion in inhomogeneous media*, J. Phys. A, **38** (2005), L679–L684. 1

[4] Z.-Q. Chen, M. M. Meerschaert, E. Nane, *Space-time fractional diffusion on bounded domains*, J. Math. Anal. Appl., **393** (2012), 479–488. 1

[5] L. Chen, R. H. Nochetto, E. Otárola, A. J. Salgado, *A PDE approach to fractional diffusion: a posteriori error analysis*, J. Comput. Phys., **293** (2015), 339–358. 1

[6] M. Cui, *Compact finite difference method for the fractional diffusion equation*, J. Comput. Phys., **228** (2009), 7792–7804. 1

[7] K. Diethelm, J. M. Ford, N. J. Ford, M. Weilbeer, *Pitfalls in fast numerical solvers for fractional differential equations*, J. Comput. Appl. Math., **186** (2006), 482–503.

[8] K. Diethelm, N. J. Ford, A. D. Freed, *Detailed error analysis for a fractional Adams method*, Numer. Algorithms, **36** (2004), 31–52. 1

[9] F. Ding, T. Chen, *Iterative least-squares solutions of coupled Sylvester matrix equations*, Systems Control Lett., **54** (2005), 95–107. 6

[10] F. Ding, T. Chen, *On iterative solutions of general coupled matrix equations*, SIAM J. Control Optim., **44** (2006), 2269–2284. 6

[11] R. Du, W. R. Cao, Z. Z. Sun, *A compact difference scheme for the fractional diffusion-wave equation*, Appl. Math. Model., **34** (2010), 2998–3007. 1

[12] R. Gorenflo, F. Mainardi, *Random walk models for space-fractional diffusion processes*, Fract. Calc. Appl. Anal., **1** (1998), 167–191. 1

[13] R. Gorenflo, F. Mainardi, *Approximation of Lévy-Feller diffusion by random walk*, Z. Anal. Anwendungen, **18** (1999), 231–246.

[14] R. Gorenflo, F. Mainardi, D. Moretti, P. Paradisi, *Time fractional diffusion: a discrete random walk approach*, Nonlinear Dynam., **29** (2002), 129–143.

[15] R. Gorenflo, F. Mainardi, D. Moretti, G. Pagnini, P. Paradisi, *Discrete random walk models for space-time fractional diffusion*, Chem. Phys., **284** (2002), 521–541. 1

[16] R. Hilfer, *Applications of fractional calculus in physics*, World Scientific Publishing, (2000). 1

[17] J. Huang, F. Liu, *The space–time fractional diffusion equation with Caputo derivatives*, J. Appl. Math. Comput., **19** (2005), 179–190. 1

[18] I. S. Jesus, J. A. T. Machado, J. B. Cunha, *Fractional electrical impedances in botanical elements*, J. Vib. Control, **14** (2008), 1389–1402. 1

[19] A. Kittisopaporn, P. Chansangiam, *Gradient-descent iterative algorithm for solving a class of linear matrix equations with applications to heat and Poisson equations*, Adv. Difference Equ., **2020** (2020), 24 pages. 1

[20] V. V. Kulish, J. L. Lage, *Application of fractional calculus to fluid mechanics*, J. Fluids Eng., **124** (2002), 803–806. 1

[21] P. Kumar, O. P. Agrawal, *Numerical scheme for the solution of fractional differential equations of order greater than one*, J. Comput. Nonlinear Dyn., **1** (2006), 178–185. 1

[22] H. Lütkepohl, *Handbook of matrices*, John Wiley & Sons, (1996). 3.1, 4.2, 4.2

[23] R. L. Magin, O. Abdullah, D. Baleanu, X. J. Zhou, *Anomalous diffusion expressed through fractional order differential operators in the Bloch-Torrey equation*, J. Magn. Reson., **190** (2008), 255–270. 1

[24] M. M. Meerschaert, D. A. Benson, H.-P. Scheffler, B. Baeumer, *Stochastic solution of space-time fractional diffusion equations*, Phys. Rev. E, **65** (2002), 4 pages. 1

[25] K. Miller, B. Roos, *An Introduction to the Fractional calculus and Fractional differential Equations*, John Wiley & Sons, New York, (1993). 1, 6

[26] J. Mua, B. Ahmad, S. Huang, *Existence and regularity of solutions to time-fractional diffusion equations*, Comput. Math. Appl., **73** (2017), 985–996. 1

[27] R. H. Nochetto, E. Otárola, A. J. Salgado, *A PDE approach to fractional diffusion in general domains: a prior error analysis*, Found. Comput. Math., **15** (2015), 733–791. 1

[28] I. Podlubny, I. Petráš, B. M. Vinagre, P. O'Leary, L. Dorčák, *Analogue realizations of fractional-order controllers*, Nonlinear Dynam., **29** (2002), 281–296. 1

[29] F. Santamaria, S. Wils, E. De Schutter, G. J. Augustine, *Anomalous diffusion in Purkinje cell dendrites caused by spines*, Neuron, **52** (2006), 635–648. 1

[30] R. Scherer, S. L. Kalla, Y. Tang, J. Huang, *The Grünwald-Letnikov method for fractional differential equations*, Comput. Math. Appl., **62** (2011), 902–917. 1, 1

[31] Z.-Z. Sun, X. Wu, *A fully discrete difference scheme for a diffusion-wave system*, Appl. Numer. Math., **56** (2006), 193–209. 1

[32] C. Tadjeran, M. M. Meerschaert, *A second-order accurate numerical method for the two dimensional fractional diffusion equation*, J. Comput. Phys., **220** (2007), 813–823. 1

[33] S. Umarov, S. Steinberg, *Variable order differential equations with piecewise constant order-function and diffusion with changing modes*, Z. Anal. Anwend., **28** (2009), 431–450. 1

[34] D. M. Young, *Iterative Methods for solving partial differential equations of elliptic type*, Trans. Amer. Math. Soc., **76** (1954), 92–111. 6

[35] D. M. Young, *Iterative Solution of Large Linear Systems*, Academic Press, New York-London, (1971). 6