Mathematics **2015**, *3*, 1032-1044; doi:10.3390/math3041032

OPEN ACCESS

mathematics

ISSN 2227-7390

www.mdpi.com/journal/mathematics

Article

A Fast $O(N \log N)$ Finite Difference Method for the One-Dimensional Space-Fractional Diffusion Equation

Treena Basu

Department of Mathematics, Occidental College, Los Angeles, CA 90041, USA;
E-Mail: basu@oxy.edu; Tel.: +1-803-397-9736

Academic Editor: Indranil SenGupta

Received: 4 August 2015 / Accepted: 15 October 2015 / Published: 27 October 2015

Abstract: This paper proposes an approach for the space-fractional diffusion equation in one dimension. Since fractional differential operators are non-local, two main difficulties arise after discretization and solving using Gaussian elimination: how to handle the memory requirement of $O(N^2)$ for storing the dense or even full matrices that arise from application of numerical methods and how to manage the significant computational work count of $O(N^3)$ per time step, where N is the number of spatial grid points. In this paper, a fast iterative finite difference method is developed, which has a memory requirement of $O(N)$ and a computational cost of $O(N \log N)$ per iteration. Finally, some numerical results are shown to verify the accuracy and efficiency of the new method.

Keywords: circulant and toeplitz matrices; fast finite difference methods; fast fourier transform; fractional diffusion equations

1. Introduction

The history of fractional calculus is almost as long as integer calculus, however it is only in the last few decades that it has gained much importance. The modeling of a variety of non-classic phenomena, *i.e.*, anomalous diffusion using fractional differential equations have proven to be promising to describe processes with memory and hereditary in geophysics [1], physics [2], chemistry [3], biology [4] and even finance and economics [5]. The primary advantage of such modeling lies in the introduction of a parameter, namely the fractional order of the equation, which can be used to model non-Markovian behavior of spatial or temporal processes. While analytical methods, such as the Fourier transform method, the Laplace transform methods, and the Mellin transform method, have been developed to seek

closed-form analytical solutions for fractional partial differential equations [6], there are very few cases in which the closed-form analytical solutions are available, just like in the case of integer-order partial differential equations. Numerical methods for the fractional partial differential equations, such as finite difference methods [7], finite element methods [8,9], spectral methods [10], and discontinuous Galerkin methods [11,12] have recently been developed and remains a relatively new topic of research because of the difficulties encountered.

Due to the non-local nature of fractional differential operators, numerical methods for fractional diffusion equations [9,13,14] raise numerical difficulties that were not encountered in the numerical methods for second-order diffusion equations. A significant obstacle that is the direct result of this non-local behavior is that these methods generate discrete systems with full or dense coefficient matrices. Meerschaert and Tadjeran [15,16] utilized a shifted Grünwald-Letnikov difference approximation to develop an implicit Euler finite difference method for space-fractional diffsuion equation in one-dimension. Further, they proved that the method is unconditionally stable and has first-order convergence in space and time. However, these methods were solved via Gaussian elimination, consequently $O(N^3)$ account of operations and $O(N^2)$ account of storage are required to solve a problem of size N.

This work focuses on the development of a fast iterative finite difference method for the accurate and efficient solution of the one-dimensional space-fractional diffusion equation. The immense computational cost and storage requirement for the one-dimensional space fractional diffusion equation was broken down recently by the authors of [17]. In [17] they proved that the stiffness matrix of [15,16] can be decomposed as a sum of diagonal matrix multiplied by a Toepltiz matrix. They utilize this decomposition and applied an operator splitting technique to the one-dimensional space-fractional diffusion equation to develop a fast operator-splitting finite difference method for the space-fractional diffusion equation in one space dimension. However, this method has a computational work account of $O(N \log^2 N)$ per iteration and has a memory need of $O(N \log N)$ per time step, due to the use of the banded coefficient matrix. While this is a vast improvement from the traditional methods solved via Gaussian elimination, there is room for improvement. In this paper the proposed method retains the same accuracy as the regular finite difference methods solved via Gaussian elimination and the resulting fast algorithm has a computational cost of $O(N \log N)$ per iteration at each time step and a memory requirement of only $O(N)$ per time step.

The rest of this paper is organized as follows. Section 2 outlines the space-fractional diffusion equation we attempt to solve and presents the corresponding Meerschaert-Tadjeran finite difference method. Section 3, begins by discussing the impact of the significantly increased computational work and memory requirement of the traditional implicit finite difference method. Then we continue with the development of the fast conjugate gradient squared finite difference formulation. This section concludes by describing how to efficiently store the stiffness matrix and how to implement a fast matrix-vector multiplication to speed up the iterative scheme. Our work in this section establishes that the fast method has a computational work of $O(N \log N)$ per iteration and a memory requirement of $O(N)$ per time step, while retaining the same accuracy as the traditional finite difference methods. This is followed by numerical experiments in Section 4 and concluding remarks in Section 5.

2. The Implicit Finite Difference Method for Time-Dependent Space Fractional Diffusion Equation

Fractional order partial differential equations are generalizations of classical partial differential equations. Fractional derivatives in space are used to model anomalous diffusion, where particles spread either faster or slower than the classical model predicts. When a fractional derivative of order α replaces the second derivative in a diffusion model, it leads to enhanced diffusion if $1 < \alpha < 2$ (a process known as superdiffusion) or leads to subdiffusion, if $0 < \alpha < 1$.

For a one-dimensional fractional diffusion model with constant coefficients, analytic solutions are available [18] using Fourier transform methods. However many practical problems require a model with variable coefficients [19,20]. In [18], a space-fractional diffusion equation was used to describe Lévy flights.

We now proceed to develop a fast numerical method for space-fractional diffusion equations. Consider the following initial-boundary value problem of a class of time-dependent space-fractional diffusion of order $1 < \alpha < 2$ [15,16,18]

$$\begin{gathered} \frac{\partial u(x,t)}{\partial t} - d_+(x,t)\frac{\partial^\alpha u(x,t)}{\partial_+ x^\alpha} - d_-(x,t)\frac{\partial^\alpha u(x,t)}{\partial_- x^\alpha} = f(x,t) \\ x_L < x < x_R, \;\; 0 < t \le T \\ u(x_L,t) = 0, \quad u(x_R,t) = 0, \qquad 0 \le t \le T \\ u(x,0) = u_0(x), \qquad x_L \le x \le x_R \end{gathered} \tag{1}$$

The case $1 < \alpha < 2$ has useful applications [21]. It is also a physically meaningful case, as explained in [22]. A two-sided fractional partial differential equation allows modeling different flow regime impacts from either side. Here the left-sided $(+)$ and the right-sided $(-)$ fractional derivatives $\frac{\partial^\alpha u(x,t)}{\partial_+ x^\alpha}$ and $\frac{\partial^\alpha u(x,t)}{\partial_- x^\alpha}$ can be defined in the (computationally feasible) Grünwald-Letnikov form [23]

$$\begin{aligned} \frac{\partial^\alpha u(x,t)}{\partial_+ x^\alpha} &= \lim_{h\to 0^+} \frac{1}{h^\alpha} \sum_{k=0}^{\lfloor (x-x_L)/h \rfloor} g_k^{(\alpha)} u(x-kh,t) \\ \frac{\partial^\alpha u(x,t)}{\partial_- x^\alpha} &= \lim_{h\to 0^+} \frac{1}{h^\alpha} \sum_{k=0}^{\lfloor (x_R-x)/h \rfloor} g_k^{(\alpha)} u(x+kh,t) \end{aligned} \tag{2}$$

where $\lfloor x \rfloor$ represents the floor of x and $g_k^{(\alpha)} = (-1)^k \binom{\alpha}{k}$ with $\binom{\alpha}{k}$ being the fractional binomial coefficients. We note that the Grünwald weights $g_k^{(\alpha)}$ can be evaluated using the recurrence relation

$$g_0^{(\alpha)} = 1, \qquad g_k^{(\alpha)} = \Big(1 - \frac{\alpha+1}{k}\Big) g_{k-1}^{(\alpha)} \quad \text{for} \quad k \ge 1 \tag{3}$$

and satisfy the following properties [15,16,23]

$$\begin{cases} g_0^{(\alpha)} = 1, \qquad g_1^{(\alpha)} = -\alpha < 0, \qquad 1 \ge g_2^{(\alpha)} \ge g_3^{(\alpha)} \ge \cdots \ge 0 \\ \sum_{k=0}^{\infty} g_k^{(\alpha)} = 0, \qquad \sum_{k=0}^{m} g_k^{(\alpha)} \le 0 \quad (m \ge 1) \end{cases} \tag{4}$$

We also note that the left-handed fractional derivative of *u* at a point depends on all function values to the left of that point. Similarly, the right-handed fractional derivatives of *u* at a point depends on all function values to the right of this point. In other words, fractional derivatives are non-local operators.

This paper focuses on the development of a fast numerical method for problem Equation (1). We refer to [24], for the existence and uniqueness of the weak solution to fractional partial differential equations.

Let N and M be positive integers and $h = (x_R - x_L)/N$ and $\Delta t = T/M$ be the sizes of spatial grid and time step, respectively. We define a spatial and temporal partition $x_i = x_L + ih$ for $i = 0, 1, \ldots, N$ and $t^m = m\Delta t$ for $m = 0, 1, \ldots, M$. Let $u_i^m = u(x_i, t^m)$, $d_{+,i}^m = d_+(x_i, t^m)$, $d_{-,i}^m = d_-(x_i, t^m)$, and $f_i^m = f(x_i, t^m)$.

We discertize the first-order time derivative in Equation (1) by a standard first-order time difference quotient, but for the discretization of the fractional spatial derivative we use the shifted Grünwald approximations. Meerschaert and Tadjeran [15,16] showed that a fully implicit finite difference scheme with a direct truncation of the series in Equation (2) turns out to be unstable! Using the following shifted Grünwald approximations

$$\begin{aligned} \frac{\partial^\alpha u(x_i, t^m)}{\partial_+ x^\alpha} &= \frac{1}{h^\alpha} \sum_{k=0}^{i+1} g_k^{(\alpha)} u_{i-k+1}^m + O(h) \\ \frac{\partial^\alpha u(x_i, t^m)}{\partial_- x^\alpha} &= \frac{1}{h^\alpha} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} u_{i+k-1}^m + O(h) \end{aligned} \tag{5}$$

they proved that the corresponding implicit finite difference scheme

$$\frac{u_i^{m+1} - u_i^m}{\Delta t} - \frac{d_{+,i}^{m+1}}{h^\alpha} \sum_{k=0}^{i+1} g_k^{(\alpha)} u_{i-k+1}^{m+1} - \frac{d_{-,i}^{m+1}}{h^\alpha} \sum_{k=0}^{N-i+1} g_k^{(\alpha)} u_{i+k-1}^{m+1} = f_i^{m+1} \tag{6}$$

is unconditionally stable and convergent. Numerical experiments show that this scheme generates very satisfactory numerical approximations.

Let $\boldsymbol{u}^m = [u_1^m, u_2^m, \ldots, u_{N-1}^m]^T$, $\boldsymbol{f}^m = [f_1^m, f_2^m, \ldots, f_{N-1}^m]^T$, $A^m = [a_{i,j}^m]_{i,j=1}^{N-1}$, and I be the identity matrix of order $N - 1$. Then the numerical scheme Equation (6) can be expressed in the following matrix form

$$(I + \frac{\Delta t}{h^\alpha} A^{m+1}) \boldsymbol{u}^{m+1} = \boldsymbol{u}^m + \Delta t \boldsymbol{f}^{m+1} \tag{7}$$

Here the entries of matrix A^{m+1} are given by

$$a_{i,j}^{m+1} = \begin{cases} -(d_{+,i}^{m+1} + d_{-,i}^{m+1}) g_1^{(\alpha)}, & j = i \\ -(d_{+,i}^{m+1} g_2^{(\alpha)} + d_{-,i}^{m+1} g_0^{(\alpha)}), & j = i - 1 \\ -(d_{+,i}^{m+1} g_0^{(\alpha)} + d_{-,i}^{m+1} g_2^{(\alpha)}), & j = i + 1 \\ -d_{+,i}^{m+1} g_{i-j+1}^{(\alpha)}, & j < i - 1 \\ -d_{-,i}^{m+1} g_{j-i+1}^{(\alpha)}, & j > i + 1 \end{cases} \tag{8}$$

It is clear that $a_{i,j}^{m+1} \leq 0$ for all $i \neq j$. We further conclude from Equations (4) and (8) that the coefficient matrix $I + \frac{\Delta t}{h^\alpha} A^{m+1}$ is a nonsingular, strictly diagonally dominant M-matrix.

Equation (8) implies that the scheme has a dense coefficient matrix, which has a memory requirement of $O(N^2)$ and computational work of $O(N^3)$ per time step. Thus the non-local nature of the fractional derivatives results in a full coefficient matrix of the system. This is in contrast to numerical methods for second-order diffusion equations which usually generate banded coefficient matrices of $O(N)$ nonzero

entries and can be solved by fast solution methods such as multigrid methods, domain decomposition methods, and wavelet methods. Therefore the development of fast and robust numerical methods with efficient storage for the space-fractional diffusion equation is crucial for the applications of fractional diffusion equations.

3. The Fast Conjugate Gradient Squared Method

Since the stiffness matrix A^{m+1} is dense, solving Equation (7) using Gaussian elimination requires computational work of $O(N^3)$ operations per iteration and memory storage of $O(N^2)$ per time step. For example, each time we reduce the size of the spatial mesh by half, the total number of unknowns per time step is doubled. As a result, the required memory increases 4 times, and the computational work increases 8 times. If the time step size is reduced by half too, then we would expect the overall consumed CPU time for solving the finite difference method to increases 16 times. Because of the significantly increased computational work and memory requirement of the numerical schemes for space-fractional diffusion equations, development of fast and reliable numerical methods with efficient storage mechanism has been of recent interest.

The goal of this paper is to develop a fast solution technique for the one-dimensional space-fractional diffusion equation via finite difference method Equation (6). Let us begin by recalling the conjugate gradient squared iterative scheme to solve the system Equation (7): $(I+\frac{\Delta t}{h^\alpha}A^{m+1})u^{m+1} = u^m + \Delta t f^{m+1}$ which can be expressed as follows

At each time step t^{m+1}, we choose $u^{(0)} = u^m$
Compute $r^{(0)} = u^m + \Delta t f^{m+1} - (u^{(0)} + \frac{\Delta t}{h^\alpha}A^{m+1}u^{(0)})$
Choose $\tilde{r}$ (for example, $\tilde{r} = r^{(0)}$)
for $i = 1, 2, \cdots$
  $\rho_{i-1} = \tilde{r}^T r^{(i-1)}$
  if $\rho_{i-1} = 0$ the method fails
    if $i = 1$
      $w^{(1)} = r^{(0)}$
      $p^{(1)} = w^{(1)}$
    else
      $\nu_{i-1} = \rho_{i-1}/\rho_{i-2}$
      $w^{(i)} = r^{(i-1)} + \nu_{i-1}q^{(i-1)}$
      $p^{(i)} = w^{(i)} + \nu_{i-1}(q^{(i-1)} + \nu_{i-1}p^{(i-1)})$
    end if
    $\hat{v} = p^{(i)} + \frac{\Delta t}{h^\alpha}A^{m+1}p^{(i)}$

    $\mu_i = \rho_{i-1}/\tilde{r}^T\hat{v}$
    $q^{(i)} = w^{(i)} - \mu_i\hat{v}$
    $u^{(i)} = u^{(i-1)} + \mu_i(w^{(i)} + q^{(i)})$
    $\hat{q} = (w^{(i)} + q^{(i)}) + \frac{\Delta t}{h^\alpha}A^{m+1}(w^{(i)} + q^{(i)})$

$r^{(i)} = r^{(i-1)} - \mu_i \hat{q}$

$\delta = \|u^m + \Delta t f^{m+1} - (u^{(i)} + \frac{\Delta t}{h^\alpha} A^{m+1} u^{(i)})\|$

Check for convergence; continue if necessary

end

$u^{m+1} = u^{(i)}$

Since the finite difference method has a nonsymmetric coefficient matrix in general, we need to use a nonsymmetric conjugate gradient method. Here we take the conjugate gradient squared method to solve the implicit Euler finite difference method Equation (6).

Notice that in the above algorithm, at each time step t^{m+1}, the evaluation of the matrix-vector multiplication $A^{m+1}p^{(i)}$, $A^{m+1}(w^{(i)} + q^{(i)})$ and $A^{m+1}u^{(i)}$ costs $O(N^2)$ operations, while all others are already of optimal order computational cost $O(N)$. Further, the storage of the stiffness matrix A^{m+1} requires $O(N^2)$ of memory, while all other operations require only $O(N)$ of memory. Our immediate goal is to develop a fast conjugate gradient squared method for the efficient solution and storage of the system Equation (7).

In light of our goal, the rest of this section addresses the following important issues: (i) an efficient storage of the coefficient matrix A^{m+1} with memory requirement of $O(N)$ and (ii) how to perform an efficient matrix-vector multiplication $A^{m+1}u$ with a general vector u in $O(N \log N)$ operations.

3.1. An Efficient $O(N)$ Storage of the Stiffness Matrix

To develop a fast solution method with minimal memory requirement, we carefully explore the structure of the coefficient matrices.

Theorem 1. *The total memory requirement for storing the coefficient matrix A^{m+1} is $O(N)$.*

Proof. We conclude from Equation (8) that the stiffness matrix A^{m+1} can be decomposed as follows

$$A^{m+1} = -\text{diag}\left(d_+^{m+1}\right) A_L - \text{diag}\left(d_-^{m+1}\right) A_R \tag{9}$$

Here $\text{diag}\left(d_+^{m+1}\right)$, $\text{diag}\left(d_-^{m+1}\right)$, are diagonal matrices of order $N-1$ with their ith entries $d_{+,i}^{m+1}$, $d_{-,i}^{m+1}$, for $i = 1, 2, \ldots, N-1$. The matrices A_L and A_R are matrices of order $N-1$ and are defined by

$$A_L = \begin{bmatrix} g_1^{(\alpha)} & g_0^{(\alpha)} & 0 & \ldots & 0 & 0 \\ g_2^{(\alpha)} & g_1^{(\alpha)} & g_0^{(\alpha)} & \ddots & \ddots & 0 \\ \vdots & g_2^{(\alpha)} & g_1^{(\alpha)} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ g_{N-2}^{(\alpha)} & \ddots & \ddots & \ddots & g_1^{(\alpha)} & g_0^{(\alpha)} \\ g_{N-1}^{(\alpha)} & g_{N-2}^{(\alpha)} & \ldots & \ldots & g_2^{(\alpha)} & g_1^{(\alpha)} \end{bmatrix} \quad A_R = \begin{bmatrix} g_1^{(\alpha)} & g_2^{(\alpha)} & \ldots & \ldots & g_{N-2}^{(\alpha)} & g_{N-1}^{(\alpha)} \\ g_0^{(\alpha)} & g_1^{(\alpha)} & g_2^{(\alpha)} & \ldots & \ddots & g_{N-2}^{(\alpha)} \\ 0 & g_0^{(\alpha)} & g_1^{(\alpha)} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & \ddots & g_1^{(\alpha)} & g_2^{(\alpha)} \\ 0 & 0 & \ldots & 0 & g_0^{(\alpha)} & g_1^{(\alpha)} \end{bmatrix} \tag{10}$$

Thus, instead of storing the full matrix A^{m+1} which have $(N-1)^2$ parameters we need only store the $3N-2$ parameters, $d_+^{m+1} = \left[d_{+,1}^{m+1}, d_{+,2}^{m+1}, \ldots, d_{+,N-1}^{m+1}\right]^T$, $d_-^{m+1} = \left[d_{-,1}^{m+1}, d_{-,2}^{m+1}, \ldots, d_{-,N-1}^{m+1}\right]^T$, and $g^{(\alpha)} = \left[g_0^{(\alpha)}, g_1^{(\alpha)}, \ldots, g_{N-1}^{(\alpha)}\right]^T$. In particular, the fractional binomial coefficient vector $g^{(\alpha)}$ depends

only on the size of the spatial partition and the order of the anomalous diffusion but is independent of time or space. So it can be preprocessed and stored in advance. □

3.2. Toeplitz and Circulant Matrix

In order to explain the fast algorithm, we define the terms Toeplitz matrix and circulant matrix. A Toeplitz matrix is a matrix in which each descending diagonal from left to right is constant. Clearly, both matrices A_L and A_R are Toeplitz matrices. Also, $A_R = (A_L)^T$ is the transpose of A_L.

A circulant matrix is a matrix in which each row vector is rotated one element to the right relative to the preceding row vector. It is clear that a circulant matrix is a Toeplitz matrix, but the converse is not true. Note that the Toeplitz matrices A_L and A_R can be embedded into $(2N-2) \times (2N-2)$ circulant matrices $C_{2N-2,L}$ and $C_{2N-2,R}$ as:

$$C_{2N-2,L} = \begin{bmatrix} A_L & B_L \\ B_L & A_L \end{bmatrix}, C_{2N-2,R} = \begin{bmatrix} A_R & B_R \\ B_R & A_R \end{bmatrix} \tag{11}$$

where

$$B_L = \begin{bmatrix} 0 & g_{N-1}^{(\alpha)} & \dots & \dots & g_3^{(\alpha)} & g_2^{(\alpha)} \\ 0 & 0 & g_{N-1}^{(\alpha)} & \dots & \ddots & g_3^{(\alpha)} \\ 0 & 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \ddots & 0 & g_{N-1}^{(\alpha)} \\ g_0^{(\alpha)} & 0 & \dots & 0 & 0 & 0 \end{bmatrix} \tag{12}$$

and

$$B_R = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 & g_0^{(\alpha)} \\ g_{N-1}^{(\alpha)} & 0 & 0 & \dots & \ddots & 0 \\ g_{N-2}^{(\alpha)} & g_{N-1}^{(\alpha)} & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ g_3^{(\alpha)} & \dots & 0 & \ddots & 0 & 0 \\ g_2^{(\alpha)} & g_3^{(\alpha)} & \dots & 0 & g_{N-1}^{(\alpha)} & 0 \end{bmatrix}. \tag{13}$$

Again, note that $B_R = (B_L)^T$ is the transpose of B_L.

It is known that a circulant matrix C_n can be decomposed as follows [25,26]

$$C_n = F_n^{-1} \operatorname{diag}(F_n c)\, F_n \tag{14}$$

where $c = [c_0, c_{n-1}, c_{n-2}, \cdots, c_2, c_1]^T$ is the first column vector of C_n and F_n is the $n \times n$ discrete Fourier transform matrix in which the (j,l)-entry $F_n(j,l)$ of the matrix F_n is given by

$$F_n(j,l) = \frac{1}{\sqrt{n}} \exp\Big(-\frac{2\pi \,\mathrm{i}\, j\, l}{n}\Big) \qquad 0 \le j,l \le n-1 \tag{15}$$

where $\mathrm{i} = \sqrt{-1}$.

We make use of the above decomposition property of circulant matrices to efficiently execute all matrix-vector multiplications $A^{m+1}p^{(i)}$, $A^{m+1}(w^{(i)}+q^{(i)})$ and $A^{m+1}u^{(i)}$ of the conjugate gradient squared method above.

3.3. A Fast $O(N \log N)$ Matrix-Vector Multiplication Algorithm

We now shift our focus to the efficient operation of the numerical scheme. To efficiently execute the matrix-vector multiplications $A^{m+1}p^{(i)}$, $A^{m+1}(w^{(i)}+q^{(i)})$ and $A^{m+1}u^{(i)}$ of the conjugate gradient squared method, we use the following $O(N \log N)$ algorithm, based on the decomposition Equation (9) of the matrix A^{m+1}, the diagonalization Equation (14) of a circulant matrix and the embedding Equation (11).

Theorem 2. *Let A^{m+1} be the stiffness matrix as Equation (8). Let u be any N dimensional vector. Then $A^{m+1}u$ can be performed in $O(N \log N)$ operations.*

Proof. We explain the operation count of $O(N \log N)$ by executing the following steps:

1. Introduce two $(2N-2) \times (2N-2)$ matrices and one $2N-2$ vector

$$C_{2N-2,L} = \begin{bmatrix} A_L & B_L \\ B_L & A_L \end{bmatrix}, C_{2N-2,R} = \begin{bmatrix} A_R & B_R \\ B_R & A_R \end{bmatrix}, u_{2N-2} = \begin{bmatrix} u \\ 0 \end{bmatrix} \tag{16}$$

   Here B_L and B_R are defined as in Equations (12) and (13) and A_L and A_R as in Equation (10) respectively. It is clear that

$$C_{2N-2,L}u_{2N-2} = \begin{bmatrix} A_L u \\ B_L u \end{bmatrix}, \qquad C_{2N-2,R}u_{2N-2} = \begin{bmatrix} A_R u \\ B_R u \end{bmatrix} \tag{17}$$

   Thus, the matrix-vector products $A_L u$ and $A_R u$ can be obtained as the first half of the matrix-vector products $C_{2N-2,L}u_{2N-2}$ and $C_{2N-2,R}u_{2N-2}$, respectively.
2. Evaluate the matrix-vector products $w_{2N-2} = F_{2N-2}u_{2N-2}$ in $O(N \log N)$ operations. In fact, $F_{2N-2}u_{2N-2}$ is the discrete Fourier transform of u_{2N-2}, which can be achieved in $O((2N)\log(2N)) = O(N \log N)$ operations via the fast Fourier transform (FFT).
3. Similarly evaluate $v_{2N-2,L} = F_{2N-2}c_{2N-2,L}$ and $v_{2N-2,R} = F_{2N-2}c_{2N-2,R}$ in $O(N \log N)$ operations, where $c_{2N-2,L}$ and $c_{2N-2,R}$ are the first column vectors of $C_{2N-2,L}$ and $C_{2N-2,R}$, respectively.
4. Evaluate the Hadamard products $z_{2N-2,L} = w_{2N-2} \cdot v_{2N-2,L} = [w_1 v_{1,L}, \cdots, w_{2N-2}v_{2N-2,L}]^T$ and $z_{2N-2,R} = w_{2N-2} \cdot v_{2N-2,R} = [w_1 v_{1,R}, \cdots, w_{2N-2}v_{2N-2,R}]^T$ in $O(N)$ operations.
5. Evaluate $y_{2N-2,L} = F^{-1}_{2N-2}z_{2N-2,L}$ and $y_{2N-2,R} = F^{-1}_{2N-2}z_{2N-2,R}$ in $O(N \log N)$ operations via inverse FFT. Combining Equations (16) and (17) yields that

$$\begin{aligned} y_{2N-2,L} &= \begin{bmatrix} y_L \\ y_L' \end{bmatrix} = C_{2N-2,L}u_{2N-2} = \begin{bmatrix} A_L u \\ B_L u \end{bmatrix} \\ y_{2N-2,R} &= \begin{bmatrix} y_R \\ y_R' \end{bmatrix} = C_{2N-2,R}u_{2N-2} = \begin{bmatrix} A_R u \\ B_R u \end{bmatrix} \end{aligned} \tag{18}$$

6. Evaluate the Hadamard products $u_L = d_+^{m+1} \cdot y_L$ and $u_R = d_-^{m+1} \cdot y_R$ in $O(N)$ operations. Use Equation (9) to evaluate $A^{m+1}u = -u_L - u_R$ in $O(N)$ operations.

□

4. Numerical Experiments

In this section we carry out numerical experiments to study the performance of the fast conjugate gradient squared finite difference method developed in this paper and to compare its performance with the finite difference methods with full coefficient matrices which were developed in [15,16].

We consider the fractional diffusion equation Equation (1) with an anomalous diffusion of order $\alpha = 1.8$ and the left-sided and right-sided diffusion coefficients

$$d_+(x,t) = 1.32\Gamma(1.2)x^{1.8}, \quad d_-(x,t) = 1.32\Gamma(1.2)(1-x)^{1.8} \tag{19}$$

The spatial domain is $[x_L, x_R] = [0,1]$, the time interval is $[0,T] = [0,1]$. The source term and the initial condition are given by

$$\begin{aligned} f(x,t) &= -16e^{1-t}\Big[x^2(1-x^2) + 2.64(x^2 + (1-x)^2) \\ &\quad -13.2\big(x^3 + (1-x)^3\big) + 12\big(x^4 + (1-x)^4\big)\Big] \\ u_0(x) &= 16ex^2(1-x)^2 \end{aligned} \tag{20}$$

The true solution to the corresponding fractional diffusion equation Equation (1) is given by [16]

$$u(x,t) = 16e^{1-t}x^2(1-x)^2 \tag{21}$$

In the numerical experiments, we solve the problem by both the fast conjugate squared (iterative) finite difference method and the regular finite difference method Equation (6) and denote their solutions by u_{FIFD}^m and u_{FD}^m, respectively. Let u^m be the numerical solution u_{FIFD}^m or u_{FD}^m at time step t^m and $u(x,t^m)$ be the true solution to problem Equation (1).

In Table 1 we present the errors $\|u_{FIFD}^M - u(\cdot,t^M)\|_{L^\infty}$ and $\|u_{FD}^M - u(\cdot,t^M)\|_{L^\infty}$ for different spatial mesh sizes and time steps. These results are very encouraging and show that fast conjugate gradient squared finite difference method developed in this paper generates numerical solutions with same accuracy as the regular finite difference method, despite the fact that the former has significantly reduced the storage and computational cost of the latter from $O(N^2)$ and $O(N^3)$ to $O(N)$ and $O(N\log N)$, respectively. We also present a representative plot with $N = M = 128$ in Figure 1, which shows that the fast conjugate gradient squared finite difference solution and the regular finite difference solution both sit on the curve of the true solution without stark differences.

The numerical experiment carried out in this section shows the significant reduction in computational time which coincides with the theoretical analysis. For example, with $N = 1024$ nodes the fast conjugate gradient finite difference scheme developed in this paper has about 280 times of CPU reduction than the standard finite difference scheme solved with Gaussian elimination. This is on top of the significant reduction in storage. This demonstrates the strong potential of the method.

Table 1. Comparison of the fast iterative finite difference (FIFD) method with the regular finite difference (FD) method in the simulation of the fractional diffusion problem with a known analytical solution.

	$N = M$	ERROR $\|\cdot\|_{L^\infty}$	CPU (Seconds)
FD	2^6	1.38160×10^{-2}	7.41
	2^7	6.46640×10^{-3}	5.79×10
	2^8	3.07995×10^{-3}	4.38×10^2
	2^9	1.41210×10^{-3}	3.52×10^3
	2^{10}	6.60815×10^{-4}	2.27×10^4
FIFD	2^6	4.91910×10^{-3}	.04
	2^7	2.46266×10^{-3}	1.30
	2^8	1.23210×10^{-3}	5.12
	2^9	6.16248×10^{-4}	20.14
	2^{10}	3.08172×10^{-4}	80.89

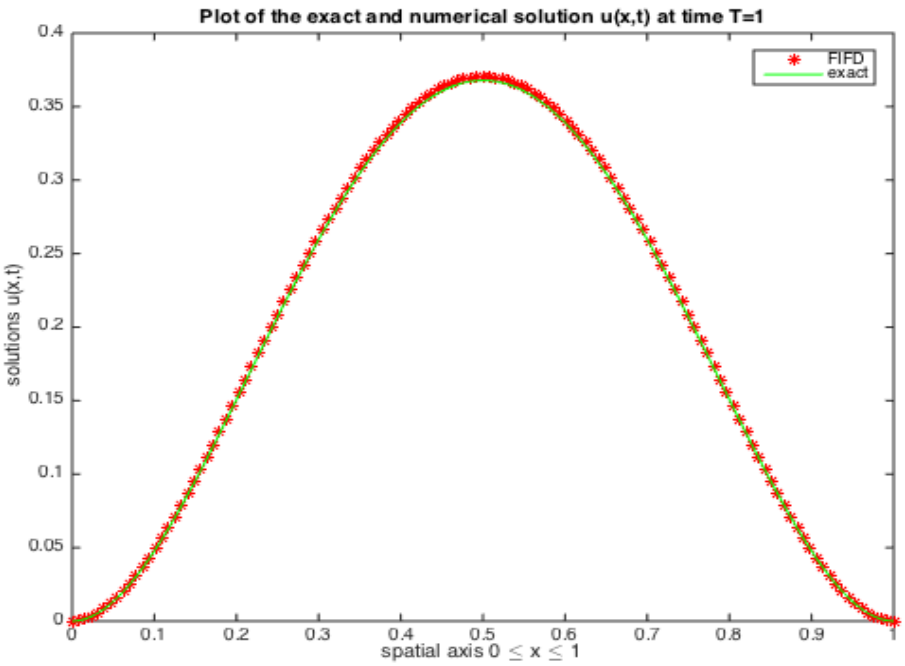

Figure 1. The true solution u (marked by "—"), the fast iterative finite difference solution u_{FIFD} (marked by "$*$"), in §4 at time $T = 1$ with $h = \frac{1}{128}$ and $\Delta t = \frac{1}{128}$.

5. Concluding Remarks and Future Work

This paper develops a fast solution method for the implicit finite difference scheme Equation (6) for the one-dimensional space-fractional diffusion equation developed by Meerschaert and Tadjeran in [15,16]. The fast method consists of carefully analyzing the structure of the coefficient matrix resulting from the finite difference method, delicately decomposing the coefficient matrix into a combination of sparse and structured dense matrices and applying an iterative scheme, in this case the conjugate gradient squared method. Over the past decade many numerical methods have been developed for space-fractional

diffusion equations, however all of them present major computational obstacles in realistic numerical simulation because of the significant computational cost and memory requirement. The fast conjugate gradient squared method developed in this paper keeps the same accuracy as the finite difference method [15,16] with Guassian elimination but has a memory requirement of only $O(N)$ per time step and a computational work of $O(N \log N)$ per iteration.

Numerical computations were carried out using MATLAB. Furthermore, in order to use the fast conjugate gradient squared finite difference method, users need only rewrite a module to replace the matrix-vector multiplication module in traditional conjugate gradient method software. And this fast matrix-vector multiplication is based on FFT, which is readily available in MATLAB. Thus, the fast method virtually does not require any additional coding work to implement.

The idea of fast solution developed in this paper can be applied to other numerical methods. A fast solution method for a second-order Crank-Nicolson finite difference method was developed in [27] for space-fractional diffusion equations in one dimension.

The reader should also note that a large diffusion coefficient could potentially lead to a large condition number of the coefficient matrix. This would in turn increase the number of iterations in the conjugate gradient squared method. Circulant preconditioners [28,29] and multigrid methods [30] have been developed for some model problems which have shown significant improvements, under special conditions. The trade-off in forming and applying a preconditioner also needs to be examined. This can be an avenue for further research for this class of problems.

Acknowledgments

The author would like to express sincere thanks to the referees for their very helpful comments and suggestions, which greatly improved the quality of this work.

Conflicts of Interest

The author declares no conflict of interest

References

1. Ganti, V.; Meerschaert, M.M.; Georgiou, E.F.; Viparelli, E.; Parker, G. Normal and anomalous diffusion of gravel tracer particles in rivers. *J. Geophys. Res.* **2010**, *115*, doi:10.1029/2008JF001222.
2. Debnath, L.; Bhatta, D. Solutions to few linear fractional inhomogenous partial differential equations in fluid mechanics. *Fract. Calc. Appl. Anal.* **2004**, *7*, 22–36.
3. Kirchner, J.W.; Feng, X.; Neal, C. Fractal stream chemistry and its implications for containant transport in catchments. *Nature* **2000**, *403*, 524–526.
4. Gal, N.; Weihs, D. Experimental evidence of strong anomalous diffusion in living cells. *Phys. Rev. E* **2010**, *81*, doi: 10.1103/PhysRevE.81.020903
5. Raberto, M.; Scalas, E.; Mainardi, F. Waiting-times and returns in high-frequency financial data: An empirical study. *Physica* **2002**, *314*, 749–755.

6. Debnath, L. *Linear Partial Differential Equations for Scientists and Engineers*, 4rth ed.; Birkhäuser, Boston, MA, USA; Basel, Switzerland; Berlin, Germany, 2007
7. Sousa, E. Finite difference approximates for a fractional advection diffusion problem. *J. Comput. Phys.* **2009**, *228*, 4038–4054.
8. Deng, W. Finite element method for the space and time fractional Fokker-Planck equation. *SIAM J. Numer. Anal.* **2008**, *47*, 204–226.
9. Roop, J.P. Computational aspects of FEM approximation of fractional advection dispersion equations on bounded domains in R^2. *J. Comput. Appl. Math.* **2006**, *193*, 243–268.
10. Lin, Y.; Xu, C. Finite difference/spectral approximations for the time-fractional diffusion equation. *J. Comput. Phys.* **2007**, *225*, 1533–1552.
11. Mustapha, K.; McLean, W. Superconvergence of a discontinuous Galerkin method for fractional diffusion and wave equations. *SIAM J. Numer. Anal.* **2013**, *51*, 491–515.
12. Xu, Q.; Hesthaven, J.S. Discontinuous Galerkin method for fractional convection-diffusion equations. *SIAM J. Numer. Anal.* **2013**, *52*, 405–423.
13. Ervin, V.J.; Heuer, N.; Roop, J.P. Numerical approximation of a time dependent, nonlinear, space-fractional diffusion equation. *SIAM J. Numer. Anal.* **2007**, *45*, 572–591.
14. Meerschaert, M.M.; Scheffler, H.P.; Tadjeran, C. Finite difference methods for two-dimensional fractional dispersion equation. *J. Comput. Phys.* **2006**, *211*, 249–261.
15. Meerschaert, M.M.; Tadjeran, C. Finite difference approximations for fractional advection-dispersion flow equations. *J. Comput. Appl. Math.* **2004**, *172*, 65–77.
16. Meerschaert, M.M.; Tadjeran, C. Finite difference approximations for two-sided space-fractional partial differential equations. *Appl. Numer. Math.* **2006**, *56*, 80–90.
17. Wang, H.; Wang, K.; Sircar, T. A direct $O(N \log^2 N)$ finite difference method for fractional diffusion equations. *J. Comput. Phys.* **2010**, *229*, 8095–8104.
18. Chaves, A. Fractional diffusion equation to describe Lévy flights. *Phys. Lett. A* **1998**, *239*, 13–16.
19. Barkai, E.; Metzler, R.; Klafter, J. From continuous time random walks to the fractional Fokker-Plank equation. *Phy. Rev. E* **2000**, *61*, 132–138.
20. Chechkin, A.V.; Klafter, J.; Sokolov, I.M. Fractional Fokker-Plank equation for ultra-slow kinetics. *Europhys. Lett.* **2003**, *63*, 326–332.
21. Benson, D.; Wheatcraft, S.W.; Meerschaert, M.M. Application of a fractional advection-dispersion equation. *Water Resour. Res.* **2000**, *36*, 1403–1413.
22. Schumer, R.; Benson, D.A.; Meerschaert, M.M.; Wheatcraft, S.W. Eulerian derivation of the fractional advection-dispersion equation. *J. Contam. Hydrol.* **2001**, *38*, 69–88.
23. Podlubny, I. *Fractional Differential Equations*; Academic Press: New York, NY, USA, 1999.
24. Li, X.; Xu, C. The existence and uniqueness of the week solution of the space-time fractional diffusion equation and a spectral method approximation. *Commun. Comput. Phys.* **2010**, *8*, 1016–1051.
25. Davis, P.J. *Circulant Matrices*; Wiley-Intersciences: New York, NY, USA, 1979.
26. Gray, R.M. Toeplitz and Circulant Matrices: A Review. *Found. Trends Commun. Inf. Theory* **2006**, *2*, 155–239.

27. Basu, T.S.; Wang, H. A fast second-order finite difference method for space-fractional diffusion equations. *Int. J. Numer. Anal. Model.* **2012**, *9*, 658–666.
28. Lei, S.L.; Sun, H.W. A circulant preconditioner for fractional diffusion equations. *J. Comput. Phys.* **2013**, *242*, 715–725.
29. Pan, J.; Ke, R.; Ng, M.; Sun, H.W. Preconditioning Techniques for Diagonal-Times-Toeplitz Matrices in Fractional Diffusion Equations. *SIAM J. Sci. Comput.* **2014**, *36*, 2698–2719.
30. Pang, H.; Sun, H.W. Multigrid method for fractional diffusion equations. *J. Comput. Phys.* **2012**, *231*, 693–703.