FAST PROCEDURES FOR CAPUTO FRACTIONAL DERIVATIVE
AND ITS APPLICATIONS TO ORDINARY AND PARTIAL
DIFFERENTIAL EQUATIONS∗

ZHENGGUANG LIU†, ALJIE CHENG‡, XIAOLI LI§, AND HONG WANG¶

Abstract. In this paper, we develop fast procedures for solving linear systems arising from
discretization of ordinary and partial differential equations with Caputo fractional derivative w.r.t
time variable. First, we consider a finite difference scheme to solve a two-sided fractional ordinary
equation. Furthermore, we present a fast solution technique to accelerate Toeplitz matrix-vector
multiplications arising from finite difference discretization. This fast solution technique is based on
a fast Fourier transform and depends on the special structure of coefficient matrices, and it helps
to reduce the computational work from $O(N^3)$ required by traditional methods to $O(N\log^2 N)$
and the memory requirement from $O(N^2)$ to $O(N)$ without using any lossy compression, where $N$ is
the number of unknowns. Two finite difference schemes to solve time fractional hyperbolic equations with
different fractional order $\gamma$ are considered. We present a fast solution technique depending on the
special structure of coefficient matrices by rearranging the order of unknowns. It helps to reduce the
computational work from $O(N^2 M)$ required by traditional methods to $O(N \log^2 N)$ and the memory
requirement from $O(N M)$ to $O(N)$ without using any lossy compression, where $N = \tau^{-1}$ and $\tau$
is the size of time step, $M = h^{-1}$ and $h$ is the size of space step. Importantly, a fast method is
employed to solve the classical time fractional diffusion equation with a lower cost at $O(M N \log^2 N)$,
where the direct method requires an overall computational complexity of $O(N^2 M)$. Moreover, the
applicability and accuracy of the scheme are demonstrated by numerical experiments to support our
theoretical analysis.

Key words. Fast procedures, Finite difference methods, Time fractional differential equations,
Toeplitz matrix, Fast Fourier transform.

AMS subject classifications. 65M06, 65M12, 65M15, 26A33

1. Introduction. In recent years, many problems in physical science, electromagnetism, electrochemistry, diffusion and general transport theory can be solved by
the fractional calculus approach, which gives attractive applications as a new modeling
tool in a variety of scientific and engineering fields. Roughly speaking, the fractional
models can be classified into two principal kinds: space-fractional differential equation
and time-fractional one. Numerical methods and theory of solutions for fractional dif-
ferential equations have been studied extensively by many researchers which mainly
cover finite element methods [37, 10, 12, 35], mixed finite element methods [38, 20, 19],
finite difference methods [25, 26, 24, 8, 22, 21], finite volume (element) methods [5, 16],
(local) discontinuous Galerkin (L)DG methods [33], spectral methods [14, 13] and so
on.

Let $\frac{C}{0} D_t^\gamma u(t)$ denote the time fractional derivative operator based on Caputo’s
definition, given by

$$\frac{C}{0} D_t^\gamma u(t) = \frac{1}{\Gamma(n-\gamma)} \int_0^t (t-s)^{n-1-\gamma} u^{(n)}(s) ds, \quad n-1 < \gamma < n.$$
From later on, for simplicity, we use \( \partial D^\gamma_t u(t) \) in stead of \( \partial_t^\gamma D^\gamma_t u(t) \).

In this work, we focus on the fractional cases \( 0 < \gamma < 2 \). This nonlocal definition is the limiting equation that governs continuous time random walks with heavy tailed random waiting times. In most cases, it is difficult, or infeasible, to find the analytical solution or good numerical solution of the problems. Numerical solutions or approximate analytical solutions become necessary. Liu et al. [17] give a radial basis functions (RBFs) meshless approach for modeling a fractal mobile/immobile transport model. Numerical simulation of the fractional order mobile/immobile advection-dispersion model is considered by Liu et al. [15]. Furthermore, Zhang and Liu [36] present a novel numerical method for the time variable fractional order mobile–immobile advection–dispersion problem. The finite difference schemes are used by Ashyralyev and Cakir [1] for solving one-dimensional fractional parabolic partial differential equations. They [2] also give the finite difference method (FDM) for fractional parabolic equations with Neumann boundary conditions. In [18], finite difference/finite element method for a nonlinear time-fractional fourth-order reaction–diffusion problem has been considered.

Many articles consider fast conjugate gradient methods based on fast Fourier transform to solve space fractional equations. For example, Wang and Basu [28] presented a fast finite difference method for two-dimensional space-fractional diffusion equations. A fast characteristic finite difference method for fractional advection–diffusion equations was considered in [32]. Chen et al. [4] provided a fast semi-implicit difference method for a nonlinear two-sided space-fractional diffusion equation with variable diffusivity coefficients. For time fractional equations, unless we extract the Toeplitz structure for stiffness matrix, we can not use fast Fourier transform to speed up the evaluation. However, Ke et al. [11] studied a fast direct method for block triangular Toeplitz-like with tri-diagonal block systems for time-fractional partial differential equations. They reduce the computational work from \( O(N^3) \) required by traditional methods to \( O(MN\log^2 M) \) and the memory requirement from \( O(N^2) \) to \( O(MN) \), where \( M \) is the number of blocks in the system and \( N \) is the size of each block. Jiang et al. [9] presented a fast evaluation scheme of the Caputo derivative to solve the fractional diffusion equations. The new method requires \( O(N_s N_{exp}) \) storage and \( O(N_s N_T N_{exp}) \) work with \( N_s \) the total number of points in space, \( N_T \) the total number of time steps and \( N_{exp} \) the number of exponentials. Fu and Wang [29] developed a fast space-time finite difference method for space-time fractional diffusion equations by fully utilizing the mathematical structure of the scheme. In addition, their method has approximately linear computational complexity, i.e., has a computational cost of \( O(MN\log(MN)) \) per Krylov subspace iteration. Our goals are to study a fast direct method to solve both ordinary and partial fractional differential equations based on Caputo fractional derivative. We consider a finite difference scheme to solve a two-sided fractional ordinary equation. Furthermore, we present a fast solution technique to accelerate Toeplitz matrix-vector multiplications arising from finite difference discretization. This fast solution technique is based on a fast Fourier transform and depends on the special structure of coefficient matrices, and it helps to reduce the computational work from \( O(N^3) \) required by traditional methods to \( O(N\log^2 N) \) and the memory requirement from \( O(N^2) \) to \( O(N) \) without using any lossy compression, where \( N \) is the number of unknowns. For time fractional partial differential hyperbolic equations with different fractional order \( \gamma \), two finite difference schemes are considered. We present a fast solution technique depending on the special structure of coefficient matrices by rearranging the order of unknowns in space and time directions. It helps to reduce the computational work from \( O(N^2 M) \) required
by traditional methods to $O(N M \log^2 N)$ and the memory requirement from $O(N M)$ to $O(N)$ without using any lossy compression, where $N = \tau^{-1}$ and $\tau$ is the size of time step, $M = h^{-1}$ and $h$ is the size of space step. Importantly, a fast method is employed to solve the classical time fractional diffusion equation with a lower cost at $O(M N \log^2 N)$, where the direct method requires an overall computational complexity of $O(N^2 M)$.

This paper is organized as follows. In section 2, we analyze the structures of stiffness matrix for two-sided ordinary differential fractional differential equations and present a fast finite difference scheme. In section 3, we introduce fast procedures for finite difference method for several different kinds of time fractional partial differential equations with the Caputo fractional derivative. Then, in section 4, some numerical experiments for the finite difference discretization are carried out.

2. A fast procedure of finite difference scheme for two-sided fractional ordinary differential equation. We consider the following two-sided fractional ordinary differential equation involving Caputo operators with general boundary conditions [7]:

$$(2) \quad 0D^\gamma_t u(t) + D^\gamma_1 u(t) + u(t) = f(t), \quad t \in (0, 1),$$

subject to the following condition:

$$u(0) = u(1) = 0.$$

The objective of this section is to consider the finite difference method for equations (2). First, for the convenience of theoretical analysis, we introduce the following lemma:

**Lemma 1.** Denote

$$G_m = (m + 1)^{1-\gamma} - m^{1-\gamma}, \quad m \geq 0, \quad 0 < \gamma < 1,$$

then, we have

$$(3) \quad G_1 \leq G_2 \leq \cdots \leq G_N.$$

Define $\Omega_\tau = \{t_n, \quad t_n = i \tau, \quad \tau = 1/N, \quad 0 \leq i \leq N\}$ to be a uniform mesh of interval $[0, 1]$. The values of the function $u$ at the grid points are denoted $u_j = u(x_j)$. We also use $u_h(x_i) = u_i$ for grid function $u_h$ if no confusion occurs. Define

$$\|u_h\|_\infty = \max_{1 \leq i \leq N-1} |u_i|, \quad \|u_h\|^2 = \sum_{i=1}^{N-1} \tau u_i^2.$$

**Lemma 2.** Suppose that $u \in C^2[0, 1]$, then, the time fractional Caputo derivative $0D^\gamma_t u(t)$ for $0 < \gamma < 1$ at $t = t_n$ is discretized by

$$(4) \quad 0D^\gamma_t u(t_n) = \frac{\tau^{-\gamma}}{\Gamma(2 - \gamma)} \left[ G_0 u_n - \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) u_k + G_n u_0 \right] + O(\tau^{2-\gamma}),$$

and $iD^\gamma_1 u(t)$ for $0 < \gamma < 1$ at $t = t_n$ is discretized by

$$(5) \quad iD^\gamma_1 u(t_n) = \frac{\tau^{-\gamma}}{\Gamma(2 - \gamma)} \left[ G_0 u_n - \sum_{m=n+1}^{N-1} (G_{m-n-1} - G_{m-n}) u_m + G_{N-n-1} u_N \right] + O(\tau^{2-\gamma}).$$
\[ \begin{align*}
\text{Proof.} \quad \text{The discrete formulation (4) has been proved in many articles such as [14]. Next, we will only prove equation (5). Firstly, we give the definition of right-Caputo derivative [7]:}
\end{align*} \]

\[ (6) \quad \tau D_1^\gamma u(t) = -\frac{1}{\Gamma(1-\gamma)} \int_t^1 u'(s)(s-t)^{-\gamma} ds, \quad 0 < \gamma < 1. \]

Then, a discrete approximation to Caputo derivative (6) at \( t_n \) can be obtained by the following approximation when \( 1 \leq n \leq N - 1 \):

\[ \begin{align*}
\tau D_1^\gamma u(t_n) &= -\frac{1}{\Gamma(1-\gamma)} \int_{t_n}^1 u'(s)(s-t_n)^{-\gamma} ds \\
&= -\frac{1}{\Gamma(1-\gamma)} \sum_{m=n}^{N-1} \int_{m\tau}^{(m+1)\tau} \frac{u'(s)}{(s-n\tau)^\gamma} ds \\
&= -\frac{1}{\Gamma(2-\gamma)} \sum_{m=n}^{N-1} \left[ \frac{u_{m+1} - u_m}{\tau} \right] (s-n\tau)^{-\gamma} ds + r_n^n \\
&= \frac{\tau^{-\gamma}}{\Gamma(2-\gamma)} \left[ G_0u_n - \sum_{m=n+1}^{N-1} \left( G_{m-n-1} - G_{m-n} \right) u_m + G_{N-n-1} u_N \right] + r_n^n,
\end{align*} \]

where

\[ r_n^n \leq -c_u \sum_{m=n}^{N-1} \int_{m\tau}^{(m+1)\tau} [(2m+1)\tau - 2s] (s-n\tau)^{-\gamma} ds + O(\tau^2), \]

where \( c_u \) is a constant depending only on \( u \). For \( r_n^n \), it is easy to obtain

\[\begin{align*}
&\sum_{m=n}^{N-1} \int_{m\tau}^{(m+1)\tau} [(2m+1)\tau - 2s] (s-n\tau)^{-\gamma} ds \\
&= \frac{\tau^{2-\gamma}}{1-\gamma} \sum_{m=n}^{N-1} \left[ (2m+1)G_{m-n} - 2(m+1)(m-n+1)^{1-\gamma} + 2m(m-n)^{1-\gamma} \right] \\
&\quad - \frac{2\tau^{2-\gamma}}{(1-\gamma)(2-\gamma)} \sum_{m=n}^{N-1} [(m-n+1)^{2-\gamma} - (m-n)^{2-\gamma}] \\
&= \frac{h^{2-\gamma}}{1-\gamma} \left( S_n - \frac{2}{2-\gamma} (N-n)^{2-\gamma} \right),
\end{align*}\]

where

\[ S_n = (N-n)^{1-\gamma} + 2[(N-n-1)^{1-\gamma} + (N-n-2)^{1-\gamma} + \cdots + 2^{1-\gamma} + 1^{1-\gamma}]. \]

From [14], it is easy to obtain

\[ \left| S_n - \frac{1}{2-\gamma} (N-n)^{2-\gamma} \right| \leq C. \]
where $C$ is a constant independent of $\gamma$ and $n$. It means that $|r^n\tau| \leq C\tau^{2-\gamma}$. The proof is completed.

Then, we get the following finite difference scheme:

**Scheme 1:**

\[
(7) \quad \frac{\tau^{-\gamma}}{\Gamma(2-\gamma)} \left[ 2G_0 u_n - \sum_{k=1, k \neq n}^{N-1} (G_{|n-k-1|} - G_{|n-k|}) u_k + G_n u_0 + G_{N-n-1} u_N \right] + u_n = f_n, \quad 1 \leq n \leq N - 1.
\]

From lemma 2, it is easy to get the compatible condition

\[
\lim_{\tau \to 0} \| R_h(u) \| = 0.
\]

Furthermore, the right function $f(t)$ is smooth enough, then we know that the finite difference equation is stable and the finite difference solution $u_h$ is convergent and the convergence rate is $(2 - \alpha)$.

**2.1. Solvability of the finite difference scheme.**

**Theorem 3.** The finite difference scheme (7) is uniquely solvable and the stiffness matrix is a symmetric Toeplitz matrix.

**Proof.** Let $\mu = \frac{\tau^{-\gamma}}{\Gamma(2-\gamma)}$. Noting that $u_0 = u_N = 0$, then the discretized scheme (7) can be rewritten as

\[
(8) \quad (1 + 2\mu) u_n - \mu \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) u_k - \mu \sum_{k=n+1}^{N-1} (G_{k-n-1} - G_{k-n}) u_k = f_n.
\]

The discretized system for finite difference method can be expressed in the following matrix form

\[
(9) \quad Au = f,
\]

where we denote

\[
u = (u_1, u_2, \ldots, u_{N-2}, u_{N-1})^T, \quad f = (f_1, f_2, \ldots, f_{N-2}, f_{N-1})^T.
\]

Define $W_k = G_k - G_{k-1}$, then, it is easy to obtain that

\[
A = \begin{pmatrix}
1 + 2\mu & \mu W_1 & \mu W_2 & \cdots & \mu W_{N-2} \\
\mu W_1 & 1 + 2\mu & \mu W_1 & \cdots & \mu W_{N-3} \\
\mu W_2 & \mu W_1 & 1 + 2\mu & \cdots & \mu W_{1} \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
\mu W_{N-2} & \mu W_{N-3} & \cdots & \mu W_1 & 1 + 2\mu
\end{pmatrix}
\]

Thus, we obtain $A_{i,j} = A_{i+1,j+1}$ which means $A$ is a Toeplitz matrix.
By the Gerschgorin circle theorem, the stiffness matrix $A$ is invertible. This invertibility guarantees the solvability of the discretized scheme. This completes the proof.

2.2. A fast procedure for finite difference method. Let $u_0$ be an initial guess. Then compute $r_0 = f - Au_0$, $d_1 = r_0$ and

$$\begin{align*}
\omega_1 &= r_0^T r_0 / d_1^T Ad_1 \\
u_1 &= u_0 + \omega_1 d_1 \\
r_1 &= r_0 - \omega_1 Ad_1
\end{align*}$$

for $k = 2, 3, \ldots$

$$\begin{align*}
\gamma_k &= r_{k-1}^T r_{k-1} / r_{k-2}^T r_{k-2} \\
d_k &= r_{k-1} + \gamma_k d_{k-1} \\
\omega_k &= r_{k-1}^T r_{k-1} / d_k^T Ad_k \\
u_k &= u_{k-1} + \omega_k d_k \\
r_k &= r_{k-1} - \omega_k Ad_k
\end{align*}$$

Check for convergence, continue if necessary.

end

$$u = u_k.$$ 

To reduce the computational work and memory requirement, we need only to accelerate the matrix-vector multiplication $Ad$ for any vector $d$ and store $A$ efficiently.

Without loss of generally, let $g(x) = 0$, then $u_i = 0$, $\forall i \leq 0$ and $i \geq N$. Then the stiffness matrix $A$ becomes a $(N-1) \times (N-1)$ Topelitz matrix. Let $a_{j-i}$ denote the common entry in the $(j-i)$-th descending diagonal of $A$ from left to right. Namely, $A_{i,j} = a_{j-i}, \forall j \geq i$.

The stiffness matrix $A$ can be embedded into a $(2N-2) \times (2N-2)$ circulate matrix $C$ as follow

$$C = \begin{pmatrix} A & B \\ B & A \end{pmatrix}, \quad B = \begin{pmatrix} 0 & a_{N-2} & \cdots & a_2 & a_1 \\ a_{N-2} & 0 & a_{N-2} & \cdots & a_2 \\ \vdots & a_{N-2} & 0 & \ddots & \vdots \\ a_2 & \vdots & \ddots & \ddots & a_{N-2} \\ a_1 & a_2 & \cdots & a_{N-2} & 0 \end{pmatrix}.$$ 

The circulate matrix $C$ has the following decomposition

$$C = F^{-1} \text{diag}(Fc)F,$$

where $c$ is the first column vector of $C$ and $F$ is the $2(N-1) \times 2(N-1)$ discrete Fourier transform matrix. Denote that $w = (d,d)^T$, then it is well known that the matrix-vector multiplication $Fw$ for $w \in \mathbb{R}^{2N-2}$ can be carried out in $O(N\log N)$ operations.
via the fast Fourier transform. Equation (10) shows that $Cw$ can be evaluated in $O(N\log N)$ operations. So, we know that $Ad$ can be evaluated in $O(N\log N)$ operations for any $d \in \mathbb{R}^{N-1}$. The overall computation cost of the fast conjugate method is $O(N\log^2 N)$.

3. A fast finite difference scheme for time fractional partial differential equation. In this subsection, we present some finite difference schemes and analyse the structures of stiffness matrices for several partial fractional equations. For convenience of theoretical analysis, we now denote

$$G_k = (k + 1)^{1-\gamma} - k^{1-\gamma}, \quad 0 < \gamma < 1,$$

$$M_k = (k + 1)^{2-\gamma} - k^{2-\gamma}, \quad 1 < \gamma < 2,$$

and

$$\frac{d}{\tau}u^n = \frac{u^n - u^{n-1}}{\tau}.$$

From [14] and [38], it is not difficult to verify that, when $\tau \to 0$,

$$1 = G_0 > G_1 > G_2 > \cdots > G_n > \cdots \to \tau^\gamma \to 0,$$

and

$$1 = M_0 > M_1 > M_2 > \cdots > M_n > \cdots \to \tau^{\gamma-1} \to 0.$$

We consider the following fractional partial differential hyperbolic equation

$$\frac{C_0}{C_1}D_t^\gamma u(x, t) + a\frac{\partial u(x, t)}{\partial x} = f(x, t), \quad x \in (0, L), \quad t \in (0, T], \quad 0 < \gamma < 1,$$

subject to the initial condition:

$$u(x, 0) = u_0(x), \quad x \in [0, L],$$

with the boundary conditions

$$u(0, t) = 0, \quad t \in [0, T], \quad a > 0,$$

$$u(L, t) = 0, \quad t \in [0, T], \quad a < 0.$$

Without loss of generality, we set $a = 1$. For above partial differential equation, we have the following discrete formula:

**Scheme 2:** Suppose $u \in C^{2,2}_{x,t}([0, L] \times [0, T])$, the time fractional Caputo derivative

$$\frac{C_0}{C_1}D_t^\gamma u(x, t)$$

for $0 < \gamma < 1$ at $(x_{i-\frac{1}{2}}, t_n)$ is discretized by [14]

$$\frac{C_0}{C_1}D_t^\gamma u(x_{i-\frac{1}{2}}, t_n) = \frac{\tau^{-\gamma}}{2\Gamma(2-\gamma)} \left[ G_0 u_i^n - \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) u_i^k + G_n u_i^0 \right]
+ \frac{\tau^{-\gamma}}{2\Gamma(2-\gamma)} \left[ G_0 u_{i-1}^n - \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) u_{i-1}^k + G_n u_{i-1}^0 \right]
+ O(\tau^{2-\gamma} + h^2).$$

*This manuscript is for review purposes only.*
For $\frac{\partial u}{\partial x}$, we have

\begin{equation}
\frac{\partial u(x_i, t^n)}{\partial x} = \frac{u^n_i - u^n_{i-1}}{h} + O(h^2),
\end{equation}

Replacing the function $u^n_i$ with its numerical approximation $U^n_i$, we get the following difference scheme:

\begin{align*}
\left( \frac{G_0 \tau^{-\gamma}}{2 \Gamma(2 - \gamma)} + \frac{1}{h} \right) U^n_i &= \frac{\tau^{-\gamma}}{2 \Gamma(2 - \gamma)} \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) U^k_i \\
&\quad - \frac{\tau^{-\gamma}}{2 \Gamma(2 - \gamma)} G_n U^0_{i-1} - \frac{\tau^{-\gamma}}{2 \Gamma(2 - \gamma)} G_n U^1_i + f^n_{i-\frac{1}{2}}.
\end{align*}

We have the following two procedures to solve the above scheme. Denote $c = \frac{\tau^{-\gamma}}{2 \Gamma(2 - \gamma)}$.

**Direct Procedure:** let

\begin{equation*}
U^n = (U^n_1, U^n_2, \ldots, U^n_{M-2}, U^n_{M-1})^T,
\end{equation*}

\begin{equation*}
F^n = (f^n_{\frac{1}{2}} - cG_n U^n_0 - cG_n U^n_0, f^n_{\frac{1}{2}} - cG_n U^n_2 - cG_n U^n_1, \ldots, f^n_{M-\frac{1}{2}} - cG_n U^n_{M-2} - cG_n U^n_{M-3}, f^n_{M-\frac{1}{2}} - cG_n U^n_{M-1} - cG_n U^n_{M-2})^T,
\end{equation*}

then we obtain the matrix form of the finite difference formulation as follows:

\begin{equation*}
AU^n = c \sum_{k=1}^{n-1} B_k U^k + F^n.
\end{equation*}

Noting that $U^n = (U^n_1, U^n_2, \ldots, U^n_{M-1})^T$, then the stiffness matrix $A$ becomes the following formulation:

\begin{equation*}
A = \begin{pmatrix}
\frac{1}{h} + cG_0 & 0 & \cdots & 0 & 0 \\
cG_0 - \frac{1}{h} & \frac{1}{h} + cG_0 & \cdots & 0 & 0 \\
0 & cG_0 - \frac{1}{h} & \ddots & \vdots & \vdots \\
\vdots & \vdots & \ddots & \ddots & 0 \\
0 & 0 & \cdots & cG_0 - \frac{1}{h} & \frac{1}{h} + cG_0
\end{pmatrix},
\end{equation*}

and define $b_k = G_{n-k} - G_{n-k+1}$, we obtain

\begin{equation*}
B_k = \begin{pmatrix}
b_k & 0 & \cdots & 0 & 0 \\
b_k & b_k & \cdots & 0 & 0 \\
0 & b_k & \ddots & \vdots & \vdots \\
\vdots & \vdots & \ddots & \ddots & 0 \\
0 & 0 & \cdots & b_k & b_k
\end{pmatrix}.
\end{equation*}
New Procedure: let

\[ U_i = (U_i^1, U_i^2, \ldots, U_i^{N-1}, U_i^N)^T , \]

\[
F_{i-\frac{1}{2}} = (f_{i-\frac{1}{2}}^1 - cG_1 U_i^0 - cG_1 U_{i-1}^0, f_{i-\frac{1}{2}}^2 - cG_2 U_i^0 - cG_2 U_{i-1}^0, \ldots, f_{i-\frac{1}{2}}^{N-1} - cG_{N-1} U_i^0 - cG_{N-1} U_{i-1}^0, f_{i-\frac{1}{2}}^N - cG_N U_i^0 - cG_N U_{i-1}^0)^T ,
\]

then we obtain a new matrix form of the finite difference formulation as follows:

\[
\tilde{A}U_i = BU_{i-1} + F_{i-\frac{1}{2}}.
\]

By rearranging the order of unknowns, we use \( U_i = (U_i^1, U_i^2, \ldots, U_i^{N-1}, U_i^N)^T \) instead of \( U^n = (U_i^n, U_i^{n+1}, \ldots, U_M^N)^T \), then the new stiffness matrix \( \tilde{A} \) can be expressed as

\[
\tilde{A} = \begin{pmatrix}
\frac{1}{h} + cG_0 & 0 & \cdots & 0 & 0 \\
cG_1 - cG_0 & \frac{1}{h} + cG_0 & 0 & \cdots & 0 \\
cG_2 - cG_1 & cG_1 - cG_0 & \frac{1}{h} + cG_0 & \ddots & \vdots \\
& \ddots & \ddots & \ddots & \ddots \\
cG_{N-1} - cG_{N-2} & cG_{N-2} - cG_{N-3} & \cdots & cG_1 - cG_0 & \frac{1}{h} + cG_0
\end{pmatrix},
\]

and

\[
B = \begin{pmatrix}
\frac{1}{h} - cG_0 & 0 & \cdots & 0 & 0 \\
cG_0 - cG_1 & \frac{1}{h} - cG_0 & 0 & \cdots & 0 \\
cG_1 - cG_2 & cG_1 - cG_2 & \frac{1}{h} - cG_0 & \ddots & \vdots \\
& \ddots & \ddots & \ddots & \ddots \\
cG_{N-2} - cG_{N-1} & cG_{N-3} - cG_{N-2} & \cdots & cG_0 - cG_1 & \frac{1}{h} - cG_0
\end{pmatrix}.
\]

By similar analysis as Theorem 3, it is easy to obtain that \( \tilde{A} \) and \( B \) are both asymmetric Toeplitz matrices. From Figure 1, we know that the solution order has been changed which leads to the change of the structure of stiffness matrix.

Next, we consider the following fractional diffusion-wave equation

\[
\frac{D_t^\gamma}{D_t^\gamma} u(x, t) + a \frac{\partial u(x, t)}{\partial x} = f(x, t), \quad x \in (0, L), \quad t \in (0, T], \quad 1 < \gamma < 2,
\]

subject to the initial condition:

\[
u(x, 0) = u_0(x), \quad \frac{\partial u(x, 0)}{\partial t} = \phi(x), \quad x \in [0, L],
\]

with the boundary conditions

\[
u(0, t) = 0, \quad t \in [0, T], \quad a > 0,
\]

\[
u(L, t) = 0, \quad t \in [0, T], \quad a < 0,
\]
without loss of generality, we define $a = 1$.

**Scheme 3**: Suppose $u \in C^{2,3}_{x,t}([0, L] \times [0, T])$, the time fractional Caputo derivative $D_0^\gamma u(x, t)$ for $1 < \gamma < 2$ at $(x_i, t_{n-\frac{1}{2}})$ is discretized by [27]

\[
C_0^\gamma D_t^\gamma u(x_i, t_{n-\frac{1}{2}}) = \frac{\tau^{1-\gamma}}{\Gamma(3-\gamma)} \left[ M_0 d_i u_i^{n-1/2} - \sum_{k=1}^{n-1} (M_{n-k-1} - M_{n-k}) d_i u_i^{k-1/2} \right] \\
- \frac{\tau^{1-\gamma}}{2\Gamma(3-\gamma)} M_{n-1} \phi_i + O(\tau^{3-\gamma}).
\]  

Combining equation (13) with equation (15), we obtain the following finite dif-
ference discretization formulation

\[
\frac{\tau^{1-\gamma}}{\Gamma(3-\gamma)} \left[ M_0 d_t u_i^n - \frac{n-1}{2} \sum_{k=1}^{n-1} (M_{n-k-1} - M_{n-k}) d_t u_i^{k-1/2} \right] \\
- \frac{\tau^{1-\gamma}}{2\Gamma(3-\gamma)} M_{n-1} \phi_i + \frac{u_i^n - u_i^{n-1}}{h} = f_i^{n-1/2} + O(\tau^{3-\alpha} + h).
\]

(16)

Define \( c = \frac{\tau^{1-\gamma}}{\Gamma(3-\gamma)} \), some simplification leads to

\[
c M_0 u_i^n + c(M_1 - 2M_0) u_i^{n-1} + c \sum_{k=1}^{n-2} (M_{n-k-2} - 2M_{n-k-1} + M_{n-k}) u_i^k \\
+ c(M_{n-2} - M_{n-1}) u_i^0 - c\tau M_{n-1} \phi_i + \frac{u_i^n - u_i^{n-1}}{h} = f_i^{n-1/2} + O(\tau^{3-\alpha} + h).
\]

(17)

Direct Procedure: let

\[
U^n = (U^n_1, U^n_2, \ldots, U^n_{M-2}, U^n_{M-1})^T,
\]

\[
F^{n-\frac{1}{2}} = \begin{pmatrix}
    f_1^{n-\frac{1}{2}} + c\tau M_{n-1} \phi_1 - c(M_{n-2} - M_{n-1}) u_i^0 \\
    f_2^{n-\frac{1}{2}} + c\tau M_{n-1} \phi_2 - c(M_{n-2} - M_{n-1}) u_i^0 \\
    \vdots \\
    f_{M-1}^{n-\frac{1}{2}} + c\tau M_{n-1} \phi_{M-1} - c(M_{n-2} - M_{n-1}) u_i^0
\end{pmatrix},
\]

then we obtain the matrix form of the finite difference formulation as follows:

\[
AU^n = c \sum_{k=1}^{n-1} B_k U^k + F^{n-\frac{1}{2}}.
\]

Noting that \( U^n = (U^n_1, U^n_2, \ldots, U^n_{M-1})^T \), the stiffness matrix \( A \) can be expressed as:

\[
A = \begin{pmatrix}
    \frac{1}{h} + cM_0 & 0 & \cdots & 0 & 0 \\
    -\frac{1}{h} & \frac{1}{h} + cM_0 & \cdots & 0 & 0 \\
    0 & -\frac{1}{2h} & \ddots & \vdots & \vdots \\
    \vdots & \vdots & \ddots & \ddots & 0 \\
    0 & 0 & \cdots & -\frac{1}{h} & \frac{1}{h} + cM_0
\end{pmatrix},
\]

and if we denote \( b_k = -(M_{n-k-2} - 2M_{n-k-1} + M_{n-k}) \), then we have, for \( 1 \leq k \leq n-2 \),

\[
B_k = \begin{pmatrix}
    b_k & 0 & \cdots & 0 & 0 \\
    b_k & b_k & \cdots & 0 & 0 \\
    0 & b_k & \ddots & \vdots & \vdots \\
    \vdots & \vdots & \ddots & \ddots & 0 \\
    0 & 0 & \cdots & b_k & b_k
\end{pmatrix}.
\]
For $k = n - 1$, we have
\[
B_{n-1} = \begin{pmatrix}
2cM_0 - cM_1 & 0 & \cdots & 0 & 0 \\
0 & 2cM_0 - cM_1 & \cdots & 0 & 0 \\
0 & 0 & \ddots & \vdots & \vdots \\
\vdots & \vdots & \ddots & \ddots & 0 \\
0 & 0 & \cdots & 0 & 2cM_0 - cM_1
\end{pmatrix}.
\]

**New Procedure:** By similar analysis as Scheme 3, we obtain the matrix form of the finite difference formulation as follows:
\[
\tilde{A}U_i = \frac{1}{h}U_{i-1} + F_i,
\]
where
\[
U_i = (U^1_i, U^2_i, \ldots, U^{N-1}_i, U^N_i)^T,
\]
and
\[
F_i = \begin{pmatrix}
f_i^1 + c\tau M_0 \phi_i \\
f_i^2 + c\tau M_1 \phi_i - c(M_0 - M_1)u^0_i \\
\vdots \\
f_i^{N-3} + c\tau M_{N-2} \phi_i - c(M_{N-3} - M_{N-2})u^0_i \\
f_i^{N-2} + c\tau M_{N-1} \phi_i - c(M_{N-2} - M_{N-1})u^0_i
\end{pmatrix}.
\]

Denote $W_j = M_{j-2} - 2M_{j-1} + M_j$. More importantly, the stiffness matrix $\tilde{A}$ can be expressed as
\[
\tilde{A} = \begin{pmatrix}
\frac{1}{h} + cM_0 & 0 & \cdots & 0 & 0 \\
cM_1 - 2cM_0 & \frac{1}{h} + cM_0 & 0 & \cdots & 0 \\
cW_2 & cM_1 - 2cM_0 & \frac{1}{h} + cM_0 & \ddots & \vdots \\
\vdots & \vdots & \ddots & \ddots & 0 \\
cW_{N-1} & cW_{N-2} & \cdots & cM_1 - 2cM_0 & \frac{1}{h} + cM_0
\end{pmatrix}.
\]

By simple analysis, we can prove that $\tilde{A}$ is a Toeplitz matrix.

Next, we consider the following time fractional diffusion equation
\[
(18) \quad C_0 D_t^\gamma u(x, t) = \frac{\partial^2 u(x, t)}{\partial x^2} + f(x, t), \quad x \in (0, L), \quad t \in (0, T], \quad 0 < \gamma < 1,
\]
subject to the initial condition:
\[
u(x, 0) = u_0(x), \quad x \in [0, L],
\]
and the boundary conditions
\[
u(0, t) = u(L, t) = 0, \quad t \in [0, T].
\]
Many numerical approaches have been designed to solve above time fractional diffusion equation. Among them, we consider the classical implicit finite difference scheme, which has been proved to have unconditional stability and $L^2$ norm convergence. The classical finite difference scheme given by Lin and Xu [14] for the one-dimensional fractional diffusion equation has the following form:

**Scheme 4:**

\[
\frac{\tau^{-\gamma}}{\Gamma(2 - \gamma)} \left[ G_0 u_i^n - \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) u_i^k + G_n u_i^0 \right] \\
= \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{h^2} + f_i^n + O(\tau^{2-\gamma} + h^2).
\]

Then, we can rewrite the finite difference scheme to the following matrix form:

\[
AU^n = \frac{\tau^{-\gamma}}{\Gamma(2 - \gamma)} \left\{ \sum_{k=1}^{n-1} (G_{n-k-1} - G_{n-k}) U^k + G_n U^0 \right\} + F^n,
\]

where $U^n = (U^n_1, U^n_2, \ldots, U^n_{M-1})^T$, $F^n = (F^n_1, F^n_2, \ldots, F^n_{M-1})^T$, and the stiffness matrix $A$ becomes the following formulation:

\[
A = \begin{pmatrix}
\frac{\tau}{\pi} + cG_0 & -\frac{1}{\pi} & 0 & \cdots & 0 & 0 \\
-\frac{1}{\pi} & \frac{2}{\pi} + cG_0 & -\frac{1}{\pi} & \cdots & 0 & 0 \\
0 & -\frac{1}{\pi} & \frac{2}{\pi} + cG_0 & -\frac{1}{\pi} & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & -\frac{1}{\pi} & \frac{1}{\pi} + cG_0 & -\frac{1}{\pi} \\
0 & 0 & \cdots & -\frac{1}{\pi} & \frac{1}{\pi} + cG_0 & \frac{1}{\pi} + cG_0
\end{pmatrix}.
\]

The linear system (20) can be solved by the time-marching method. However, it needs an average of $O(MN)$ operations to obtain a RHS vector and the time-marching method for system (20) requires an overall computational complexity of $O(MN^2)$. Next, we will consider a fast method to solve the linear system (20). First, we define

\[
\bar{U} = (U_1, U_2, \ldots, U_{M-2}, U_{M-1})^T,
\]

where

\[
U_i = (U^1_i, U^2_i, \ldots, U^{N-1}_i, U^n_i).
\]

Then, the linear system (20) can be rewritten as

\[
\bar{A}\bar{U} = \bar{F},
\]

where the stiffness matrix $\bar{A}$ can be written as the following block matrix

\[
\bar{A} = \begin{pmatrix}
\bar{A} & \bar{B} & 0 & \cdots & 0 & 0 \\
\bar{B} & \bar{A} & \bar{B} & \cdots & 0 & 0 \\
0 & \bar{B} & \bar{A} & \bar{B} & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & \cdots & \bar{B} & \bar{A} & \bar{B} \\
0 & 0 & \cdots & \bar{B} & \bar{A}
\end{pmatrix}.
\]

This manuscript is for review purposes only.
By simple calculation, it is easy to obtain

\[
\hat{A} = \begin{pmatrix}
\frac{2}{h^2} + cG_0 & 0 & \cdots & 0 & 0 \\
cG_1 - cG_0 & \frac{1}{h} + cG_0 & 0 & \cdots & 0 \\
cG_2 - cG_1 & cG_1 - cG_0 & \frac{1}{h} + cG_0 & \ddots & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots \\
cG_{N-1} - cG_{N-2} & cG_{N-2} - cG_{N-3} & \cdots & cG_1 - cG_0 & \frac{1}{h} + cG_0 \\
\end{pmatrix},
\]

where \( \hat{A} \) is the block lower Toeplitz matrix. Here the block matrix \( \hat{B} \) is

\[
\hat{B} = \begin{pmatrix}
-\frac{1}{h^2} & 0 & 0 & \cdots & 0 \\
0 & -\frac{1}{h^2} & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots \\
0 & \cdots & \cdots & -\frac{1}{h^2} & 0 \\
0 & \cdots & \cdots & 0 & -\frac{1}{h^2} \\
\end{pmatrix}.
\]

By making use of block Toeplitz structure in (20), the proposed procedure can be employed to solve the equations with a lower cost at \( O(MN \log^2 N) \) by using fast Fourier transforms.

### 3.1. The fast finite difference method.

For general real nonsymmetric linear system \( Au = b \), we introduce the generalized minimum residual (GMRES) method [34]. The GMRES method was proposed by Saad and Schultz in 1986, which is one of the most important Krylov subspace methods for real nonsymmetric linear system [23]. Let \( u_0 \) be an initial guess. Then the following GMRES algorithm can be used to solve \( Au = b \).

\[
\begin{align*}
& r_0 = b - Ax_0, \quad \beta = \|r_0\|_2, \quad v_1 = r_0 / \beta \\
& \text{for } j = 1 : k \\
& \quad w_j = Av_j \\
& \quad \text{for } i = 1 : j \\
& \quad \quad h_{ij} = w_j^T v_i \\
& \quad \quad w_j = w_j - h_{ij} v_i \\
& \quad \text{end} \\
& \quad h_{j+1,j} = \|w_j\|_2 \quad \text{if } h_{j+1,j} = 0, \text{ set } k = j \text{ and go to (*)} \\
& \quad v_{j+1} = w_j / h_{j+1,j} \\
& \quad \text{end} \\
& \omega_k = r_{k-1}^T r_{k-1} / d_k^T A d_k \\
& \text{(*) compute } y_k \text{ the minimizer of } \|\beta e_1 - \tilde{H} k y\|_2 \\
& u_k = u_0 + V y_k \\
& u = u_k
\end{align*}
\]
where $\tilde{H}_k = (h_{ij})_{1 \leq i,j \leq k}$. The matrix $V = (v_1, v_2, \ldots, v_k) \in \mathbb{R}^{n \times k}$ with $k \leq n$ is a matrix with orthonormal columns. In order to avoid a large storage and computational cost for the orthogonalization, the GMRES method is usually restarted after each $m$ ($m \ll n$) iteration steps (refer to the GMRES(m) method).

To reduce the computational work and memory requirement, we need only to accelerate the matrix-vector multiplication $Ad$ for any vector $d$ and store $A$ efficiently. Let $a_{j-i}$ denote the common entry in the $(j-i)$-th descending diagonal of $A$ from left to right. Namely, $A_{i,j} = a_{j-i}, \forall j \geq i$.

The stiffness matrix $A$ can be embedded into a $2N \times 2N$ circulate matrix $C$ as follow[3, 6, 30]

$$C = \begin{pmatrix} A & B \\ B & A \end{pmatrix}, \quad B = \begin{pmatrix} a_0 & a_{N-1} & \cdots & a_2 & a_1 \\ a_{N-1} & 0 & a_{N-2} & \cdots & a_2 \\ \vdots & a_{N-2} & 0 & \ddots & \vdots \\ a_2 & \vdots & \ddots & \ddots & a_{N-1} \\ a_1 & a_2 & \cdots & a_{N-1} & 0 \end{pmatrix}. $$

The circulate matrix $C$ has the following decomposition [3, 30, 31]

$$C = F^{-1} \text{diag}(Fc)F, \quad (22)$$

where $c$ is the first column vector of $C$ and $F$ is the $2N \times 2N$ discrete Fourier transform matrix. Denote that $w = (d, d)^T$, then it is well known that the matrix-vector multiplication $Fw$ for $w \in \mathbb{R}^{2N}$ can be carried out in $O(N \log N)$ operations via the fast Fourier transform. Equation (22) shows that $Cw$ can be evaluated in $O(N \log N)$ operations. So, we know that $Ad$ can be evaluated in $O(N \log N)$ operations for any $d \in \mathbb{R}^N$. The overall computation cost of the fast conjugate method is $O(N \log^2 N)$.

4. Numerical results. In this section, some computational experiments have been carried out. Here we use Gaussian elimination (Gauss), the conjugate gradient (CG) method, and the fast conjugate gradient (FCG) method to solve two-sided fractional ordinary differential equation and direct method and fast method including fast generalized minimum residual method to solve time fractional partial differential equations for comparison. These methods were implemented in Matlab, and the numerical experiments were run on a 8-GB memory computer.

Example 1: we consider the following two-sided ordinary fractional differential equation with the real solution $u(t) = t(1-t)$:

$$\begin{align*}
0D_t^\gamma u(t) + tD_t^\gamma u(t) + u(t) &= f(t), \quad t \in (0, 1), \\
u(0) = u(1) = 0,
\end{align*}$$

with forcing function

$$f(t) = \frac{t^{1-\gamma}}{\Gamma(2-\gamma)} - \frac{2t^{2-\gamma}}{\Gamma(3-\gamma)} + \frac{2t^{2-\gamma}1}{\Gamma(3-\gamma)}(\gamma t - 2t^2 - \gamma + 2t)(1-t)^{-\gamma} + t(1-t).$$

In the tables and figures, we provide the $L^2(\Omega)$ norm of the error and the corresponding rates of convergence for a sequence of grid sizes. From Table 1, one can see that Scheme 1 for the ordinary differential model (2) converge at the optimal rates of $O(h^{2-\gamma})$. In Table 2, we present the CPU time consumed by the Gauss, CG and FCG.
methods. We observe that the FCG solver results in a significantly less computational
time, compared to the Gauss and CG solvers. Moreover, as the mesh size $h$ is reduced
to $h = 2^{-15}$, both Gauss and CG methods run out of memory, whereas the FCG
method solves the problem still consuming relatively little computational time.

### Table 1
The $L_2$ errors and convergence rates for Scheme 1 for different values of $\gamma$.

| $\tau$  | $\gamma = 0.1$ | Rate | $\gamma = 0.5$ | Rate | $\gamma = 0.9$ | Rate |
|---------|----------------|------|----------------|------|----------------|------|
| $1/2^5$ | $7.8554e-5$    | -    | $1.5571e-3$    | -    | $1.9214e-2$    | -    |
| $1/2^6$ | $2.3116e-5$    | 1.7648 | $5.7041e-4$    | 1.4488 | $9.6294e-3$    | 1.0424 |
| $1/2^7$ | $6.7000e-6$    | 1.7867 | $2.0606e-4$    | 1.4689 | $4.6751e-3$    | 1.0681 |
| $1/2^8$ | $1.9201e-6$    | 1.8030 | $7.3842e-5$    | 1.4806 | $2.2298e-3$    | 1.0958 |
| $1/2^9$ | $5.4550e-7$    | 1.8155 | $2.6334e-5$    | 1.4875 | $1.0530e-3$    | 1.0872 |
| $1/2^{10}$ | $1.5388e-7$ | 1.8258 | $9.3635e-6$    | 1.4918 | $4.9455e-4$    | 1.0903 |
| $1/2^{11}$ | $4.3158e-8$ | 1.8341 | $3.3230e-6$    | 1.4946 | $2.3155e-4$    | 1.0948 |
| $1/2^{12}$ | $1.2045e-8$ | 1.8412 | $1.1779e-6$    | 1.4963 | $1.0823e-4$    | 1.0972 |

### Table 2
The CPU time consumed by the Gauss, CG and FCG methods for different mesh sizes for the
Scheme 1.

| $\tau$  | Gauss method | CG method | FCG method |
|---------|--------------|-----------|------------|
| $2^{-9}$ | 1.20 s -     | 0.21 s 62 | 0.01 s 62  |
| $2^{-10}$ | 9.90 s -    | 1.27 s 75 | 0.06 s 75  |
| $2^{-11}$ | 103 s -     | 5.58 s 91 | 0.09 s 91  |
| $2^{-12}$ | 931 s -     | 23.8 s 110 | 0.57 s 110 |
| $2^{-13}$ | 7328 s -    | 105 s 133 | 1.29 s 133  |
| $2^{-14}$ | >10 h -    | 515 s 159 | 2.24 s 159  |
| $2^{-15}$ | Out of memory - | Out of memory - | 4.58 s 192 |
| $2^{-16}$ | N/A -     | N/A - 230 | 9.81 s 230  |
| $2^{-17}$ | N/A -     | N/A - 276 | 27.2 s 276  |
| $2^{-18}$ | N/A -     | N/A - 331 | 80.9 s 331  |
| $2^{-19}$ | N/A -     | N/A - 415 | 281 s 415  |
| $2^{-20}$ | N/A -     | N/A - 498 | 451 s 498  |
| $2^{-21}$ | N/A -     | N/A - 599 | 977 s 599  |
| $2^{-22}$ | N/A -     | N/A - 741 | 2548 s 741  |

**Example 2**: we consider the following partial fractional differential equation with
the real solution $u(x, t) = t \sin(\pi x)$:

\[
\begin{align*}
\frac{C_0}{\Gamma(2-\gamma)} D^\gamma_t u(x, t) + \frac{\partial u(x, t)}{\partial x} &= f(x, t), \quad x \in (0, 1), \ t \in (0, 1], \ 0 < \gamma < 1, \\
u(x, 0) &= 0, \quad x \in [0, 1], \\
u(0, t) &= \nu(t, 1) = 0, \quad t \in [0, 1],
\end{align*}
\]

with forcing function

\[
f(x, t) = \frac{t^{1-\gamma}}{\Gamma(2-\gamma)} \sin(\pi x) + \pi t \cos(\pi x).
\]
We check the errors and convergence rates for Scheme 2 by using Example 2. We fixed the temporal mesh size \( \tau = 1/10 \) and refine the spatial mesh size \( h \) from \( \frac{1}{2} \) to \( \frac{1}{256} \). Table 3 shows the numerical results for two different fractional values: \( \gamma = 0.1 \) and \( \gamma = 0.9 \). While Table 3 shows that the convergence order in space is \( O(h^2) \) for both the direct method and our fast procedure. From Table 4, we observe that the fast procedure is much faster than the direct method.

### Table 3

The \( L_2 \) errors and convergence rates for direct method and fast method with different values of \( \gamma \).

| \( h \)       | \( \gamma = 0.1 \) |                | \( \gamma = 0.9 \) |
|-------------|------------------|-----------------|---------------------|
|             | Direct method    | Fast method     | Direct method       | Fast method        |
|             | \( L_2 \) error  | Rate            | \( L_2 \) error     | Rate               |
| 1/2\(^4\)  | 7.8115e-2        | -               | 8.5609e-2           | -                  |
| 1/2\(^5\)  | 1.9620e-2        | 1.99            | 2.1600e-2           | 1.98               |
| 1/2\(^6\)  | 4.9189e-3        | 1.99            | 5.4270e-3           | 1.99               |
| 1/2\(^8\)  | 1.2316e-3        | 2.00            | 1.3602e-3           | 2.00               |
| 1/2\(^9\)  | 3.0816e-4        | 2.00            | 3.4051e-4           | 2.00               |
| 1/2\(^10\) | 7.7072e-5        | 2.00            | 8.5185e-5           | 2.16               |

### Table 4

The CPU time consumed by direct and fast methods with different mesh sizes for Scheme 2.

| \( h = \tau \) | 1/2\(^2\) | 1/2\(^3\) | 1/2\(^5\) | 1/2\(^11\) | 1/2\(^13\) |
|---------------|----------|----------|----------|-----------|-----------|
| Direct method (CPU) | 7.14s   | 57.7s    | 471s     | 3826s     | \( >10h \) | \( >2d \) |
| Fast method (CPU)   | 0.17s   | 0.43s    | 1.49s    | 5.56s     | 399s      | 1191s     |

### Example 3:

We consider the following partial fractional differential equation with the real solution \( u(x, t) = t^3x(1-x) \):

\[
\begin{align*}
\mathcal{C}_0 D_t^\gamma u(x, t) + \frac{\partial u(x, t)}{\partial x} &= f(x, t), \quad x \in (0, 1), \quad t \in (0, 1], \quad 1 < \gamma < 2, \\
u(x, 0) &= 0, \quad \frac{\partial u(x, 0)}{\partial t} = 0, \quad x \in [0, 1], \\
u(0, t) &= 0, \quad t \in [0, 1],
\end{align*}
\]

with forcing function

\[
f(x, t) = \frac{6t^{3-\gamma}}{\Gamma(4-\gamma)} (x - x^2) + t^3(1 - 2x).
\]

Then, we check the errors and convergence rates for Scheme 3 by using Example 3. We take \( h = \tau \) from \( \frac{1}{8} \) to \( \frac{1}{256} \) to check the convergence rates. Table 5 shows the numerical results for two different fractional values: \( \gamma = 1.1 \) and \( \gamma = 1.9 \). While Table 5 shows that the convergence order in space is \( O(h) \) for both the direct method and our fast method. From Table 6, we observe that our fast procedure is much faster than the direct procedure.

### Example 4:

We consider the following time fractional diffusion equation with the
real solution $u(x,t) = t^3 \sin(\pi x)$:

$$
\begin{aligned}
\mathcal{C}D_t^{\gamma}u(x,t) &= \frac{\partial^2 u(x,t)}{\partial x^2} + f(x,t), \quad x \in (0,1), \quad t \in (0,1), \quad 0 < \gamma < 1, \\
u(x,0) &= 0, \quad x \in [0,1], \\
u(0,t) &= 0, \quad t \in [0,1],
\end{aligned}
$$

with forcing function

$$
f(x,t) = \frac{6t^{3-\gamma}}{\Gamma(4-\gamma)} \sin(\pi x) + \pi^2 t^3 \sin(\pi x).
$$

Errors and convergence rates are considered for Scheme 4 by using Example 4. We take $h = \tau$ from $\frac{1}{2^2}$ to $\frac{1}{2^8}$ to check the convergence rates. Table 7 shows the numerical results for two different fractional values: $\gamma = 0.1$ and $\gamma = 0.9$. While Table 7 shows that the direct method and our fast procedure have the same convergence order. From Table 8, we can also observe that our fast method is much faster than the direct method.

5. Conclusions. For time fractional equations, unless we extract the Toeplitz structure for stiffness matrix, we can not use fast Fourier transform to speed up the evaluation. In this article, we study a fast procedure to solve both ordinary and partial fractional differential equations based on Caputo fractional derivative. For differential equations with Caputo fractional time derivative, we give several different finite difference schemes and analyse the stiffness matrices. We present a fast solution technique depending on the special structure of coefficient matrices by rearranging the order of unknowns in time and space directions. We observe that our fast procedure is much faster than the direct procedure.

Acknowledgments. We would like to acknowledge the assistance of volunteers in putting together this example manuscript and supplement. This work was supported in part by the National Natural Science Foundation of China under Grants 91630207, 11471194 and 11571115, by the National Science Foundation under Grant DMS-1216923, by the OSD/ARO MURI Grant W911NF-15-1-0562, by the National Science and Technology Major Project of China under Grants 2011ZX05052 and 2011ZX05011-004, and by Shandong Provincial Natural Science Foundation, China under Grant ZR2011AM015, and by Taishan Scholars Program of Shandong Province of China.

| $h$ | $\gamma = 1.1$ | $\gamma = 1.9$ |
|-----|----------------|----------------|
|     | Direct method  | Fast method    | Direct method | Fast method    |
|     | $L_2$ error    | Rate           | $L_2$ error    | Rate           |
| $1/2^1$ | 3.0960e-2 | -              | 3.0960e-2 | -              |
| $1/2^2$ | 1.6276e-2 | 0.93           | 1.6275e-2 | 0.93           |
| $1/2^3$ | 8.3729e-3 | 0.96           | 8.3729e-3 | 0.96           |
| $1/2^4$ | 4.2530e-3 | 0.98           | 4.2529e-3 | 0.98           |
| $1/2^5$ | 2.1445e-3 | 0.99           | 2.1444e-3 | 0.99           |
| $1/2^6$ | 1.0769e-3 | 0.99           | 1.0769e-3 | 0.99           |
| $1/2^7$ | 0.5693e-4 | 1.00           | 0.5694e-4 | 1.00           |

This manuscript is for review purposes only.
Table 6
The CPU time consumed by direct and fast methods with different mesh sizes for Scheme 3.

| $h = \tau$ | 1/2$^9$ | 1/2$^8$ | 1/2$^7$ | 1/2$^6$ | 1/2$^5$ | 1/2$^4$ |
|------------|---------|---------|---------|---------|---------|--------|
| Direct method (CPU) | 0.62s | 4.77s | 38.0s | 302s | >10h | >2d |
| Fast method (CPU) | 0.14s | 0.23s | 0.57s | 2.01s | 6.67s | 27.4s |

REFERENCES

[1] A. Ashyralyev and Z. Cakir, On the numerical solution of fractional parabolic partial differential equations with the dirichlet condition, Discrete Dynamics in Nature and Society, 2012 (2012), http://dx.doi.org/10.1155/2012/696179.

[2] A. Ashyralyev and Z. Cakir, Fdm for fractional parabolic equations with the neumann condition, Advances in Difference Equations, 2013 (2013), pp. 1–16.

[3] A. Böttcher and B. Silbermann, Introduction to large truncated Toeplitz matrices, Springer Science & Business Media, New York, 1999.

[4] S. Chen, F. Liu, X. Jiang, I. Turner, and V. Anh, A fast semi-implicit difference method for a nonlinear two-sided space-fractional diffusion equation with variable diffusivity coefficients, Applied Mathematics and Computation, 257 (2015), pp. 591–601.

[5] A. Cheng, H. Wang, and K. Wang, A eulerian–lagrangian control volume method for solute transport with anomalous diffusion, Numerical Methods for Partial Differential Equations, 31 (2015), pp. 253–267.

[6] R. M. Gray, Toeplitz and circulant matrices: A review, vol. 77, Now Publishers, 2006.

[7] M. Hernández-Hernández, V. N. Kolokoltsov, et al., On the solution of two-sided fractional ordinary differential equations of caputo type, Fractional Calculus and Applied Analysis, 19 (2016), pp. 1393–1413.

[8] J. Huang, Y. Tang, L. Vázquez, and J. Yang, Two finite difference schemes for time fractional diffusion-wave equation, Numerical Algorithms, 64 (2013), pp. 707–720.

[9] S. Jiang, J. Zhang, Q. Zhang, and Z. Zhang, Fast evaluation of the caputo fractional derivative and its applications to fractional diffusion equations, arXiv preprint arXiv:1511.03453, (2015).

[10] Y. Jiang and J. Ma, High-order finite element methods for time-fractional partial differential equations, Journal of Computational and Applied Mathematics, 235 (2011), pp. 3285–3290.

[11] R. Ke, M. K. Ng, and H.-W. Sun, A fast direct method for block triangular toeplitz-like with tri-diagonal block systems from time-fractional partial differential equations, Journal of Computational Physics, 303 (2015), pp. 203–211.

[12] W. Li and X. Da, Finite central difference/finite element approximations for parabolic integro-differential equations, Computing, 90 (2010), pp. 89–111.

[13] Y. Lin, X. Li, and C. Xu, Finite difference/spectral approximations for the fractional cable equation, Mathematics of Computation, 80 (2011), pp. 1369–1396.

[14] Y. Lin and C. Xu, Finite difference/spectral approximations for the time-fractional diffusion equation, Journal of Computational Physics, 225 (2007), pp. 1533–1552.

Table 7
The $L_2$ errors and convergence rates for direct method and fast method with different values of $\gamma$ for Scheme 4.

| $h = \tau$ | $\gamma = 0.1$ | $\gamma = 0.9$ |
|------------|---------------|---------------|
| Direct method | Fast method | Direct method | Fast method |
| $L_2$ error | Rate | $L_2$ error | Rate | $L_2$ error | Rate | $L_2$ error | Rate |
| 1/2$^1$ | 8.4669e-3 | - | 8.4669e-3 | - | 2.3623e-2 | - | 2.3623e-2 | - |
| 1/2$^2$ | 2.1203e-3 | 1.99 | 2.1203e-3 | 1.99 | 9.5705e-3 | 1.30 | 9.5705e-3 | 1.30 |
| 1/2$^3$ | 5.3323e-4 | 1.99 | 5.3323e-4 | 1.99 | 4.1202e-3 | 1.21 | 4.1202e-3 | 1.21 |
| 1/2$^4$ | 1.3428e-4 | 1.98 | 1.3428e-4 | 1.98 | 1.8378e-3 | 1.16 | 1.8378e-3 | 1.16 |
| 1/2$^5$ | 3.3841e-5 | 1.98 | 3.3796e-5 | 1.99 | 8.3651e-4 | 1.13 | 8.3140e-4 | 1.14 |
| 1/2$^6$ | 8.5339e-6 | 1.99 | 8.4517e-6 | 2.00 | 3.8563e-4 | 1.12 | 3.5499e-4 | 1.22 |
Table 8
The CPU time consumed by direct and fast methods with different mesh sizes for Scheme 4 with \( \gamma = 0.1 \).

| \( h = \tau \) | 1/2^6 | 1/2^7 | 1/2^8 | 1/2^9 | 1/2^10 |
|---|---|---|---|---|---|
| Direct method (CPU) | 0.09s | 0.73s | 6.09s | 57.0s | 641s |
| Fast method (CPU) | 0.00s | 0.07s | 0.26s | 2.27s | 15.6s |

[15] F. Liu, P. Zhuang, and K. Burrage, Numerical methods and analysis for a class of fractional advection–dispersion models, Computers & Mathematics with Applications, 64 (2012), pp. 2990–3007.

[16] F. Liu, P. Zhuang, I. Turner, K. Burrage, and V. Anh, A new fractional finite volume method for solving the fractional diffusion equation, Applied Mathematical Modelling, 38 (2014), pp. 3871–3878.

[17] Q. Liu, F. Liu, I. Turner, Y. Anh, and Y. Gu, A rbf meshless approach for modeling a fractal mobile/immobile transport model, Applied Mathematics and Computation, 226 (2014), pp. 336–347.

[18] Y. Liu, Y. Du, H. Li, S. He, and W. Gao, Finite difference/finite element method for a nonlinear time-fractional fourth-order reaction–diffusion problem, Computers & Mathematics with Applications, 70 (2015), pp. 573–591.

[19] Y. Liu, Y. Du, H. Li, and J. Wang, An \( h^{-1} \)-galerkin mixed finite element method for time fractional reaction–diffusion equation, Journal of Applied Mathematics and Computing, 47 (2015), pp. 103–117.

[20] Y. Liu, Z. Fang, H. Li, and S. He, A mixed finite element method for a time-fractional fourth-order partial differential equation, Applied Mathematics and Computation, 243 (2014), pp. 703–717.

[21] Z. Liu and X. Li, A parallel cgs block-centered finite difference method for a nonlinear time-fractional parabolic equation, Computer Methods in Applied Mechanics and Engineering, 308 (2016), pp. 330–348.

[22] Z. Liu and X. Li, A crank–nicolson difference scheme for the time variable fractional mobile–immobile advection–dispersion equation, Journal of Applied Mathematics and Computing, (2017), pp. 1–20, http://dx.doi.org/10.1007/s12190-016-1079-7.

[23] Y. Saad and M. H. Schultz, Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on scientific and statistical computing, 7 (1986), pp. 856–869.

[24] E. Sousa, Finite difference approximations for a fractional advection diffusion problem, Journal of Computational Physics, 228 (2009), pp. 4038–4054.

[25] E. Sousa, A second order explicit finite difference method for the fractional advection diffusion equation, Computers & Mathematics with Applications, 64 (2012), pp. 3141–3152.

[26] E. Sousa, An explicit high order method for fractional advection diffusion equations, Journal of Computational Physics, 278 (2014), pp. 257–274.

[27] Z.-z. Sun and X. Wu, A fully discrete difference scheme for a diffusion-wave system, Applied Numerical Mathematics, 56 (2006), pp. 193–209.

[28] H. Wang and T. S. Basu, A fast finite difference method for two-dimensional space-fractional diffusion equations, SIAM Journal on Scientific Computing, 34 (2012), pp. A2444–A2458.

[29] H. Wang et al., A preconditioned fast finite difference method for space-time fractional partial differential equations, Fractional Calculus and Applied Analysis, 20 (2017), pp. 88–116.

[30] H. Wang and H. Tian, A fast galerkin method with efficient matrix assembly and storage for a peridynamic model, Journal of Computational Physics, 231 (2012), pp. 7730–7738.

[31] H. Wang and H. Tian, A fast and faithful collocation method with efficient matrix assembly for a two-dimensional nonlocal diffusion model, Computer Methods in Applied Mechanics and Engineering, 273 (2014), pp. 19–36.

[32] K. Wang and H. Wang, A fast characteristic finite difference method for fractional advection–diffusion equations, Advances in water resources, 34 (2011), pp. 810–816.

[33] L. Wei and Y. He, Analysis of a fully discrete local discontinuous galerkin method for time-fractional fourth-order problems, Applied Mathematical Modelling, 38 (2014), pp. 1511–1522.

[34] J. Xiao-qing, Preconditioning Techniques for Toeplitz Systems, Higher Education Press, Beijing, 2010.

[35] F. Zeng, C. Li, F. Liu, and I. Turner, The use of finite difference/element approaches for ...
solving the time-fractional subdiffusion equation, SIAM Journal on Scientific Computing, 35 (2013), pp. A2976–A3000.

[36] H. Zhang, F. Liu, M. S. Phanikumar, and M. M. Meerschaert, A novel numerical method for the time variable fractional order mobile–immobile advection–dispersion model, Computers & Mathematics with Applications, 66 (2013), pp. 693–701.

[37] N. Zhang, W. Deng, and Y. Wu, Finite difference/element method for a two-dimensional modified fractional diffusion equation, Adv. Appl. Math. Mech, 4 (2012), pp. 496–518.

[38] Y. Zhao, P. Chen, W. Bu, X. Liu, and Y. Tang, Two mixed finite element methods for time-fractional diffusion equations, Journal of Scientific Computing, (2015), pp. 1–22.