A FAST HIGH ORDER METHOD FOR TIME FRACTIONAL DIFFUSION EQUATION WITH NON-SMOOTH DATA

HAILI QIAO AND AIJIE CHENG

School of Mathematics, Shandong University
Jinan, Shandong 250100, China

(Communicated by Song Jiang)

Abstract. In this paper, we consider the time fractional diffusion equation with Caputo fractional derivative. Due to the singularity of the solution at the initial moment, it is difficult to achieve an ideal convergence order on uniform meshes. Therefore, in order to improve the convergence order, we discrete the Caputo time fractional derivative by a new $L^1 - 2$ format on graded meshes, while the spatial derivative term is approximated by the classical central difference scheme on uniform meshes. We analyze the approximation about the time fractional derivative, and obtain the time truncation error, but the stability analysis remains an open problem. On the other hand, considering that the computational cost is extremely large, we present a reduced-order finite difference extrapolation algorithm for the time-fraction diffusion equation by means of proper orthogonal decomposition (POD) technique, which effectively reduces the computational cost. Finally, several numerical examples are given to verify the convergence of the scheme and the effectiveness of the reduced order extrapolation algorithm.

1. Introduction. In recent years, fractional differential equations have been widely used in various fields of science and engineering [20]. Fractional equations are usually used to describe dynamic systems, which have been used to represent many natural processes in physics. For example, the fractional diffusion equation, the first-order time partial derivative in the standard diffusion equation is replaced by a fractional derivative of order $\alpha \in (0, 1)$ (in the sense of Riemann-Liouville or Caputo) to obtain the time fractional diffusion equation (TFDE). From a physical point of view, the generalized diffusion equation is obtained by replacing the classical Fick’s law with the fractional Fick’s law, which describes the transportation process with long memory [10]. The fractional derivative represents the degree of memory in the diffusion material, which can describe many non-Markov random walks [12]. In the complex or disordered system of fractal media, due to geometric or energy reasons, the path is limited, and abnormal diffusion occurs. The random walk model is beyond the classic Brownian motion. At this time, it is more effective to use the time fractional diffusion equation to describe this model. For example, Metzler and Klafter [18] proved that the fractional diffusion equation describes the...
Consider the following time fractional diffusion problem
\[
\frac{\partial^\alpha u(x, t)}{\partial t^\alpha} + \mathcal{L}u(x, t) = f(x, t), \quad (x, t) \in \Omega \times (0, T],
\]
\[
u(x, t) = 0, \quad (x, t) \in \partial\Omega \times [0, T],
\]
\[
u(x, 0) = u_0(x), \quad x \in \Omega.
\]
Here \(0 < \alpha < 1\), \(\Omega \subset R^d (d = 1, 2)\) represents a bounded area; if \(d = 1\), \(\mathcal{L}u(x, t) = -\frac{\partial^2 u}{\partial x^2}(x, t), \Omega = [0, L];\) if \(d = 2\), \(\mathcal{L}u(x, t) = -\frac{\partial^2 u}{\partial x^2}(x, y, t) - \frac{\partial^2 u}{\partial y^2}(x, y, t), \Omega = [0, L] \times [0, L]\). \(\frac{\partial^\alpha}{\partial t^\alpha}u(x, t)\) denotes the Caputo fractional derivative for the time variable \(t\), defined as follows
\[
\frac{\partial^\alpha}{\partial t^\alpha}u(x, t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{\partial u(x, s)}{\partial s} (t-s)^{-\alpha} ds.
\]

As a hot topic in these years, many scholars have studied the numerical solution problem of model (1). Under the assumption that the solution satisfies certain smooth conditions at the initial time, Lin and Xu [11] derived the classic L1 format for the Caputo time fractional derivative term on uniform grid, that is, at each interval, a piecewise linear interpolation function is used to approximate the integrand function. The space derivative term is discretized by the Legendre spectral method, and the convergence order is \(O(\tau^{2-\alpha} + K^{-m})\), \(\tau, K, m\) represent the time step, polynomial degree, and regularity of the exact solution, respectively. The L1 format has been widely used in practice, see [32, 2, 29, 7]. In [8], the L1 - 2 format is proposed on uniform grid for the Caputo fractional derivative term, i.e., the integrable function is approximated by the linear interpolation function in the first time grid layer, and the quadrature interpolation function is used to approximate the integrable function in the other time grid layers, which effectively improve the convergence order to \((3-\alpha)\). The properties of coefficients and the local truncation error are analyzed. Lv[16] provided comprehensive analysis for the L1 - 2 format, including stability analysis and global error estimates, and as continuation of the work a fast evaluation method was introduced and analysed [31]. Alikhanov [1] constructed the L2 - 1 format (also known as the fractional C-N format) for fractional derivative terms on a uniform grid, where \(\sigma = 1 - \frac{\alpha}{2}\) is called the offset parameter. In the integration interval \([t_k, t_{k+1}]\), a linear interpolation function is used to approximate the integrand function, and the quadratic interpolation function is used to approximate the integrand function in the integration interval \([t_{k-1}, t_k], k \geq 1\). When the format is applied to the problem (1), the diffusion term is discretized at the non-integer node \(t_{k+\sigma}\) by the two-node approximation of \(t_k\) and \(t_{k+1}\). Theoretical analysis proved that convergence order is \(O(\tau^{2})\) in the time-direction.

However, it’s an ideal assumption that the solution satisfies certain smoothness at the initial moment. Usually, the solution of the problem (1) has singularity at the initial moment [22, 23], that is, when the time \(t\) approaches 0, the partial derivative \(\frac{\partial u}{\partial t}\) tends to \(\infty\). In this case, the numerical format proposed in the above articles will reduce the convergence order. Considering the singularity of the solution at the initial time, some scholars put forward the discrete problem on the non-uniform...
grid, so as to obtain the ideal convergence order. For example, in [24], the Caputo fractional derivative term is discretized by $L1$ format on a graded meshes, and the diffusion term is approximated by the central difference quotient formula on a uniform grid. The convergence order is $O(N^{-\min(r\alpha,2)} + h^2)$, where $r$ is the parameter of the graded meshes. The Caputo fractional derivative term is discretized by the $L2−1_\sigma$ format on the fitted meshes, and the diffusion term is discretized at the non-integral time node $t_{k+\sigma}$ by the two-nodes linear combination of $t_k$ and $t_{k+1}$. In [3], the spatial derivative is discretized by the spectral method, the time derivative term is discretized by the $L2−1_\sigma$ format on graded meshes, and the diffusion term is discretized at the non-integral time node $t_{k+\sigma}$, with three-points approximation regarding $t_{k-1}$, $t_k$ and $t_{k+1}$, the time convergence order is improved to be $O(N^{-\min(r\alpha,3-\alpha)})$. Some scholars still discretize the time fractional derivatives on a uniform grid, and then modify some initial steps to capture the singularities of the solution near the origin [6, 27, 25], or use some non-polynomial (or singular) basis functions to capture the singularity of the solution [30, 28, 17, 5] and obtain ideal convergence orders, we do not list them one by one.

In this paper, on the one hand, in order to obtain a higher convergence order scheme for the singularity problem at the initial moment, we use the $L1−2$ scheme to discretize the time fractional derivative term on graded meshes, and the diffusion term is discretized on uniform grid by second-order central difference quotient. Due to the difficulty of global error analysis and stability analysis, we only consider the local truncation error estimation. On the other hand, observing that the computational cost is extremely large, then we optimize the discrete format of the model (1) by means of singular value decomposition (SVD) and proper orthogonal decomposition (POD) to obtain a reduced-order finite difference extra-calculation, which effectively reduces the computational cost. For a detailed introduction of POD, please refer to [13, 14, 20, 15].

The rest of this paper is organized as follows. In Section 2, we discretize the problem (1). In Section 3, we estimate the local truncation error for the time derivative term. In Section 4, we optimize the numerical solution format obtained in Section 2, and derive a reduced-order extrapolation algorithm based on singular value decomposition and POD. In Section 5, several numerical examples are used to verify the convergence of the scheme and the effectiveness of the reduced-order scheme. Finally, a conclusion is made in the last section.

2. The discrete problem. In this section, the Caputo fractional derivative term is discretized by the $L1−2$ format on the graded meshes, and the diffusion term is approximated by the classical second-order central difference quotient on a uniform grid, then the discrete scheme of the problem (1) is derived.

We take the positive integers $M$, $N$, and let $h = \frac{L}{M}$ represent the space step, then obtain the set of space nodes $\Omega_h = \{x_m | x_m, m = 0, 1, 2, \cdots, M\}$, for $d = 1$, and $\Omega_h = \{x_m | (x_i, y_j), i, j = 0, 1, 2, \cdots, M, m = j(M + 1) + i\}$, for $d = 2$. We define $\Omega_h = \{x_m | x_m, m = 1, 2, \cdots, M − 1\}$, for $d = 1$, and $\Omega_h = \{x_m | (x_i, y_j), i, j = 1, 2, \cdots, M − 1, m = (j − 1)(M − 1) + i\}$, for $d = 2$. For the time interval, we use the set $\Omega_\tau = \{t_n | t_n = T \cdot \left(\frac{n}{N}\right)^r, n = 0, 1, 2, \cdots, N\}$, where $r \geq 1$. Define $\tau_n = t_n − t_{n−1}$, $n = 1, 2, \cdots, N$. Let $u^n_m$ and $U^n_m$ represent the true and approximate solutions of the function $u(x, t)$ at the grid node $(x_m, t_n)$, respectively.
The linear interpolation of the function \( v(t) \) on the interval \([t_{k-1}, t_k], k \geq 1\) can be expressed as

\[
I_{1,k}v(t) = \frac{t-t_k}{t_{k-1}-t_k} v(t_{k-1}) + \frac{t-t_{k-1}}{t_k-t_{k-1}} v(t_k).
\]

and the quadratic interpolation on the interval \([t_{k-1}, t_k], k \geq 2\) can be expressed as

\[
I_{2,k}v(t) = \frac{(t-t_{k-1})(t-t_k)}{(t_k-2-t_{k-1})(t_{k-2}-t_k)} v(t_{k-2}) + \frac{(t-t_{k-2})(t-t_k)}{(t_{k-1}-t_{k-2})(t_{k-1}-t_k)} v(t_{k-1})
\]

+ \(\frac{(t-t_{k-2})(t-t_{k-1})}{(t_{k-1}-t_{k-2})(t_{k-1}-t_{k-1})} v(t_k)\)

According to Taylor expansion, we have

\[
v(t) - I_{1,k}v(t) = \frac{v''(\eta_k)}{2} (t-t_{k-1})(t-t_k),
\]

\[
v(t) - I_{2,k}v(t) = \frac{v''(\phi_k)}{6} (t-t_{k-2})(t-t_{k-1})(t-t_k),
\]

\[
I_{1,k}v(t) = 1 + \frac{t-t_{k-1}}{t_k-t_{k-1}}[v(t_k) - v(t_{k-1})],
\]

\[
I_{2,k}v(t) = \frac{2t-t_{k-1}+t_k}{(t_k-2-t_{k-1})(t_{k-2}-t_k)} v(t_{k-2})
\]

+ \(\frac{2t-t_{k-2}+t_k}{(t_{k-1}-t_{k-2})(t_{k-1}-t_k)} v(t_{k-1})\)

+ \(\frac{2t-t_{k-2}+t_{k-1}}{(t_{k-1}-t_{k-2})(t_{k-1}-t_{k-1})} v(t_k)\).

We discretize the problem (1) at the node \((x_m, t_n) \in \Omega_h \times \Omega_T\). For the Caputo fractional derivative term, we use the \(L1 - 2\) format on the graded meshes

\[
\frac{C^\alpha 0 D^\alpha_t u(x_m, t_n)}{1} = \frac{1}{\Gamma(1-\alpha)} \sum_{k=1}^{n} \int_{t_{k-1}}^{t_k} (t_n-s)^{-\alpha} \frac{\partial u(x_m, s)}{\partial s} ds
\]

\[
\approx \frac{1}{\Gamma(1-\alpha)} \left[ \int_{t_0}^{t_{k-1}} (t_n-s)^{-\alpha} [I_{1,1}u(x_m, s)]' ds + \sum_{k=2}^{n} \int_{t_{k-1}}^{t_k} (t_n-s)^{-\alpha} [I_{2,k}u(x_m, s)]' ds \right]
\]

\[
= \frac{1}{\Gamma(1-\alpha)} \left[ (d_{n,0} + a_{n,n})u(x_m, t_0) + (d_{n,1} + a_{n,1} + b_{n,1})u(x_m, t_1) \right.
\]

\[
+ \sum_{k=2}^{n} (a_{n,k} + b_{n,k} + c_{n,k})u(x_m, t_k) + \left. (b_{n,n-1} + c_{n,n-1})u(x_m, t_{n-1}) + c_{n,n}u(x_m, t_n) \right]
\]

\[
:= \frac{C^\alpha 0 D^\alpha_N u_m^n},
\]

where

\[
a_{n,k} = \frac{1}{(t_{k-1}-t_{k+1})(t_k-t_{k+2})} \int_{t_{k+1}}^{t_k} (t_n-s)^{-\alpha} (2s-t_{k+1}+t_{k+2}) ds
\]
In order to simplify the discrete scheme, we let

\[
\frac{1}{1 - \alpha (t_k - t_{k+1})(t_k - t_{k+2})} \{(t_n - t_{k+1})^{1-\alpha} (t_{k+1} - t_{k+2}) \}
- (t_n - t_{k+2})^{1-\alpha} (t_{k+2} - t_{k+1}) + \frac{2}{2 - \alpha} \{(t_n - t_{k+1})^{2-\alpha} - (t_n - t_{k+2})^{2-\alpha} \},
\]

\(n = 2, 3, \ldots, N, \quad k = 0, 1, \ldots, n - 2,\)

\[
b_{n,k} = \frac{1}{(t_k - t_{k-1})(t_k - t_{k+1})} \int_{t_k}^{t_{k+1}} (t_n - s)^{-\alpha} (2s - t_{k-1} - t_{k+1}) ds
= \frac{1}{1 - \alpha (t_k - t_{k-1})(t_k - t_{k+1})} \{(t_n - t_{k-1})^{1-\alpha} (t_{k-1} - t_k) \}
- (t_n - t_{k-1})^{1-\alpha} (t_{k-1} - t_k) + \frac{2}{2 - \alpha} \{(t_n - t_{k-1})^{2-\alpha} - (t_n - t_{k-1})^{2-\alpha} \},
\]

\(n = 2, 3, \ldots, N, \quad k = 1, 2, \ldots, n - 1,\)

\[
e_{n,k} = \frac{1}{(t_k - t_{k-2})(t_k - t_{k-1})} \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha} (2s - t_{k-2} - t_{k-1}) ds
= \frac{1}{1 - \alpha (t_k - t_{k-2})(t_k - t_{k-1})} \{(t_n - t_{k-1})^{1-\alpha} (t_{k-1} - t_{k-2}) \}
- (t_n - t_{k-1})^{1-\alpha} (t_{k-1} - t_{k-2}) + \frac{2}{2 - \alpha} \{(t_n - t_{k-1})^{2-\alpha} - (t_n - t_{k-1})^{2-\alpha} \},
\]

\(n = 2, 3, \ldots, N, \quad k = 2, 3, \ldots, n,\)

\[
d_{n,0} = \frac{1}{t_0 - t_1} \int_{t_0}^{t_1} (t_n - s)^{-\alpha} ds
= \frac{1}{1 - \alpha (t_0 - t_1)} [(t_n - t_0)^{1-\alpha} - (t_n - t_1)^{1-\alpha}],
\]

\(n = 1, 2, \ldots, N,\)

\[
d_{n,1} = \frac{1}{t_1 - t_0} \int_{t_0}^{t_1} (t_n - s)^{-\alpha} ds
= \frac{1}{1 - \alpha (t_1 - t_0)} [(t_n - t_0)^{1-\alpha} - (t_n - t_1)^{1-\alpha}],
\]

\(n = 1, 2, \ldots, N.\)

In order to simplify the discrete scheme, we let

\(e_{1,0} = d_{1,0}, \quad e_{1,1} = d_{1,1},\)

\(e_{2,0} = d_{2,0} + a_{2,0}, \quad e_{2,1} = d_{2,1} + b_{2,1}, \quad e_{2,2} = c_{2,2},\)

\(e_{3,0} = d_{3,0} + a_{3,0}, \quad e_{3,1} = d_{3,1} + a_{3,1} + b_{3,1}, \quad e_{3,2} = b_{3,2} + c_{3,2}, \quad e_{3,3} = c_{3,3},\)

for \(4 \leq n \leq N,\) we define

\(e_{n,0} = d_{n,0} + a_{n,0}, \quad e_{n,1} = d_{n,1} + a_{n,1} + b_{n,1},\)

\(e_{n,n-1} = b_{n,n-1} + c_{n,n-1}, \quad e_{n,n} = c_{n,n},\)

\(e_{n,k} = a_{n,k} + b_{n,k} + c_{n,k}, \quad \text{for} \quad 2 \leq k \leq n - 2.\)

Thus

\[
C D_N^{\alpha} u_m^n = \frac{1}{\Gamma(1 - \alpha)} \sum_{k=0}^{n} e_{n,k} u_m^k.
\]

(5)
The diffusion term is approximated by the classical second-order central difference quotient, we obtain
\[
\mathcal{L} u(x_m, t_n) \approx -\frac{u_{m+1}^n - 2u_m^n + u_{m-1}^n}{h^2}, \quad d = 1, \\
\mathcal{L} u(x_m, t_n) \approx -\frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{h^2} - \frac{u_{i,j+1}^n + u_{i,j-1}^n}{h^2}, \quad d = 2.
\]

Combining the formulas (5) and (6), we get the finite difference scheme (FD) of the problem (1)
\[
C D_N^\alpha u_m^n - \delta_x^2 u_m^n = f_m^n, \quad 1 \leq m \leq \hat{M}, \quad 1 \leq n \leq N, \\
u(x_m, t_n) = 0, \quad x_m \in \Omega_h / \bar{\Omega}_h, \quad 0 \leq n \leq N, \\
u_0^n = u_0(x_m), \quad 0 \leq m \leq \hat{M}.
\]
where $\hat{M} = M - 1$, $\hat{M} = M$, for $d = 1$, and $\hat{M} = (M - 1)^2$, $\hat{M} = M(M + 2)$, for $d = 2$.

3. **Truncation error estimates.** In this section, we give the truncation error estimates for the Caputo derivative term.

According to the Taylor expansion formula, it is easy to get
\[
\tau_{k+1} = t_{k+1} - t_k = T(k + 1) - T(k) \leq CTN^{-r}k^{r-1}, \\
k = 0, 1, 2, \ldots, \hat{N} - 1.
\]

**Theorem 3.1.** Let $u(\cdot, t) \in C[0, T] \cap C^3(0, T)$. Assume that $|\frac{\partial^l u(\cdot, t)}{\partial t^l}| \leq C(1 + t^{\alpha-1})$ for $l = 0, 1, 2, 3$, there exists a constant $C$ such that for all $(x_m, t_n) \in \bar{\Omega}_h \times \Omega_T$, we have
\[
|C D_N^\alpha u(x_m, t_n) - C _0 D_t^\alpha u(x_m, t_n)| \leq C N^{-\min\{\rho, 3-\alpha\}}.
\]

**Proof.** For $n = 1, 2, \ldots, N$, $m = 1, 2, \ldots, \hat{M}$, we have
\[
C D_N^\alpha u_m^n - C _0 D_t^\alpha u(x_m, t_n) = \sum_{k=1}^n T_{nk},
\]
Here, for $k = 1$
\[
T_{n1} := \frac{1}{\Gamma(1-\alpha)} \int_{t_0}^{t_1} (t_n - s)^{-\alpha} \left[ \frac{\partial I_{1,1} u}{\partial s} (x_m, s) - \frac{\partial u}{\partial s} (x_m, s) \right] ds,
\]
for $k = 2, 3, \ldots, N$
\[
T_{nk} := \frac{1}{\Gamma(1-\alpha)} \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha} \left[ \frac{\partial I_{2,k} u}{\partial s} (x_m, s) - \frac{\partial u}{\partial s} (x_m, s) \right] ds.
\]
We have [24]
\[
|T_{n1}| \leq C n^{-\alpha}, \quad n = 1, 2, \ldots, N.
\]
For \( n = 2, 3, \ldots, N, \ k = 2, 3, \ldots, \left[ \frac{n-1}{2} \right] \)

\[
T_{nk} = \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha} \left[ \frac{\partial I_{2,k} u}{\partial s}(x_m, s) - \frac{\partial u}{\partial s}(x_m, s) \right] ds \\
\leq \alpha \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha-1} u(t_{k-1}, x_m, s) (s - t_{k-2})(s - t_{k-1})(s - t_k) ds \\
\leq C t_k^{\alpha-3} x_k^{3} \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha-1} ds \\
\leq C t_k^{\alpha-3} x_k^{4} (t_n - t_k)^{-\alpha-1},
\]

where

\[
(t_n - t_k)^{-\alpha-1} = \left( \frac{T^{n^r - k^r}}{N^{r^3}} \right)^{-\alpha-1} = T^{-\alpha-1} \left( \frac{N^r}{n^r - k^r} \right)^{\alpha+1} \\
\leq T^{-\alpha-1} \left( \frac{N^r}{n^r - \left( \frac{n-1}{2} \right)^r} \right)^{\alpha+1} \leq CT^{-\alpha-1}(N/n)^{r(1+\alpha)},
\]

Take the formula (12) into (11), we obtain

\[
T_{nk} \leq C \left( T \left( \frac{k-1}{N} \right)^{r^3} T^{4} N^{-4r} (k-1)^{4(r-1)} T^{-\alpha-1}(N/n)^{r(1+\alpha)} \right) \\
\leq C N r^{(3-\alpha)-4r+(1+\alpha)n^{-1}(1+\alpha)} k^{-r(\alpha-3)+4(r-1)} \\
\leq C n^{-r(1+\alpha)} (k-1)^{r(1+\alpha)-4},
\]

Hence

\[
\sum_{k=1}^{\left[ \frac{n-1}{2} \right]+1} T_{nk} \leq C n^{-r(1+\alpha)} \sum_{k=1}^{\left[ \frac{n-1}{2} \right]} (k-1)^{r(1+\alpha)-4} \\
\leq \begin{cases} 
C n^{-r(1+\alpha)}, & r(1+\alpha) < 3, \\
C n^{-3 \ln(n)}, & r(1+\alpha) = 3, \\
C n^{-3}, & r(1+\alpha) > 3.
\end{cases}
\]

and

\[
\sum_{k=\left[ \frac{n-1}{2} \right]+1}^{n-1} T_{nk} = \sum_{k=\left[ \frac{n-1}{2} \right]+1}^{n-1} t_k^{\alpha-3} x_k^{3} \int_{t_{k-1}}^{t_k} (t_n - s)^{-\alpha-1} ds \\
\leq C t_k^{\alpha-3} x_k^{4} \sum_{k=\left[ \frac{n-1}{2} \right]+1}^{n-1} \frac{1}{\alpha} \left[ (t_n - t_k)^{-\alpha} - (t_n - t_{k-1})^{-\alpha} \right] \\
\leq C t_k^{\alpha-3} x_k^{4} \left[ (t_n - t_{n-1})^{-\alpha} - (t_n - t_{\left[ \frac{n-1}{2} \right]})^{-\alpha} \right] \leq C t_k^{\alpha-3} x_k^{3} r_n^{-\alpha} \\
\leq C t_k^{\alpha-3} (T N^{-r^3}(k-1)^{r-1})^{3} r_n^{-\alpha} \leq C t_k^{\alpha-3} T^{3} N^{-3r n^{3(r-1)}} [T N^{-r} n^{r-1}]^{-\alpha} \\
\leq C T^{3} N^{-3r} n^{3(r-1)} T^{-\alpha} (n/N)^{r(\alpha-3)} T^{-\alpha} N^{r} n^{-\alpha(r-1)} \leq C n^{-(3-\alpha)}.
\]
Finally, we consider $T_{nn}$ for $n > 1$

$$T_{nn} = \frac{1}{\Gamma(1-\alpha)} \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} \left[ \frac{\partial I_2 u}{\partial s}(x_m, s) - \frac{\partial u}{\partial s}(x_m, s) \right] ds$$

$$\leq C t_{n-1}^{\alpha-3} t_n^{-2} \int_{t_{n-1}}^{t_n} (t_n - s)^{-\alpha} ds$$

$$\leq C t_{n-1}^{\alpha-3} t_n^{-2} (t_n - t_{n-1})^{1-\alpha}$$

$$\leq C \left[ TN^{-r}(n-1)^r \right]^{\alpha-3} \left[ TN^{-r}(n-2)^r \right] \left[ TN^{-r}(n-1)^r \right]^{1-\alpha}$$

$$\leq C N^{-(3-\alpha)}.$$  \hspace{1cm} (16)

Combining formulas (10), (14), (15) and (16), we obtain

$$|C D_N^\alpha u(x_m, t_n) - C D_0^\alpha u(x_m, t_n)| \leq C n^{-\min\{r \alpha, 3-\alpha\}}.$$ \hspace{1cm} (17)

The above truncation error estimate suggests a choice of the grading constant $r = (3 - \alpha)/\alpha$, which means the numerical scheme possess optimal order convergence, i.e., $3 - \alpha$. In the section of numerical experiments, we will find the impact of the grading constant on the accuracy.

4. Reduced-order FD extrapolation algorithm. Considering that the calculation cost is relatively large, in this section we construct a reduced order finite difference extrapolation algorithm for the problem (1) by means of the singular value decomposition (SVD) and POD technology, which effectively reduces the calculation cost.

We first solve the first $L$ time layer solutions $\{U_m^i\}_{i=1}^L$ according to the finite difference format (7), which is called a snapshot. Then we store the snapshot in the matrix $A$

$$A = \begin{pmatrix}
U_1^1 & U_2^1 & \cdots & U_L^1 \\
U_1^2 & U_2^2 & \cdots & U_L^2 \\
\vdots & \vdots & \ddots & \vdots \\
U_1^M & U_2^M & \cdots & U_L^M
\end{pmatrix}.$$

By performing singular value decomposition for the matrix $A$, we get

$$A = U \begin{pmatrix}
S & 0 \\
0 & 0
\end{pmatrix} V^T,$$

where $S = \text{diag}\{\sigma_1, \sigma_2, \ldots, \sigma_l\} \in R^{l \times l}$ is a diagonal matrix corresponding to the matrix $A$, and $\sigma_i$ ($i = 1, 2, \ldots, l$) is the positive singular value of the matrix $A$. Matrix $U = (\phi_1, \phi_2, \ldots, \phi_M) \in R^{M \times M}$, $V = (\rho_1, \rho_2, \ldots, \rho_L) \in R^{L \times L}$ are the eigenvectors of $AA^T$ and $A^TA$, respectively. The columns of these eigenvector matrices are organized such that corresponding to the singular values $\sigma_i$ are comprised in $S$ with a non-decreasing order. And the singular values of the decomposition are connected to the eigenvalues of the matrices $AA^T$ and $A^TA$ in a manner such that $\lambda_i = \sigma_i^2$, ($i = 1, 2, \ldots, l$). Since the number of mesh points is far larger than that of transient moment points, i.e., $L \ll M$, that is, the order $M$ for matrix $AA^T$ is far larger than the order $L$ for matrix $A^TA$. However, their null eigenvalues are identical, therefore, we may first solve the eigenequation corresponding to matrix
\[ A^T A \] to find the eigenvalues \( \lambda_i \), \( (i = 1, 2, \ldots, l) \) and corresponding eigenvectors \( V = (\rho_1, \rho_2, \ldots, \rho_L) \), and then by the relationship
\[
\phi_j = \frac{1}{\sigma_j} A \rho_j, \quad j = 1, 2, \ldots, l,
\]
we may obtain \( l (l < L) \) eigenvectors corresponding to the non-null eigenvalues for matrix \( AA^T \).

Let \( U^k = (U^k_1, U^k_2, \ldots, U^k_M)^T \) \((k = 1, 2, \ldots, L)\) represent \( L \) column vectors of matrix \( A \), and define the projection
\[
P_{\tilde{M}}(U^k) = \sum_{j=1}^\tilde{M} (\phi_j, U^k) \phi_j,
\]
where \( 1 \leq \tilde{M} \leq L \), and \((\cdot, \cdot)\) represents the inner product of the vector. Then, there hold that
\[ ||U^k - P_{\tilde{M}}(U^k)||_2 \leq \sigma_k. \]
Thus, \( \{ \phi_j \}_{j=1}^\tilde{M} \) is an optimal basis. According to the nature of the eigenvector, it is well known that \( \Phi = (\phi_1, \phi_2, \ldots, \phi_{\tilde{M}}) \) is an orthogonal matrix, and \( \{ \phi_j \}_{j=1}^\tilde{M} \) is a set of orthogonal bases, which is called a set of POD bases.

Next, we construct a reduced order finite difference extrapolation scheme for the problem (1).

Rewrite the discrete format (7) of the problem (1) into a matrix form
\[
\begin{align*}
&e_{n,n} U^n - B U^n = F^n - \sum_{k=0}^{n-1} e_{n,k} U^k, \quad n = 1, 2, \ldots, N,
\end{align*}
\]
where \( U^n = (U^n_1, U^n_2, \ldots, U^n_M)^T \), \( F^n = (f^n_1 + \frac{1}{h^2} U^n_0, f^n_2, f^n_3, \ldots, f^n_{M-1}, f^n_M + \frac{1}{h^2} U^n_M)^T \), and
\[
B = \frac{1}{h^2} \left( \begin{array}{cccc}
-2 & 1 & 0 & \cdots & 0 & 0 \\
1 & -2 & 1 & \cdots & 0 & 0 \\
0 & 1 & -2 & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & -2 & 1 \\
0 & 0 & 0 & \cdots & 1 & -2
\end{array} \right).
\]

\( U^n \) in the above formula are approximately replaced with \( U^n = \Phi \beta^n_M \), then we get
\[
\begin{align*}
e_{n,n} \beta^n_M - \Phi^T B \Phi \beta^n_M &= \Phi^T F^n - \sum_{k=0}^{n-1} e_{n,k} \beta^n_M, \quad n = L + 1, L + 2, \ldots, N. \quad (18)
\end{align*}
\]
For \( n = 0, 1, 2, \ldots, L \), we let
\[
\beta^n_M = \Phi^T U^n. \quad (19)
\]
We solve the equations (18) and (19) to get \( \beta^n_M \) \((n = 0, 1, 2, \ldots, N)\), then we solve
\[
U^{*n} = \Phi \beta^n_M, \quad n = 1, 2, \ldots, N. \quad (20)
\]
We call the formulas (18)-(20) as the reduced order finite difference extrapolation scheme (abbreviated as RFD) based on SVD and POD.
Remark 1. the finite difference format (FD) contains $\hat{M}$ unknowns at each time level, while the reduced order extrapolated finite difference format contains only $\tilde{M}$ ($\tilde{M} \ll \hat{M}$) unknowns, which greatly reduce the amount of calculation. In the numerical simulation, we choose $L = 20$, and $\tilde{M} = 5$.

5. Numerical experiments. In this section we use several numerical examples to verify the previous theoretical analysis. It is well known that for the spatial second-order central difference scheme, the convergence order is $O(M^{-2})$, so we do not verify it. The following examples only verify the time convergence rates. We use Matlab R2014a to calculate numerical examples on a computer with a memory 4G processor 2.60Ghz.

Example 1. Consider

$$\frac{\partial}{\partial \alpha} D^\alpha_t u(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) = f(x,t), \quad (x,t) \in (0,\pi) \times (0,1],$$

$$u(0,t) = u(\pi,t) = 0, \quad t \in [0,1],$$

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

Suppose the exact solution is $u(x,t) = \sin(x)(1 + t^{\alpha} + t^{2\alpha} + t^{1+\alpha})$. It is obvious that the $u(x,t)$ is singular at the initial moment, and the corresponding $f(x,t)$ can be solved according to the equation (1).

We fix the number of space and time splits $M = N = 500$, let $\alpha = 0.6$, $r = \frac{3-\alpha}{\alpha}$. The numerical format (7) and reduced-order format (18)-(20) are used to calculate Example 1. Figure 1 plots the absolute error of the numerical solution obtained by the two formats. From Figure 1, We find that the magnitude of the difference between the calculation results of the two numerical schemes is $10^{-13}$, which means that the difference is very small.

![Figure 1](image.png)

**Figure 1.** The absolute error of solutions obtained by the two formats with $M = 500$, $N = 500$, $\alpha = 0.6$, $r = \frac{3-\alpha}{\alpha}$ for Example 1.

Fix $M = 10000$, take $\alpha = 0.6$, and $r = \frac{3-\alpha}{\alpha}$. We present the CPU time consumed by the finite difference format (7) and reduced order format (18)-(20) for Example1 with different $N = 2^5, 2^6, \ldots, 2^{10}$ in Table 1. The data shows that the calculation speed of the reduced-order algorithm is much better than traditional algorithm, for
example, when \( N = 2^{10} \), the finite difference scheme needs 7446 seconds to solve the problem, while the reduced-order scheme only needs 13.6 seconds, which verifies the effectiveness of the reduced-order algorithm.

**Table 1.** The CPU time consumed by FD and RFD with different mesh sizes with \( M = 10000, \alpha = 0.6, r = \frac{3-\alpha}{\alpha} \) for Example 1

| \( N \) | \( 2^5 \) | \( 2^6 \) | \( 2^7 \) | \( 2^8 \) | \( 2^9 \) | \( 2^{10} \) |
|---|---|---|---|---|---|---|
| FD (CPU) | 235.9671 | 460.4681 | 922.5431 | 1821.1869 | 3661.3747 | 7446.4113 |
| RFD (CPU) | 0.0624 | 0.1092 | 0.2496 | 0.9360 | 3.2448 | 13.5721 |

Due to the reduced-order algorithm can effectively reduce the calculation time, and the calculation accuracy is slightly different from the traditional algorithm, next we fix the number of spatial divisions \( M = 10000 \), and the step size of the spatial division is sufficiently small, so that the spatial discretization has a significant impact on the calculation. The effect of errors can be ignored to verify the time convergence order. In Table 2-5, we select \( r = 1, r = \frac{1}{\alpha}, r = \frac{2}{\alpha}, r = \frac{3-\alpha}{\alpha} \) implement the format (18)-(20) for different \( \alpha \) values, respectively. The tables list the maximum error and its convergence order. The datas show that the convergence order is \( \alpha, 1, 2, \) and \( 3-\alpha \) respectively for different values of \( r \).

**Example 2.** Consider the following homogeneous time fractional differential equation

\[
\begin{align*}
\frac{\partial^\alpha}{\partial t^\alpha}u(x,t) - \frac{\partial^2 u}{\partial x^2}(x,t) &= 0, \quad (x,t) \in (0, \pi) \times [0,1], \\
u(0,t) &= u(\pi,t) = 0, \quad t \in [0,1], \\
u(x,0) &= u_0(x), \quad x \in [0,\pi].
\end{align*}
\]

Here \( u_0(x) = \sin(x) \), the exact solution has the form

\[
u(x,t) = E_{\alpha,1}(-t^\alpha)\sin(x),
\]

where \( E_{\alpha,1}(z) \) represents the Mittag-Leffler function defined by[21]

\[
E_{\alpha,1}(z) = \sum_{k=0}^{\infty} \frac{z^k}{\Gamma(\alpha k + 1)}, \quad \alpha > 0.
\]
As same to Example 1, we first verify the effectiveness of the reduced-order algorithm, that is, the calculation results are not much different from the traditional finite difference method (see Figure 2), but the calculation cost is significantly reduced (see Table 6). Then, we select $r = \frac{3-\alpha}{\alpha}$. Example 2 is calculated by the reduced order format (18)-(20). The maximum error and convergence rates are listed with different $\alpha$ in Table 7.

| $N$ | $\alpha = 0.4$ | $\alpha = 0.6$ | $\alpha = 0.8$ |
|-----|----------------|----------------|----------------|
|     | $\text{Max}\_\text{err}$ | $\text{rate}$ | $\text{Max}\_\text{err}$ | $\text{rate}$ | $\text{Max}\_\text{err}$ | $\text{rate}$ |
| $2^5$ | 1.1780e-2 | 6.1861e-3 | 3.7802e-3 | | | |
| $2^6$ | 6.0418e-3 | 0.9633 | 3.1668e-3 | 0.9814 | 2.0794e-3 | 0.8623 | |
| $2^7$ | 3.0601e-3 | 1.6014e-3 | 1.0853e-3 | 0.9837 | 5.5376e-4 | 0.9708 | |
| $2^8$ | 1.5400e-3 | 0.9906 | 8.0511e-4 | 0.9921 | | |
| $2^9$ | 7.7251e-4 | 0.9553 | 2.0364e-4 | 0.9961 | 1.4039e-4 | 0.9867 | |
| $2^{10}$ | 3.8687e-4 | 0.9977 | 2.0290e-4 | 0.9981 | 1.4039e-4 | 0.9931 | |

| $N$ | $\alpha = 0.4$ | $\alpha = 0.6$ | $\alpha = 0.8$ |
|-----|----------------|----------------|----------------|
|     | $\text{Max}\_\text{err}$ | $\text{rate}$ | $\text{Max}\_\text{err}$ | $\text{rate}$ | $\text{Max}\_\text{err}$ | $\text{rate}$ |
| $2^5$ | 9.3428e-3 | 2.5627e-3 | 9.3566e-4 | | | |
| $2^6$ | 2.3654e-3 | 1.9818 | 6.5273e-4 | 1.9731 | 2.5124e-4 | 1.8969 | |
| $2^7$ | 5.9358e-4 | 1.9946 | 1.6398e-4 | 1.9929 | 6.4353e-5 | 1.9632 | |
| $2^8$ | 1.4888e-4 | 1.9953 | 4.1050e-5 | 1.9981 | 1.6234e-5 | 1.9880 | |
| $2^9$ | 3.7600e-5 | 1.9854 | 1.0266e-5 | 1.9995 | 4.0700e-6 | 1.9967 | |
| $2^{10}$ | 9.7725e-6 | 1.9439 | 2.5667e-6 | 1.9999 | 1.0181e-6 | 1.9992 | |

| $N$ | $\alpha = 0.4$ | $\alpha = 0.6$ | $\alpha = 0.8$ |
|-----|----------------|----------------|----------------|
|     | $\text{Max}\_\text{err}$ | $\text{rate}$ | $\text{Max}\_\text{err}$ | $\text{rate}$ | $\text{Max}\_\text{err}$ | $\text{rate}$ |
| $2^5$ | 8.1920e-3 | 1.9021e-3 | 8.5782e-4 | | | |
| $2^6$ | 1.3564e-3 | 2.5944 | 3.6421e-4 | 2.3848 | 2.0153e-4 | 2.0897 | |
| $2^7$ | 2.2375e-4 | 2.5999 | 6.9143e-5 | 2.3971 | 4.5568e-5 | 2.1449 | |
| $2^8$ | 3.6784e-5 | 2.6048 | 1.3105e-5 | 2.3995 | 1.0109e-5 | 2.1724 | |
| $2^9$ | 5.9417e-6 | 2.6301 | 2.4831e-6 | 2.3999 | 2.2188e-6 | 2.1877 | |
| $2^{10}$ | 8.5465e-7 | 2.7975 | 4.7049e-7 | 2.3999 | 4.8440e-7 | 2.1955 | |

Table 3. $M = 10000$, $r = 1/\alpha$, the $L^\infty$ error and convergence rates for Example 1 on graded meshes

Table 4. $M = 10000$, $r = 2/\alpha$, the $L^\infty$ error and convergence rates for Example 1 on graded meshes

Table 5. $M = 10000$, $r = (3-\alpha)/\alpha$, the $L^\infty$ error and convergence rates for Example 1 on graded meshes
Next, we use the L1 format [24] to discretize the time fractional derivative on the graded mesh, and use the central difference quotient to approximate the spatial second derivative term on the uniform grid to solve Example 2. Fix $M = 10000$, select the graded mesh parameter $r = \frac{3-\alpha}{\alpha}$ to get the optimal convergence order of the format. List the calculation results in the Table 8. Compared with our proposed scheme, our calculation format has obtained a higher convergence order and more accurate numerical results. In addition, we follow the $L^2 - l_\sigma$ in the article [4] to discretize the fractional derivative term on the graded mesh, and select
the graded mesh parameter as \( r = \frac{2}{\alpha} \), the spatial derivative term is discretized by the finite difference method, and the diffusion term is discretized at the non-integral time node \( t_{k+\sigma} \) by the two-nodes linear combination of \( t_k \) and \( t_{k+1} \). The results obtained from the calculation of Example 2 are listed in the Table 9. The data shows that our calculation results are similar to the results obtained by \( L^2 - 1_{\sigma} \), but higher convergence orders can be obtained.

### Table 8. \( L^1 \) format, \( M = 10000 \), \( r = (2 - \alpha)/\alpha \) the \( L^\infty \) error and convergence rates for Example 2 on graded meshes

| \( N \) | \( \alpha = 0.4 \) | \( \alpha = 0.6 \) | \( \alpha = 0.8 \) |
|---|---|---|---|
| \( \text{Max} \text{err} \) | rate | \( \text{Max} \text{err} \) | rate | \( \text{Max} \text{err} \) | rate |
| 2\(^5\) | 2.0114e-3 | 3.4273e-3 | 5.1455e-3 | 2.3995e-3 | 1.1006 |
| 2\(^6\) | 7.1710e-4 | 1.4880 | 1.3922e-3 | 1.2997 | 2.3995e-3 | 1.1006 |
| 2\(^7\) | 2.4927e-4 | 1.5245 | 5.5229e-4 | 1.3339 | 1.1033e-3 | 1.1209 |
| 2\(^8\) | 8.5555e-5 | 1.5428 | 1.1033e-3 | 1.1209 |
| 2\(^9\) | 9.8709e-6 | 1.5619 | 3.2075e-5 | 1.3798 | 1.0172e-4 | 1.1562 |
| 2\(^10\) | 9.8709e-6 | 1.5619 | 3.2075e-5 | 1.3798 | 1.0172e-4 | 1.1562 |

### Table 9. \( L^2 - 1_{\sigma} \) format, \( M = 10000 \), \( r = 2/\alpha \) the \( L^\infty \) error and convergence rates for Example 2 on graded meshes

| \( N \) | \( \alpha = 0.4 \) | \( \alpha = 0.6 \) | \( \alpha = 0.8 \) |
|---|---|---|---|
| \( \text{Max} \text{err} \) | rate | \( \text{Max} \text{err} \) | rate | \( \text{Max} \text{err} \) | rate |
| 2\(^5\) | 2.3712e-4 | 1.9954e-4 | 1.2946e-4 | 1.9291 | 3.5332e-5 | 1.8735 |
| 2\(^6\) | 6.1882e-5 | 1.9380 | 5.2396e-5 | 1.9291 | 3.5332e-5 | 1.8735 |
| 2\(^7\) | 1.5706e-5 | 1.9782 | 1.3317e-5 | 1.9762 | 9.2562e-6 | 1.9325 |
| 2\(^8\) | 3.9412e-6 | 1.9946 | 3.3471e-6 | 1.9923 | 2.3634e-6 | 1.9695 |
| 2\(^9\) | 9.8622e-7 | 1.9986 | 8.3790e-7 | 1.9981 | 5.9543e-7 | 1.9889 |
| 2\(^10\) | 2.4661e-7 | 1.9997 | 2.0955e-7 | 1.9995 | 1.4920e-7 | 1.9967 |

### Example 3. Consider

\[
\frac{\partial}{\partial t} D^\alpha_t u(x, y, t) - \frac{\partial^2 u}{\partial x^2}(x, y, t) - \frac{\partial^2 u}{\partial y^2}(x, y, t) = f(x, y, t), \quad (x, y) \in \Omega, \quad t \in (0, 1],
\]

\[
u(x, y, t) = 0, \quad (x, y) \in \partial\Omega, \quad t \in [0, 1],
\]

\[
u(x, y, 0) = u_0(x, y), \quad (x, y) \in \Omega.
\]

where \( \Omega = [0, \pi] \times [0, \pi] \). We assume that the exact solution of the problem is

\[
u(x, y, t) = \left(1 + \frac{\sin(x) + \sin(y)}{2^{1+\alpha}}\right) \sin(x) \sin(y).
\]

According to the equation (1), the corresponding \( f(x, y, t) \) and \( u_0(x, y) \) can be defined.

Similar to one-dimensional problem, we first select the number of space and time splits \( M = N = 200 \), let \( \alpha = 0.6 \), \( r = \frac{3-\alpha}{\alpha} \). The numerical format (7) and reduced-order format (18)-(20) are used to calculate Example 3. Figure 3 plot the absolute
errors of the numerical solution obtained by the two schemes at $t = 0.2, 0.4, 0.6, 0.8$, respectively. We observe that the difference between the calculation results is very small.

Then, we select $M = 100$, take $\alpha = 0.6$, and $r = \frac{3-\alpha}{\alpha}$. We present the CPU time consumed by the finite difference format (7) and the reduced order format (18)-(20) for Example 3 with different $N = 2^5, 2^6, \ldots, 2^8$ in Table 10. The data shows that the calculation speed of the reduced-order algorithm is much better than traditional algorithm, which verifies the effectiveness of the reduced-order algorithm.

Finally, we select $M = 500$ and $r = \frac{3-\alpha}{\alpha}$. Example 3 is calculated by the reduced order format (18)-(20) for different $\alpha$ values. The maximum error and convergence rates are listed in Table 11, and we get the the convergence order of $3 - \alpha$.

![Figure 3. The absolute error of solutions obtained by FD and RFD $M = 200$, $N = 200$, $\alpha = 0.6$, $r = \frac{3-\alpha}{\alpha}$ for Example 3.](image)

| $N$     | $2^5$       | $2^6$       | $2^7$       | $2^8$       |
|---------|-------------|-------------|-------------|-------------|
| FD (CPU)| 2082.2857   | 4202.6201   | 8426.9712   | 16835.3939  |
| RFD(CPU)| 0.0468      | 0.0624      | 0.1872      | 0.8112      |

Table 10. The CPU time consumed by FD and RFD with different mesh sizes with $M = 100$, $\alpha = 0.6$, $r = \frac{3-\alpha}{\alpha}$ for Example 3.
Table 11. \( M = 500, r = (3-\alpha)/\alpha \), the \( L^\infty \) error and convergence rates for Example 3 on graded meshes

| \( N \) | \( \alpha = 0.4 \) | \( \alpha = 0.6 \) | \( \alpha = 0.8 \) |
|---|---|---|---|
| | \( \text{Max err} \) | rate | \( \text{Max err} \) | rate | \( \text{Max err} \) | rate |
| 2\(^5\) | 9.1847e-3 | 2.1075e-3 | 9.0565e-4 | | |
| 2\(^6\) | 1.5275e-3 | 2.5880 | 2.3730 | 2.1443e-4 | 2.0785 | | |
| 2\(^7\) | 2.5217e-4 | 2.5987 | 7.7358e-5 | 2.3949 | 4.8728e-5 | 2.1377 | | |
| 2\(^8\) | 4.1476e-5 | 2.6040 | 1.4667e-5 | 2.3990 | 1.0858e-5 | 2.1660 | | |

6. Conclusion. We consider the time fractional diffusion equation with Caputo fractional derivative, which solution is singular at the initial moment. The fractional derivative term is discretized by the \( L^1-2 \) format on the graded meshes then a high order format is constructed. The local truncation error is estimated. Numerical tests show that this format possesses convergence rate of \( 3-\alpha \), which reaches the best order among all of the existing finite difference schemes. In addition, considering that the computation cost of finite difference format is relatively large, we optimize the direct discrete scheme by means of SVD and POD, and get the reduced-order finite difference extrapolation algorithm, which greatly reduces the amount of unknowns of each time layer. Numerical experiments verify the convergence order of the numerical simulation scheme and the effectiveness of the reduction algorithm. The difference between the calculation results of the reduced order extrapolated finite difference format and the finite difference format is very small, and the calculation cost is significantly reduced. Regrettably, at present, we can not provide a proof for the stability of the numerical scheme. This is because of the complexity of the coefficients in the numerical scheme and we will investigate the stability and convergence analysis for the scheme in the future.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grants 91630207, 11971272.

REFERENCES

[1] A. A. Alikhanov, A new difference scheme for the time fractional diffusion equation, *J. Comput. Phys.*, **280** (2015), 424–438.
[2] H. Brunner, L. Ling and M. Yamamoto, Numerical simulations of 2D fractional subdiffusion problems, *J. Comput. Phys.*, **229** (2010), 6613–6622.
[3] H. Chen and M. Stynes, A high order method on graded meshes for a time-fractional diffusion problem, *Finite Difference Methods*, 15–27, Lecture Notes in Comput. Sci., 11386, Springer, Cham, 2019.
[4] H. Chen and M. Stynes, Error analysis of a second-order method on fitted meshes for a time-fractional diffusion problem, *J. Sci. Comput.*, **79** (2019), 624–647.
[5] N. J. Ford, M. L. Morgado and M. Rebelo, Nonpolynomial collocation approximation of solutions to fractional differential equations, *Fract. Calc. Appl. Anal.*, **16** (2013), 874–891.
[6] N. J. Ford and Y. Yan, An approach to construct higher order time discretisation schemes for time fractional partial differential equations with nonsmooth data, *Fract. Calc. Appl. Anal.*, **20** (2017), 1076–1105.
[7] G.-H. Gao and Z.-Z. Sun, A compact finite difference scheme for the fractional sub-diffusion equations, *J. Comput. Phys.*, **230** (2011), 586–595.
[8] G.-H. Gao, Z.-Z. Sun and H.-W. Zhang, A new fractional numerical differentiation formula to approximate the Caputo fractional derivative and its applications, *J. Comput. Phys.*, 259 (2014), 33–50.

[9] M. Giona, S. Cerbelli and H. E. Roman, Fractional diffusion equation and relaxation in complex viscoelastic materials, *Physica A: Statistical Mechanics and its Applications*, 191 (1992), 449–453.

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

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

[12] F. Liu, S. Shen, V. Anh and I. Turner, Analysis of a discrete non-Markovian random walk approximation for the time fractional diffusion equation, *ANZIAM J.*, 46 (2004), 488–504.

[13] Z. Luo, J. Chen, J. Zhu, R. Wang and I. M. Navon, An optimizing reduced order FDS for the tropical Pacific Ocean reduced gravity model, *Internat. J. Numer. Methods Fluids*, 55 (2007), 143–161.

[14] Z. Luo, H. Li, P. Sun and J. Gao, A reduced-order finite difference extrapolation algorithm based on POD technique for the non-stationary navier–stokes equations, *Appl. Math. Model.*, 37 (2013), 5464–5473.

[15] Z. Luo, F. Teng and H. Xia, A reduced-order extrapolated Crank–Nicolson finite spectral element method based on POD for the 2D non-stationary Boussinesq equations, *J. Math. Anal. Appl.*, 471 (2019), 564–583.

[16] C. Lv and C. Xu, Error analysis of a high order method for time-fractional diffusion equations, *SIAM J. Sci. Comput.*, 38 (2016), A2699–A2724.

[17] Z. Mao and J. Shen, Efficient spectral–Galerkin methods for fractional partial differential equations with variable coefficients, *J. Comput. Phys.*, 307 (2016), 243–261.

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

[19] R. R. Nigmatullin, The realization of the generalized transfer equation in a medium with fractal geometry, *Physica Status Solidi (B)*, 133 (1986), 425–430.

[20] K. B. Oldham and J. Spanier, *The Fractional Calculus*, Mathematics in Science and Engineering, Vol. 111. Academic Press [A subsidiary of Harcourt Brace Jovanovich, Publishers], New York-London, 1974.

[21] I. Podlubny, *Fractional Differential Equations*, in Mathematics in Science and Engineering, 1999.

[22] K. Sakamoto and M. Yamamoto, Initial value/boundary value problems for fractional diffusion-wave equations and applications to some inverse problems, *J. Math. Anal. Appl.*, 382 (2011), 426–447.

[23] M. Stynes, Too much regularity may force too much uniqueness, *Pract. Calc. Appl. Anal.*, 19 (2016), 1554–1562.

[24] M. Stynes, E. O’Riordan and J. L. Gracia, Error analysis of a finite difference method on graded meshes for a time-fractional diffusion equation, *SIAM J. Numer. Anal.*, 55 (2017), 1057–1079.

[25] Y. Xing and Y. Yan, A higher order numerical method for time fractional partial differential equations with nonsmooth data, *J. Comput. Phys.*, 357 (2018), 305–323.

[26] B. Xu and X. Zhang, A reduced fourth-order compact difference scheme based on a proper orthogonal decomposition technique for parabolic equations, *Bound. Value Probl.*, 2019 (2019), 130.

[27] Y. Yang, Y. Yan and N. J. Ford, Some time stepping methods for fractional diffusion problems with nonsmooth data, *Comput. Methods Appl. Math.*, 18 (2018), 129–146.

[28] F. Zeng, Z. Zhang and G. E. Karniadakis, A generalized spectral collocation method with tunable accuracy for variable-order fractional differential equations, *SIAM J. Sci. Comput.*, 37 (2015), A2710–A2732.

[29] Y.-N. Zhang and Z.-Z. Sun, Alternating direction implicit schemes for the two-dimensional fractional sub-diffusion equation, *J. Comput. Phys.*, 230 (2011), 8713–8728.

[30] Z. Zhang, F. Zeng and G. E. Karniadakis, Optimal error estimates of spectral Petrov–Galerkin and collocation methods for initial value problems of fractional differential equations, *SIAM J. Numer. Anal.*, 53 (2015), 2074–2096.

[31] H. Zhu and C. Xu, A fast high order method for the time-fractional diffusion equation, *SIAM J. Numer. Anal.*, 57 (2019), 2829–2849.
[32] P. Zhuang and F. Liu, Implicit difference approximation for the time fractional diffusion equation, *J. Appl. Math. Comput.*, 22 (2006), 87-99.

Received May 2020; revised December 2020.

E-mail address: QHLmath@163.com

E-mail address: aijie@sdu.edu.cn