INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 21, Number 3, Pages 431–458

doi: 10.4208/ijnam2024-1017

FAST NUMERICAL SOLVERS FOR SUBDIFFUSION PROBLEMS WITH SPATIAL INTERFACES

BOYANG YU, YONGHAI LI, AND JIANGGUO LIU*

Abstract. This paper develops novel fast numerical solvers for subdiffusion problems with spatial interfaces. These problems are modeled by partial differential equations that contain both fractional order and conventional first order time derivatives. The former is non-local and approximated by L1 and L2 discretizations along with fast evaluation algorithms based on approximation by sums of exponentials. This results in an effective treatment of the "long-tail" kernel of subdiffusion. The latter is local and hence conventional implicit Euler or Crank-Nicolson discretizations can be used. Finite volumes are utilized for spatial discretization based on consideration of local mass conservation. Interface conditions for mass and fractional fluxes are incorporated into these fast solvers. Computational complexity and implementation procedures are briefly discussed. Numerical experiments demonstrate accuracy and efficiency of these new fast solvers.

Key words. Caputo and Riemann-Liouville derivatives, fast numerical solvers, fractional order fluxes, interface problems, subdiffusion, sum of exponentials (SOE).

1. Introduction

Anomalous diffusion happens in many physical, chemical, and biological processes [7, 8, 30, 33, 37, 44]. It is known that subdiffusion is a measure of cytoplasmic crowdedness in living cells [44] and anomalous diffusion in cardiac tissues is an index of myocardial microstructure [7].

Mathematically, subdiffusion is modeled by partial differential equations with fractional order time derivatives in Caputo, Riemann-Liouville, or other forms [8, 30].

The work in [11] offers a kind of guide to identify some common pitfalls in fractional-order differential problems. Approximation of fractional derivatives by polynomial interpolation is the common idea: L1, L2 schemes [30] and fractional linear multistep methods [25, 26, 27, 28]. [11] also discusses that the effect of the solution regularity on the accuracy of the numerical scheme for fractional-derivative problems briefly. Furthermore, because of the non-locality of the fractional-order operator, nested mesh techniques [14, 10], the fast fourier transform algorithm [15, 16] and kernel compression scheme [2, 3, 4] are mentioned for a fast, efficient and reliable treatment of fractional-derivative problems.

The conventional discretization methods applied to the frational partial differential equations [24, 23, 39, 40, 36, 48] also involve computation for the entire time period and/or across the whole domain and hence are very expensive. Therefore, fast PDE numerical solvers have been developed to overcome these disadvantages [12, 17, 36, 43, 46, 47, 48, 38]. Various techniques have been developed, e.g., approximation of kernels by sums of exponentials [17, 36, 46], and parallelization in time [45].

Subdiffusion may happen simultaneously in subdomains that are separated by spatial interfaces. Time-fractional anomalous diffusion models are used in [13]

Received by the editors on July 19, 2023 and, accepted on April 16, 2024.
2000 *Mathematics Subject Classification.* 35R11, 65M08, 65Y04, 65Y20.
*Corresponding author.

for simulations of transport processes in heterogeneous binary media for which interface conditions are established. Subdiffusive flow in a composite medium with a communicating interface has been investigated in [34]. In [42], 1-dim moving interface problems governed by subdiffusion is investigated. More work can be found in [9].

However, fast numerical solvers for subdiffusion problems with spatial interfaces are not yet available in the literature, to the best of our knowledge. There are well developed numerical methods in [13] for subdiffusion problems with spatial interfaces and nonlinear terms, but no discussion on fast solvers. Fast solvers are undoubtedly important for this type of problems. Our paper intends to fill this gap.

Our fast solvers rely on a special treatment of the convolutional kernel. Specifically, the integral for the Caputo derivative is split as a *recent past term* and a *history term* (the so-called "long-tail"). Based on approximation of the negative kernel by a sum of exponentials [5, 6], a recurrence formula is established for the history term and efficient time-marching schemes are developed (See Sections 3 and 5 for details). Moreover, interface conditions are naturally incorporated. It is interesting to notice that similar notion for handling kernels was applied in an early work on parabolic problems [41].

The rest of this paper is organized as follows. Section 2 describes the governing equations and interface conditions. Section 3 briefly reviews the L1, L2 temporal discretizations on graded meshes and then establishes fast evaluation algorithms based on approximation of negative power kernels by sums of exponentials. Section 4 focuses on spatial discretization using finite volumes. Section 5 develops fast numerical solvers for subdiffusion problems with spatial interfaces by combining the implicit Euler or Crank-Nicolson discretization for the conventional 1st order time derivative with the fast L1 or L2 evaluation algorithms for the fractional order derivatives. Stability analysis of the direct/fast L1 + back-Euler solvers for a simplified model is presented in Section 6. Section 7 discusses computational complexity and implementation of the fast solvers. Section 8 presents numerical experiments. The paper is concluded with remarks in Section 9.

2. Mathematical Models for Subdiffusion Problems with Spatial Interfaces

In essence, such problems involve subdiffusion with different diffusion indexes in subdomains that are separated by interfaces. For ease of representation, we consider two 1-dim or 2-dim subdomains Ω_1, Ω_2 that are separated by one interface Γ. Specifically, we consider the following governing equations

$$
\begin{cases}
\partial_t u - {}_0^R\mathrm{D}_t^{1-\alpha_1} \nabla \cdot (A_1(\boldsymbol{x})\nabla u) = f_1(\boldsymbol{x}, t) & \text{in } \Omega_1 \times (0, T], \\
\partial_t u - {}_0^R\mathrm{D}_t^{1-\alpha_2} \nabla \cdot (A_2(\boldsymbol{x})\nabla u) = f_2(\boldsymbol{x}, t) & \text{in } \Omega_2 \times (0, T],
\end{cases} \tag{1}
$$

where $\Omega = \Omega_1 \cup \Omega_2$ is an open bounded connected domain in $\mathbb{R}^d (d = 1, 2)$,

(i) The time-fractional indices $\alpha_i \in (0, 1)$ for $i = 1, 2$;
(ii) $A_i(\boldsymbol{x}) = \left[a_{j,k}^{(i)}(\boldsymbol{x})\right]_{1 \le j,k \le d}, i = 1, 2$ are the diffusion tensors that are symmetric, bounded, and uniformly positive-definite on Ω_i;
(iii) The source terms $f_i \in L^2(\Omega)$ for $i = 1, 2$.

Note that in Equation (1), ${}_0^R\mathrm{D}_t^\alpha\, u$ denotes the Riemann-Liouville derivative that is defined as

$$ {}_0^R\mathrm{D}_t^\alpha\, u(\boldsymbol{x},t) = \frac{1}{\Gamma(1-\alpha)}\frac{\partial}{\partial t}\int_0^t \frac{u(\boldsymbol{x},s)}{(t-s)^\alpha}ds, \tag{2} $$

where $\Gamma(\cdot)$ is the Gamma function. Note also that the Caputo derivative is defined as

$$ {}_0^C\mathrm{D}_t^\alpha\, u(\boldsymbol{x},t) = \frac{1}{\Gamma(1-\alpha)}\int_0^t \frac{\partial_s u(\boldsymbol{x},s)}{(t-s)^\alpha}ds. \tag{3} $$

For subdiffusion problems, the time-nonlocal dependence between the flux J and the concentration gradient with a "long-tail" power kernel is reflected by the following definition [13, 32, 33]

$$ J_i(\boldsymbol{x},t) = -\,{}_0^R\mathrm{D}_t^{1-\alpha_i}\left(A_i(\boldsymbol{x})\nabla u\cdot\boldsymbol{n}_i(\boldsymbol{x})\right), \quad i=1,2, \tag{4} $$

where $\boldsymbol{n}_i(\boldsymbol{x})$ is the unit vector normal of $\partial\Omega_i$ outward to Ω_i at point $\boldsymbol{x}$.

In this paper, we focus on the following time-fractional diffusion equations [20, 21, 29]

$$ \partial_t u = {}_0^R\mathrm{D}_t^{1-\alpha_i}\,\nabla\cdot(A_i(\boldsymbol{x})\nabla u) + f_i(\boldsymbol{x},t) \quad \text{in } \Omega_i\times(0,T], \quad i=1,2, \tag{5} $$

subject to the ***initial and Dirichlet boundary conditions*** listed below

$$ u(\boldsymbol{x},0) = u_0(\boldsymbol{x}), \quad \boldsymbol{x}\in\Omega, \tag{6} $$

$$ u(\boldsymbol{x},t) = u_b(\boldsymbol{x},t), \quad \boldsymbol{x}\in\Gamma=\partial\Omega. \tag{7} $$

Furthermore, the ***interface conditions*** are posed as, for $(\boldsymbol{x},t)\in\Gamma_{int}\times(0,T]$, (note $\Gamma_{int}=\partial\Omega_1\cap\partial\Omega_2$)

$$ u(\boldsymbol{x},t)\big|_{\Omega_1} = u(\boldsymbol{x},t)\big|_{\Omega_2}, \tag{8} $$

and

$$ {}_0^R\mathrm{D}_t^{1-\alpha_1}(A_1(\boldsymbol{x})\nabla u(\boldsymbol{x},t)\cdot\boldsymbol{n}_1(\boldsymbol{x}))\big|_{\Omega_1} = {}_0^R\mathrm{D}_t^{1-\alpha_2}(A_2(\boldsymbol{x})\nabla u(\boldsymbol{x},t)\cdot\boldsymbol{n}_2(\boldsymbol{x}))\big|_{\Omega_2}. \tag{9} $$

For a time-fractional or conventional diffusion equation, u can be understood as the concentration of some substance. Mass continuity condition is same for these two types of diffusion equations. Now the definition of flux is different. For the conventional diffusion, the flux continuity condition reads as

$$ A_1(\boldsymbol{x})\nabla u(\boldsymbol{x},t)\cdot\boldsymbol{n}_1(\boldsymbol{x})\big|_{\Omega_1} = A_2(\boldsymbol{x})\nabla u(\boldsymbol{x},t)\cdot\boldsymbol{n}_2(\boldsymbol{x})\big|_{\Omega_2}. $$

But for subdiffusion problems, we use the fractional order fluxes in Equation (9). See [13] also.

Recall the relationship between the Caputo and Riemann-Liouville derivatives reads as [30]

$$ {}_0^C\mathrm{D}_t^{1-\alpha}\, u(\boldsymbol{x},t) = {}_0^R\mathrm{D}_t^{1-\alpha}\left(u(\boldsymbol{x},t)-u(\boldsymbol{x},0)\right), \tag{10} $$

which can be rewritten as

$$ {}_0^R\mathrm{D}_t^{1-\alpha}\, u(\boldsymbol{x},t) = {}_0^C\mathrm{D}_t^{1-\alpha}\, u(\boldsymbol{x},t) + {}_0^R\mathrm{D}_t^{1-\alpha}\, u(\boldsymbol{x},0) = {}_0^C\mathrm{D}_t^{1-\alpha}\, u(\boldsymbol{x},t) + \frac{u(\boldsymbol{x},0)}{\Gamma(\alpha)}t^{\alpha-1}. \tag{11} $$

Accordingly, there holds, for $i=1,2$,

$$ {}_0^R\mathrm{D}_t^{1-\alpha_i}\left(\nabla\cdot(A_i(\boldsymbol{x})\nabla u)\right) = {}_0^C\mathrm{D}_t^{1-\alpha_i}\left(\nabla\cdot(A_i(\boldsymbol{x})\nabla u)\right) + \frac{\nabla\cdot(A_i(\boldsymbol{x})\nabla u(\boldsymbol{x},0))}{\Gamma(\alpha_i)}t^{\alpha_i-1}. \tag{12} $$

3. Time-fractional Derivatives and Fast Evaluation Algorithms

This section briefly reviews the L1 and L2 discretizations of the Caputo derivative [23, 36, 39, 48]. The negative power kernel in the definition of the Caputo derivative can be approximated by sums of exponentials (SOE), which lead to fast evaluation algorithms. The Prony's reduction algorithm further reduces the numbers of terms of exponentials needed for approximation.

3.1. Solution Regularity and Graded Meshes. Consider a subdiffusion benchmark problem [13]

$$\frac{du}{dt} = -\kappa\, {}_0^R\mathrm{D}_t^{1-\alpha}\, u, \text{ in } (0,T], \quad u(0) = u_0, \tag{13}$$

which has the solution $u(t) = u_0 E_\alpha(-\kappa t^\alpha) = u_0 \sum_{j=0}^{\infty} \frac{(-\kappa)^j t^{\alpha j}}{\Gamma(\alpha j+1)}$, This analytical solution can be cast into the following form:

$$u(t) = u_1(t) + u_2(t),\ u_1(t) = \sum_{j=0}^{m} c_j t^{\sigma_j},\ u_2(t) = c_{m+1} t^{\sigma_{m+1}} + \xi(t) t^{\sigma_{m+2}}, \tag{14}$$

where c_j is a real constant, $\sigma_j = \alpha j$, $\xi(t)$ a continuous function. For the low regularity part $u_1(t)$, we have

$$|u_1'(t)| \le C t^{\alpha-1}, \quad \forall t \in (0,T]. \tag{15}$$

It is important to note that the essential feature of the typical solution of (13) is that $u(t)$ has an initial layer at $t = 0$ and $u'(t)$ blows up as $t \to 0^+$. The L1 and L2 formulas for the low regularity part $u_1(t)$ will fail on the uniform grid when σ_j is small, which leads to a large discretisation error near $t = 0$ and the error accumulates globally.

To address this issue, let $N_T > 0$ be the number of temporal partitions. A graded mesh $t_n = T(n/N_T)^r$ for $n = 0, 1, \ldots, N_T$ is applied to L1 and L2 formulas, the mesh grading $r \ge 1$ is chosen by the user. The graded mesh with $r = 2, N_T = 10$ shown in Figure 1 is relatively fine near the initial time. The work in [35, 18] show clearly how the regularity of the solution and the grading of the mesh affect the order of convergence.

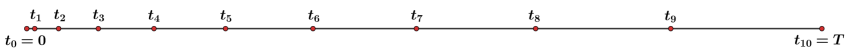


FIGURE 1. A graded mesh with $r = 2, N_T = 10$.

3.2. L1 Discretization of Fractional Order Derivatives. The Caputo or Riemann-Liouville derivative is actually an integral. The L1 discretization of such an integral is based on a piecewise linear approximation of function $u(\cdot)$ in the integrand. Assume the time interval $[0,T]$ has a partition $t_n = T(n/N_T)^r$ for $n = 0, 1, \ldots, N_T$. Let $\tau_n = t_n - t_{n-1}, \tau_{n,k} = t_n - t_k$ for $n > k \ge 0, n = 1, 2, \ldots, N_T$. For convenience, we denote $u(t_n)$ as $u^{(n)}$. The piecewise linear approximant is expressed as

$$(\Pi_{1,k} u)(t) = u^{(k-1)} \frac{t_k - t}{\tau_k} + u^{(k)} \frac{t - t_{k-1}}{\tau_k}, \quad \forall t \in [t_{k-1}, t_k], \quad 1 \le k \le N_T. \tag{16}$$

Then its derivative is a piecewise constant

$$(\Pi_{1,k} u)'(t) = \frac{u^{(k)} - u^{(k-1)}}{\tau_k}, \quad \forall t \in (t_{k-1}, t_k). \tag{17}$$

This implies that, for any t_n with $1 \le n \le N_T$,

$$
\begin{aligned}
{}_0^C\mathrm{D}_t^{1-\alpha} u(t_n) &= \frac{1}{\Gamma(\alpha)} \int_0^{t_n} \frac{u'(s)}{(t_n - s)^{1-\alpha}} ds \approx \frac{1}{\Gamma(\alpha)} \sum_{k=1}^{n} \int_{t_{k-1}}^{t_k} \frac{(\Pi_{1,k} u)'(s)}{(t_n - s)^{1-\alpha}} ds \\
&= \frac{1}{\Gamma(\alpha)} \sum_{k=1}^{n} \frac{u^{(k)} - u^{(k-1)}}{\tau_k} \int_{t_{k-1}}^{t_k} \frac{ds}{(t_n - s)^{1-\alpha}} =: \mathrm{D}_{L1}^{1-\alpha} u^{(n)}.
\end{aligned}
$$

Direct calculations of the above simple integrals yield, for $1 \le n \le N_T$,

$$
\mathrm{D}_{L1}^{1-\alpha} u^{(n)} = \frac{1}{\Gamma(1+\alpha)} \sum_{k=1}^{n} \frac{u^{(k)} - u^{(k-1)}}{\tau_k} (\tau_{n,k-1}^{\alpha} - \tau_{n,k}^{\alpha}). \tag{18}
$$

Accordingly, we approximate the Riemann-Liouville derivative as

$$
{}_0^R\mathrm{D}_t^{1-\alpha} u(t_n) \approx \frac{1}{\Gamma(1+\alpha)} \sum_{k=1}^{n} \frac{u^{(k)} - u^{(k-1)}}{\tau_k} (\tau_{n,k-1}^{\alpha} - \tau_{n,k}^{\alpha}) + \frac{u^{(0)}}{\Gamma(\alpha)} t_n^{\alpha-1}. \tag{19}
$$

3.3. L2 Discretization of Time-fractional Derivatives. Now we consider the L2 discretization of a time-fractional order derivative based on the quadratic interpolation of $u(\cdot)$ in the integrand. Again, let $t_n = T(n/N)^r$ for $0 \le n \le N_T$, $t_{n-\frac{1}{2}} = \frac{t_n + t_{n-1}}{2}$. And $\tau_n = t_n - t_{n-1}, \tau_{n,k} = t_n - t_k, \tau_{n-\frac{1}{2},k} = t_{n-\frac{1}{2}} - t_k$ for $n > k \ge 0, n = 1, 2, \ldots, N_T$. For any $t \in [t_{k-1}, t_{k+1}]$, define

$$
\begin{aligned}
(\Pi_{2,k} u)(t) =& u^{(k-1)} \frac{(t - t_k)(t - t_{k+1})}{\tau_k(\tau_k + \tau_{k+1})} - u^{(k)} \frac{(t - t_{k-1})(t - t_{k+1})}{\tau_k \tau_{k+1}} \\
&+ u^{(k+1)} \frac{(t - t_{k-1})(t - t_k)}{\tau_{k+1}(\tau_k + \tau_{k+1})}.
\end{aligned}
$$

Going through similar calculation details, we obtain, for $n \ge 2$,

$$
\begin{cases}
\mathrm{D}_{L2}^{1-\alpha} u^{(\frac{1}{2})} := \dfrac{1}{\Gamma(\alpha)} \displaystyle\int_{t_0}^{t_{\frac{1}{2}}} \frac{(\Pi_{1,1} u)'(s)}{(t_{\frac{1}{2}} - s)^{1-\alpha}} ds = \frac{\tau_1^{\alpha-1}}{2^{\alpha}\Gamma(1+\alpha)} (u^{(1)} - u^{(0)}), \\
\mathrm{D}_{L2}^{1-\alpha} u^{(n-\frac{1}{2})} := \dfrac{1}{\Gamma(\alpha)} \left(\displaystyle\int_{t_{n-1}}^{t_{n-\frac{1}{2}}} \frac{(\Pi_{2,n-1} u)'(s)}{(t_{n-\frac{1}{2}} - s)^{1-\alpha}} ds + \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} \frac{(\Pi_{2,k} u)'(s)}{(t_{n-\frac{1}{2}} - s)^{1-\alpha}} ds \right) \\
\qquad\qquad = \dfrac{1}{\Gamma(\alpha)} \left(a_n^{(\alpha)} u^{(n-2)} + b_n^{(\alpha)} u^{(n-1)} + c_n^{(\alpha)} u^{(n)} \right. \\
\qquad\qquad \left. + \sum_{k=1}^{n-1} a_{n,k}^{(\alpha)} u^{(k-1)} + b_{n,k}^{(\alpha)} u^{(k)} + c_{n,k}^{(\alpha)} u^{(k+1)} \right),
\end{cases} \tag{20}
$$

where

$$
\begin{cases}
a_n^{(\alpha)} = \dfrac{1}{\tau_{n-1}(\tau_n + \tau_{n-1})} \left(- \dfrac{2}{1+\alpha} \tau_{n-\frac{1}{2},n-1}^{1+\alpha} + \dfrac{\tau_{n-\frac{1}{2},n-1} + \tau_{n-\frac{1}{2},n}}{\alpha} \tau_{n-\frac{1}{2},n-1}^{\alpha} \right), \\
b_n^{(\alpha)} = -\dfrac{1}{\tau_n \tau_{n-1}} \left(- \dfrac{2}{1+\alpha} \tau_{n-\frac{1}{2},n-1}^{1+\alpha} + \dfrac{\tau_{n-\frac{1}{2},n-2} + \tau_{n-\frac{1}{2},n}}{\alpha} \tau_{n-\frac{1}{2},n-1}^{\alpha} \right), \\
c_n^{(\alpha)} = \dfrac{1}{\tau_n(\tau_n + \tau_{n-1})} \left(- \dfrac{2}{1+\alpha} \tau_{n-\frac{1}{2},n-1}^{1+\alpha} + \dfrac{\tau_{n-\frac{1}{2},n-2} + \tau_{n-\frac{1}{2},n-1}}{\alpha} \tau_{n-\frac{1}{2},n-1}^{\alpha} \right),
\end{cases} \tag{21}
$$

and

$$
\left\{
\begin{aligned}
a_{n,k}^{(\alpha)} &= \frac{1}{\tau_{n-1}(\tau_n+\tau_{n-1})}\bigg(\frac{2}{1+\alpha}(\tau_{n-\frac{1}{2},k}^{1+\alpha}-\tau_{n-\frac{1}{2},k-1}^{1+\alpha})\\
&\quad -\frac{\tau_{n-\frac{1}{2},n-1}+\tau_{n-\frac{1}{2},n}}{\alpha}(\tau_{n-\frac{1}{2},k}^{\alpha}-\tau_{n-\frac{1}{2},k-1}^{\alpha})\bigg),\\
b_{n,k}^{(\alpha)} &= -\frac{1}{\tau_n\tau_{n-1}}\bigg(\frac{2}{1+\alpha}(\tau_{n-\frac{1}{2},k}^{1+\alpha}-\tau_{n-\frac{1}{2},k-1}^{1+\alpha})\\
&\quad -\frac{\tau_{n-\frac{1}{2},n-2}+\tau_{n-\frac{1}{2},n}}{\alpha}(\tau_{n-\frac{1}{2},k}^{\alpha}-\tau_{n-\frac{1}{2},k-1}^{\alpha})\bigg),\\
c_{n,k}^{(\alpha)} &= \frac{1}{\tau_n(\tau_n+\tau_{n-1})}\bigg(\frac{2}{1+\alpha}(\tau_{n-\frac{1}{2},k}^{1+\alpha}-\tau_{n-\frac{1}{2},k-1}^{1+\alpha})\\
&\quad -\frac{\tau_{n-\frac{1}{2},n-2}+\tau_{n-\frac{1}{2},n-1}}{\alpha}(\tau_{n-\frac{1}{2},k}^{\alpha}-\tau_{n-\frac{1}{2},k-1}^{\alpha})\bigg).
\end{aligned}
\right. \tag{22}
$$

Accordingly, the L2 approximation to the Riemann-Liouville derivative is established as

$$
{}_0^R\mathrm{D}_t^{1-\alpha}\,u(t_{n-\frac{1}{2}}) \approx \mathrm{D}_{L2}^{1-\alpha}\,u^{(n-\frac{1}{2})} + \frac{u^{(0)}}{\Gamma(\alpha)}t_{n-\frac{1}{2}}^{\alpha-1}. \tag{23}
$$

3.4. Approximation of a Negative Power Kernel by Sum of Exponentials. The computational cost for numerically solving time-fractional PDEs are huge if the direct L1 or L2 discretization formula is used. Fast evaluation algorithms have been developed thanks to approximation by a sum of exponentials (SOE) to the negative power kernel in the definition of the Caputo derivative.

Lemma 1. *(Approximation of a negative power by SOE* [6]*). For any exponent* $\beta \in (0,1)$*, an error tolerance* $\varepsilon \in (0, e^{-1})$*, and a cut-off time* $\delta \in (0,1)$*, there exist constants* L*,* $\mathcal{M}$*,* $\mathcal{N}$ *such that*

$$
\Big|t^{-\beta} - S(t;\mathcal{M},\mathcal{N},L)\Big| \Big/ t^{-\beta} \le \varepsilon, \quad \forall t \in [\delta, 1], \tag{24}
$$

where $\mathcal{M}$*,* $\mathcal{N}$ *are usually taken as integers,*

$$
S(t;\mathcal{M},\mathcal{N},L) = \frac{L}{\Gamma(\beta)}\sum_{j=\mathcal{M}+1}^{\mathcal{N}} e^{j\beta L}e^{-e^{jL}t}, \tag{25}
$$

$$
\left\{
\begin{aligned}
&L \le \frac{2\pi}{\log 3 + \beta\log(\cos 1)^{-1} + \log(\varepsilon^{-1})},\\
&\mathcal{M} \ge L^{-1}\Big(\beta^{-1}\log\varepsilon + \beta^{-1}\log\Gamma(1+\beta)\Big),\\
&\mathcal{N} \le L^{-1}\Big(\log(\delta^{-1}) + \log\log(\varepsilon^{-1}) + \log\beta + 1\Big).
\end{aligned}
\right. \tag{26}
$$

The number of terms in the sum $S(t;\mathcal{M},\mathcal{N},L)$ *may be estimated as*

$$
\mathcal{N}-\mathcal{M} \le 10^{-1}(2\log\varepsilon^{-1} + \log\beta + 2)\Big(\log(\delta^{-1}) + \beta^{-1}\log\varepsilon^{-1} + \log\log\varepsilon^{-1} + 2\Big). \tag{27}
$$

Note that the Prony's algorithm can be used [5, 6] to reduce the number of terms in $S(t;\mathcal{M},\mathcal{N},L)$. In practical computation, the sum $S(t;\mathcal{M},\mathcal{N},L)$ is usually split into two sums $S(t;\mathcal{M},-1,L)$ and $S(t;-1,\mathcal{N},L)$. Then we apply the Prony's

TABLE 1. Numbers of exponential terms needed to approximate the negative power $t^{-0.5}$.

	SOE			Reduced SOE		
ε \ δ	10^{-2}	10^{-3}	10^{-4}	10^{-2}	10^{-3}	10^{-4}
10^{-3}	29	32	35	15	18	21
10^{-6}	87	92	98	26	31	37
10^{-9}	175	183	192	36	44	51

algorithm to $S(t;\mathcal{M},-1,L)$ to obtain a new approximation $S_r(t)$. Finally, we obtain a reduced approximation

$$t^{-\beta} \approx S(t;-1,\mathcal{N},L) + S_r(t).$$

Remark 1. *The reduction needs several "good" properties, for which details can be found in* [6]*. These properties are satisfied in our numerical experiments and thus the algorithm is stable.*

Lemma 2. *(Approximation to a negative power by reduced SOE). For any fractional exponent $\beta \in (0,1)$, an error tolerance $\varepsilon \in (0, e^{-1}]$, and a cut-off time $\delta \in (0,1]$, according to the Prony's algorithm and Lemma 1, there exist a positive integer N_{exp}, positive constants λ_j and positive weights θ_j for $j = 1,2,\ldots,N_{exp}$, such that the relative error*

$$\left| t^{-\beta} - \sum_{j=1}^{N_{exp}} \theta_j e^{-\lambda_j t} \right| \Big/ t^{-\beta} \le \varepsilon, \quad \forall t \in [\delta, 1]. \tag{28}$$

The effectiveness of the reduction algorithm is demonstrated by Table 1, which lists the number of terms in the approximation of $t^{-\beta}$ with $\beta = 0.5$. It is obvious that the latter use much less terms while attaining the same levels of errors.

3.5. A Fast Evaluation Algorithm for the L1 Discretization. Recall the L1 discretization yields

$$D_{L1}^{1-\alpha} u^{(n)} = \frac{1}{\Gamma(\alpha)} \sum_{k=1}^{n} \int_{t_{k-1}}^{t_k} \frac{(\Pi_{1,k}u)'(s)}{(t_n - s)^{1-\alpha}} ds,$$

which leads to a direct evaluation algorithm stated in (18) that is computationally expensive, since computation of $u^{(n)}$ for any $1 \le n \le N_T$ needs to go through the whole history to the very beginning.

By splitting the above sum into two parts,

$$\begin{aligned} D_{L1}^{1-\alpha} u^{(n)} =& \frac{1}{\Gamma(\alpha)} \int_{t_{n-1}}^{t_n} \frac{(\Pi_{1,k}u)'(s)}{(t_n - s)^{1-\alpha}} ds + \frac{1}{\Gamma(\alpha)} \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} \frac{(\Pi_{1,k}u)'(s)}{(t_n - s)^{1-\alpha}} ds \\ =:& I_R(t_n) + I_H(t_n), \end{aligned} \tag{29}$$

we observe that the ***recent past*** term $I_R(t_n)$ on $[t_{n-1}, t_n]$ can still be handled by the standard L1 approximation as

$$I_R(t_n) = \frac{u^{(n)} - u^{(n-1)}}{\Gamma(\alpha)\tau_n} \int_{t_{n-1}}^{t_n} \frac{ds}{(t_n - s)^{1-\alpha}} = \frac{u^{(n)} - u^{(n-1)}}{\Gamma(1+\alpha)\tau_n^{1-\alpha}}. \tag{30}$$

The ***history*** term $I_H(t_n)$ on $[0, t_{n-1}]$ causes the "long-tail" but can be reformulated. Based on approximation of a negative power kernel by a reduced SOE, a recurrence

relation formula can be established to significantly reduce the computational cost of the L1 discretization.

Applying Lemma 2 (approximation of a negative kernel by a sum of exponentials), we obtain

$$\begin{aligned} I_H(t_n) &= \frac{T^{\alpha-1}}{\Gamma(\alpha)} \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} (\Pi_{1,k}u)'(s) \Big(\frac{t_n - s}{T}\Big)^{\alpha-1} ds \\ &\approx \frac{T^{\alpha-1}}{\Gamma(\alpha)} \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} (\Pi_{1,k}u)'(s) \sum_{j=1}^{N_{exp}} \theta_j e^{-\lambda_j (t_n - s)/T} ds \\ &= \frac{T^{\alpha-1}}{\Gamma(\alpha)} \sum_{j=1}^{N_{exp}} \theta_j \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} (\Pi_{1,k}u)'(s)\, e^{-\lambda_j (t_n - s)/T} ds. \end{aligned}$$

For convenience, we denote, for $1 \le n \le N_T$ and $1 \le j \le N_{exp}$,

$$\omega_{FL1}^{(n,j)} = \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} (\Pi_{1,k}u)'(s)\, e^{-\lambda_j (t_n - s)/T} ds. \tag{31}$$

Now we split the above sum and perform direct calculations to obtain

$$\begin{aligned} \omega_{FL1}^{(n,j)} =& \sum_{k=1}^{n-2} \int_{t_{k-1}}^{t_k} (\Pi_{1,k}u)'(s)\, e^{-\lambda_j (t_n - s)/T} ds + \int_{t_{n-2}}^{t_{n-1}} (\Pi_{1,n-1}u)'\, e^{-\lambda_j (t_n - s)/T} ds \\ =& e^{-\lambda_j (\tau_n/T)} \sum_{k=1}^{n-2} \int_{t_{k-1}}^{t_k} (\Pi_{1,k}u)'(s)\, e^{-\lambda_j (t_{n-1} - s)/T} ds \\ &+ \int_{t_{n-2}}^{t_{n-1}} \frac{u^{(n-1)} - u^{(n-2)}}{\tau_{n-1}} e^{-\lambda_j (t_n - s)/T} ds, \end{aligned}$$

which yields, for $2 \le n \le N_T$ and $1 \le j \le N_{exp}$,

$$\omega_{FL1}^{(n,j)} = e^{-\lambda_j (\tau_n/T)} \omega_{FL1}^{(n-1,j)} + \frac{e^{-\lambda_j (\tau_n/T)} - e^{-\lambda_j (\tau_{n,n-2}/T)}}{\lambda_j/T} \frac{u^{(n-1)} - u^{(n-2)}}{\tau_{n-1}}. \tag{32}$$

By default, $\omega_{FL1}^{(1,j)} = 0$ for $1 \le j \le N_{exp}$. As we shall see later, this nice recurrence relation (or time-marching) formula significantly expedites the L1 discretization of time-fractional order derivatives.

Now we put all these together to establish a fast L1 evaluation algorithm for the Caputo derivative.

- For $n = 1$, the fast algorithm uses the usual L1 evaluation formula

$$\mathrm{D}_{FL1}^{1-\alpha} u^{(1)} = \frac{1}{\Gamma(1+\alpha)} \frac{u^{(1)} - u^{(0)}}{\tau_1^{1-\alpha}}. \tag{33}$$

- For $n = 2, 3, \cdots, N_T$, we have

$$\mathrm{D}_{FL1}^{1-\alpha} u^{(n)} = \frac{T^{\alpha-1}}{\Gamma(\alpha)} \sum_{j=1}^{N_{exp}} \theta_j \omega_{FL1}^{(n,j)} + \frac{1}{\Gamma(1+\alpha)} \frac{u^{(n)} - u^{(n-1)}}{\tau_n^{1-\alpha}}, \tag{34}$$

where the auxiliary quantity $\omega_{FL1}^{(n,j)}$ satisfies a recurrence formula stated above but slightly reformulated as below.

(35)
$$\begin{cases} \omega_{FL1}^{(n,j)} = e^{-\lambda_j(\tau_n/T)}\omega_{FL1}^{(n-1,j)} + \dfrac{e^{-\lambda_j(\tau_n/T)} - e^{-\lambda_j(\tau_{n,n-2}/T)}}{\lambda_j\tau_{n-1}/T}\left(u^{(n-1)} - u^{(n-2)}\right), \\ \omega_{FL1}^{(1,j)} = 0, \qquad \forall 1 \le j \le N_{exp}. \end{cases}$$

3.6. A Fast Evaluation Algorithm for the L2 Discretization at the Midpoint. Similarly, we first split the integral

$$\begin{aligned} D_{L2}^{1-\alpha}u^{(n-\frac{1}{2})} =& \frac{1}{\Gamma(\alpha)}\int_{t_{n-1}}^{t_{n-\frac{1}{2}}} \frac{(\Pi_{2,n-1}u)'(s)}{(t_{n-\frac{1}{2}}-s)^{1-\alpha}}ds \\ &+ \frac{1}{\Gamma(\alpha)}\sum_{k=1}^{n-1}\int_{t_{k-1}}^{t_k} \frac{(\Pi_{2,k}u)'(s)}{(t_{n-\frac{1}{2}}-s)^{1-\alpha}}ds,\ n \ge 2 \end{aligned}$$

as a recent past term and a history term.

The recent past term can be handled directly and we end up with

$$\frac{1}{\Gamma(\alpha)}\int_{t_{n-1}}^{t_{n-\frac{1}{2}}} \frac{(\Pi_{2,n-1}u)'(s)}{(t_{n-\frac{1}{2}}-s)^{1-\alpha}}ds = \frac{1}{\Gamma(\alpha)}\left(a_n^{(\alpha)}u^{(n-2)} + b_n^{(\alpha)}u^{(n-1)} + c_n^{(\alpha)}u^{(n)}\right). \tag{36}$$

For the history term, we apply the reduced SOE approximation to obtain

$$\begin{aligned} &\frac{1}{\Gamma(\alpha)}\sum_{k=1}^{n-1}\int_{t_{k-1}}^{t_k} \frac{(\Pi_{2,k}u)'(s)}{(t_{n-\frac{1}{2}}-s)^{1-\alpha}}ds \\ =&\frac{T^{\alpha-1}}{\Gamma(\alpha)}\sum_{k=1}^{n-1}\int_{t_{k-1}}^{t_k} (\Pi_{2,k}u)'(s)\left(\frac{t_{n-\frac{1}{2}}-s}{T}\right)^{\alpha-1}ds \\ \approx&\frac{T^{\alpha-1}}{\Gamma(\alpha)}\sum_{k=1}^{n-1}\int_{t_{k-1}}^{t_k} (\Pi_{2,k}u)'(s)\sum_{j=1}^{N_{exp}}\theta_j e^{-\lambda_j(t_{n-\frac{1}{2}}-s)/T}ds \\ =&\frac{T^{\alpha-1}}{\Gamma(\alpha)}\sum_{j=1}^{N_{exp}}\theta_j\,\omega_{FL2}^{(n,j)}. \end{aligned} \tag{37}$$

where

$$\omega_{FL2}^{(n,j)} = \sum_{k=1}^{n-1}\int_{t_{k-1}}^{t_k} (\Pi_{2,k}u)'(s)\,e^{-\lambda_j(t_{n-\frac{1}{2}}-s)/T}ds. \tag{38}$$

Again we split the above sum. Some diligent Calculus exercises yield a recurrence formula

(39)
$$\begin{cases} \omega_{FL2}^{(n,j)} = e^{-\lambda_j(\tau_n+\tau_{n-1})/(2T)}\omega_{FL2}^{(n-1,j)} + A_j^{(\alpha)}u^{(n-2)} + B_j^{(\alpha)}u^{(n-1)} + C_j^{(\alpha)}u^{(n)}, \\ \omega_{FL2}^{(1,j)} = 0, \quad \text{for } 1 \le j \le N_{exp}, \end{cases}$$

where, after introduction of $\rho_j = \lambda_j/T$,

$$
\left\{
\begin{aligned}
A_j^{(\alpha)} &= \frac{1}{\rho_j \tau_{n-1}(\tau_n + \tau_{n-1})}\Big((-\tau_n - 2/\rho_j)e^{-\rho_j \tau_{n-\frac{1}{2},n-1}} \\
&\quad +(\tau_{n,n-2} + \tau_{n-1} + 2/\rho_j)e^{-\rho_j \tau_{n-\frac{1}{2},n-2}}\Big), \\
B_j^{(\alpha)} &= -\frac{1}{\rho_j \tau_{n-1}\tau_n}\Big((\tau_{n-1} - \tau_n - 2/\rho_j)e^{-\rho_j \tau_{n-\frac{1}{2},n-1}} \\
&\quad +(\tau_{n,n-2} + 2/\rho_j)e^{-\rho_j \tau_{n-\frac{1}{2},n-2}}\Big), \\
C_j^{(\alpha)} &= \frac{1}{\rho_j \tau_n(\tau_n + \tau_{n-1})}\Big((\tau_{n-1} - 2/\rho_j)e^{-\rho_j \tau_{n-\frac{1}{2},n-1}} \\
&\quad +(\tau_{n-1} + 2/\rho_j)e^{-\rho_j \tau_{n-\frac{1}{2},n-2}}\Big),
\end{aligned}
\right. \tag{40}
$$

Finally, we obtain a fast evaluation algorithm for the L2 discretization as follows.

$$
\left\{
\begin{aligned}
&\mathrm{D}_{FL2}^{1-\alpha} u^{(n-\frac{1}{2})} = \frac{T^{1-\alpha}}{\Gamma(\alpha)} \sum_{j=1}^{N_{exp}} \theta_j \omega_{FL2}^{(n,j)} + \frac{1}{\Gamma(\alpha)} \left(a_n^{(\alpha)} u^{(n-2)} + b_n^{(\alpha)} u^{(n-1)} + c_n^{(\alpha)} u^{(n)}\right), \\
&\quad \text{for } \ n = 2, 3, \cdots, N_T; \\
&\mathrm{D}_{FL2}^{1-\alpha} u^{(\frac{1}{2})} = \frac{\tau_1^{\alpha-1}}{2^\alpha \Gamma(1+\alpha)}(u^{(1)} - u^{(0)}),
\end{aligned}
\right. \tag{41}
$$

4. Spatial Discretization by Finite Volumes

Among many possible choices for spatial discretization, we prefer the node-oriented finite volume method, which produces locally mass-conservative numerical solutions. For ease of presentation, we focus on 2-dimensional problems.

4.1. A Primary Triangular Mesh and the P_1 Trial Space. Let Ω be a polygonal domain and $\mathcal{T}_h = \{K\}$ be a quasi-uniform triangular mesh on $\overline{\Omega}$, where K represents a typical triangular element. Let h be the largest diameter of all triangles. Let $\mathcal{P}_h$ be the set of all vertices in the triangular mesh and N_P be the number of vertices.

We use piecewise linear shape functions to form the trial space

$$
U_h = \{u_h \in C(\overline{\Omega}) : u_h|_K \in P_1(K), \forall K \in \mathcal{T}_h\}, \tag{42}
$$

where $P_1(K)$ is the space of all linear (first-order) Lagrange polynomials on K. It is obvious that $u_h|_K$ is determined by the values at the three vertices. Based on this, we construct global basis functions $\phi_j(\boldsymbol{x})$ centered at the j-th node for any $1 \le j \le N_P$.

For time-dependent problems with a given temporal partition $t_n = T(n/N_T)^r, 0 \le n \le N_T$, we consider approximation

$$
u(\boldsymbol{x}, t_n) \approx \sum_{j=1}^{N_P} u_j^{(n)} \phi_j(\boldsymbol{x}). \tag{43}
$$

For convenience, we formulate $u_j^{(n)}(1 \le j \le N_P)$ as a size-N_P column vector and denote it as $\boldsymbol{u}_h^{(n)}$.

4.2. The Dual Mesh and Test Space of Constants. For the node-oriented finite volume methods, the discrete equations are actually established on the dual volumes. A dual volume is usually a polygon centered at a given node and enclosed by a zig-zag line segment connecting the midpoints of the adjacent edges. Illustrated in Figure 2 is a dual volume centered at the node P_6, which is adjacent to 5 other nodes via 5 edges. The dual volume is enclosed by a 10-piece zigzag line segment.

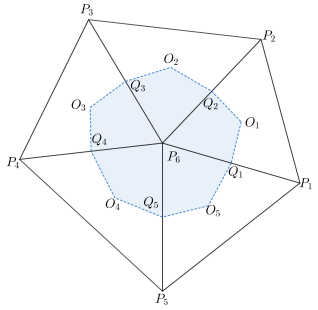


FIGURE 2. A dual element/volume surrounding a primal node.

The dual mesh consists of all such dual volumes and is denoted as $\mathcal{T}_h^* = \{K_j^*, j = 1, \ldots, N_P\}$, where K_j^* is the small polygon surrounding the j-th node P_j. The test space is thus defined as

$$V_h(\overline{\Omega}) = \{v_h \in L^2(\overline{\Omega}) : v_h|_{K_j^*} = \text{constant}, \forall K_j^* \in \mathcal{T}_h^*\}. \tag{44}$$

Equivalently,

$$V_h(\overline{\Omega}) = \text{Span}\{\psi_P; \forall P \in \mathcal{P}_h\}, \tag{45}$$

where ψ_P is the characteristic function for K_P^*. In the following, we abbreviate ψ_{P_j} to ψ_j which is the characteristic function for $K_{P_j}^*$.

5. Fast Numerical Solvers for Subdiffusion Problems with Spatial Interfaces

This section develops fast numerical solvers for the PDEs in (5) in two dimensions. The conventional first order time derivative will be discretized by the implicit Euler or Crank-Nicolson methods, whereas the fractional order time derivative is handled by the L1 or L2 discretization along with fast evaluation algorithms. Spatial discretization is based on $\mathcal{P}_1$-type node-oriented finite volumes. Essentially, these solvers are time-marching schemes that possess mass and fractional flux continuity across the interfaces and local mass conservation on dual volumes.

For ease of writing, we define the symbol $\boldsymbol{n}$ inside the integral as the outer unit vector normal of the integral domain. We integrate both sides of Equation (5) over

a typical dual volume K_P^* to obtain, for any $P \in \mathcal{P}_h$,

$$\begin{aligned}\int_{K_P^*} \partial_t u_h d\boldsymbol{x} =& {}_0^R\mathrm{D}_t^{1-\alpha_1} \int_{K_P^*\cap\Omega_1} \nabla\cdot(A_1(\boldsymbol{x})\nabla u_h)\, d\boldsymbol{x} \\ &+ {}_0^R\mathrm{D}_t^{1-\alpha_2} \int_{K_P^*\cap\Omega_2} \nabla\cdot(A_2(\boldsymbol{x})\nabla u_h)\, d\boldsymbol{x} \\ &+ \int_{K_P^*\cap\Omega_1} f_1(\boldsymbol{x},t)d\boldsymbol{x} + \int_{K_P^*\cap\Omega_2} f_2(\boldsymbol{x},t)d\boldsymbol{x} \\ =& {}_0^R\mathrm{D}_t^{1-\alpha_1} \sum_{j=1}^{N_P} u_j \int_{\partial(K_P^*\cap\Omega_1)} (A_1(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x}))\cdot\boldsymbol{n} ds \\ &+ {}_0^R\mathrm{D}_t^{1-\alpha_2} \sum_{j=1}^{N_P} u_j \int_{\partial(K_P^*\cap\Omega_2)} (A_2(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x}))\cdot\boldsymbol{n} ds \\ &+ \int_{K_P^*\cap\Omega_1} f_1(\boldsymbol{x},t)d\boldsymbol{x} + \int_{K_P^*\cap\Omega_2} f_2(\boldsymbol{x},t)d\boldsymbol{x}.\end{aligned} \tag{46}$$

As for $P \in \Gamma_{int}$, we combine (46) and (9) to obtain

$$\begin{aligned}\int_{K_P^*} \partial_t u_h d\boldsymbol{x} =& {}_0^R\mathrm{D}_t^{1-\alpha_1} \sum_{j=1}^{N_P} u_j \int_{\partial K_P^*\cap\overline{\Omega}_1} (A_1(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x}))\cdot\boldsymbol{n} ds \\ &+ {}_0^R\mathrm{D}_t^{1-\alpha_2} \sum_{j=1}^{N_P} u_j \int_{\partial K_P^*\cap\overline{\Omega}_2} (A_2(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x}))\cdot\boldsymbol{n} ds \\ &+ \int_{K_P^*\cap\Omega_1} f_1(\boldsymbol{x},t)d\boldsymbol{x} + \int_{K_P^*\cap\Omega_2} f_2(\boldsymbol{x},t)d\boldsymbol{x}.\end{aligned} \tag{47}$$

Based on (46) and (47), we derive the following time-fractional system

$$\boldsymbol{M}\frac{d\boldsymbol{u}_h}{dt} + {}_0^R\mathrm{D}_t^{1-\alpha_1}(\boldsymbol{K}_1\boldsymbol{u}_h) + {}_0^R\mathrm{D}_t^{1-\alpha_2}(\boldsymbol{K}_2\boldsymbol{u}_h) = \boldsymbol{f}(t), \tag{48}$$

where $\boldsymbol{M}$ is the mass matrix, corresponding to the first order time derivative term, $\boldsymbol{K}_1, \boldsymbol{K}_2$ are the stiffness matrices derived from the first two terms on the right sides of (46) and (47), respectively, $\boldsymbol{f}(t)$ is the contribution from the source term, and $\boldsymbol{u}_h$ is a column vector of size N_P consisting of the nodal unknowns. Here are some details for the entries of the above matrices and vectors:

$$[\boldsymbol{K}_1]_{i,j} = -\int_{\partial K_{P_i}^*\cap\overline{\Omega}_1} (A_1(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x}))\cdot\boldsymbol{n}\psi_i(\boldsymbol{x})\, ds, \tag{49}$$

$$[\boldsymbol{K}_2]_{i,j} = -\int_{\partial K_{P_i}^*\cap\overline{\Omega}_2} (A_2(\boldsymbol{x})\nabla\phi_j(\boldsymbol{x}))\cdot\boldsymbol{n}\psi_i(\boldsymbol{x})\, ds, \tag{50}$$

$$[\boldsymbol{M}]_{i,j} = \langle\psi_i(\boldsymbol{x}),\phi_j(\boldsymbol{x})\rangle_\Omega, \quad [\boldsymbol{f}(t)]_i = \langle\psi_i(\boldsymbol{x}),f_1(\boldsymbol{x},t)\rangle_{\Omega_1} + \langle\psi_i(\boldsymbol{x}),f_2(\boldsymbol{x},t)\rangle_{\Omega_2}, \tag{51}$$

for $i,j = 1,\ldots,N_P$. Note that N_P is the number of nodes in the chosen triangular mesh.

Because of the relationship (10), Equation (48) is converted to

$$\boldsymbol{M}\frac{d\boldsymbol{u}_h}{dt} + {}_0^C\mathrm{D}_t^{1-\alpha_1}(\boldsymbol{K}_1\boldsymbol{u}_h) + {}_0^C\mathrm{D}_t^{1-\alpha_2}(\boldsymbol{K}_2\boldsymbol{u}_h) + \frac{t^{\alpha_1-1}}{\Gamma(\alpha_1)}\boldsymbol{K}_1\boldsymbol{u}_h^{(0)} + \frac{t^{\alpha_2-1}}{\Gamma(\alpha_2)}\boldsymbol{K}_2\boldsymbol{u}_h^{(0)} = \boldsymbol{f}(t). \tag{52}$$

5.1. A Fast Solver Based on Back Euler and the Fast L1 Algorithm. Applying the implicit Euler and L1 discretization to the previous equation, we obtain

$$\boldsymbol{M}\frac{\boldsymbol{u}_h^{(n)}-\boldsymbol{u}_h^{(n-1)}}{\tau_n}+\boldsymbol{K}_1\,\mathrm{D}_{L1}^{1-\alpha_1}\,\boldsymbol{u}_h^{(n)}+\boldsymbol{K}_2\,\mathrm{D}_{L1}^{1-\alpha_2}\,\boldsymbol{u}_h^{(n)}+\frac{t_n^{\alpha_1-1}}{\Gamma(\alpha_1)}\boldsymbol{K}_1\boldsymbol{u}_h^{(0)} + \frac{t_n^{\alpha_2-1}}{\Gamma(\alpha_2)}\boldsymbol{K}_2\boldsymbol{u}_h^{(0)}=\boldsymbol{f}^{(n)}, \tag{53}$$

where $\boldsymbol{f}^{(n)}=\boldsymbol{f}(t_n)$, and for $i=1,2$,

$$\mathrm{D}_{L1}^{1-\alpha_i}\,\boldsymbol{u}_h^{(n)}=\frac{1}{\Gamma(1+\alpha_i)}\sum_{k=1}^{n}\frac{\boldsymbol{u}_h^{(k+1)}-\boldsymbol{u}_h^{(k)}}{\tau_{k+1}}(\tau_{n,k}^{\alpha_i}-\tau_{n,k+1}^{\alpha_i}). \tag{54}$$

Furthermore, utilizing the L1 fast evaluation algorithm yields

$$\boldsymbol{M}\frac{\boldsymbol{u}_h^{(n)}-\boldsymbol{u}_h^{(n-1)}}{\tau_n}+\boldsymbol{K}_1\,\mathrm{D}_{FL1}^{1-\alpha_1}\,\boldsymbol{u}_h^{(n)}+\boldsymbol{K}_2\,\mathrm{D}_{FL1}^{1-\alpha_2}\,\boldsymbol{u}_h^{(n)}+\frac{t_n^{\alpha_1-1}}{\Gamma(\alpha_1)}\boldsymbol{K}_1\boldsymbol{u}_h^{(0)} + \frac{t_n^{\alpha_2-1}}{\Gamma(\alpha_2)}\boldsymbol{K}_2\boldsymbol{u}_h^{(0)}=\boldsymbol{f}^{(n)}. \tag{55}$$

Suppose θ_j,λ_j,N_1 and η_j,μ_j,N_2 are the reduced SOE parameters for approximation of t^{α_1-1} and t^{α_2-1}, respectively. Then we have, for $n\geq 2$,

$$\begin{cases}\mathrm{D}_{FL1}^{1-\alpha_1}\,\boldsymbol{u}_h^{(n)}=\dfrac{T^{\alpha_1-1}}{\Gamma(\alpha_1)}\displaystyle\sum_{j=1}^{N_1}\theta_j\mathbf{v}_{FL1}^{(n,j)}+\frac{1}{\Gamma(1+\alpha_1)}\frac{\boldsymbol{u}_h^{(n)}-\boldsymbol{u}_h^{(n-1)}}{\tau_n^{1-\alpha_1}},\\ \mathrm{D}_{FL1}^{1-\alpha_2}\,\boldsymbol{u}_h^{(n)}=\dfrac{T^{\alpha_2-1}}{\Gamma(\alpha_2)}\displaystyle\sum_{j=1}^{N_2}\eta_j\mathbf{w}_{FL1}^{(n,j)}+\frac{1}{\Gamma(1+\alpha_2)}\frac{\boldsymbol{u}_h^{(n)}-\boldsymbol{u}_h^{(n-1)}}{\tau_n^{1-\alpha_2}},\end{cases} \tag{56}$$

where $\mathbf{v}_{FL1}^{(n,j)},\mathbf{w}_{FL1}^{(n,j)}$ are auxiliary sequences of N_P-dimensional vectors that follow the following recurrence relations (or time-marching formulas)

$$\begin{cases}\mathbf{v}_{FL1}^{(n,j)}=e^{-\lambda_j\tau_n/T}\mathbf{v}_{FL1}^{(n-1,j)}+\dfrac{e^{-\lambda_j\tau_n/T}-e^{-\lambda_j\tau_{n,n-2}/T}}{\lambda_j\tau_{n-1}/T}\left(\boldsymbol{u}_h^{(n-1)}-\boldsymbol{u}_h^{(n-2)}\right),\\ \qquad\text{for } 1\leq j\leq N_1,\\ \mathbf{w}_{FL1}^{(n,j)}=e^{-\mu_j\tau_n/T}\mathbf{w}_{FL1}^{(n-1,j)}+\dfrac{e^{-\mu_j\tau_n/T}-e^{-\mu_j\tau_{n,n-2}/T}}{\mu_j\tau_{n-1}/T}\left(\boldsymbol{u}_h^{(n-1)}-\boldsymbol{u}_h^{(n-2)}\right),\\ \qquad\text{for } 1\leq j\leq N_2,\end{cases} \tag{57}$$

and by default (for $n=1$),

$$\mathbf{v}_{FL1}^{(1,j)}=\mathbf{0},\ \text{ for } 1\leq j\leq N_1,\qquad \mathbf{w}_{FL1}^{(1,j)}=\mathbf{0},\ \text{ for } 1\leq j\leq N_2. \tag{58}$$

As for $n=1$, we have,

$$\mathrm{D}_{FL1}^{1-\alpha_i}\,\boldsymbol{u}_h^{(1)}=\mathrm{D}_{L1}^{1-\alpha_i}\,\boldsymbol{u}_h^{(1)}=\frac{1}{\Gamma(1+\alpha_i)}\frac{\boldsymbol{u}_h^{(1)}-\boldsymbol{u}_h^{(0)}}{\tau_1^{1-\alpha_i}},\qquad i=1,2. \tag{59}$$

Combining (55), (56), and (59), we obtain a fast solver for the fully discrete system.

- For $n \geq 2$,

(60)
$$\Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_n^{\alpha_i}}{\Gamma(1+\alpha_i)} \boldsymbol{K}_i\Big)\boldsymbol{u}_h^{(n)} = \Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_n^{\alpha_i}}{\Gamma(1+\alpha_i)} \boldsymbol{K}_i\Big)\boldsymbol{u}_h^{(n-1)} + \tau_n \boldsymbol{f}^{(n)}$$
$$-\tau_n \frac{T^{\alpha_1-1}}{\Gamma(\alpha_1)} \sum_{j=1}^{N_1} \theta_j \boldsymbol{K}_1 \mathbf{v}_{FL1}^{(n,j)} - \tau_n \frac{T^{\alpha_2-1}}{\Gamma(\alpha_2)} \sum_{j=1}^{N_2} \eta_j \boldsymbol{K}_2 \mathbf{w}_{FL1}^{(n,j)} - \tau_n \sum_{i=1}^{2} \frac{t_n^{\alpha_i-1}}{\Gamma(\alpha_i)} \boldsymbol{K}_i \boldsymbol{u}_h^{(0)}.$$

  Here $\tau_n^{\alpha_i}\boldsymbol{K}_i$ reflects ***subdiffusion***, whereas the first two terms in the 2nd line demonstrate ***facilitation*** by the auxiliary sequences $\mathbf{v}_{FL1}^{(n,j)}, \mathbf{w}_{FL1}^{(n,j)}$.

- For $n = 1$,

(61)
$$\Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_1^{\alpha_i}}{\Gamma(1+\alpha_i)} \boldsymbol{K}_i\Big)\boldsymbol{u}_h^{(1)} = \Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_1^{\alpha_i}}{\Gamma(1+\alpha_i)} \boldsymbol{K}_i\Big)\boldsymbol{u}_h^{(0)}$$
$$-\tau_1 \sum_{i=1}^{2} \frac{t_1^{\alpha_i-1}}{\Gamma(\alpha_i)} \boldsymbol{K}_i \boldsymbol{u}_h^{(0)} + \tau_1 \boldsymbol{f}^{(1)}.$$

5.2. A Fast Solver Combining Crank-Nicolson Discretization and the Fast L2 Algorithm. Now we combine the Crank-Nicolson discretization and the fast L2 evaluation algorithm to obtain

(62)
$$\boldsymbol{M}\boldsymbol{u}_h^{(n)} = \boldsymbol{M}\boldsymbol{u}_h^{(n-1)} + \tau_n \sum_{i=1}^{2} \boldsymbol{K}_i \, \mathrm{D}_{L2}^{1-\alpha_i} \, \boldsymbol{u}_h^{(n-\frac{1}{2})}$$
$$-\tau_n \sum_{i=1}^{2} \frac{t_{n-\frac{1}{2}}^{\alpha_i-1}}{\Gamma(\alpha_i)} \boldsymbol{K}_i \boldsymbol{u}_h^{(0)} + \tau_n \boldsymbol{f}^{(n-\frac{1}{2})},$$

where for $i = 1, 2$,

(63)
$$\begin{cases} \mathrm{D}_{L2}^{1-\alpha_i} \, \boldsymbol{u}_h^{(n-\frac{1}{2})} = \dfrac{1}{\Gamma(\alpha_i)} \left(a_n^{(\alpha_i)} \boldsymbol{u}_h^{(n-2)} + b_n^{(\alpha_i)} \boldsymbol{u}_h^{(n-1)} + c_n^{(\alpha_i)} \boldsymbol{u}_h^{(n)}\right) \\ \qquad\qquad + \dfrac{1}{\Gamma(\alpha_i)} \displaystyle\sum_{k=1}^{n-1} \left(a_{n,k}^{(\alpha_i)} \boldsymbol{u}_h^{(k-1)} + b_{n,k}^{(\alpha_i)} \boldsymbol{u}_h^{(k)} + c_{n,k}^{(\alpha_i)} \boldsymbol{u}_h^{(k+1)}\right), \quad \forall 2 \leq n \leq N_T, \\ \mathrm{D}_{L2}^{1-\alpha_i} \, \boldsymbol{u}_h^{(\frac{1}{2})} = \dfrac{\tau_1^{\alpha_i-1}}{2^{\alpha_i}\Gamma(1+\alpha_i)} (\boldsymbol{u}_h^{(1)} - \boldsymbol{u}_h^{(0)}). \end{cases}$$

The direct L2 terms can be replaced by the fast L2 evaluation formulas and we end up with

(64)
$$\boldsymbol{M} \frac{\boldsymbol{u}_h^{(n)} - \boldsymbol{u}_h^{(n-1)}}{\tau_n} + \sum_{i=1}^{2} \boldsymbol{K}_i \, \mathrm{D}_{FL2}^{1-\alpha_i} \, \boldsymbol{u}_h^{(n-\frac{1}{2})}$$
$$= -\sum_{i=1}^{2} \frac{t_{n-\frac{1}{2}}^{\alpha_i-1}}{\Gamma(\alpha_i)} \boldsymbol{K}_i \boldsymbol{u}_h^{(0)} + \boldsymbol{f}^{(n-\frac{1}{2})}.$$

Let θ_j, λ_j, N_1 and η_j, μ_j, N_2 be the reduced SOE parameters for approximation of t^{α_1-1} and t^{α_2-1}, respectively. Then we have (for $n = 1$),

(65)
$$\mathrm{D}_{FL2}^{1-\alpha_i} \, \boldsymbol{u}_h^{(1)} = \mathrm{D}_{L2}^{1-\alpha_i} \, \boldsymbol{u}_h^{(1)} = \frac{\tau_1^{\alpha_i-1}}{2^{\alpha_i}\Gamma(1+\alpha_i)} (\boldsymbol{u}_h^{(1)} - \boldsymbol{u}_h^{(0)}), \quad i = 1, 2.$$

But for $n \geq 2$,

$$
\left\{
\begin{aligned}
\mathrm{D}_{FL2}^{1-\alpha_1}\, \boldsymbol{u}_h^{(n)} &= \frac{T^{\alpha_1-1}}{\Gamma(\alpha_1)} \sum_{j=1}^{N_1} \theta_j \mathbf{v}_{FL2}^{(n,j)} \\
&\quad + \frac{1}{\Gamma(\alpha_1)} \left(a_n^{(\alpha_1)} \boldsymbol{u}_h^{(n-2)} + b_n^{(\alpha_1)} \boldsymbol{u}_h^{(n-1)} + c_n^{(\alpha_1)} \boldsymbol{u}_h^{(n)} \right), \\
\mathrm{D}_{FL2}^{1-\alpha_2}\, \boldsymbol{u}_h^{(n)} &= \frac{T^{\alpha_2-1}}{\Gamma(\alpha_2)} \sum_{j=1}^{N_2} \eta_j \mathbf{w}_{FL2}^{(n,j)} \\
&\quad + \frac{1}{\Gamma(\alpha_2)} \left(a_n^{(\alpha_2)} \boldsymbol{u}_h^{(n-2)} + b_n^{(\alpha_2)} \boldsymbol{u}_h^{(n-1)} + c_n^{(\alpha_2)} \boldsymbol{u}_h^{(n)} \right),
\end{aligned}
\right. \tag{66}
$$

where the auxiliary sequences satisfy the following recurrence formulas, for $n \geq 2$,

$$
\left\{
\begin{aligned}
\mathbf{v}_{FL2}^{(n,j)} &= e^{-\lambda_j(\tau_n+\tau_{n-1})/(2T)} \mathbf{v}_{FL2}^{(n-1,j)} + A_j^{(\alpha_1)} \boldsymbol{u}_h^{(n-2)} + B_j^{(\alpha_1)} \boldsymbol{u}_h^{(n-1)} + C_j^{(\alpha_1)} \boldsymbol{u}_h^{(n)}, \\
\mathbf{w}_{FL2}^{(n,j)} &= e^{-\mu_j(\tau_n+\tau_{n-1})/(2T)} \mathbf{w}_{FL2}^{(n-1,j)} + A_j^{(\alpha_2)} \boldsymbol{u}_h^{(n-2)} + B_j^{(\alpha_2)} \boldsymbol{u}_h^{(n-1)} + C_j^{(\alpha_2)} \boldsymbol{u}_h^{(n)}.
\end{aligned}
\right. \tag{67}
$$

For $n \geq 2$, let

$$
\left\{
\begin{aligned}
\overline{\mathbf{v}}_{FL2}^{(n,j)} &= e^{-\lambda_j(\tau_n+\tau_{n-1})/(2T)} \mathbf{v}_{FL2}^{(n-1,j)} + A_j^{(\alpha_1)} \boldsymbol{u}_h^{(n-2)} + B_j^{(\alpha_1)} \boldsymbol{u}_h^{(n-1)}, \\
\overline{\mathbf{w}}_{FL2}^{(n,j)} &= e^{-\mu_j(\tau_n+\tau_{n-1})/(2T)} \mathbf{w}_{FL2}^{(n-1,j)} + A_j^{(\alpha_2)} \boldsymbol{u}_h^{(n-2)} + B_j^{(\alpha_2)} \boldsymbol{u}_h^{(n-1)},
\end{aligned}
\right. \tag{68}
$$

we have

$$
\left\{
\begin{aligned}
\mathbf{v}_{FL2}^{(n,j)} &= C_j^{(\alpha_1)} \boldsymbol{u}_h^{(n)} + \overline{\mathbf{v}}_{FL2}^{(n,j)}, \\
\mathbf{w}_{FL2}^{(n,j)} &= C_j^{(\alpha_2)} \boldsymbol{u}_h^{(n)} + \overline{\mathbf{w}}_{FL2}^{(n,j)}.
\end{aligned}
\right. \tag{69}
$$

For $n = 1$, by default,

$$
\mathbf{v}_{FL2}^{(1,j)} = \overline{\mathbf{v}}_{FL2}^{(1,j)} = \mathbf{0}, \ \text{ for } 1 \leq j \leq N_1; \qquad \mathbf{w}_{FL2}^{(1,j)} = \overline{\mathbf{w}}_{FL2}^{(1,j)} = \mathbf{0}, \ \text{ for } 1 \leq j \leq N_2. \tag{70}
$$

Combining (64), (65), and (66), we obtain a fast solver expressed as a time-marching scheme.

- For $n = 1$,

$$
\begin{aligned}
\Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_1^{\alpha_i-1}}{2^{\alpha_i}\Gamma(1+\alpha_i)} \boldsymbol{K}_i \Big) \boldsymbol{u}_h^{(1)} &= \Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_1^{\alpha_i-1}}{2^{\alpha_i}\Gamma(1+\alpha_i)} \boldsymbol{K}_i \Big) \boldsymbol{u}_h^{(0)} \\
&\quad - \tau_1 \sum_{i=1}^{2} \frac{t_{\frac{1}{2}}^{\alpha_i-1}}{\Gamma(\alpha_i)} \boldsymbol{K}_i \boldsymbol{u}_h^{(0)} + \tau_1 \boldsymbol{f}^{(\frac{1}{2})},
\end{aligned} \tag{71}
$$

- For $n \geq 2$,

(72)
$$\begin{aligned}
&\Big(\boldsymbol{M} + \sum_{i=1}^{2} \frac{\tau_n c_n^{(\alpha_i)}}{\Gamma(\alpha_i)} \boldsymbol{K}_i + \frac{\tau_n T^{\alpha_1 - 1}}{\Gamma(\alpha_1)} \boldsymbol{K}_1 \sum_{j=1}^{N_1} \theta_j C_j^{(\alpha_1)} + \frac{\tau_n T^{\alpha_2 - 1}}{\Gamma(\alpha_2)} \boldsymbol{K}_2 \sum_{j=1}^{N_2} \eta_j C_j^{(\alpha_2)}\Big) \boldsymbol{u}_h^{(n)} \\
&= \boldsymbol{M} \boldsymbol{u}_h^{(n-1)} - \sum_{i=1}^{2} \frac{\tau_n}{\Gamma(\alpha_i)} \boldsymbol{K}_i \Big(a_n^{(\alpha_i)} \boldsymbol{u}_h^{(n-2)} + b_n^{(\alpha_i)} \boldsymbol{u}_h^{(n-1)} \Big) \\
&- \tau_n \frac{T^{\alpha_1 - 1}}{\Gamma(\alpha_1)} \boldsymbol{K}_1 \sum_{j=1}^{N_1} \theta_j \overline{\mathbf{v}}_{FL2}^{(n,j)} - \tau_n \frac{T^{\alpha_2 - 1}}{\Gamma(\alpha_2)} \boldsymbol{K}_2 \sum_{j=1}^{N_2} \eta_j \overline{\mathbf{w}}_{FL2}^{(n,j)} \\
&- \tau_n \sum_{i=1}^{2} \frac{t_{n-\frac{1}{2}}^{\alpha_i - 1}}{\Gamma(\alpha_i)} \boldsymbol{K}_i \boldsymbol{u}_h^{(0)} + \tau_n \boldsymbol{f}^{(n-\frac{1}{2})},
\end{aligned}$$

where $\mathbf{v}_{FL2}^{(n,j)}, \mathbf{w}_{FL2}^{(n,j)}$ are the auxiliary sequences defined in (67) and (70). Observations on subdiffusion and facilitation by auxiliary sequences stated right after formula (60) apply here as well.

6. Stability Analysis

This section focuses on stability analysis of the direct/fast L1 + back-Euler solver with the time mesh grading parameter $r = 1$. Here we consider a simplified model

$$\partial_t u = {}_0^R \mathrm{D}_t^{1-\alpha_i} \nabla \cdot (A_i \nabla u) + f_i(\boldsymbol{x}, t) \quad \text{in } \Omega_i \times (0, T], \quad i = 1, 2, \tag{73}$$

subject to the zero initial condition

$$u(\boldsymbol{x}, 0) = 0, \quad \boldsymbol{x} \in \Omega, \tag{74}$$

where the diffusion tensors $A_i, i = 1, 2$ are constants.

Recall the direct/fast L1 discretization operator with a uniform time partition: For $1 \leq n \leq N_T$,

$$\begin{aligned}
\mathrm{D}_{L1}^{1-\alpha} u^{(n)} &= \frac{1}{\Gamma(1+\alpha)} \sum_{k=1}^{n} \frac{u^{(k)} - u^{(k-1)}}{\tau} ((t_n - t_{k-1})^{\alpha} - (t_n - t_k)^{\alpha}) \\
&= \frac{1}{\Gamma(1+\alpha)} \sum_{k=1}^{n} \frac{u^{(k)} - u^{(k-1)}}{\tau^{1-\alpha}} \left((n-k+1)^{\alpha} - (n-k)^{\alpha} \right),
\end{aligned} \tag{75}$$

$$\begin{aligned}
\mathrm{D}_{FL1}^{1-\alpha} u^{(n)} &= \frac{T^{\alpha-1}}{\tau \Gamma(\alpha)} \int_{t_{n-1}}^{t_n} \left(\frac{t_n - s}{T} \right)^{\alpha - 1} ds \left(u^{(n)} - u^{(n-1)} \right) \\
&+ \frac{T^{\alpha-1}}{\tau \Gamma(\alpha)} \sum_{k=1}^{n-1} \int_{t_{k-1}}^{t_k} \sum_{j=1}^{N_{exp}} \theta_j e^{-\lambda_j (t_n - s)/T} ds \left(u^{(k)} - u^{(k-1)} \right),
\end{aligned} \tag{76}$$

where τ is the time step size. We prove the positive-definiteness of this operator.

Theorem 1. *Consider the operator $\mathrm{D}_{L1}^{1-\alpha}$ defined above with $\alpha \in (0, 1)$. For any positive integer N_T and a real vector $[g^{(0)}, g^{(1)}, \ldots, g^{(N_T)}] \in \mathcal{R}^{N_T+1}$ with $g^{(0)} = 0$, there holds*

$$\sum_{n=1}^{N_T} \left(\mathrm{D}_{L1}^{1-\alpha} g^{(n)} \right) g^{(n)} \geq 0. \tag{77}$$

Proof. To simplify writing, we denote

$$b_j = (j+1)^\alpha - j^\alpha, \quad j \geq 0, \tag{78}$$

$$\omega_0 = b_0, \qquad \omega_k = -(b_{k-1} - b_k), \quad k \geq 1. \tag{79}$$

For $n = 0$, let

$$\mathrm{D}_{L1}^{1-\alpha} g^{(0)} = w_0 g^{(0)} = 0. \tag{80}$$

For $n \geq 1$, applying Formula (75) and the fact that $g^{(0)} = 0$, we obtain

$$\mathrm{D}_{L1}^{1-\alpha} g^{(n)} = \frac{1}{\Gamma(1+\alpha)\tau^{1-\alpha}} \sum_{k=0}^{n} \omega_k g^{(n-k)}. \tag{81}$$

Then the following equation holds

$$\sum_{n=1}^{N_T} \left(\mathrm{D}_{L1}^{1-\alpha} g^{(n)} \right) g^{(n)} = \sum_{n=0}^{N_T} \left(\mathrm{D}_{L1}^{1-\alpha} g^{(n)} \right) g^{(n)} = \sum_{n=0}^{N_T} \left(\sum_{k=0}^{n} \omega_k g^{(n-k)} \right) g^{(n)}. \tag{82}$$

One can verify that validity of (77) is equivalent to proof of the positive-definiteness of matrix

$$W = \begin{pmatrix} \omega_0 & \frac{\omega_1}{2} & \frac{\omega_2}{2} & \cdots & \frac{\omega_{N_T}}{2} \\ \frac{\omega_1}{2} & \omega_0 & \frac{\omega_1}{2} & \ddots & \vdots \\ \frac{\omega_2}{2} & \frac{\omega_1}{2} & \ddots & \ddots & \frac{\omega_2}{2} \\ \vdots & \ddots & \ddots & \omega_0 & \frac{\omega_1}{2} \\ \frac{\omega_{N_T}}{2} & \cdots & \frac{\omega_2}{2} & \frac{\omega_1}{2} & \omega_0 \end{pmatrix}. \tag{83}$$

It is straightforward to see

$$\begin{gathered} b_j > 0, \quad j \geq 0, \\ 1 = b_0 > b_1 > \cdots > b_k > b_{k+1} > \cdots \end{gathered} \tag{84}$$

and the following property holds

$$\sum_{k=0}^{N_T} \omega_k = b_{N_T} > 0, \qquad \omega_0 = 1, \qquad \omega_k < 0, \;\; k \geq 1. \tag{85}$$

Finally, we obtain

$$w_0 + \sum_{k=1}^{N_T} \frac{w_k}{2} > 0, \tag{86}$$

which implies W is positive-definite and the inequality (77) holds. □

Theorem 2. *Let $\mathrm{D}_{FL1}^{1-\alpha}$ be the fast L1 discretization operator with a uniform partition. Assume the relative error of the reduced SOE approximation ε is sufficiently small. Then for any positive integer N_T and a real vector $[g^{(0)}, g^{(1)}, \ldots, g^{(N_T)}] \in \mathcal{R}^{N_T+1}$ with $g^{(0)} = 0$, there holds*

$$\sum_{n=1}^{N_T} \left(\mathrm{D}_{FL1}^{1-\alpha} g^{(n)} \right) g^{(n)} \geq 0, \quad \forall \alpha \in (0,1). \tag{87}$$

Proof. It is similar to that of Theorem 1. According to Formula (76), the fast L1 discretization operator $\mathrm{D}_{FL1}^{1-\alpha}$ can be formulated in a manner analogous to Equation (81)

$$\mathrm{D}_{FL1}^{1-\alpha} g^{(n)} = \frac{T^{\alpha-1}}{\tau\Gamma(\alpha)} \sum_{k=0}^{n} \overline{\omega}_k g^{(n-k)}. \tag{88}$$

For sufficiently small ε, we apply the Mean Value Theorem to obtain

$$\sum_{k=0}^{N_T} \overline{\omega}_k > 0, \qquad \overline{\omega}_0 > 0, \qquad \overline{\omega}_k < 0, \;\; k \geq 1. \tag{89}$$

This implies the inequality (87), as expected. □

For the spatial discretization based on finite volumes, we present here new formulations of the direct/fast L1 + back-Euler solvers that are equivalent to the previous ones but more convenient for analysis. For any test function ψ_P (characteristic function on K_P^*), we define

$$\begin{aligned} \mathcal{A}(\mathrm{D}_M^{1-\boldsymbol{\alpha}} u_h^{(n)}, \psi_P) = &\int_{\partial(K_P^* \cap \Omega_1)} A_1 \left(\mathrm{D}_M^{1-\alpha_1} \nabla u_h^{(n)} \right) \cdot \boldsymbol{n} \, ds \\ &+ \int_{\partial(K_P^* \cap \Omega_2)} A_2 \left(\mathrm{D}_M^{1-\alpha_2} \nabla u_h^{(n)} \right) \cdot \boldsymbol{n} \, ds, \quad M \in \{L1, FL1\}, \end{aligned} \tag{90}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$. As for $P \in \Gamma_{int}$, we use the interface conditions to obtain

$$\begin{aligned} \mathcal{A}(\mathrm{D}_M^{1-\boldsymbol{\alpha}} u_h^{(n)}, \psi_P) = &\int_{\partial K_P^* \cap \overline{\Omega}_1} A_1 \left(\mathrm{D}_M^{1-\alpha_1} \nabla u_h^{(n)} \right) \cdot \boldsymbol{n} \, ds \\ &+ \int_{\partial K_P^* \cap \overline{\Omega}_2} A_2 \left(\mathrm{D}_M^{1-\alpha_2} \nabla u_h^{(n)} \right) \cdot \boldsymbol{n} \, ds, \quad M \in \{L1, FL1\}. \end{aligned}$$

Let Π_h^* be the projection operator from U_h to V_h defined as

$$\Pi_h^* u_h = \sum_{P \in \mathcal{P}_h} u_h(P) \psi_P. \tag{91}$$

The numerical solvers (53)-(55) for the simplified model (73)(74) read as: Seek $u_h \in U_h$ such that, $\forall\, v_h \in U_h, M \in \{L1, FL1\}$,

$$\left(\frac{u_h^{(n)} - u_h^{(n-1)}}{\tau}, \Pi_h^* v_h \right) + \mathcal{A}(\mathrm{D}_M^{1-\boldsymbol{\alpha}} u_h^{(n)}, \Pi_h^* v_h) = (f^{(n)}, \Pi_h^* v_h), \tag{92}$$

where $f^{(n)} = f(\boldsymbol{x}, t_n) = \begin{cases} f_1(\boldsymbol{x}, t_n), & \forall \boldsymbol{x} \in \Omega_1, \\ f_2(\boldsymbol{x}, t_n), & \forall \boldsymbol{x} \in \Omega_2. \end{cases}$

Theorem 3. *For any $u_h \in U_h$, the aforementioned finite volume form satisfies*

$$\sum_{n=0}^{N_T} \mathcal{A}(\mathrm{D}_M^{1-\boldsymbol{\alpha}} u_h^{(n)}, \Pi_h^* u_h^{(n)}) \geq 0, \quad M \in \{L1, FL1\}. \tag{93}$$

Proof. Firstly, we define another numerical form as follows. For $u_h, v_h \in U_h$,

$$\begin{aligned} \mathcal{A}'(\mathrm{D}_M^{1-\boldsymbol{\alpha}} u_h^{(n)}, v_h) = &\int_{\Omega_1} A_1 \left(\mathrm{D}_M^{1-\alpha_1} \nabla u_h^{(n)} \right) \cdot \nabla v_h dX \\ &+ \int_{\Omega_2} A_2 \left(\mathrm{D}_M^{1-\alpha_2} \nabla u_h^{(n)} \right) \cdot \nabla v_h dX. \end{aligned} \tag{94}$$

Then we have

$$\begin{aligned}\sum_{n=0}^{N_T}\mathcal{A}'(\mathrm{D}_M^{1-\boldsymbol{\alpha}}u_h^{(n)},u_h^{(n)}) &= \sum_{n=0}^{N_T}\int_{\Omega_1}A_1\left(\mathrm{D}_M^{1-\alpha_1}\nabla u_h^{(n)}\right)\cdot\nabla u_h^{(n)}dX\\ &+\sum_{n=0}^{N_T}\int_{\Omega_2}A_2\left(\mathrm{D}_M^{1-\alpha_2}\nabla u_h^{(n)}\right)\cdot\nabla u_h^{(n)}dX\\ &=: I_1+I_2.\end{aligned}\tag{95}$$

According to Theorems 1 and 2, it is easy to verify that $I_1\geq 0, I_2\geq 0$. Thus the following estimate holds

$$\sum_{n=0}^{N_T}\mathcal{A}'(\mathrm{D}_M^{1-\boldsymbol{\alpha}}u_h^{(n)},u_h^{(n)})\geq 0.\tag{96}$$

For the constant diffusion coefficients A_1, A_2, the equation $\mathcal{A}'(\mathrm{D}_M^{1-\boldsymbol{\alpha}}u_h^{(n)},v_h) = \mathcal{A}(\mathrm{D}_M^{1-\boldsymbol{\alpha}}u_h^{(n)},\Pi_h^*v_h)$ holds, see [22]. Finally, we obtain

$$\sum_{n=0}^{N_T}\mathcal{A}(\mathrm{D}_M^{1-\boldsymbol{\alpha}}u_h^{(n)},\Pi_h^*u_h^{(n)}) = \sum_{n=0}^{N_T}\mathcal{A}'(\mathrm{D}_M^{1-\boldsymbol{\alpha}}u_h^{(n)},u_h^{(n)})\geq 0,\tag{97}$$

as claimed. □

Theorem 4. *The direct L1 + back-Euler solver for the simplified model (73)(74) with a uniform temporal partition is unconditionally stable and the numerical solution u_h satisfies*

$$||u_h^{(N_T)}||_0\leq C\tau\sum_{n=1}^{N_T}||f^{(n)}||_0.\tag{98}$$

Proof. For the projection operator Π_h^*, we have the following properties (see [22])

$$(u_h,\Pi_h^*\overline{u}_h) = (\overline{u}_h,\Pi_h^*u_h)\,,\quad \forall u_h,\overline{u}_h\in U_h,\tag{99}$$

$$c_1||u_h||_0\leq (u_h,\Pi_h^*u_h)^{\frac{1}{2}}\leq c_2||u_h||_0,\quad \forall u_h\in U_h.\tag{100}$$

Setting $v_h = 2\Pi_h^*u_h^{(n)}$ in equation (92) with $M = L1$, we get the following equation

$$\left(u_h^{(n)}-u_h^{(n-1)},2\Pi_h^*u_h^{(n)}\right)+2\tau\mathcal{A}(\mathrm{D}_{L1}^{1-\boldsymbol{\alpha}}u_h^{(n)},\Pi_h^*u_h^{(n)}) = 2\tau(f^{(n)},\Pi_h^*u_h^{(n)}).\tag{101}$$

According to the properties stated in (99)(100), we know

$$\begin{aligned}\left(u_h^{(n)}-u_h^{(n-1)},2\Pi_h^*u_h^{(n)}\right) &= \left(u_h^{(n)},\Pi_h^*u_h^{(n)}\right)-\left(u_h^{(n-1)},\Pi_h^*u_h^{(n-1)}\right)\\ &+c||u_h^{(n)}-u_h^{(n-1)}||_0^2,\end{aligned}\tag{102}$$

where $c>0$ is a constant. Summing from $n=1$ to N_T and using Theorem 3 with $M = L1$, we obtain

$$\left(u_h^{(N_T)},\Pi_h^*u_h^{(N_T)}\right)-\left(u_h^{(0)},\Pi_h^*u_h^{(0)}\right)\leq 2\tau\sum_{n=1}^{N_T}(f^{(n)},\Pi_h^*u_h^{(n)}).\tag{103}$$

Choose n^* such that $||u_h^{(n^*)}||_0 = \max_{1\le n\le N_T} ||u_h^{(n)}||_0$. Then we have

$$\begin{aligned}||u_h^{(n^*)}||_0^2 \le& C\left(||u_h^{(0)}||_0^2 + 2\tau\sum_{n=1}^{n^*}||f^{(n)}||_0||u_h^{(n)}||_0\right)\\ \le& C||u_h^{(n^*)}||_0\left(||u_h^{(0)}||_0 + 2\tau\sum_{n=1}^{N_T}||f^{(n)}||_0\right).\end{aligned}$$

Finally, the numerical solution satisfies

$$||u_h^{(N_T)}||_0 \le ||u_h^{(n^*)}||_0 \le C\left(||u_h^{(0)}||_0 + 2\tau\sum_{n=1}^{N_T}||f^{(n)}||_0\right) = C\tau\sum_{n=1}^{N_T}||f^{(n)}||_0, \tag{104}$$

and therefore, the numerical scheme is unconditionally stable. □

Corollary 1. *The fast L1 + back-Euler solver for the simplified model (73)(74) with a uniform temporal partition is unconditionally stable and the numerical solution u_h satisfies*

$$||u_h^{(N_T)}||_0 \le C\tau\sum_{n=1}^{N_T}||f^{(n)}||_0. \tag{105}$$

Remark 2. *For more general cases with a nonzero initial condition and/or a graded time mesh, $\sum_{n=1}^{N_T}\tau_n\mathcal{A}(\mathrm{D}_{L1}^{1-\boldsymbol{\alpha}}u_h^{(n)}, \Pi_h^* u_h^{(n)}) \ge 0$ needs to be proved if the current approach is adopted. However, this property may not hold in general. New approaches or techniques need to be developed for future work.*

7. Implementation and Computational Complexity of the Fast Solvers

This section provides some details on implementation and computational complexity of our fast solvers. For ease of presentation, we focus on the fast L1 + back-Euler solver.

7.1. Implementation of the Fast Solvers. Algorithm 1 shown below can be efficiently executed via vectorial operations in `Matlab` or `C/C++`. The linear systems in (60)(61) can be solved using `Matlab` built-in solvers or those in popular `C/C++` libraries.

Remark 3. *For time-fractional problems, graded temporal meshes were proposed in* [18, 35]. *In particular, for L1 temporal discretization, the optimal grading parameter is $r = \frac{2-\alpha}{\alpha}$ (with $\alpha \in (0,1)$ being the order of the Caputo derivative), see* [35] *(Remark 5.6 therein). For L2 temporal discretization, the optimal grading parameter is $r = \frac{3-\alpha}{\alpha}$, see* [18] *(Remark 4.3 therein). Following these guidelines, we set $\alpha_{min} = \min(\alpha_1, \alpha_2)$. Then we set $r = (2-\alpha_{min})/\alpha_{min}$ for L1 discretization, but $r = (3-\alpha_{min})/\alpha_{min}$ for L2 discretization. These choices are used in our numerical experiments (reported in Section 8), which indeed demonstrate the desired convergence rates.*

Remark 4. *To ensure stability of the L2 formula on a graded mesh, we usually adopt a modified L2 formula. Given a positive integer N_0,*

- *If $n \le N_0$,*

$$\mathrm{D}_{L2}^{1-\alpha}\, u^{(n-\frac{1}{2})} = \frac{1}{\Gamma(\alpha)}\Big(\int_{t_{n-1}}^{t_{n-\frac{1}{2}}}\frac{(\Pi_{1,n}u)'(s)}{(t_{n-\frac{1}{2}}-s)^{1-\alpha}}ds + \sum_{k=1}^{n-1}\int_{t_{k-1}}^{t_k}\frac{(\Pi_{1,k}u)'(s)}{(t_{n-\frac{1}{2}}-s)^{1-\alpha}}ds\Big), \tag{106}$$

Algorithm 1 Fast L1 + back-Euler solver

1: Generate a triangular mesh and its corresponding dual mesh on the computational domain. Good-quality triangular meshes can be generated by `distmesh2d` (a `Matlab` package) [31].
2: Compute the mass matrix $\boldsymbol{M}$ and stiffness matrices $\boldsymbol{K}_1, \boldsymbol{K}_2$ in formulas (49), (50), and (51). The 1-point quadrature for triangles and dual edges is accurate enough for calculation of the integrals in the mass matrix and stiffness matrices.
3: For temporal discretization, choose an integer $N_T > 0$ as the number of steps, a value for the grading parameter r, an expected accuracy $\varepsilon = (1/N_T)^2$, and a cut-off time $\sigma = \tau_2/T$.
4: Determine the parameter values of $N_{exp}, \lambda_j, \theta_j$ for reduced SOE.
5: Compute the numerical solution $\boldsymbol{u}_h^{(1)}$ using formula (61). Similarly, the 1-point quadrature can be used to compute $\boldsymbol{f}^{(1)}$.
6: **for** $n = 2, 3, \cdots, N_T$ **do**
7: Compute two auxiliary sequences of vectors $\mathbf{v}_{FL1}^{(n,j)} (1 \le j \le N_1)$ and $\mathbf{w}_{FL1}^{(n,j)} (1 \le j \le N_2)$ using Fomula (57).

$$\begin{cases} \mathbf{v}_{FL1}^{(n,j)} & \longleftarrow & \{\mathbf{v}_{FL1}^{(n-1,j)},\ \boldsymbol{u}_h^{(n-1)},\ \boldsymbol{u}_h^{(n-2)}\} \\ \mathbf{w}_{FL1}^{(n,j)} & \longleftarrow & \{\mathbf{w}_{FL1}^{(n-1,j)},\ \boldsymbol{u}_h^{(n-1)},\ \boldsymbol{u}_h^{(n-2)}\} \end{cases}$$

8: Compute the numerical solution $\boldsymbol{u}_h^{(n)}$ using Formula (60)

$$\boldsymbol{u}_h^{(n)} \quad \longleftarrow \quad \{\mathbf{v}_{FL1}^{(n,j)},\ \mathbf{w}_{FL1}^{(n,j)},\ \boldsymbol{u}_h^{(n-1)}\}.$$

9: **end for**

- *If* $n > N_0$,

(107)

$$\mathrm{D}_{L2}^{1-\alpha} u^{(n-\frac{1}{2})} = \frac{1}{\Gamma(\alpha)} \left(\int_{t_{n-1}}^{t_{n-\frac{1}{2}}} \frac{(\Pi_{2,n-1} u)'(s)}{(t_{n-\frac{1}{2}} - s)^{1-\alpha}} ds + \sum_{k=N_0+1}^{n-1} \int_{t_{k-1}}^{t_k} \frac{(\Pi_{2,k} u)'(s)}{(t_{n-\frac{1}{2}} - s)^{1-\alpha}} ds \right)$$
$$+ \frac{1}{\Gamma(\alpha)} \left(\sum_{k=1}^{N_0} \int_{t_{k-1}}^{t_k} \frac{(\Pi_{1,k} u)'(s)}{(t_{n-\frac{1}{2}} - s)^{1-\alpha}} ds \right).$$

Note N_0 is chosen as a small positive integer (2 or 3) in actual calculations. The default 16-digit precision of `Matlab` is insufficient to obtain satisfactory numerical results for the direct/fast L2 + Crank-Nicolson solver. The specific package "variable-precision arithmetic", namely, `vpa`, has been used for improvement of calculation accuracy.

7.2. Efficiency of the Fast Solvers. We focus on the solver that combines the back-Euler and fast L1 algorithm. Consider two subdomains with subdiffusion exponents α_1, α_2. We denote

$$\alpha_{min} = \min(\alpha_1, \alpha_2), \qquad \alpha_{max} = \max(\alpha_1, \alpha_2).$$

Recall that N_T (first defined in Section 3.1) is the number of temporal partitions. It is known from the discussion in Section 3 that we need to set the error $\varepsilon \le \mathcal{O}((T/N_T)^{1+\alpha_{min}})$. But for practical computation, we choose $\varepsilon = (1/N_T)^2$ to obtain better accuracy. For the (fast L1 + back-Euler) solver, the number of terms needed

for the reduced SOE approximation follows

(108)

$$\begin{aligned} N_{max} =& \max\{N_1, N_2\} \\ \le& 10^{-1}(2\log\varepsilon^{-1}+2)\Big(\log N_T + (1-\alpha_{max})^{-1}\log(\varepsilon^{-1}) + \log\log\varepsilon^{-1} + 2\Big). \end{aligned}$$

For $\varepsilon = (1/N_T)^2$, we obtain

$$N_{max} \le \mathcal{O}((\log N_T)^2).$$

The fast L1 evaluation algorithm for the Caputo derivative requires $\mathcal{O}(N_P)$ operations for $\mathbf{v}_{FL1}^{(n,j)}$ for each fixed n (a step in time-marching) and each fixed j (one term in the reduced SOE) by the recurrence relation formula. The back Euler method needs $\mathcal{O}(1)$ for each time step. Recall the number of spatial nodes is N_P. So the entire fast solver requires only

$$\mathcal{O}\Big(N_P N_T (\log N_T)^2\Big), \qquad \mathcal{O}\Big(N_P (\log N_T)^2\Big), \tag{109}$$

for operations and memory/storage, respectively.

As comparison, we point out that the (direct L1 + back-Euler) solver needs $\mathcal{O}(N_P\, N_T^2)$ for operations and $\mathcal{O}(N_P N_T)$ for memory/storage.

8. Numerical Experiments

This section presents numerical results for two examples to demonstrate efficiency and accuracy of the fast solvers. Since no exact solutions are known in the elementary form, we use instead reference (numerical) solutions on very fine meshes for computation of errors.

All numerical results below are obtained by using `Matlab R2023a` on a laptop with AMD Ryzen 5 5600H CPU@3.30GHz and 16GB RAM.

Example 1. For this 1d problem, the two subdomains are $\Omega_1 = (0, 0.5)$, $\Omega_2 = (0.5, 1)$, the interface is the point $x = 0.5$, $T = 1$. The PDEs and boundary and initial conditions are listed below.

$$\begin{cases} \partial_t u - D_{RL}^{1-0.3} u_{xx} = 0.1\, t^2 \quad \text{in } [0, 0.5] \times (0, 1], \\ \partial_t u - 0.2\, D_{RL}^{1-0.5} u_{xx} = 0.2\, t^2 \quad \text{in } [0.5, 1] \times (0, 1], \\ u(0, t) = u(1, t) = 0, \quad \forall t \in (0, 1], \\ u(x, 0) = \sin(\pi x), \quad \forall x \in (0, 1). \end{cases} \tag{110}$$

As discussed in Remark 4, graded temporal meshes are used and values of optimal mesh grading parameter r are chosen. Specifically for this problem, $\alpha_1 = 0.3, \alpha_2 = 0.5$, and hence $\min(\alpha_1, \alpha_2) = 0.3$.

- $r = (2-0.3)/0.3 = 5.66$ is chosen for the L1 + back-Euler (L1-BE) solvers;
- $r = (3 - 0.3)/0.3 = 9$ is chosen for the L2 + Crank-Nicolson (L2-CN) solvers.

A graded mesh with $r = 5.66$ is shown in Figure 3. Numerical results in Tables 2,3,4,5 demonstrate the convergence rates of our fast solvers. Figures 4,5 illustrate effects of the fast solvers for the 1d interface problem.

FIGURE 3. The graded mesh for the L1-BE solver.

TABLE 2. Ex.1 with back-Euler: L_2-norm errors and convergence rates in space at $T = 1$.

		Direct L1 solver		Fast L1 solver	
N_P	N_T	Error	Rate	Error	Rate
10	50	1.1441×10^{-4}	–	1.1890×10^{-4}	–
20		2.6203×10^{-5}	2.1265	2.7560×10^{-5}	2.1091
40		5.9708×10^{-6}	2.1337	6.3450×10^{-6}	2.1189
80		1.3981×10^{-6}	2.0944	1.4919×10^{-6}	2.0885

TABLE 3. Ex.1 with back-Euler: L_2-norm errors and convergence rates in time at $T = 1$.

		Direct L1 solver		Fast L1 solver	
N_P	N_T	Error	Rate	Error	Rate
50	32	1.5539×10^{-2}	–	1.3783×10^{-2}	–
	64	7.7970×10^{-3}	0.9950	7.0623×10^{-3}	0.9647
	128	3.9162×10^{-3}	0.9934	3.6220×10^{-3}	0.9634
	256	1.9670×10^{-3}	0.9935	1.8497×10^{-3}	0.9695

TABLE 4. Ex.1 with Crank-Nicolson: L_2-norm errors and convergence rates in space at $T = 1$.

		Direct L2 solver		Fast L2 solver	
N_P	N_T	Error	Rate	Error	Rate
10	50	1.4265×10^{-4}	–	1.4267×10^{-4}	–
20		3.6093×10^{-5}	1.9827	3.6096×10^{-5}	1.9827
40		9.0087×10^{-6}	2.0023	9.0054×10^{-6}	2.0029
80		2.2699×10^{-6}	1.9886	2.2604×10^{-6}	1.9941

TABLE 5. Ex.1 with Crank-Nicolson: L_2-norm errors and convergence rates in time at $T = 1$.

		Direct L2 solver		Fast $L2$ solver	
N_P	N_T	Error	Rate	Error	Rate
50	32	4.5594×10^{-3}	–	3.4363×10^{-3}	–
	64	1.0550×10^{-3}	2.1115	8.1483×10^{-4}	2.0763
	128	2.5084×10^{-4}	2.0724	2.1304×10^{-4}	1.9353
	256	6.0185×10^{-5}	2.0593	5.3790×10^{-5}	1.9857

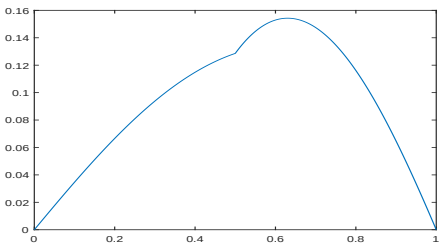


FIGURE 4. Ex.1: Numerical solution obtained from the fast L1 solver (combined with back-Euler) with $N_P = N_T = 500$.

Example 2. We consider a 2d problem on the unit square $\Omega = (0,1)^2$. The interface is the circle centered at $(\frac{1}{2}, \frac{1}{2})$ with radius $\frac{1}{4}$. In other words, $\Omega_2 =$

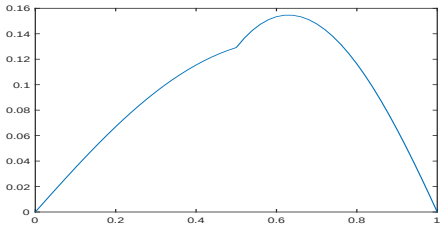


FIGURE 5. Ex.1: Numerical solution obtained from the fast L2 solver (combined Crank-Nicolson) with $N_P = N_T = 50$.

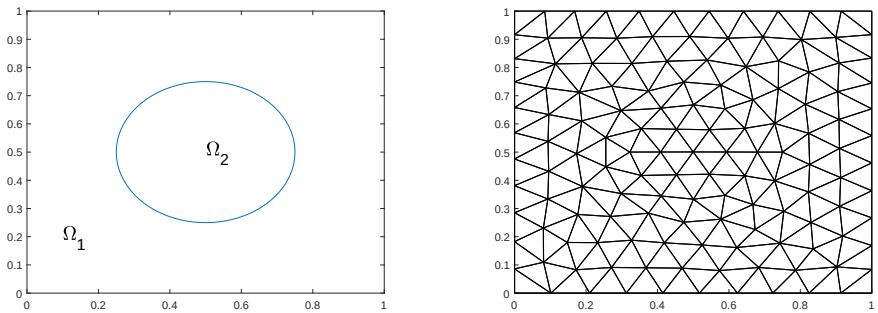
TABLE 6. Ex.2 with back-Euler: L_2-norm errors and convergence rates in time at $T = 1$.

$h \approx 0.1$	Direct L1 solver		Fast L1 solver	
N_T	Error	Rate	Error	Rate
32	9.8284×10^{-3}	–	9.8284×10^{-3}	–
64	4.7199×10^{-3}	1.0582	4.7203×10^{-3}	1.0581
128	2.3027×10^{-3}	1.0354	2.2998×10^{-3}	1.0374
256	1.1370×10^{-3}	1.0181	1.1550×10^{-3}	0.9936

$\{(x_1, x_2) | (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.25^2\}$, $\Omega_1 = \Omega \backslash \Omega_2$. The final time $T = 1$. The PDEs and boundary and initial conditions are as follows.

$$\begin{cases} \partial_t u - D_{RL}^{1-0.3} \Delta u = 0.1\, t^2 \quad \text{in } \Omega_1 \times (0, 1], \\ \partial_t u - 0.2\, D_{RL}^{1-0.5} \Delta u = 0.2\, t^2 \quad \text{in } \Omega_2 \times (0, 1], \\ u(\partial\Omega, t) = 0, \quad u(x, y, 0) = \sin(\pi x) \sin(\pi y), \end{cases} \tag{111}$$

For this 2d problem, graded temporal meshes are used also. The choices for the optimal grading parameter r are the same as those for Example 1. The spatial mesh shown in Figure 6(b) has 233 triangles, 139 nodes, and a mesh size $h \approx 0.1$. Tables 6 and 8 exhibit accuracy of the fast solvers, whereas Tables 7 and 9 demonstrate significant advantages of the fast solvers over the direct solvers. Figures 7 and 8 illustrate effects of the interface conditions.



(a) (b)

FIGURE 6. Ex.2: (a) The computational domains; (b) An interface-fitted triangular mesh.

TABLE 7. Ex.2: Comparison of CPU time (in seconds) for direct and fast L1 solvers.

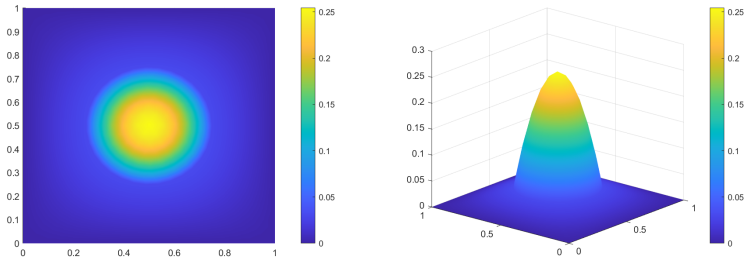
$h \approx 0.05$	Direct L1 solver	Fast L1 solver
N_T	CPU time	CPU time
5000	123.19	9.03
10000	465.09	17.64
15000	1020.37	27.32
20000	1794.54	36.60

TABLE 8. Ex.2 with Crank-Nicolson: L_2-norm errors and convergence rates in time at $T = 1$.

$h \approx 0.1$	Direct L2 solver		Fast L2 solver	
N_T	Error	Rate	Error	Rate
32	3.3945×10^{-3}	–	2.5448×10^{-3}	–
64	7.2219×10^{-4}	2.2327	5.3769×10^{-4}	2.2427
128	1.6637×10^{-4}	2.1179	1.3670×10^{-4}	1.9757
256	3.9393×10^{-5}	2.0784	3.4738×10^{-5}	1.9764

TABLE 9. Ex.2: Comparison of CPU time (in seconds) for direct and fast L2 solvers.

$h \approx 0.05$	Direct L2 solver	Fast L2 solver
N_T	CPU time	CPU time
5000	116.37	11.74
10000	432.79	24.15
15000	934.94	36.14
20000	1632.73	48.12

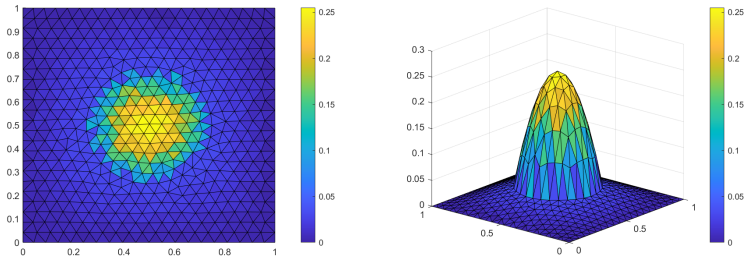


(a) Solution as a 2d image (b) Solution as a surface

FIGURE 7. Ex.2: Numerical solution by the L1 + back-Euler fast solver with $N_T = 500, h = 0.05$.

9. Concluding Remarks

In this paper, we have developed two fast numerical solvers for subdiffusion problems with spatial interfaces. Such problems are modeled by partial differential equations that contain both the conventional 1st order time derivative and fractional order time derivatives, in addition to appropriate interface conditions on the mass and fractional order fluxes. Specifically, the back Euler or Crank-Nicolson



(a) Solution as a 2d image (b) Solution as a meshed surface

FIGURE 8. Ex.2: Numerical solution by the L2 + Crank-Nicolson fast solver with $N_T = 50, h = 0.05$.

discretization is used for the conventional 1st order time derivative, whereas the L1 or L2 discretization along with their fast evaluation algorithms are utilized for the Caputo derivative in the model. Procedures for efficient implementation of these solvers are briefly discussed. Numerical experiments have demonstrated accuracy and efficiency of these new fast solvers.

It can be proved that our fast solvers are unconditionally stable by establishing positive-definiteness of the fast L1 discretization operator. The closeness of the direct and fast L1 discretization operators can also be established and then utilized to prove the convergence of our fast solver. Due to page limitation, the analysis details are omitted.

We point out that the L2-1_σ discretization [1] cannot be used for such problems, although it has a high order truncation error $\mathcal{O}((1/N_T)^{3-\alpha})$. Since the intermediate time moments are $t_{k+\sigma}$ with $\sigma = 1 - \frac{\alpha}{2}$, the temporal time nodes will be different for different spatial domains separated by interfaces.

Nevertheless, the methodology developed in this paper can be applied to nonlinear problems, e.g., the time-fractional Allen-Cahn equation. It will also be interesting to develop fast solvers for convection-subdiffusion problems with interfaces. These are currently under investigation and will be reported in our future work.

The model (5)-(9) can be generalized to allow moving interfaces, which bear greater significance in practical problems. The Arbitrary Lagrangian-Eulerian (ALE) method proves an effective approach for handling moving interfaces. It was successfully employed in [19] to tackle a Stokes/Parabolic moving interface problem. New techniques similar to those in [19] can be developed to determine the ALE mapping and domainwise velocities, in combination with finite volume methods. However, the time-fractional operators are non-local. At each step of computation, the domain velocities of all historical layers may be involved. New techniques need to be developed and will be reported in our future work.

Acknowledgments

Y. Li was partially supported by The National Natural Science Foundation of China (No.12071177). J. Liu was partially supported by US National Science Foundation under grant DMS-2208590.

References

[1] A. Alikhanov, A new difference scheme for the time fractional diffusion equation, J. Comput. Phys., 280 (2015), pp. 424–438.

[2] D. Baffet, A Gauss–Jacobi kernel compression scheme for fractional differential equations, Journal of Scientific Computing, 79 (2019), pp. 227–248.
[3] D. Baffet and J. S. Hesthaven, High-order accurate adaptive kernel compression time-stepping schemes for fractional differential equations, J. Sci. Comput., 72 (2017), pp. 1169–1195.
[4] D. Baffet and J. S. Hesthaven, A kernel compression scheme for fractional differential equations, SIAM J. Numer. Anal., 55 (2017), pp. 496–520.
[5] G. Beylkin and L. Monzón, On approximation of functions by exponential sums, Appl. Comput. Harmon. Anal., 19 (2005), pp. 17–48.
[6] G. Beylkin and L. Monzón, Approximation by exponential sums revisited, Appl. Comput. Harmon. Anal., 28 (2010), pp. 131–149.
[7] A. Bueno-Orovio, I. Teh, J. E. Schneider, K. Burrage, and V. Grau, Anomalous diffusion in cardiac tissue as an index of myocardial microstructure, IEEE Trans. Med. Img., 35 (2016), pp. 2200–2207.
[8] M. Caputo and M. Fabrizio, The kernel of the distributed order fractional derivatives with an application to complex materials, Fractal and Fractional MDPI, 1 (2017), p. 1010013.
[9] A. Delic and B. Jovanovic, Numerical approximation of an interface problem for fractional in time diffusion equation, Appl. Math. Comput., (2014).
[10] K. Diethelm and A. D. Freed, An efficient algorithm for the evaluation of convolution integrals, Computers & Mathematics with Applications, 51 (2006), pp. 51–72.
[11] K. Diethelm, R. Garrappa, and M. Stynes, Good (and not so good) practices in computational methods for fractional calculus, Mathematics, 8 (2020), p. 324.
[12] S. Duo, L. Ju, and Y. Zhang, A fast algorithm for solving the space-time fractional diffusion equation, Comput. Math. Appl., 75 (2018), pp. 1929–1941.
[13] L. Feng, I. Turner, P. Perré, and K. Burrage, An investigation of nonlinear time-fractional anomalous diffusion models for simulating transport processes in heterogeneous binary media, Commun. Nlin. Sci. Numer. Simul., 92 (2021), p. 105454.
[14] N. J. Ford and A. C. Simpson, The numerical solution of fractional differential equations: speed versus accuracy, Numerical Algorithms, 26 (2001), pp. 333–346.
[15] E. Hairer, C. Lubich, and M. Schlichte, Fast numerical solution of nonlinear Volterra convolution equations, SIAM journal on scientific and statistical computing, 6 (1985), pp. 532–541.
[16] E. Hairer, C. Lubich, and M. Schlichte, Fast numerical solution of weakly singular Volterra integral equations, Journal of computational and applied mathematics, 23 (1988), pp. 87–98.
[17] S. Jiang, J. Zhang, Q. Zhang, and Z. Zhang, Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations, Commun. Comput. Phys., 21 (2017), pp. 650–678.
[18] N. Kopteva, Error analysis of an L2-type method on graded meshes for a fractional-order parabolic problem, Mathematics of Computation, 90 (2021), pp. 19–40.
[19] R. Lan and P. Sun, A monolithic arbitrary Lagrangian–Eulerian finite element analysis for a Stokes/parabolic moving interface problem, Journal of Scientific Computing, 82 (2020), p. 59.
[20] T. Langlands and B. Henry, The accuracy and stability of an implicit solution method for the fractional diffusion equation, J. Comput. Phys., 205 (2005), pp. 719–736.
[21] C. Li, W. Deng, and Y. Wu, Numerical analysis and physical simulations for the time fractional radial diffusion equation, Comput. Math. Appl., 62 (2011), pp. 1024–1037.
[22] R. Li, Z. Chen, and W. Wu, Generalized difference methods for differential equations: numerical analysis of finite volume methods, CRC Press, 2000.
[23] Y. Lin and C. Xu, Finite difference/spectral approximations for the time-fractional diffusion equation, J. Comput. Phys., 225 (2007), pp. 1533–1552.
[24] J. Liu, H. Li, and Y. Liu, A new fully discrete finite difference/element approximation for fractional cable equation, Journal of Applied Mathematics and Computing, 52 (2016), pp. 345–361.
[25] C. Lubich, Discretized fractional calculus, SIAM Journal on Mathematical Analysis, 17 (1986), pp. 704–719.
[26] C. Lubich, Convolution quadrature and discretized operational calculus. I, Numerische Mathematik, 52 (1988), pp. 129–145.
[27] C. Lubich, Convolution quadrature and discretized operational calculus. II, Numerische Mathematik, 52 (1988), pp. 413–425.
[28] C. Lubich, Convolution quadrature revisited, BIT Numerical Mathematics, 44 (2004), pp. 503–514.

[29] K. Mustapha, An implicit finite-difference time-stepping method for a sub-diffusion equation with spatial discretization by finite elements, IMA Journal of Numerical Analysis, 31 (2011), pp. 719–739.
[30] K. B. Oldham and J. Spanier, Theory and applications of differentiation and integration to arbitrary order, The Fractional Calculus; Academic Press: New York, NY, USA; London, UK, (1974).
[31] P.-O. Persson and G. Strang, A simple mesh generator in MATLAB, SIAM Review, 46 (2004), pp. 329–345.
[32] Y. Povstenko, Central symmetric solution to the Neumann problem for a time-fractional diffusion-wave equation in a sphere, Nlin. Anal. Real World Appl., 13 (2012), pp. 1229–1238.
[33] Y. Povstenko and T. Kyrylych, Fractional thermoelasticity problem for an infinite solid with a penny-shaped crack under prescribed heat flux across its surfaces, Philos. Trans. Royal Soc. A, 378 (2020).
[34] R. Raghavan and C.-C. Chen, Subdiffusive flow in a composite medium with a communicating (absorbing) interface, Oil Gas Sci. Technol., 75 (2020), p. Article 26.
[35] 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 Journal on Numerical Analysis, 55 (2017), pp. 1057–1079.
[36] H. Sun and W. Cao, A fast temporal second-order difference scheme for the time-fractional subdiffusion equation, Numer. Meth. PDEs, 37 (2021), pp. 1825–1846.
[37] H.-G. Sun, W. Chen, C. Li, and Y.-Q. Chen, Fractional differential models for anomalous diffusion, Physica A, 389 (2010), pp. 2719–2724.
[38] J. Sun, D. Nie, and W. Deng, Fast algorithms for convolution quadrature of Riemann-Liouville fractional derivative, Applied Numerical Mathematics, 145 (2019), pp. 384–410.
[39] J. Tan and J. Liu, An efficient numerical solver for anisotropic subdiffusion problems, J. Comput. Appl. Math., 364 (2020), p. Article 112318.
[40] J. Tan and J. Liu, Solving anisotropic subdiffusion problems in annuli and shells, J. Comput. Appl. Math., 401 (2022), p. Article 113764.
[41] J. Tausch, Fast Nyström methods for parabolic boundary integral equations, Springer, Berlin, Heidelberg, 2012, pp. 185–219.
[42] C. J. Vogl, M. J. Miksis, and S. H. Davis, Moving boundary problems governed by anomalous diffusion, Proc. R. Soc. A, 468 (2012), pp. 3348–3369.
[43] H. Wang and T. S. Basu, A fast finite difference method for two-dimensional space-fractional diffusion equations, SIAM J. Sci. Comput., 34 (2012), pp. A2444–A2458.
[44] M. Weiss, M. Elsner, F. Kartberg, and T. Nilsson, Anomalous subdiffusion is a measure for cytoplasmic crowding in living cells, Biophys. J., 87 (2004), pp. 3518–3524.
[45] S. Wu and Z. Zhou, A parallel-in-time algorithm for higher-order BDF methods for diffusion and subdiffusion equations, SIAM J. Sci. Comput., 43 (2021), pp. A3627–A3656.
[46] Y. Yan, Z. Sun, and J. Zhang, Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations: a second-order scheme, Commun. Comput. Phys., 22 (2017), pp. 1028–1048.
[47] F. Zeng, Z. Zhang, and G. E. Karniadakis, Fast difference schemes for solving high-dimensional time-fractional subdiffusion equations, J. Comput. Phys., 307 (2016), pp. 15–33.
[48] H. Zhu and C. Xu, A fast high order method for the time-fractional diffusion equation, SIAM J. Numer. Anal., 57 (2019), pp. 2829–2849.

School of Mathematics, Jilin University, Changchun, Jilin 130012, China
E-mail: yuby21@mails.jlu.edu.cn

School of Mathematics, Jilin University, Changchun, Jilin 130012, China
E-mail: yonghai@jlu.edu.cn
URL: https://math.jlu.edu.cn/info/1061/9163.htm

Department of Mathematics, Colorado State University, Fort Collins, CO 80523-1874, USA
E-mail: liu@math.colostate.edu
URL: https://www.math.colostate.edu/∼liu/