A unified fast method for the fractional initial-boundary value problems

Fanhai Zeng
Shandong University

Collaborators: Yuxiang Huang, Ling Guo, and Zheng Yang

August 2022

Outlines

- Background
- Construction of the fast method, applications, and convergence
- Numerical examples

Background

Develop the fast method to calculate the convolution

$$k * u(t) = \int_0^t k(t-s)u(s)\,\mathrm{d}s. \tag{1}$$

The convolution (1) arises in

- the integral models, i.e., the fractional differential equations, describing the anomalous diffusion [1].
- wave propagation on bounded domains, which requires to impose transparent (nonreflecting) boundary conditions [2] [3] [4] [5]

$$\hat{u}_j(x,t) = -\int_0^t f_j(t-s)\partial_\nu \hat{u}_j(x,s)\,\mathrm{d}s, \quad x \in \partial\Omega.$$

1D Schrodinger EQ: $1/2$-order Caputo fractional derivative

[1] R. Metzler, J.H. Jeon, A.G. Cherstvy, and E. Barkai, Phys. Chem. Chem. Phys., 2014.

[2] C. Lubich and A. Schadle, SIAM J. Sci. Comput., 2002.

[3] L. Banjai, M. López-Fernández, and A. Schädle, SIAM J. Numer. Math., 2017.

[4] S. Jiang and Greengard, Comput. Math. Appl., 2004

[5] B. Li, J. Zhang, C. Zheng, , SIAM J. Numer. Math., 2018.

The fractional operator: choosing $k(t) = k_\alpha(t) = \frac{t^{\alpha-1}}{\Gamma(\alpha)}$ yields

$$k_\alpha * u(t) = \int_0^t \frac{(t-s)^{\alpha-1}}{\Gamma(\alpha)} u(s)\,\mathrm{d}s. \tag{2}$$

- The **Riemann–Liouville (RL) fractional integral** of order α for $\alpha > 0$;
- The **RL fractional derivative** of order $-\alpha > 0$ for $\alpha < 0$ in the sense of **Hardamard finite-part integral**.

$$\begin{aligned} \textbf{An example:} \qquad k_{-\alpha} * u(t) &= P.V. \left[\int_0^t \frac{(t-s)^{-\alpha-1}}{\Gamma(-\alpha)} u(s)\,\mathrm{d}s \right] \\ &= \frac{d}{dt} \left[(k_{1-\alpha} * u)(t) \right], \quad 0 < \alpha < 1. \end{aligned} \tag{3}$$

- Variable-order fractional operators: $\boxed{\alpha = \alpha(t)}$

$$k_{\alpha(t)} * u(t) = \int_0^t \frac{(t-s)^{\alpha(t)-1}}{\Gamma(\alpha(t))} u(s)\,\mathrm{d}s. \tag{4}$$

- How about $\boxed{\alpha = \alpha(s)}$ in (4)?

Numerical methods

- **Interpolation**: uniform or nonuniform grid points $\{t_0, t_1, ..., t_M\}$

$$(k_\alpha * u)(t_n) \approx (k_\alpha * I_\tau u)(t_n) = \sum_{k=0}^{n} \omega_{n,k} u(t_k), \quad 1 \le n \le M. \tag{5}$$

- **Convolution quadrature (CQ, fractional linear multistep method)**: uniform grid points $t_k = k\tau$, $\tau > 0$ is a step size

$$(k_\alpha * u)(t_n) \approx \underbrace{\sum_{k=0}^{n} \omega_{n-k} u(t_k)}_{\text{convolution part}} + \underbrace{\sum_{j=0}^{m} W_{n,j} u(t_j)}_{\text{corrections}}, \quad 1 \le n \le M, \tag{6}$$

where ω_n are the coefficients of the Taylor expansion of the generating function,

$$\omega(z) = \sum_{n=0}^{\infty} \omega_n z^n,$$

which can be derived from the generating function of the **linear multistep method**.

Storage and computational cost

Direct calculation of

$$k_\alpha * u(t_n) \approx \sum_{k=0}^{n} w_{n,k} u(t_k), \quad 1 \le n \le M \tag{7}$$

needs

1) storage: $O(M) \to O(1)$ as $\alpha \to \pm 1$. For example, for linear interpolation and $\alpha \to -1$, (7) reduces to

$$\sum_{k=0}^{n} w_{n,k} u(t_k) \to \frac{u(t_n) - u(t_{n-1})}{t_n - t_{n-1}}.$$

2) computational cost: $O(M^2) \to O(M)$ as $\alpha \to \pm 1$

Fast calculation yields

1) storage: $O(\log(M))$ or $O(Q)$; $\quad \log(M) \ll M,\ Q \ll M.$
2) computational cost: $O(M \log(M))$ or $O(QM)$.

Sum-of-exponentials

The basic idea is to seek a suitable **sum-of-exponentials** to approximate the kernel function $k_\alpha(t) = t^{\alpha-1}/\Gamma(\alpha)$, i.e.,

$$k_\alpha(t) = \frac{t^{\alpha-1}}{\Gamma(\alpha)} = \sum_{j=1}^{Q} w_j e^{\lambda_j t} + O(\varepsilon t^{\alpha-1}), \tag{8}$$

where $\varepsilon > 0$ is a given precision.

- **Global** approximation: (8) holds for $\forall t \in [\delta, T]$, $\delta, T > 0$.
- **Local** approximation: (8) holds for $\forall t \in I_\ell \subset [\delta, T]$, i.e., $I_\ell = [B^{\ell-1}\tau, 2B^\ell\tau]$, $B > 1$ is a positive integer, satisfying

$$[\delta, T] \subset I_1 \cup I_2 \cup I_3 \cup \cdots \cup I_L, \qquad L = \left\lceil \log_B^{(T/\tau)} \right\rceil = O(\log(T/\tau)).$$

The method based on the numerical Laplace transform inversion

The Laplace transform of the kernel function $k_\alpha(t) = t^{\alpha-1}/\Gamma(\alpha)$ is

$$K_\alpha(\lambda) = \mathcal{L}[k_\alpha](\lambda) = \lambda^{-\alpha}.$$

The inverse Laplace transform yields [Lubich and Schädle, SISC, 2002; López-Fernández, Lubich, and Schädle, SISC, 2008]

$$\begin{aligned} k_\alpha(t) =& \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} K_\alpha(\lambda) e^{t\lambda} \mathrm{d}\lambda = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \lambda^{-\alpha} e^{t\lambda} \mathrm{d}\lambda \\ =& \frac{1}{2\pi \mathrm{i}} \int_{-\pi}^{\pi} (z(\theta))^{-\alpha} z'(\theta) e^{tz(\theta)} \mathrm{d}\theta \qquad (\lambda = z(\theta)) \\ \approx& \sum_{j=-N+1}^{N} w_j^{(\ell)} e^{\lambda_j^{(\ell)} t} \qquad t \in I_\ell = [B^{\ell-1}\tau, 2B^\ell \tau], \end{aligned} \tag{9}$$

where $w_j^{(\ell)} = \frac{1}{2\pi \mathrm{i}} (\lambda_j^{(\ell)})^{-\alpha} \partial_\theta z(\theta_j, \mu_\ell)$, $\lambda_j^{(\ell)} = z(\theta_j, \mu_\ell)$, $\theta_j = \frac{(2j-1)\pi}{2N}$.

- The optimal Talbot contour with $z(\theta, \mu_\ell)$ given by [Weideman, SINUM, 2006]

$$z(\theta, \mu_\ell) = \mu_\ell(-0.4814 + 0.6443(\theta \cot(\theta) + \mathrm{i}0.5653\theta)), \tag{10}$$

where $\mu_\ell = N/T_\ell$, $T_\ell = 2B^\ell\tau$, $B > 1$ is a positive integer.

- The hyperbolic contour with $z(\theta, \mu_\ell)$ given by [Lopez-Fernandez, Lubich, et al., Numer. Math., 2005]

$$z(\theta, \mu_\ell) = \mu_\ell(1 - \sin(\psi + \mathrm{i}\theta)) + \sigma. \tag{11}$$

In numerical simulations, we can select $\sigma = 0$, $\psi = 0.4\pi$, $\mu_\ell = N/T_\ell$, $T_\ell = B^\ell\tau$, $N = \lceil -\log(\tau^{1-\alpha}\epsilon) \rceil$, ϵ is a given precision.

The method based on the multi-pole approximation

Find a rational approximation $r(\lambda)$ to the Laplace transform of

$$k_\alpha(t+\delta) = (t+\delta)^{\alpha-1}/\Gamma(\alpha), \delta \geq 0,$$

which takes the following form [Baffet and Hesthaven, SINUM, 2017]

$$r(\lambda) = \sum_{j=1}^{Q} \frac{c_j}{\lambda - \lambda_j}. \tag{12}$$

Applying the inverse Laplace transform yields

$$k_\alpha(t+\delta) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} r(\lambda) e^{t\lambda} \mathrm{d}\lambda = \sum_{j=1}^{Q} w_j e^{\lambda_j t}. \tag{13}$$

The method based on the Jacobi–Gauss quadrature

$k_\alpha(t) = \frac{t^{\alpha-1}}{\Gamma(\alpha)}$ can be expressed as

$$\begin{aligned} k_\alpha(t) =& \frac{\sin(\alpha\pi)}{\pi} \int_0^\infty \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda \qquad (\alpha < 1) \\ =& \frac{\sin(\alpha\pi)}{\pi} \int_0^A \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda + O(\varepsilon t^{\alpha-1}). \end{aligned} \tag{14}$$

Taking $A = 2^{m_2}$ yields

$$\begin{aligned} \int_0^A \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda =& \underbrace{\int_0^{2^{-m_1}} \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda}_{\text{Jacobi–Gauss}} + \sum_{k=-m_1}^{m_2-1} \underbrace{\int_{2^k}^{2^{k+1}} \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda}_{\text{Legendre–Gauss}} \\ \approx& \sum_{j=1}^{N_o} w_j^{(o)} e^{-\lambda_j^{(o)} t} + \sum_{k=-m_1}^{m_2-1} \sum_{j=1}^{N_k} w_j^{(k)} (\lambda_j^{(k)})^{-\alpha} e^{-\lambda_j^{(k)} t}. \end{aligned} \tag{15}$$

Jacobi–Gauss quadrature [Jiang, Zhang, Zhang, and Zhang, Commun. Comput. Phys., 2017]

$$
\begin{aligned}
k_\alpha(t) \approx & \frac{\sin(\alpha\pi)}{\pi} \int_0^A \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda \\
\approx & \frac{\sin(\alpha\pi)}{\pi} \left(\sum_{j=1}^{N_o} w_j^{(o)} e^{-\lambda_j^{(o)} t} + \sum_{k=-m_1}^{m_2-1} \sum_{j=1}^{N_k} w_j^{(k)} (\lambda_j^{(k)})^{-\alpha} e^{-\lambda_j^{(k)} t} \right) \qquad (16) \\
= & \sum_{j=1}^{Q} w_j e^{\lambda_j t}.
\end{aligned}
$$

Legendre–Gauss quadrature [Jing-Rebecca Li, SISC, 2010]

$$
\begin{aligned}
k_\alpha(t) \approx & \underbrace{\frac{\sin(\alpha\pi)}{\pi} \int_0^\sigma \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda}_{O(\varepsilon)} + \frac{\sin(\alpha\pi)}{\pi} \int_\sigma^A \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda \\
\approx & \frac{\sin(\alpha\pi)}{\pi} \left(\sum_{k=-m_1}^{m_2-1} \sum_{j=1}^{N_k} w_j^{(k)} (\lambda_j^{(k)})^{-\alpha} e^{-\lambda_j^{(k)} t} \right) \qquad (17) \\
= & \sum_{j=1}^{Q} w_j e^{\lambda_j t}.
\end{aligned}
$$

The method based on the Laguerre–Gauss quadrature

Rewrite

$$k_\alpha(t) = \frac{\sin(\alpha\pi)}{\pi} \int_0^\infty \lambda^{-\alpha} e^{-t\lambda} \mathrm{d}\lambda$$

as

$$\begin{aligned} k_\alpha(t) &= \frac{\sin(\alpha\pi)}{\pi} \int_0^\infty \underbrace{\lambda^{-\alpha} e^{-T_\ell \lambda}}_{\text{weight}} e^{-(t-T_\ell)\lambda} \mathrm{d}\lambda \\ &= \frac{\sin(\alpha\pi)}{\pi} \sum_{j=1}^{N} w_j^{(\ell)} e^{-(t-T_\ell)\lambda_j^{(\ell)}} + O(\varepsilon), \quad t - T_\ell \in I_\ell \qquad (18) \\ &= \sum_{j=1}^{Q} w_j e^{\lambda_j t} + O(\varepsilon). \end{aligned}$$

See [Zeng, Turner, Burrage, JSC, 2018].

The method based on the trapezoidal rule on the real line

Let $\lambda = e^x$. Then

$$k_\alpha(t) = \frac{\sin(\alpha\pi)}{\pi}\int_0^\infty \lambda^{-\alpha}e^{-t\lambda}\mathrm{d}\lambda = \int_{-\infty}^{\infty} \underbrace{\frac{\sin(\alpha\pi)}{\pi}e^{-te^x+(1-\alpha)x}}_{\phi(x,t)}\mathrm{d}x, \tag{19}$$

where $|\phi(x,t)| \to 0$ as $|x| \to \infty$ for $t > 0$. The trapezoidal rule for approximating (19) is [Trefethen, Weideman, SIAM Rev., 2014; McLean, Contemporary Computational Mathematics, 2018]

$$\begin{aligned} k_\alpha(t) &= h\sum_{j=-\infty}^{\infty}\phi(jh,t) + O(e^{-1/h}) \\ &= \underbrace{h\sum_{j=-\infty}^{-N_1-1}\phi(jh,t)}_{O(\varepsilon)} + h\sum_{j=-N_1}^{N_2}\phi(jh,t) + \underbrace{h\sum_{j=N_2+1}^{\infty}\phi(jh,t)}_{O(\varepsilon)} + O(e^{-1/h}) \\ &= \sum_{j=1}^{Q} w_j e^{\lambda_j t} + O(e^{-1/h}) + O(\varepsilon). \end{aligned} \tag{20}$$

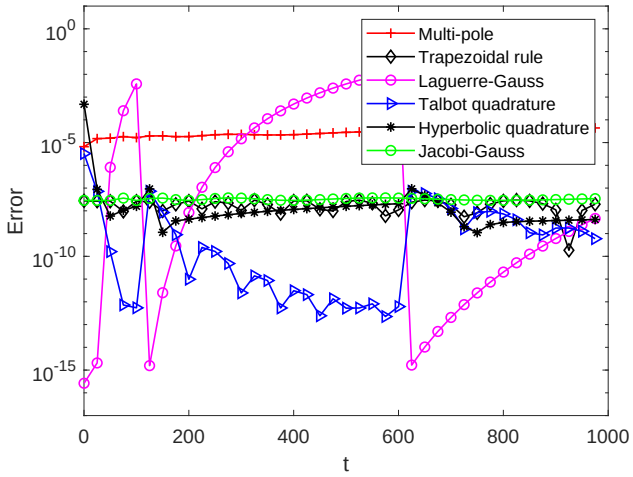
Table: Comparison of different methods for approximating $k_\alpha(t) = t^{\alpha-1}/\Gamma(\alpha)$.

Kernel approximation method	Range of α	Are λ_j dependent on α?	Extension to variable-order $\alpha = \alpha(t)$?	Global/Local approximation
Contour quadrature (9)	$\alpha \in \mathbb{R}$	No	Yes	Local, $t \in I_\ell$
Multi-pole method (13)	$\alpha < 1$	Yes	No	Global, $t \in [\delta, T]$
Jacobi–Gauss quadrature (16)	$\alpha < 1$	Yes	No	Global
Legendre–Gauss quadrature (17)	$\alpha < 1$	No	Yes	Global
Laguerre–Gauss quadrature (18)	$\alpha < 1$	Yes	No	Local
Trapezoidal rule (20)	$\alpha < 1$	No	Yes	Global

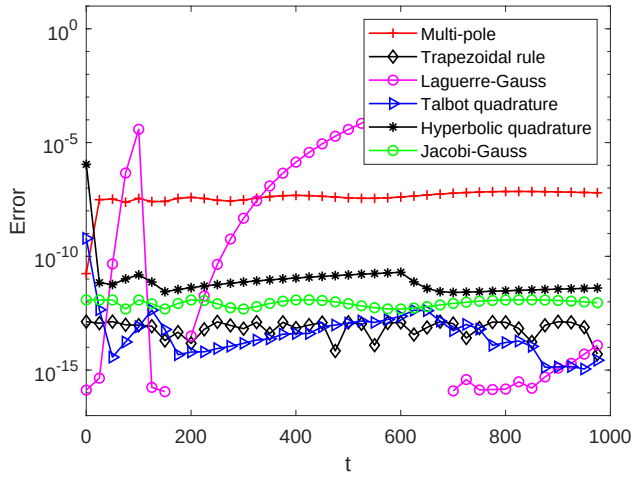
Compare the different kernel approximation methods.

Define the relative error

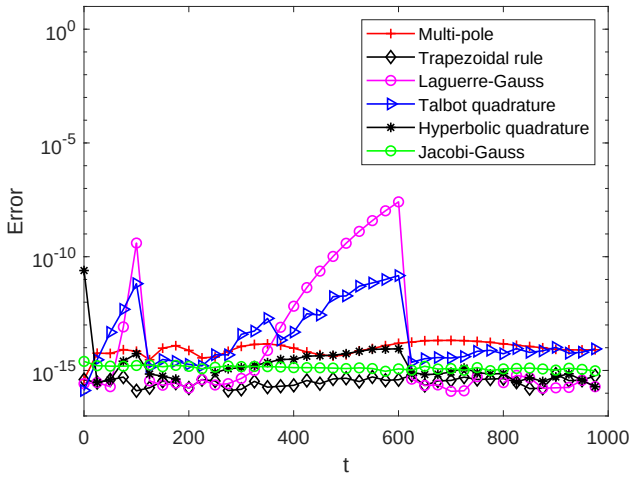
$$e(t) = \left| \frac{k_\alpha(t) - \sum_{j=1}^{Q} w_j e^{\lambda_j t}}{k_\alpha(t)} \right|.$$



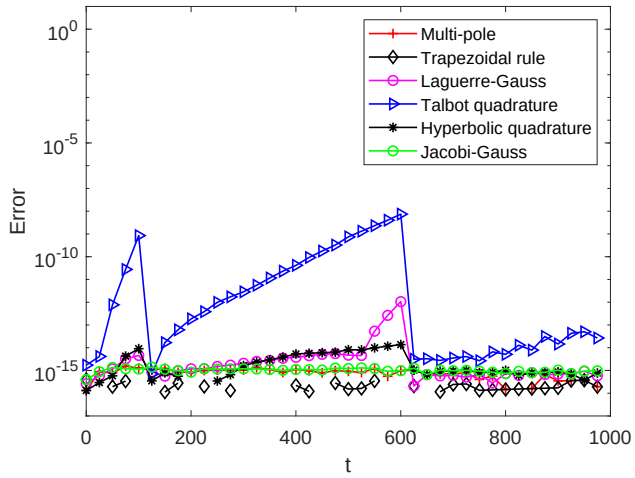
(a) $Q = 17 \times 6 = 102$.



(b) $Q = 27 \times 6 = 162$.

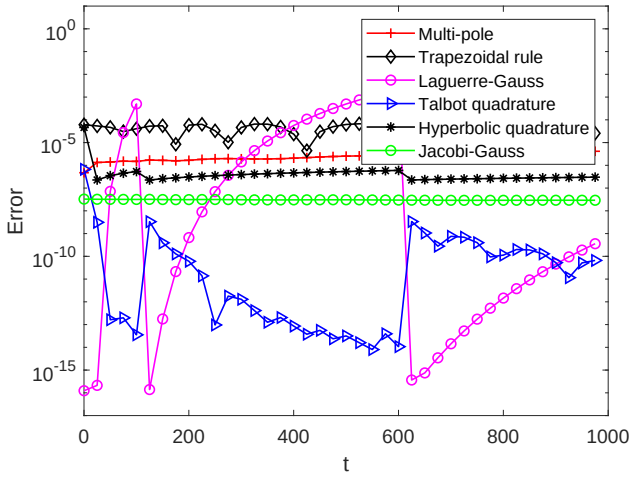


(c) $Q = 51 \times 6 = 306$.

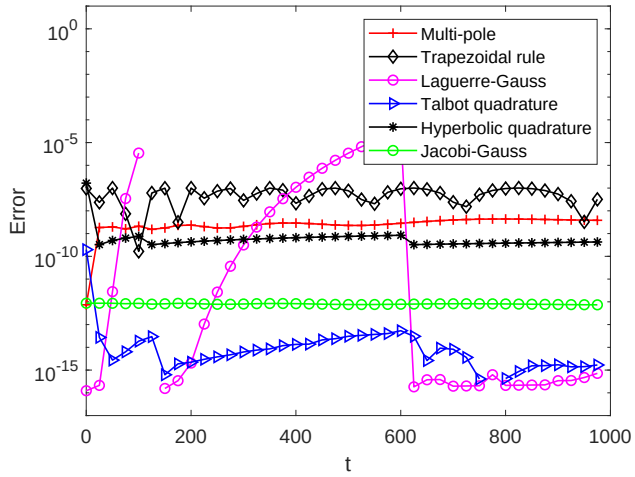


(d) $Q = 76 \times 6 = 456$.

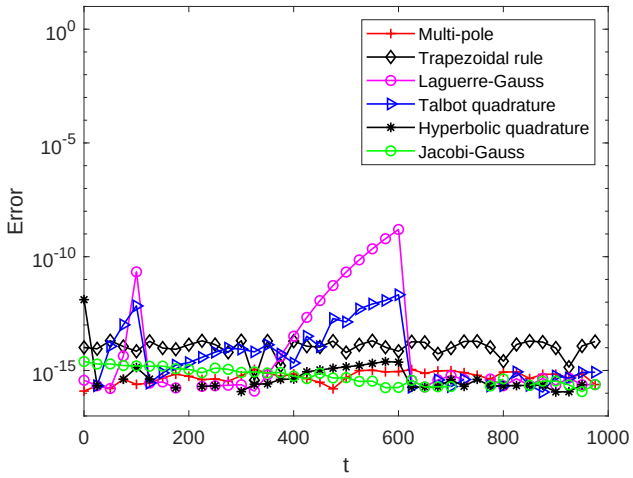
Figure: Comparison of different kernel approximation methods, $\delta = 0.1$, $T = 1000$, $\alpha = -0.4$, and $B = 5$.



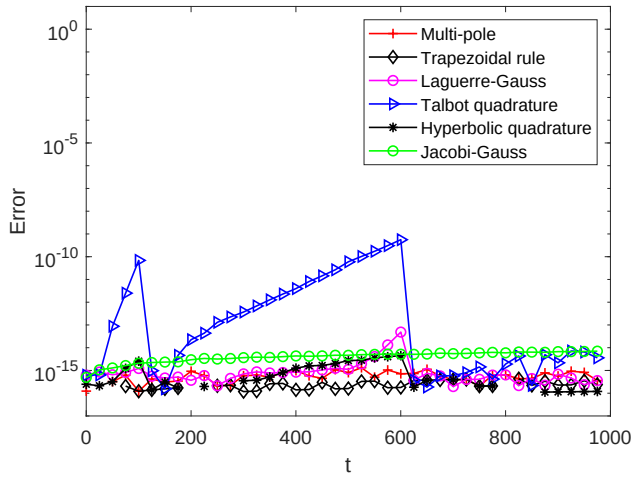
(a) $Q = 17 \times 6 = 102$.



(b) $Q = 27 \times 6 = 162$.

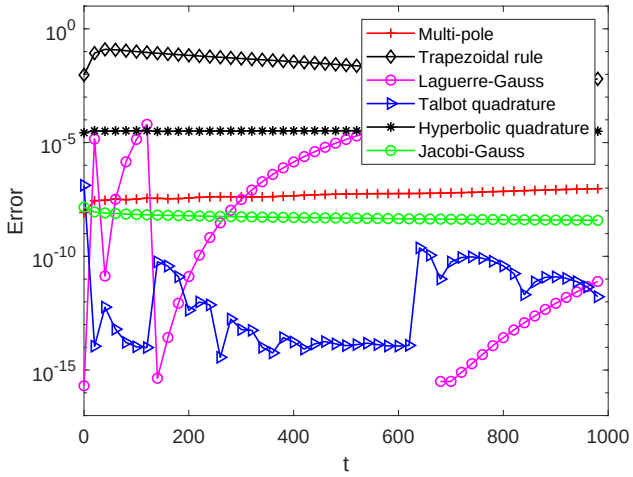


(c) $Q = 51 \times 6 = 306$.

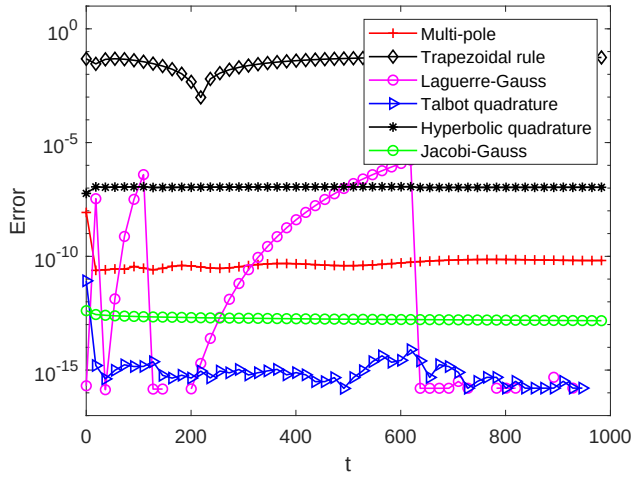


(d) $Q = 76 \times 6 = 456$.

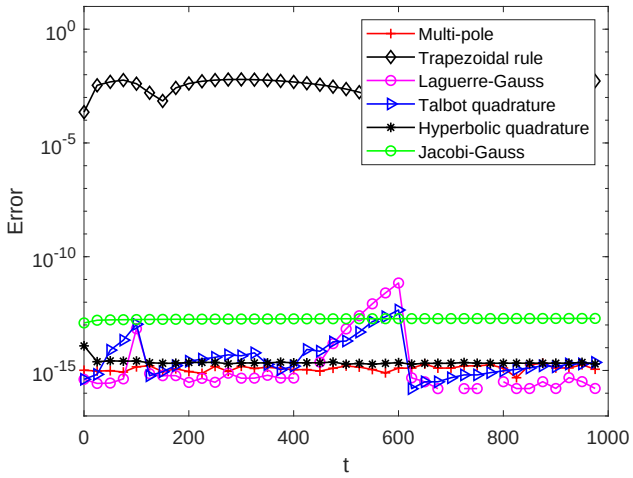
Figure: Comparison of different kernel approximation methods, $\delta = 0.1$, $T = 1000$, $\alpha = 0.4$, and $B = 5$.



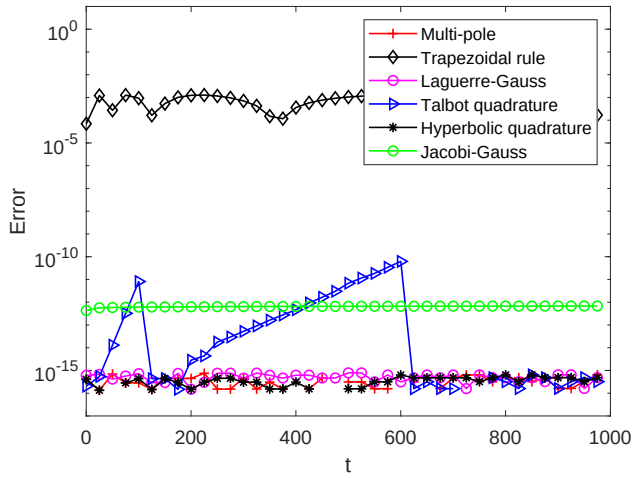
(a) $Q = 17 \times 6 = 102$.



(b) $Q = 26 \times 6 = 156$.



(c) $Q = 54 \times 6 = 324$.



(d) $Q = 76 \times 6 = 456$.

Figure: Comparison of different kernel approximation methods, $\delta = 0.1$, $T = 1000$, $\alpha = 0.95$, and $B = 5$.

An improved kernel approximation method

We already have

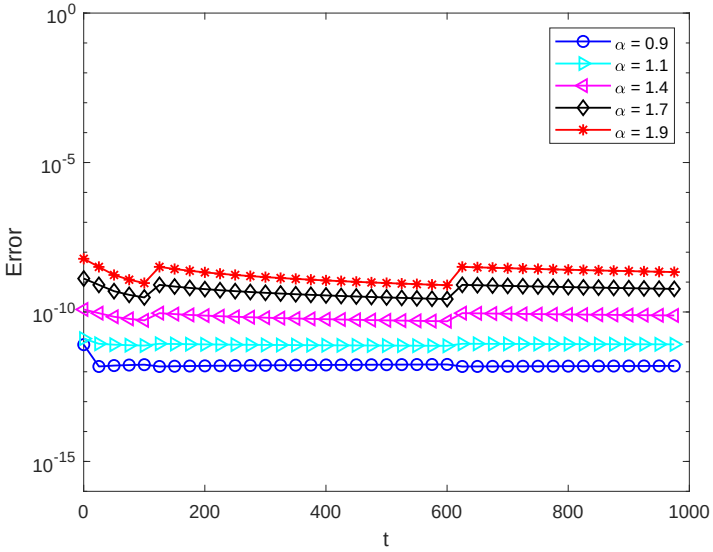
$$k_\alpha(t) = \frac{\sin(\alpha\pi)}{\pi}\int_0^\infty \lambda^{-\alpha}e^{-t\lambda}\mathrm{d}\lambda \approx \sum_{j=1}^{Q} w_j(\alpha)e^{\lambda_j(\alpha)t}, \qquad \alpha < 1. \tag{21}$$

Hence, $k_\alpha(t)$ can be approximated by

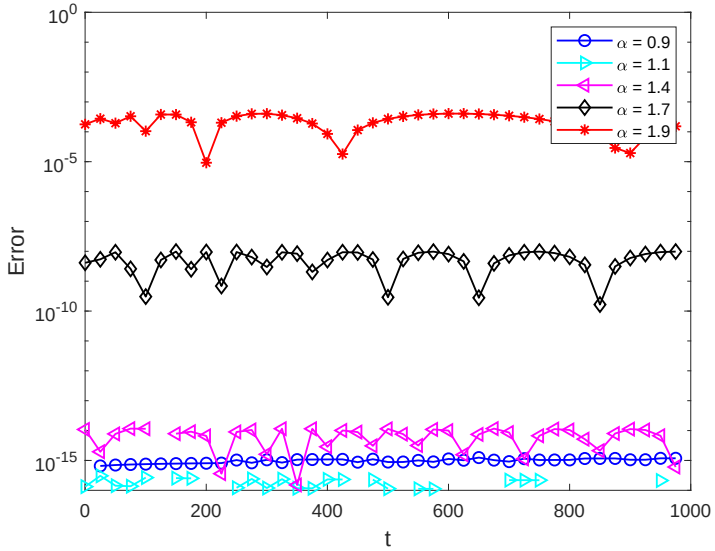
$$\begin{aligned} k_\alpha(t) =& \frac{t}{\alpha-1}k_{\alpha-1}(t) \\ \approx& \frac{t}{\alpha-1}\sum_{j=1}^{Q} w_j(\alpha-1)e^{\lambda_j(\alpha-1)t} \qquad (\alpha<2) \\ =& t\sum_{j=1}^{Q}\widehat{w}_j e^{\widehat{\lambda}_j t}, \end{aligned} \tag{22}$$

where $\widehat{w}_j = \frac{w_j(\alpha-1)}{\alpha-1}, \widehat{\lambda}_j = \lambda_j(\alpha-1)$.

$$k_\alpha(t) = \frac{t^2}{(\alpha-1)(\alpha-2)}k_{\alpha-2}(t), \qquad \alpha \in (2,3).$$



(a) The hyperbolic quadrature.



(b) The improved method.

Figure: Comparison between the hyperbolic quadrature (9) based on the contour quadrature (11) and the improved method (22), $Q = 256$.

Variable-order fractional case

Table: Comparison of different methods for approximating $k_\alpha(t) = t^{\alpha-1}/\Gamma(\alpha)$.

Kernel approximation method	Range of α	Are λ_j dependent on α?	Extension to variable-order $\alpha = \alpha(t)$?	Global/Local approximation
Contour quadrature (9)	$\alpha \in \mathbb{R}$	No	Yes	Local
Multi-pole method (13)	$\alpha < 1$	Yes	No	Global
Jacobi–Gauss quadrature (16)	$\alpha < 1$	Yes	No	Global
Legendre–Gauss quadrature (17)	$\alpha < 1$	No	Yes	Global
Laguerre–Gauss quadrature (18)	$\alpha < 1$	Yes	No	Local
Trapezoidal rule (20)	$\alpha < 1$	No	Yes	Global

Variable-order fractional case: $\alpha = \alpha(t)$

Contour quadrature: From (9), we can obtain

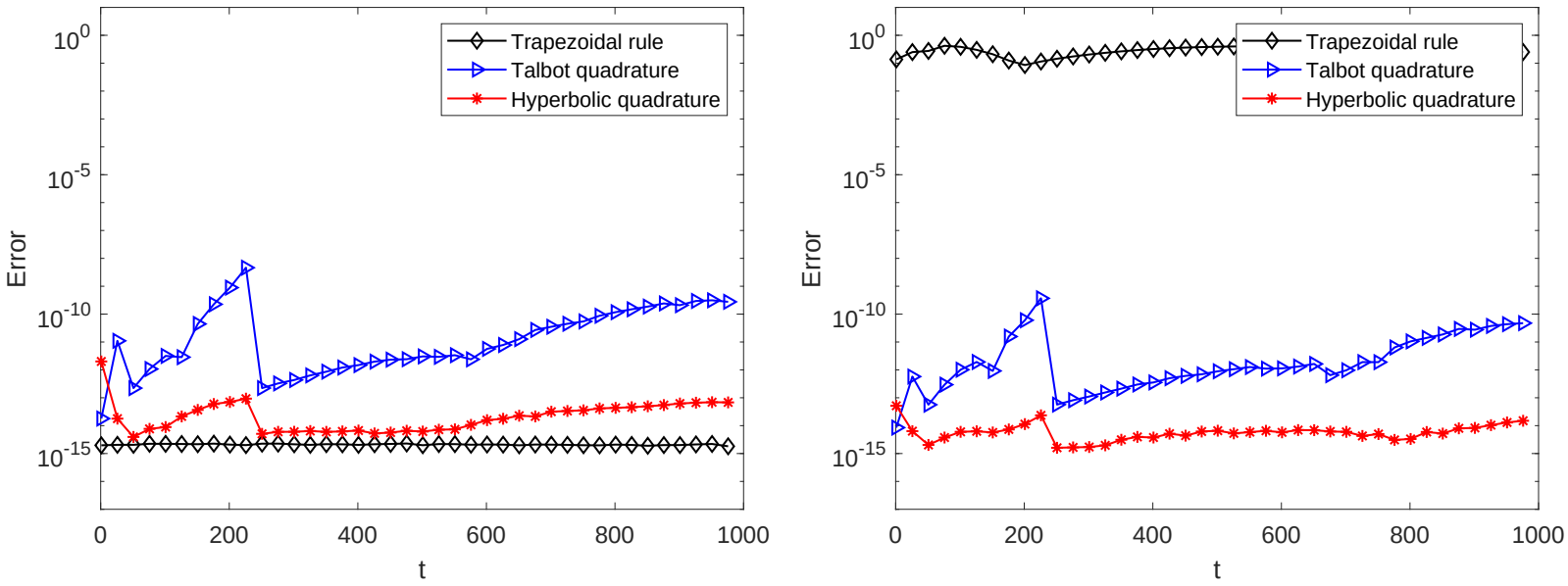
$$k_{\alpha(t)}(t) = \frac{t^{\alpha(t)-1}}{\Gamma(\alpha(t))} = \frac{1}{2\pi \mathrm{i}} \int_C \lambda^{-\alpha(t)} e^{t\lambda} \mathrm{d}\lambda. \tag{23}$$

Legendre–Gauss quadrature:

$$\begin{aligned} k_{\alpha(t)}(t) =& \frac{\sin(\alpha(t)\pi)}{\pi} \int_0^{\infty} \lambda^{-\alpha(t)} e^{-t\lambda} \mathrm{d}\lambda \\ =& \frac{\sin(\alpha(t)\pi)}{\pi} \int_{\sigma}^{A} \lambda^{-\alpha(t)} e^{-t\lambda} \mathrm{d}\lambda + O(\varepsilon) \\ =& \frac{\sin(\alpha(t)\pi)}{\pi} \sum_{j=1}^{K} \int_{\sigma_{j-1}}^{\sigma_j} \lambda^{-\alpha(t)} e^{-t\lambda} \mathrm{d}\lambda + O(\varepsilon) \end{aligned} \tag{24}$$

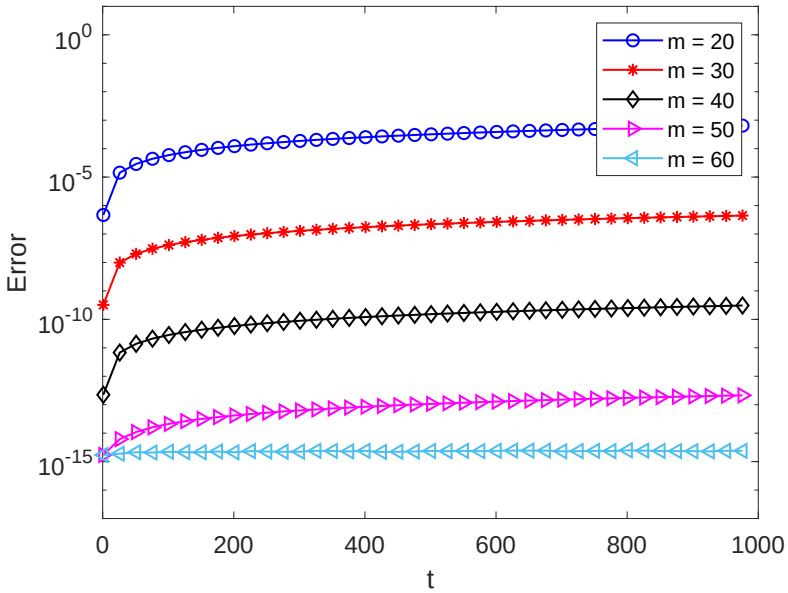
Trapezoidal rule: $\lambda = e^x$

$$\begin{aligned} k_{\alpha(t)}(t) =& \frac{\sin(\alpha(t)\pi)}{\pi} \int_0^{\infty} \lambda^{-\alpha(t)} e^{-t\lambda} \mathrm{d}\lambda \\ =& \frac{\sin(\alpha(t)\pi)}{\pi} \int_{-\infty}^{\infty} e^{(1-\alpha(t))x} e^{-te^x} \mathrm{d}x \end{aligned} \tag{25}$$

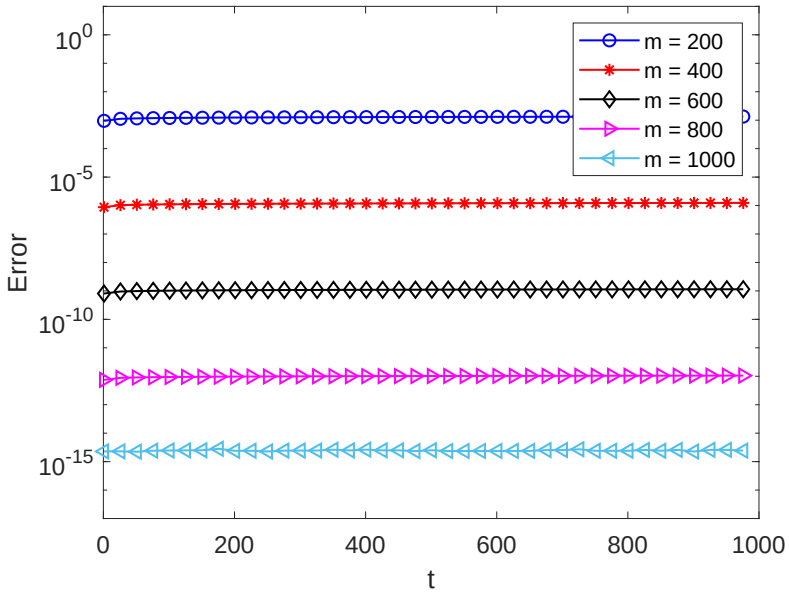


(a) $\alpha(t) = -(\frac{1}{20} + \frac{9}{10}\sin^2(10t))$. (b) $\alpha(t) = \frac{1}{20} + \frac{9}{10}\sin^2(10t)$.

Figure: Comparison of different kernel approximations for variable-order fractional orders, $\tau = 1$, $T = 1000$, and $Q = N \times L = 64 \times 4 = 256$. The trapezoidal rule shows better accuracy than the other two methods for negative $\alpha(t)$, but displays the worst accuracy when $\alpha(t) > 0$.



(a) $\alpha(t) = (\frac{1}{20} + \frac{9}{10}\sin^2(10t))$.



(b) $\alpha(t) = -(\frac{1}{20} + \frac{9}{10}\sin^2(10t))$.

Figure: The accuracy of the Legendre–Gauss quadrature for the method (17) for the variable order fractional case, $\tau = 1$, $T = 1000$, $m_2 = 6$, $N = 16$, and $Q = N \times m$.

The unified fast time-stepping method

1. Fast method for **RL fractional integral** $(k_\alpha * u)(t)$ is

$$\mathcal{F}(\alpha), \qquad \alpha > 0.$$

Fast method for **RL fractional derivative** ${}_{RL}D^{\alpha}_{0,t}u(t) = P.V.((k_{-\alpha} * u)(t))$ is

$$\mathcal{F}(-\alpha), \qquad \alpha > 0.$$

2. Fast method for the **Caputo fractional derivative**

$${}_{C}D^{\alpha}_{0,t}u(t) = (k_{m-\alpha} * u^{(m)})(t) = P.V.(k_{-\alpha} * (u-\phi)(t)), \quad m-1 < \alpha < m$$

reads as

$$\mathcal{F}(-\alpha) + \mathcal{B}(\alpha).$$

For a smooth $u(t)$, $\phi(t) = \sum_{j=0}^{m-1} \frac{u^{(j)}(0)}{\Gamma(j+1)} t^j$, so that

$$\mathcal{B}(\alpha) = -\sum_{j=0}^{m-1} \frac{t^{j-\alpha}}{\Gamma(j+1-\alpha)} u^{(j)}(0).$$

The linear interpolation operator $\Pi^1 : C[0,T] \to C[0,T]$ reads as

$$\Pi^1 u(t)|_{t\in[t_{j-1},t_j]} = \frac{t-t_j}{t_{j-1}-t_j}u(t_{j-1}) + \frac{t-t_{j-1}}{t_j-t_{j-1}}u(t_j), \tag{26}$$

Then the convolution integral $(k_\alpha * u)(t)$ can be approximated by

$$D^{-\alpha,n}u = (k_\alpha * \Pi^1 u)(t_n) = \sum_{j=1}^{n}(A_{n,j}^{(\alpha)}u^{j-1} + B_{n,j}^{(\alpha)}u^j), \tag{27}$$

where $u^j = u(t_j)$ and

$$\begin{aligned} A_{n,j}^{(\alpha)} &= \int_{t_{j-1}}^{t_j} k_\alpha(t_n - s)\frac{s-t_j}{t_{j-1}-t_j}\mathrm{d}s, \\ B_{n,j}^{(\alpha)} &= \int_{t_{j-1}}^{t_j} k_\alpha(t_n - s)\frac{s-t_{j-1}}{t_j-t_{j-1}}\mathrm{d}s. \end{aligned} \tag{28}$$

- $\alpha > 0$: (27) is the fractional trapezoidal rule.
- $\alpha < 0$: (27) is the popularly used L1 method for discretizing the RL fractional derivative.

The aim of this section is to design fast algorithms to calculate

$$(k_\alpha * \Pi^1 u)(t_n) = \int_{t_0}^{t_n} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s = \sum_{j=1}^{n}(A_{n,j}^{(\alpha)} u_{j-1} + B_{n,j}^{(\alpha)} u_j). \quad (29)$$

Divide $(k_\alpha * \Pi^1 u)(t_n)$ into two parts as

$$(k_\alpha * \Pi^1 u)(t_n) = \underbrace{\int_{t_{n-1}}^{t_n} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s}_{\text{Local part : } L^{\alpha,n}u} + \underbrace{\int_{0}^{t_{n-1}} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s}_{\text{History part : } H^{\alpha,n}u}, \quad (30)$$

where the local part $L^{\alpha,n}u$ and the history part $H^{\alpha,n}u$ can also be expressed as

$$L^{\alpha,n}u = A_{n,n}^{(\alpha)} u^{n-1} + B_{n,n}^{(\alpha)} u^n, \quad (31)$$

$$H^{\alpha,n}u = \sum_{j=1}^{n-1}(A_{n,j}^{(\alpha)} u^{j-1} + B_{n,j}^{(\alpha)} u^j). \quad (32)$$

Develop the fast memory-saving method to calculate the history part $H^{\alpha,n}u$.

The history part $H^{\alpha,n}u$ can be approximated via

$$
\begin{aligned}
H^{\alpha,n}u &= \underbrace{\sum_{j=1}^{n-1}(A_{n,j}^{(\alpha)}u^{j-1} + B_{n,j}^{(\alpha)}u^{j})}_{\text{Direct calculation}} = \int_0^{t_{n-1}} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s \\
&\approx \int_0^{t_{n-1}} \sum_{j=1}^{Q} w_j(\alpha)e^{\lambda_j(t_n-s)}\Pi^1 u(s)\mathrm{d}s \\
&= \sum_{j=1}^{Q} w_j(\alpha)e^{\lambda_j(t_n-t_{n-1})} \underbrace{\int_0^{t_{n-1}} e^{\lambda_j(t_{n-1}-s)}\Pi^1 u(s)\mathrm{d}s}_{Y_j(t_{n-1})} \\
&= \underbrace{\sum_{j=1}^{Q} w_j(\alpha)e^{\lambda_j(t_n-t_{n-1})}Y_j(t_{n-1})}_{\text{Fast calculation}} =: {}_F H^{\alpha,n}u,
\end{aligned}
\tag{33}
$$

where $Y_j(t)$ satisfies the following ODE

$$
Y_j'(t) = \lambda_j Y_j(t) + \Pi^1 u(t), \qquad Y_j(0) = 0. \tag{34}
$$

The above linear ODE has the analytical solution

$$Y_j(t) = e^{\lambda_j t} Y_j(0) + \int_0^t e^{\lambda_j (t-s)} \Pi^1 u(s) \mathrm{d}s,$$

which can be exactly solved by the following recursive relation

$$\begin{aligned} Y_j(t_m) &= e^{\lambda_j (t_m - t_{m-1})} Y_j(t_{m-1}) + \int_{t_{m-1}}^{t_m} e^{\lambda_j (t_m - t)} \Pi^1 u(t) \mathrm{d}t \\ &= \Lambda_1 Y_j(t_{m-1}) + \Lambda_2 u^{m-1} + \Lambda_3 u^m \end{aligned} \tag{35}$$

at $t = t_m$, where $Y_j(0) = 0$ and

$$\begin{cases} \Lambda_1 = e^{\lambda_j (t_m - t_{m-1})}, \\ \Lambda_2 = \dfrac{1}{\lambda_j} \left(e^z - 1 - \dfrac{e^z - z - 1}{z} \right), \quad z = \lambda_j (t_m - t_{m-1}), \\ \Lambda_3 = \dfrac{1}{\lambda_j} \dfrac{e^z - z - 1}{z}, \quad z = \lambda_j (t_m - t_{m-1}). \end{cases} \tag{36}$$

Algorithm 1: Fast calculation of the fractional operator $k_\alpha * u(t)$ based on the kernel approximation

$$k_\alpha(t) \approx \frac{t^{\alpha-1}}{\Gamma(\alpha)} = \sum_{j=1}^{Q} w_j(\alpha) e^{\lambda_j(\alpha)t}, \quad t \in [\delta, T] \tag{37}$$

Step 1. Divide $(k_\alpha * \Pi^1 u)(t_n)$ into two parts as

$$(k_\alpha * \Pi^1 u)(t_n) = \underbrace{\int_{t_{n-1}}^{t_n} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s}_{\text{Local part : } L^{\alpha,n}u} + \underbrace{\int_{0}^{t_{n-1}} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s}_{\text{History part : } H^{\alpha,n}u}.$$

Step 2. Calculate the local part $L^{\alpha,n}u$ directly by (31).

Step 3. Approximate the history part $H^{\alpha,n}u$ by

$${}_F H^{\alpha,n}u = \sum_{j=1}^{Q} w_j(\alpha) e^{\lambda_j(\alpha)(t_n - t_{n-1})} \boxed{Y_j(t_{n-1})}, \quad n \geq 2, \tag{38}$$

where ${}_F H^{\alpha,n}u = 0$ for $n \leq 1$, $Y_j(t_{n-1})$ satisfies (35).

Step 4. Output ${}_F D^{-\alpha,n}u = L^{\alpha,n}u + {}_F H^{\alpha,n}u$.

Algorithm 2: Fast calculation of the fractional operator $k_\alpha * u(t)$ based on the improved kernel approximation

$$k_\alpha(t) = \frac{t^{\alpha-1}}{\Gamma(\alpha)} = \frac{t}{\alpha-1} k_{\alpha-1}(t) \approx t \sum_{j=1}^{Q} \widehat{w}_j e^{\widehat{\lambda}_j t}, \quad t \in [\delta, T] \tag{39}$$

Step 1. Divide $(k_\alpha * \Pi^1 u)(t_n)$ into two parts as

$$(k_\alpha * \Pi^1 u)(t_n) = \underbrace{\int_{t_{n-1}}^{t_n} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s}_{\text{Local part : } L^{\alpha,n}u} + \underbrace{\int_{0}^{t_{n-1}} k_\alpha(t_n - s)\Pi^1 u(s)\mathrm{d}s}_{\text{History part : } H^{\alpha,n}u}.$$

Step 2. Calculate the local part $L^{\alpha,n}u$ directly.

Step 3. Approximate the history part $H^{\alpha,n}u$ by

$${}_F H^{\alpha,n}u = \sum_{j=1}^{Q} \widehat{w}_j(\alpha) e^{\widehat{\lambda}_j(t_n - t_{n-1})} \boxed{(t_n Y_j(t_{n-1}) - Z_j(t_{n-1}))}, \quad n \geq 2, \tag{40}$$

where ${}_F H^{\alpha,n}u = 0$ for $n \leq 1$, $Y_j(t_{n-1})$ satisfies (35).

Step 4. Output ${}_F D^{-\alpha,n}u = L^{\alpha,n}u + {}_F H^{\alpha,n}u$.

$$
\begin{aligned}
H^{\alpha,n}u &= \sum_{j=1}^{n-1}\left(A_{n,j}^{(\alpha)}u^{j-1}+B_{n,j}^{(\alpha)}u^{j}\right)=\int_0^{t_{n-1}}k_\alpha(t_n-s)\Pi^1u(s)\mathrm{d}s\\
&\approx \sum_{j=1}^{Q}\widehat{w}_j(\alpha)\int_0^{t_{n-1}}(t_n-s)e^{\widehat{\lambda}_j(t_n-s)}\Pi^1u(s)\mathrm{d}s\\
&= \sum_{j=1}^{Q}\widehat{w}_j(\alpha)e^{\widehat{\lambda}_j(t_n-t_{n-1})}\left(t_nY_j(t_{n-1})-Z_j(t_{n-1})\right)\\
&=: {}_FH^{\alpha,n}u,
\end{aligned}
\tag{41}
$$

where

$$
\begin{aligned}
Y_j(t_{n-1}) &= \int_0^{t_{n-1}}e^{\widehat{\lambda}_j(t_{n-1}-s)}\Pi^1u(s)\mathrm{d}s,\\
Z_j(t_{n-1}) &= \int_0^{t_{n-1}}e^{\widehat{\lambda}_j(t_{n-1}-s)}s\Pi^1u(s)\mathrm{d}s.
\end{aligned}
\tag{42}
$$

Application to FODEs

Consider the following fractional ODE (FODE)

$$\begin{cases} {}_C D_{0,t}^{\alpha(t)} u(t) = -u(t) + f(u,t), \quad 0 < \alpha(t) \le 1, \quad t \in (0,T], \\ u(0) = u_0, \end{cases} \tag{43}$$

where

$${}_C D_{0,t}^{\alpha(t)} u(t) = P.V.(k_{-\alpha(t)} * u(t)) - \frac{t^{-\alpha(t)}}{\Gamma(1-\alpha(t))} u(0).$$

The **fast method** for (43) reads: Find ${}_F U^n$ for $n \ge 1$ such that

$${}_F D^{\alpha_n,n}\, {}_F U - \frac{t_n^{-\alpha_n}}{\Gamma(1-\alpha_n)} {}_F U^0 = -{}_F U^n + f({}_F U^n, t_n), \quad {}_F U^0 = u_0. \tag{44}$$

The **direct method** for (43) reads: Find U^n for $n \ge 1$ such that

$$D^{\alpha_n,n} U - \frac{t_n^{-\alpha_n}}{\Gamma(1-\alpha_n)} U^0 = -U^n + f(U^n, t_n), \quad U^0 = u_0. \tag{45}$$

Error of the direct method (uniform time mesh $t_j = j\tau$)

$$R^n = {}_C D^{\alpha}_{0,t} u(t_n) - {}_C D^{\alpha}_{0,t} \Pi^1 u(t_n) = O(\tau^{2-\alpha}).$$

Theorem (Huang, Zeng, Guo, 2022)

Assume that

$$k_{-\alpha}(t) = \frac{t^{-\alpha-1}}{\Gamma(-\alpha)} = \sum_{j=1}^{Q} w_j e^{\lambda_j t} + O(\varepsilon t^{-\alpha-1}), \quad t \in [\delta, T]. \tag{46}$$

Let U^n and ${}_F U^n$ be the solutions of the direct method and fast method, respectively. If τ is sufficiently small and $\varepsilon/\tau^{\alpha} \lesssim 1$, then

$$|U^n - {}_F U^n| \lesssim \varepsilon n^{\alpha}, \quad 1 \le n \le M. \tag{47}$$

Hence,

$$|{}_F U^n - u(t_n)| \le \underbrace{|U^n - {}_F U^n|}_{\lesssim \varepsilon n^{\alpha}} + \underbrace{|U^n - u(t_n)|}_{\text{Error of the direct method} \le |R^n|}. \tag{48}$$

Semi-linear time-fractional subdiffusion equation

$$\begin{cases} {}_C D^{\alpha}_{0,t} u(x,t) = \partial_{xx} u(x,t) + f(u), \quad (x,t) \in \Omega \times (0,T], \\ u(x,0) = u_0(x), \quad x \in \bar{\Omega}, \\ u(x,t) = 0, \quad (x,t) \in \partial\Omega \times [0,T]. \end{cases} \tag{49}$$

Fast FDM for (49) is given by: For $1 \le n \le n_T$, find ${}_F U_j^n$ such that

$$\begin{cases} {}_F D^{\alpha_n,n}\, {}_F U_j^n = \delta_x^2 {}_F U_j^n + f({}_F U_j^n), \quad 1 \le j \le M-1, \\ {}_F U_0^n = {}_F U_M^n = 0, \quad {}_F U_j^0 = u_0(x_j), \quad 0 \le j \le M. \end{cases} \tag{50}$$

Direct FDM for (49) is given by: For $1 \le n \le n_T$, find U_j^n such that

$$\begin{cases} D^{\alpha_n,n} U_j^n = \delta_x^2 U_j^n + f(U_j^n), \quad 1 \le j \le M-1, \\ U_0^n = U_M^n = 0, \quad U_j^0 = u_0(x_j), \quad 0 \le j \le M. \end{cases} \tag{51}$$

Theorem

Let u and U_j^n be the solutions of (49) and (51), respectively, $u(\cdot,t) \in C^4(\Omega)$, and $f(z)$ satisfies the local Lipschitz condition. If τ is sufficiently small, then

$$\|u(\cdot,t_n) - U^n\|_{L^p} \quad \lesssim \quad \tau t_n^{\alpha-1} + h^2 t_n^{\alpha}, \quad 1 \le n \le M,$$

where $p = 2, \infty$.

See [Dongfang Li et al., JSC, 2022; Natalia Kopteva and Xiangyun Meng, SINUM, 2020].

Theorem (Huang, Zeng, Guo, 2022)

Let U_j^n and ${}_FU_j^n$ be the solutions of (51) and (50), respectively, $f(z)$ satisfies the local Lipschitz condition. If τ is sufficiently small and $\varepsilon/\tau^\alpha \lesssim 1$, then

$$\|{}_FU^n - U^n\|_{L^p} \lesssim \varepsilon n^\alpha, \quad 1 \le n \le M,$$

where $p = 2, \infty$.

Furthermore,

$$\|{}_FU^n - u(\cdot, t_n)\|_{L^p} \lesssim \left(\tau t_n^{\alpha-1} + h^2 t_n^\alpha\right) + \varepsilon n^\alpha, \quad 1 \le n \le M,$$

where $p = 2, \infty$.

Nonuniform grids

$t_j = T(j/M)^r$ $\quad$ $r = 1$ reduces to the uniform grid.

Theorem (Yang, Zeng, 2022)

Let U_j^n and ${}_FU_j^n$ be the solutions of (51) and (50), respectively, $f(z)$ satisfies the local Lipschitz condition. If $\varepsilon M^\alpha \lesssim 1$, then

$$\|{}_FU^n - U^n\|_{L^p} \lesssim \varepsilon M^\alpha t_n^{\alpha/r} \lesssim \varepsilon n^\alpha, \quad 1 \le n \le M,$$

where $p = 2, \infty$.

Furthermore,

$$\|{}_FU^n - u(\cdot, t_n)\|_{L^p} \lesssim \left(\textit{temporal error} + h^2 t_n^\alpha\right) + \varepsilon n^\alpha, \quad 1 \le n \le M.$$

Numerical results

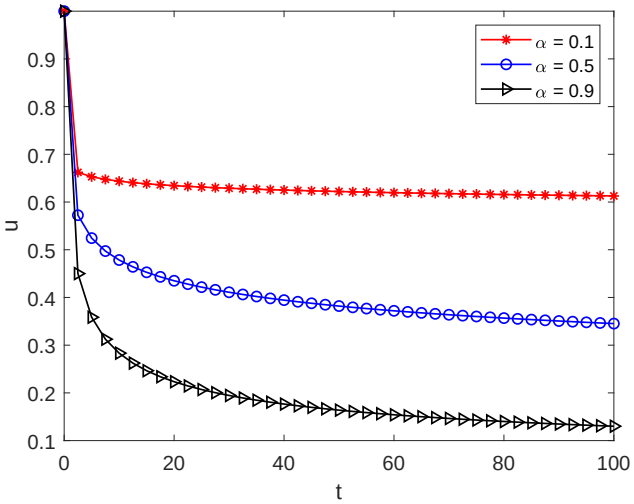
Consider the following fractional ODE

$$ {}_{C}D_{0,t}^{\alpha(t)}u(t) = -u(t) + f(u,t), \quad u(0) = u_0, \quad t \in (0,T], \tag{52} $$

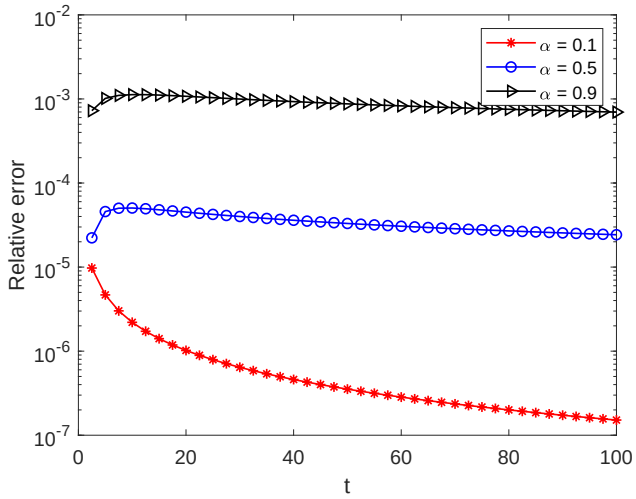
where $0 < \alpha(t) \leq 1$.

Choose the nonlinear term $f = u(1-u^2)$ and the initial value $u_0 = 1$.

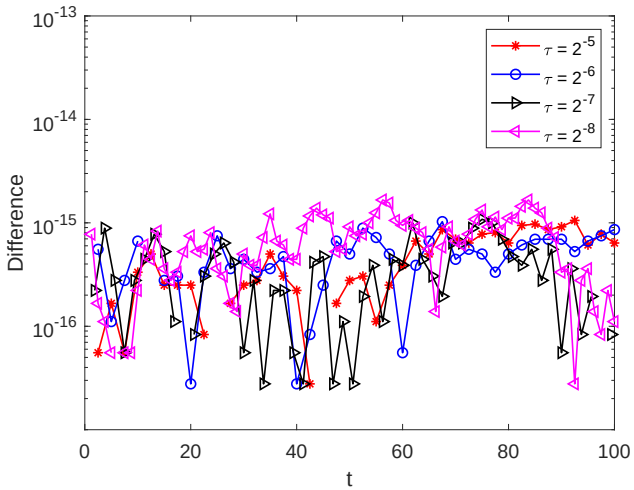
Compute the reference solutions U_{ref}^n with a smaller step size $\tau = 10^{-3}$.



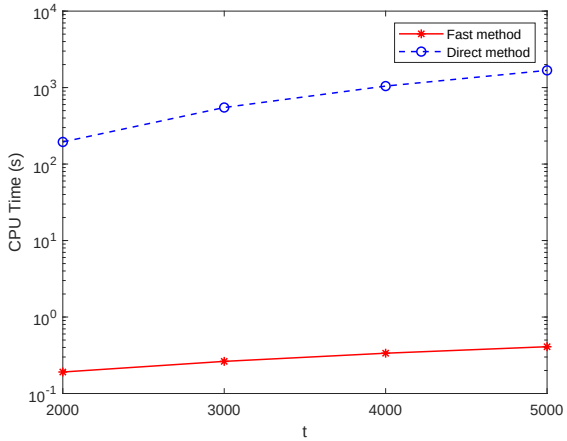
(a) Numerical solutions.



(b) The relative errors.



(c) Difference $\frac{|U^n - {}_F U^n|}{|U^n|+1}$.



(d) Computational time.

Figure: (a) Numerical solutions for $\alpha = 0.1, 0.5, 0.9$. (b) the errors between numerical solutions and reference solutions for different

Consider the following time-fractional Allen–Cahn equation

$$\begin{cases} {}_{C}D^{\alpha}_{0,t}u(\mathbf{x},t) = \gamma\Big(\varepsilon\Delta u(\mathbf{x},t) - \varepsilon^{-1}F'(u)\Big), & (\mathbf{x},t) \in \Omega \times (0,T], \\ u(\mathbf{x},0) = u_0(\mathbf{x}), & \mathbf{x} \in \Omega, \end{cases} \tag{53}$$

subject to periodic boundary conditions, where $\alpha \in (0,1]$, $\Omega \subset \mathbb{R}^d$, $\mathbf{x} = (x,y)$ for $d = 2$, $\mathbf{x} = (x,y,z)$ for $d = 3$, ε is the thickness of phase interface, γ is mobility constant, and F is defined by

$$F(u) = \frac{1}{4}(1-u^2)^2.$$

The computational domain $\Omega = (-1, 1)^2$ is partitioned uniformly into 128×128 subdomains, $\varepsilon = 0.02$, and $\gamma = 0.02$. The initial condition is taken as

$$u_0 = -\tanh\left(\left((x-0.3)^2 + y^2 - 0.2^2\right)/\varepsilon\right)\tanh\left(\left((x+0.3)^2 + y^2 - 0.2^2\right)/\varepsilon\right)$$
$$\times \tanh\left(\left(x^2 + (y-0.3)^2 - 0.2^2\right)/\varepsilon\right)\tanh\left(\left(x^2 + (y+0.3)^2 - 0.2^2\right)/\varepsilon\right).$$

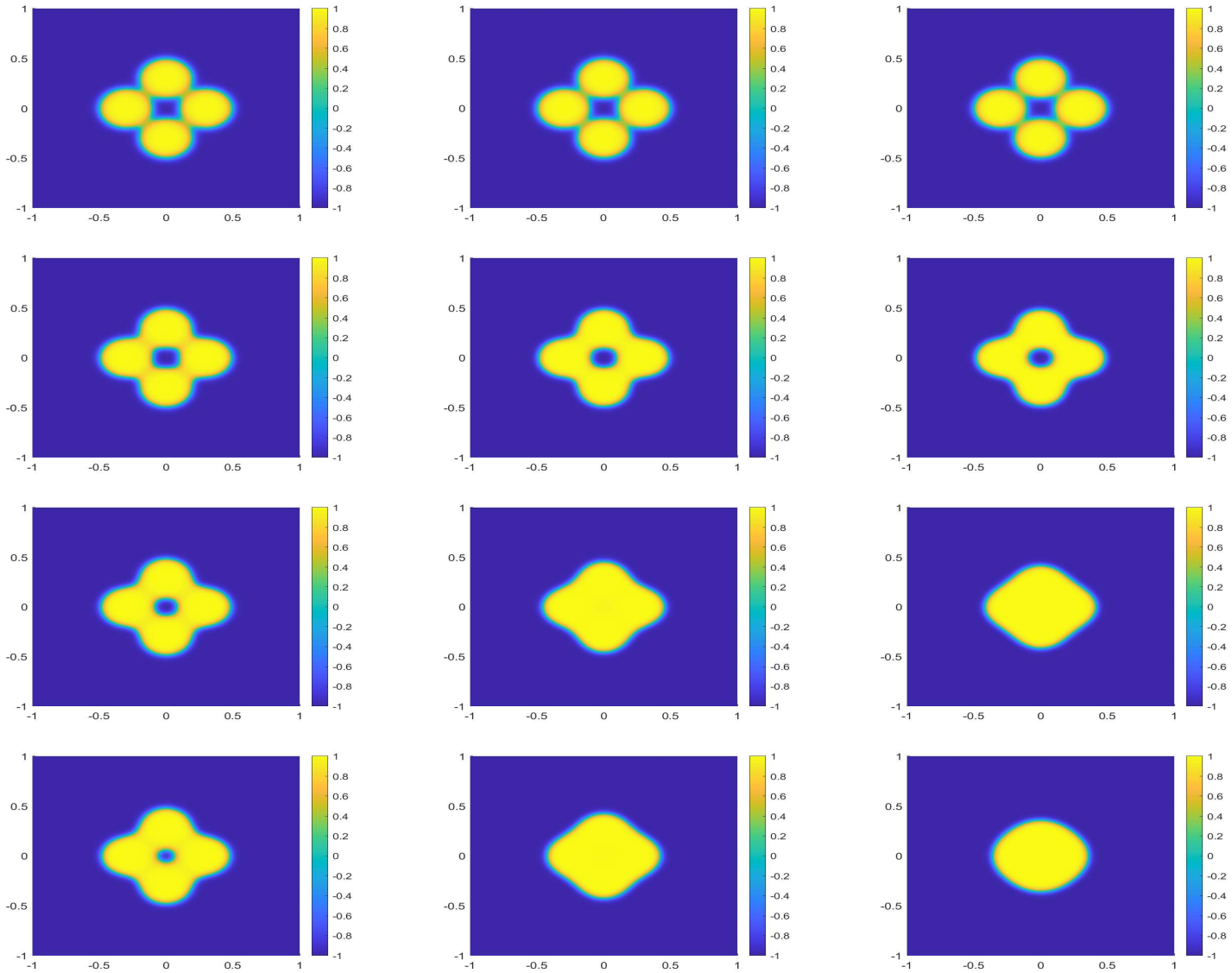


Figure: Numerical solutions of the Allen–Cahn equation (53). Solution snapshots at $t = 1, 10, 50, 100$ (from top to bottom) for three fractional orders $\alpha = 0.4, 0.7, 0.9$ (from left to right).

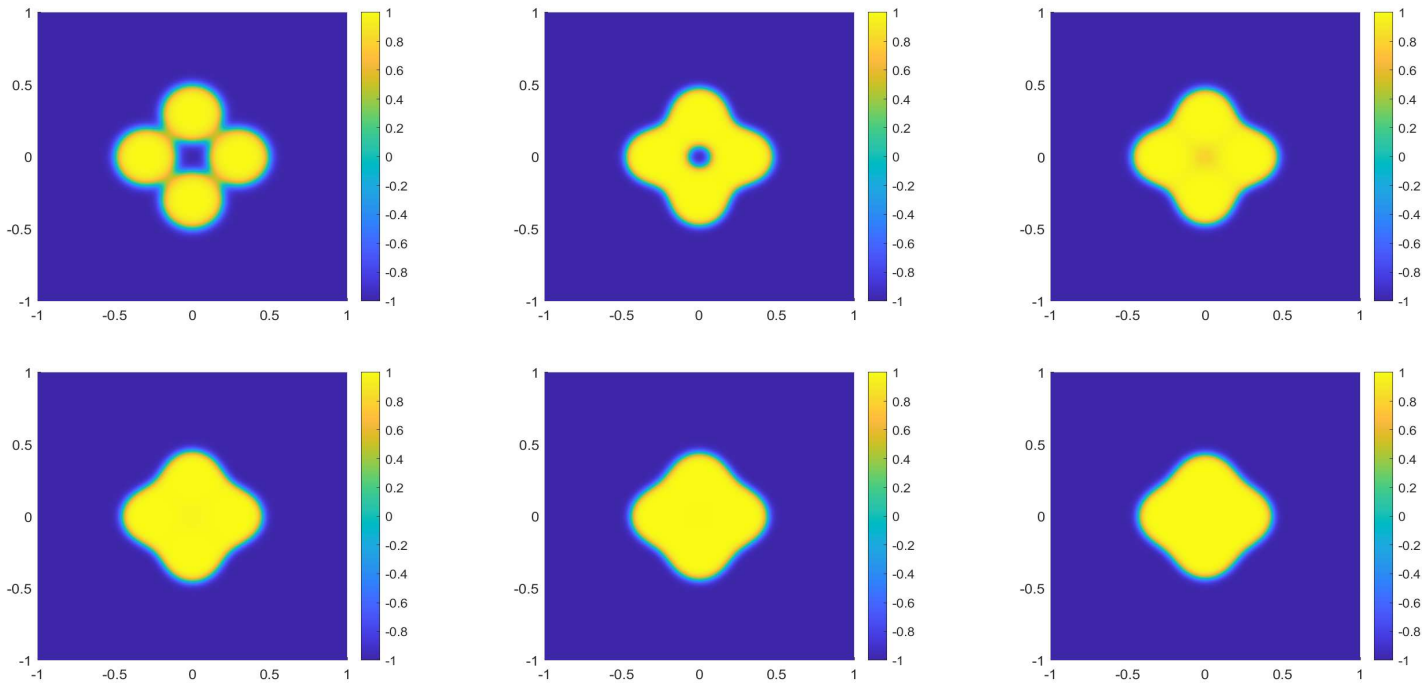
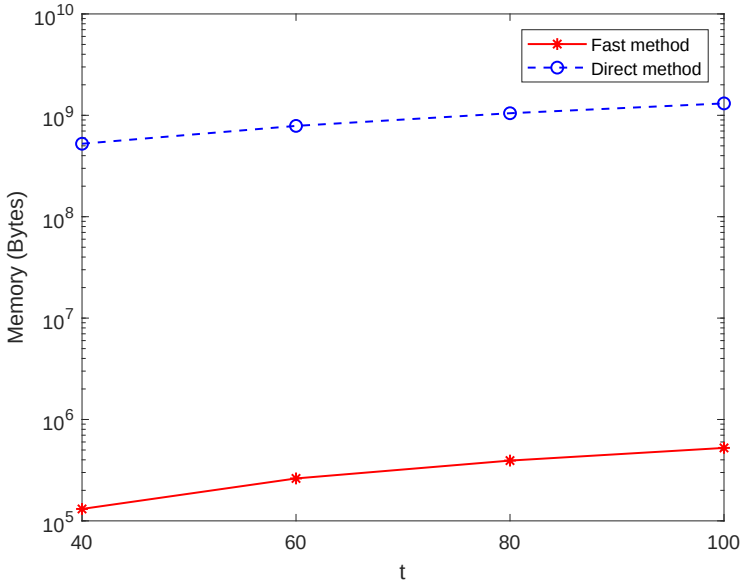
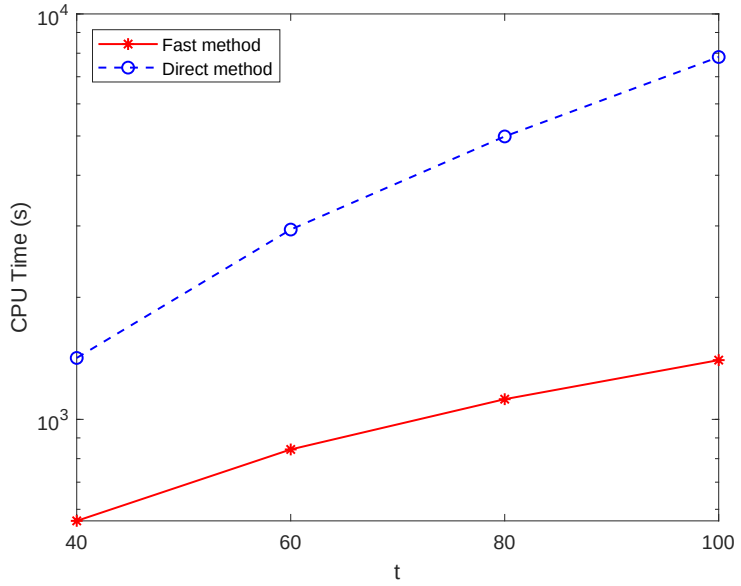


Figure: Numerical solutions of (53) with variable-order $\alpha(t) = 0.4 + \sin^2(t)/2$. Solution snapshots at $t = 1, 30, 50$ (top: from left to right) and $t = 100, 150, 200$ (bottom: from left to right).



(a) The active memory requirement.



(b) The computational time.

Figure: The comparison of the fast method (red stars) and direct method (blue circles) for solving two-dimensional Allen–Cahn equation (53) with the time step size $\tau = 0.01$, the spatial step size $h = 1/64$, and $\alpha = 0.9$. We can see that the proposed unified fast method can significantly reduce the memory and thus use much less time than that of the direct method.

Three-dimensional time-fractional Allen–Cahn equation (53) subject to the following initial condition

$$u_0 = \tanh\left(\frac{1 + 0.2\cos(6\theta) - \sqrt{x^2 + 2y^2 + z^2}}{\sqrt{2}\varepsilon}\right),$$

where $\theta = \tan^{-1}(z/x)$. The computational domain $\Omega = (-1.5, 1.5)^3$ is divided uniformly into $32 \times 32 \times 32$ subdomains, $\varepsilon = 0.05$, and $\gamma = 0.05$. The time step size is taken as $\tau = 0.01$.

The initial condition is a star-shaped ball, which becomes smaller as the time t increases for $\alpha = 0.4, 0.7, 0.9$. For a fixed time t, the ball becomes smaller as the fractional order α becomes larger.

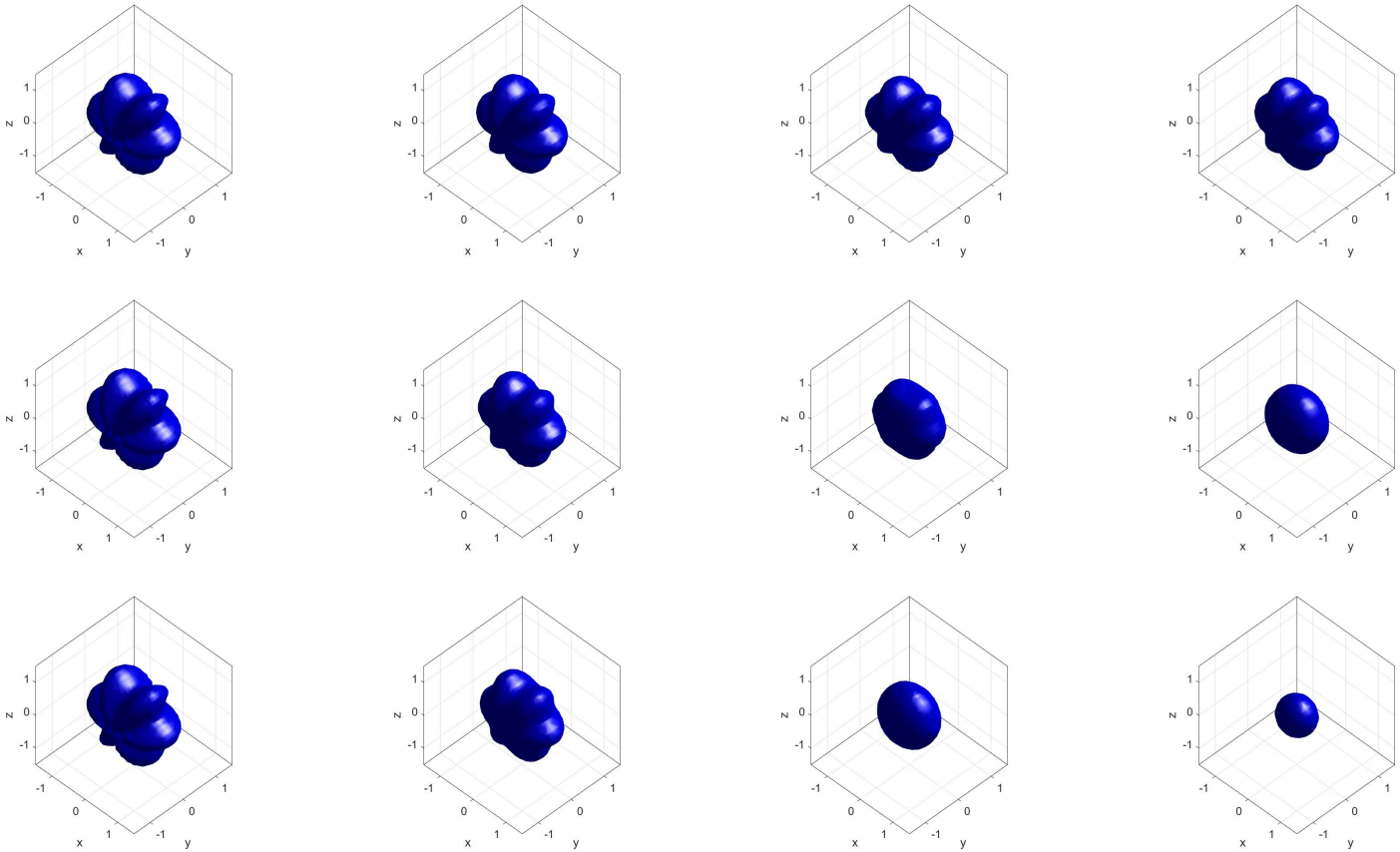


Figure: Numerical solutions of the three-dimensional Allen–Cahn equation (53). Solution snapshots at $t = 1, 10, 50, 100$ (from left to right) for three fractional orders $\alpha = 0.4, 0.7, 0.9$ (from top to bottom).

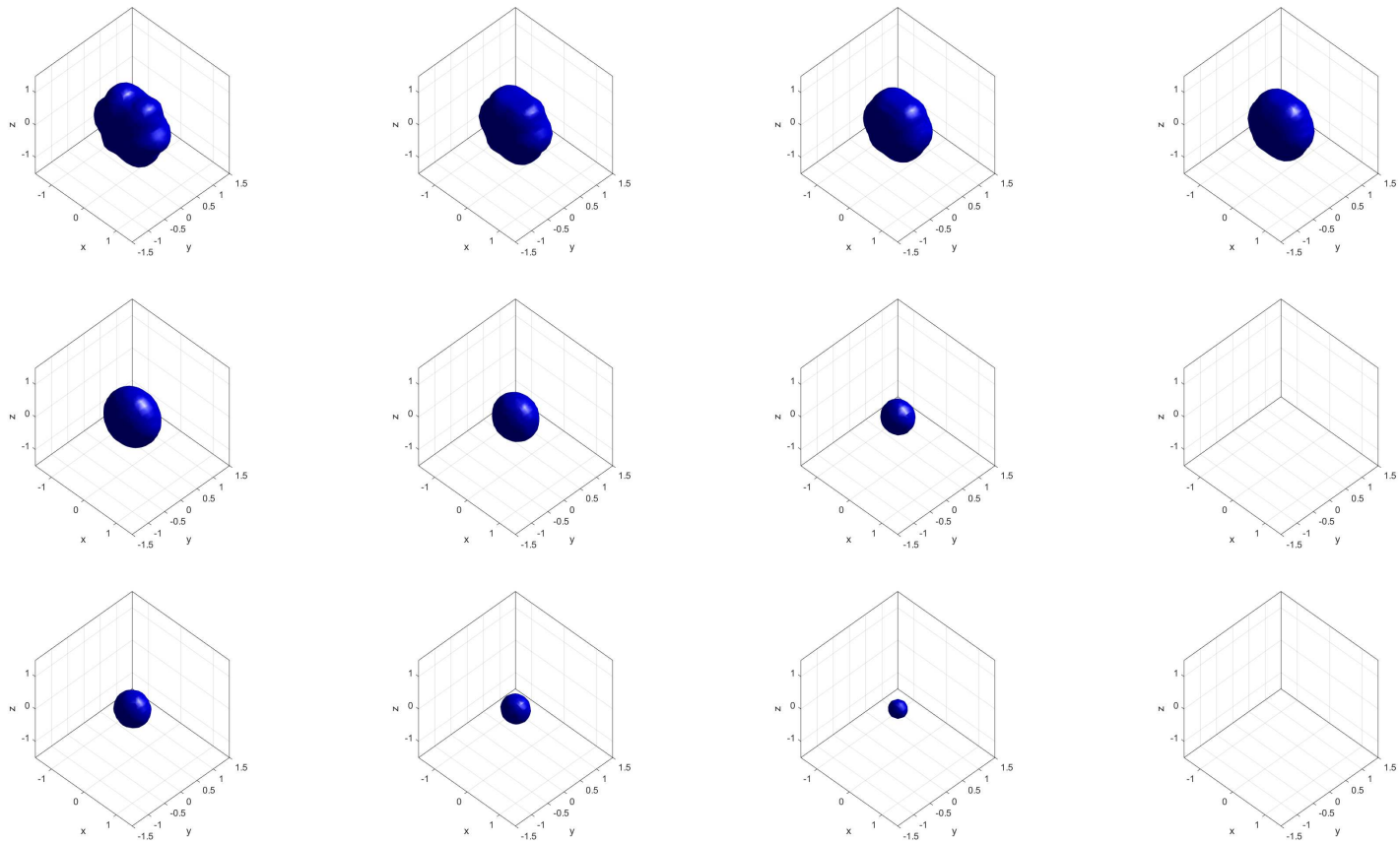


Figure: Snapshots of numerical solutions of the three-dimensional Allen–Cahn equation (53). Top: $\alpha = 0.4$, $t = 300, 500, 800, 1000$ (from left to right); center: $\alpha = 0.7$, $t = 140, 200, 250, 300$ (from left to right); bottom: $\alpha = 0.9$, $t = 110, 120, 130, 140$ (from left to right).

Consider the following one-dimensional Schrödinger equation

$$\begin{cases} \mathrm{i}u_t(x,t) = -u_{xx}(x,t), & \text{for } t>0, \quad x\in\mathbb{R}, \\ u(x,0) = u_0(x), & \text{for } x\in\mathbb{R}. \end{cases} \tag{54}$$

In applications, we are concerned with the solution on the bounded domain $x\in\Omega$, for example, $\Omega=[-3,3]$. In such a case, (54) is equivalent to a time-dependent Schrödinger equation with nonreflecting boundary conditions [**?**]

$$\begin{cases} \mathrm{i}u_t(t) = -u_{xx}(t), & \text{for } t>0, \\ u(x,t) = e^{\frac{\pi}{4}\mathrm{i}}(k_{1/2}*u_x)(t), & \text{at } x=-3, \\ u(x,t) = -e^{\frac{\pi}{4}\mathrm{i}}(k_{1/2}*u_x)(t), & \text{at } x=+3, \\ u(x,0) = u_0(x), & \text{for } x\in[-3,3], \end{cases} \tag{55}$$

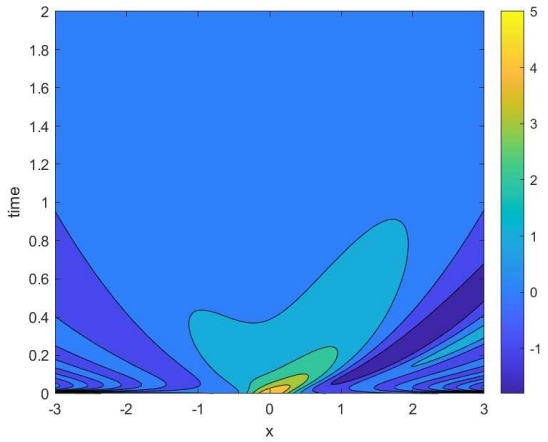
where $\mathrm{i}^2=-1$ and $u(t)=u(x,t), u_x(t)=u_x(x,t)=\partial_x u(x,t)$.

The Crank–Nicolson method is applied to the first equation in (55), the fractional integral operator in the second or third equation in (55) is discretized by the fast Algorithm 1, we obtain a semi-discrete method for (55) as follows: Find $U^n = U(\cdot, t_n)$ for $n \geq 1$ such that

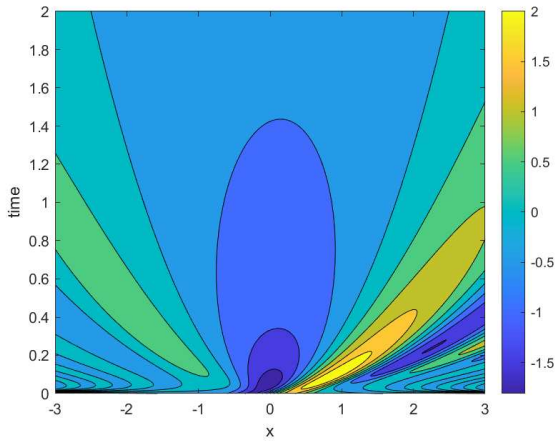
$$\begin{cases} i\dfrac{U^n - U^{n-1}}{t_n - t_{n-1}} = -\dfrac{U^n_{xx} + U^{n-1}_{xx}}{2}, & \text{for } x \in (-3, 3), \\ U^n(\pm 3) = \mp e^{\frac{\pi}{4}\mathrm{i}}{}_F D^{-1/2,n} U_x(\pm 3), & \\ U^0(x) = u_0(x), & \text{for } x \in [-3, 3]. \end{cases} \tag{56}$$

The initial data $u(x, 0) = \frac{1}{\sqrt{\xi}} e^{\mathrm{i}kx - x^2/(4\xi)}$. The exact solution of (54) is

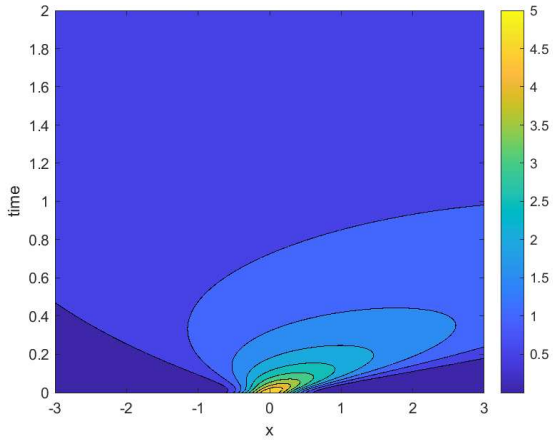
$$u(x, t) = \frac{1}{\sqrt{\xi + \mathrm{i}t}} e^{\mathrm{i}k(x-kt) - (x-2kt)^2/4(\xi+\mathrm{i}t)}, \quad \xi, k \in \mathbb{R}.$$



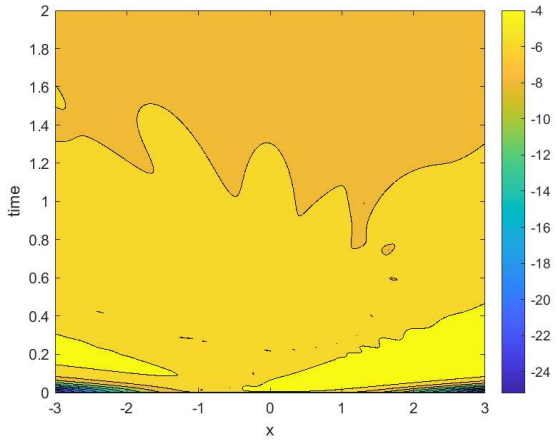
(a) Real part.



(b) Imaginary part.

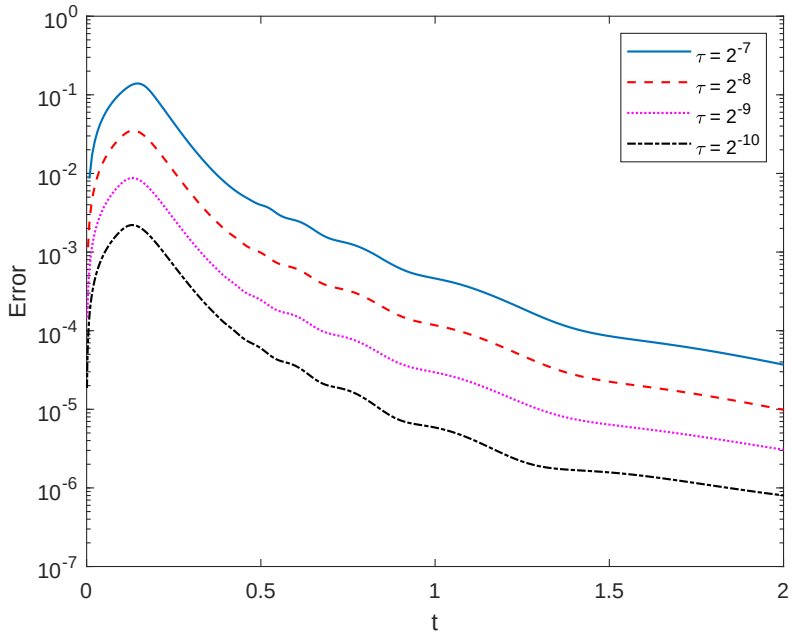


(c) Absolute values.

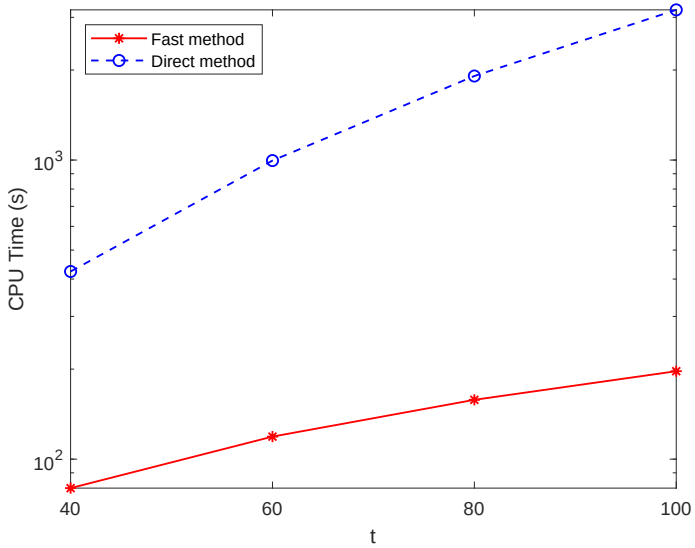


(d) Point-wise errors.

Figure: Numerical solutions of the Schrödinger equation (54) on the interval $[-3, 3]$, $h = \tau = 0.001, \xi = 0.04$, $k = 2$. (a) The real part of the numerical solution; (b) The imaginary part of the numerical solution; (c)



(a) The L^2 error at $t = 2$.



(b) The computational time.

Figure: (a) The L^2 errors of the numerical solution under different time step size; (b) Comparison of the computational time between the fast method and the direct method.

Refences

[1] Yuxiang Huang, Qiaoge Li, Rongxin Li, Fanhai Zeng, Ling Guo, A unified fast memory-saving time-stepping method for fractional operators and its applications, Numer. Math. Theor. Meth. Appl., 2022.

[2] Yuxiang Huang, Fanhai Zeng, Ling Guo, Convergence analysis of the fast L1 method for time-fractional subdiffusion equations, Appl. Math. Lett., 2022.

[3] Zheng Yang, Fanhai Zeng, A corrected L1 method for a time-fractional subdiffution equation, 2022.

Thanks!