



A preconditioned fast finite element approximation to variable-order time-fractional diffusion equations in multiple space dimensions

Jinhong Jia^a, Hong Wang^{b,*}, Xiangcheng Zheng^{b,c,*}

^a School of Mathematics and Statistics, Shandong Normal University, Jinan, Shandong 250358, China

^b Department of Mathematics, University of South Carolina, Columbia, SC 29208, USA

^c School of Mathematical Sciences, Peking University, Beijing 100871, China

ARTICLE INFO

Article history:

Received 4 November 2019

Received in revised form 15 December 2020

Accepted 2 January 2021

Available online 11 January 2021

Keywords:

Variable-order time-fractional diffusion equation

Finite element method

Divided-and-conquer algorithm

Toeplitz matrix

ABSTRACT

We develop a preconditioned fast divided-and-conquer finite element approximation for the initial-boundary value problem of variable-order time-fractional diffusion equations. Due to the impact of the time-dependent variable order, the coefficient matrix of the resulting all-at-once system does not have a Toeplitz-like structure. In this paper we derive a fast approximation of the coefficient matrix by the means of a sum of Toeplitz matrices multiplied by diagonal matrices. We show that the approximation is asymptotically consistent with the original problem, which requires $O(MN \log^3 N)$ computational complexity and $O(MN \log^2 N)$ memory with M and N being the numbers of degrees of freedom in space and time, respectively. Furthermore, a preconditioner is introduced to reduce the number of iterations caused by the bad condition number of the coefficient matrix. Numerical experiments are presented to demonstrate the effectiveness and the efficiency of the proposed method.

© 2021 IMACS. Published by Elsevier B.V. All rights reserved.

1. Introduction

Integer-order diffusion partial differential equations were derived under the assumptions that the underlying particle movements have a mean free path and a mean waiting time [27,28], which hold for the diffusive transport of solutes in homogeneous aquifers when the solute plumes were observed to exhibit a Gaussian type exponentially decaying tails [3]. However, field tests showed that the diffusive transport of solutes in heterogeneous aquifers often exhibits highly skewed power-law decaying tails [4,5,27,28]. The traditional approach is to tweak the variable parameters in pre-set integer-order diffusion equations to address the impact of heterogeneity of the media. However, this approach tends to recover a rapidly varying, scale-dependent diffusivity and may overfit the training data but yield less accurate prediction on testing data [30], as the Gaussian type solutions of integer-order diffusion equations cannot accurately capture the highly skewed power-law decaying behavior of the solute transport in heterogeneous media.

Fractional diffusion equations (FDEs) were derived so that their solutions can accurately capture the highly skewed power-law decaying tails that were observed in the solute transport in heterogeneous media [27,28]. Consequently, sig-

* Corresponding authors.

E-mail addresses: jhjia@sdnu.edu.cn (J. Jia), hwang@math.sc.edu (H. Wang), xz3@math.sc.edu (X. Zheng).

nificant research has been carried out in the numerical approximations to FDEs and the corresponding numerical analysis [7–11,14,15,17,24,26,36,40,42]. However, FDEs introduce new mathematical, numerical and computational difficulties that are not common for integer-order diffusion equations. For instance, the first-order time derivative of the solutions to the time-fractional diffusion equations (tFDEs) exhibits singularity near the initial time $t = 0$, which makes the error estimates in the literature that were proved under full regularity assumptions of the true solutions inappropriate [29,32].

It was shown that the occurrence of the nonphysical solutions to tFDEs at the initial time $t = 0$ roots from the incompatibility between the nonlocality of the power law decaying tails and the locality of the classical initial condition [2,38]. It was proved in [38] that tFDEs in which the order varies smoothly to the integer value at the time $t = 0$ can eliminate the nonphysical singularity of the solutions to constant-order tFDEs and have solutions with full regularity. Furthermore, variable-order tFDEs occur in many applications [25,33,41,44], as the order relates to the fractal dimension of the porous media determined by the Hurst index [27] that changes as the geometrical structure of the media changes. Different numerical approximations were developed and analyzed [31,34,41,44]. In particular, an optimal-order error estimate for a finite element approximation to variable-order tFDEs was proved in [43] without an artificial assumption of the smoothness of the true solutions.

Because of the memory property of time-fractional differential operators, numerical discretizations of tFDEs at each time step require the storage and numerical evaluation of the numerical solutions at all the previous time steps. Consequently, a direct numerical approximation to tFDEs typically has $O(MN)$ memory requirement and $O(MN^2)$ computational complexity, where M and N are the numbers of spatial unknowns and time steps, respectively. The significantly increased computational complexity and memory requirement in the numerical simulations of tFDEs makes the FDE modeling in real applications computationally intractable.

There has been extensive research on the development of fast numerical solution techniques for tFDEs. Motivated by the Toeplitz-like matrix splitting that was originally discovered by some of the authors in the context of spatial FDEs [37], some of the authors developed a Toeplitz-like matrix splitting based all-at-once preconditioned fast solution method to solve space-time FDEs in a space-time fully coupled manner [13]. All-at-once solution technique was subsequently developed by others [6]. The divide-and-conquer (DAC) method is a recursive solution method based on the all-at-once linear algebraic system [12,23]. These methods have a greatly reduced computational complexity of $O(MN \log^2 N)$. An alternative class of fast numerical methods is the exponential sum approximation based algorithms [16,20,39]. In these methods, a sum-of-exponentials (SOE) approximation for the singular kernel in the fractional differential operator was derived to compute the fractional derivative in $O(\log N)$ or $O(\log^2 N)$ evaluations. Consequently, these methods can solve tFDEs in $O(MN \log^2 N)$ computations.

Nevertheless, these fast methods do not apply to variable-order tFDEs. The coefficient matrices of the numerical schemes to variable-order tFDEs lose the Toeplitz-like structure, which was crucial in the development of the all-at-once methods [6,13] and DAC methods [12,23], while the exponential sum approximation based algorithms were derived via Laplace transform of the singular kernels that cannot be carried out for variable-order tFDEs [20,39].

Concerning the aforementioned issues, the main objective of this work is to find a fast numerical approximation to variable-order tFDEs via different approaches. The basic idea is to perform the Taylor expansion on each entry of the coefficient matrix, which includes the variable order $\alpha(t)$, at a fixed order $\bar{\alpha}$. Consequently, the coefficient matrix is approximated by a sum of Toeplitz matrices multiplied by diagonal matrices. We prove that it suffices to keep the first $O(\log N)$ terms in the expansion in order to retain the first order accuracy in time, which, together with the preconditioned DAC technique employing the Toeplitz structure of the approximating matrices, reduce the computational complexity from $O(MN^2)$ to $O(MN \log^3 N)$.

The rest of the paper is organized as follows. In Section 2 we present the model problem and its numerical discretization. In Section 3 we approximate the coefficient matrix by a sum of Toeplitz-like matrices and analyze its asymptotic consistency. In Section 4 we develop a fast DAC method for the approximated system, based on which a preconditioned fast DAC method is subsequently proposed. We perform numerical experiments to test the performance of the method in the last section.

2. A variable-order tFDE and its finite element approximation

In this section we go over the variable-order tFDE model and its finite element approximation.

2.1. On tFDE modeling issues

We begin with the widely used tFDE model

$$\begin{aligned} {}^C D_t^\alpha u - \nabla \cdot (\mathbf{K}(\mathbf{x}) \nabla) u &= f(\mathbf{x}, t), \quad (\mathbf{x}, t) \in \Omega \times (0, T], \quad 0 < \alpha < 1, \\ u(\mathbf{x}, 0) &= u_0(\mathbf{x}), \quad \mathbf{x} \in \Omega; \quad u(\mathbf{x}, t) = 0, \quad (\mathbf{x}, t) \in \partial\Omega \times (0, T], \end{aligned} \quad (1)$$

which was derived via a continuous time random walk (CTRW) assuming that the mean waiting time has a power-law decaying tail [27,28]. Consequently, as field tests showed that solute transport in heterogeneous aquifers often exhibits highly skewed and power-law decaying behavior and model (1) was derived under the assumption that the solutions have such behavior [5], model (1) accurately models the subdiffusive transport.

In model (1), $\Omega \in \mathbb{R}^d$ ($d = 1, 2, 3$) refers to a bounded domain with the piecewise smooth boundary $\partial\Omega$, $\mathbf{x} := (x_1, \dots, x_d)$, $\nabla := (\partial/\partial x_1, \dots, \partial/\partial x_d)^T$, $\mathbf{K}(\mathbf{x}) := (k_{i,j}(\mathbf{x}))_{i,j=1}^d$ is a symmetric and positive-definite matrix with uniform lower and upper bounds, the Caputo fractional differential operator ${}_0^C D_t^\alpha$ is defined by [27,28]

$$\begin{aligned} {}_0 I_t^\alpha g(t) &:= \frac{1}{\Gamma(\alpha)} \int_0^t \frac{g(s)}{(t-s)^{1-\alpha}} ds, \\ {}_0^C D_t^\alpha g(t) &:= {}_0 I_t^{1-\alpha} g'(t) = \frac{1}{\Gamma(1-\alpha)} \int_0^t \frac{g'(s)}{(t-s)^\alpha} ds, \end{aligned} \quad (2)$$

where $\Gamma(\cdot)$ refers to the Gamma function. The tFDE (1) was derived as the diffusion limit of a CTRW, as the number of particle jumps tends to infinity. Hence, it holds for relatively large time $t > 0$, rather than all the way up to the time $t = 0$ as often assumed in the literature. This explains why the solutions to problem (1) exhibit nonphysical singularity near the initial time $t = 0$.

The following variable-order tFDE model was analyzed in [38]

$$\begin{aligned} u_t + q(t) {}_0^R D_t^{1-\alpha(t)} u - \nabla \cdot (\mathbf{K}(\mathbf{x}) \nabla) u &= f(\mathbf{x}, t), \quad (\mathbf{x}, t) \in \Omega \times (0, T], \\ u(\mathbf{x}, 0) &= u_0(\mathbf{x}), \quad \mathbf{x} \in \Omega; \quad u(\mathbf{x}, t) = 0, \quad (\mathbf{x}, t) \in \partial\Omega \times (0, T]. \end{aligned} \quad (3)$$

Here u_t denotes the first-order partial derivative in time, the variable-order Riemann-Liouville fractional derivative is defined by [25,41,44]

$${}_0^R D_t^{1-\alpha(t)} g(t) := \left[\frac{1}{\Gamma(\alpha(t))} \frac{d}{d\xi} \int_0^\xi \frac{g(s)}{(\xi-s)^{1-\alpha(t)}} ds \right]_{\xi=t}.$$

The variable fractional integral and Caputo fractional differential operator can be defined by replacing the α in (2) by $\alpha(t)$.

A constant-order two time-scale mobile-immobile tFDE was presented in [30] to model the diffusive transport of the solute in heterogeneous porous media. In this case a large portion of solute may get absorbed to the media and deviates from the transport of the solute in the bulk fluid phase. The absorbed solute may get slowly released back to the bulk fluid phase that leads to a subdiffusive transport process. The variable-order tFDE model (3), was motivated by the work in [30] and accounts for the impact of the deformation of the porous media reflected in the change of the fractal dimension or Hurst index of the media. In the model the u_t term models the Fickian diffusive transport of the solute in the bulk fluid phase, which consists of $1/(1+q(t))$ portion of the total solute mass, and the $q(t) {}_0^R D_t^{1-\alpha} u$ term models the subdiffusive transport of the solute absorbed to the solid matrix, which is $q(t)/(1+q(t))$ portion of the total solute mass. Note that the governing tFDE (3) holds on the entire time interval including the initial time $t = 0$.

2.2. A finite element approximation

We utilize the following relation between the variable-order Riemann-Liouville and Caputo fractional derivatives [41,44]

$${}_0^R D_t^{1-\alpha(t)} g(t) = {}_0^C D_t^{1-\alpha(t)} g(t) + \frac{g(0)t^{\alpha(t)-1}}{\Gamma(\alpha(t))}, \quad t \in (0, T]$$

to rewrite the variable-order Riemann-Liouville tFDE model (3) in the following Caputo form

$$\begin{aligned} u_t + q(t) {}_0^C D_t^{1-\alpha(t)} u - \nabla \cdot (\mathbf{K}(\mathbf{x}) \nabla) u &= -\frac{q(t)u_0(\mathbf{x})t^{\alpha(t)-1}}{\Gamma(\alpha(t))} + f(\mathbf{x}, t), \\ u(\mathbf{x}, 0) &= u_0(\mathbf{x}), \quad \mathbf{x} \in \Omega; \quad u(\mathbf{x}, t) = 0, \quad (\mathbf{x}, t) \in \partial\Omega \times (0, T]. \end{aligned} \quad (4)$$

It was proved in [38] that if the variable order $\alpha(t)$ satisfies that $\alpha(0) = 1$, $\alpha'(0) = 0$ and $\lim_{t \rightarrow 0^+} \alpha'(t) \ln t = 0$ in addition to the usual smoothness assumption on the variable order $\alpha(t)$ as well as the coefficients, right-hand side and the initial data, the problem (4) has a unique solution that has the full regularity.

For $M, N \in \mathbb{N}^+$, let $t_i := i\tau$ for $i = 0, 1, \dots, N$ with $\tau = T/N$ be a uniform partition of $[0, T]$ and define a quasi-uniform partition of Ω with the diameter h and M interior nodes. Let S_h be the finite element space of continuous and piecewise-linear functions on Ω_h with the nodal basis $\{\phi_i(\mathbf{x})\}_{i=1}^M$.

We discretize u_t and ${}_0^C D_t^{1-\alpha(t)} u$ at $t = t_n$ by the backward Euler and the L^1 discretization, respectively

$$\begin{aligned} u_t(\mathbf{x}, t_n) &\approx \delta_t u(\mathbf{x}, t_n) := \frac{u(\mathbf{x}, t_n) - u(\mathbf{x}, t_{n-1})}{\tau}, \\ {}_0^C D_t^{1-\alpha(t_n)} u(\mathbf{x}, t_n) &= \frac{1}{\Gamma(\alpha(t_n))} \int_0^{t_n} \frac{u_s(\mathbf{x}, s)}{(t_n - s)^{1-\alpha(t_n)}} ds \\ &\approx \delta_t^{1-\alpha(t_n)} u(\mathbf{x}, t_n) = \frac{1}{\Gamma(\alpha(t_n))} \sum_{k=1}^n \int_{t_{k-1}}^{t_k} \frac{\delta_k u(\mathbf{x}, t_k)}{(t_n - s)^{1-\alpha(t_n)}} ds \\ &= \frac{1}{\Gamma(1 + \alpha(t_n))} \sum_{k=1}^n \left[(t_n - t_{k-1})^{\alpha(t_n)} - (t_n - t_k)^{\alpha(t_n)} \right] \delta_t u(\mathbf{x}, t_k). \end{aligned} \quad (5)$$

We multiply (4) by any $\chi_h \in S_h$, integrate the resulting equation on Ω , and replace $u_t(\mathbf{x}, t_n)$, ${}_0^C D_t^{1-\alpha(t_n)} u(\mathbf{x}, t_n)$ by $\delta_t u(\mathbf{x}, t_n)$ and $\delta_t^{1-\alpha(t_n)} u(\mathbf{x}, t_n)$, respectively, and then to replace $u(\mathbf{x}, t_n)$ by the trial function

$$u_h^n := \sum_{i=1}^M u_i^n \phi_i \in S_h, \quad (6)$$

to obtain a finite element scheme for the tfDE model (4): Find u_h^n such that the following equation holds for any $\chi_h \in S_h$

$$\begin{aligned} (\delta_t u_h^n, \chi_h) + q(t_n) (\delta_t^{1-\alpha(t_n)} u_h^n, \chi_h) + (\mathbf{K}(\cdot) \nabla u_h^n, \nabla \chi_h) \\ = - \frac{q(t_n) t_n^{\alpha(t_n)-1}}{\Gamma(\alpha(t_n))} (u_0, \chi_h) + (f(\cdot, t_n), \chi_h). \end{aligned} \quad (7)$$

Let $L_2(\Omega)$ be the space of the Lebesgue square integrable functions in Ω and $H^m(\Omega)$ be the Sobolev space of functions with derivatives of order up to m in $L_2(\Omega)$. Let

$$\check{H}^m(\Omega) := \left\{ v \in H^m(\Omega) : \nabla \cdot (\mathbf{K}(\mathbf{x}) \nabla)^j v(\mathbf{x}) = 0, \mathbf{x} \in \partial\Omega, \quad \forall j < m/2, j \in \mathbb{N} \right\}.$$

For a Hilbert space $\mathcal{X}$, we introduce the Sobolev spaces involving time [1,35]

$$\begin{aligned} H^m(0, T; \mathcal{X}) \\ := \left\{ f(\mathbf{x}, t) : \frac{\partial^l f(\cdot, t)}{\partial t^l} \in \mathcal{X}, t \in (0, T), \left\| \frac{\partial^l f(\cdot, t)}{\partial t^l} \right\|_{\mathcal{X}} \in L_2(0, T), 0 \leq l \leq m \right\} \end{aligned}$$

with the norm being defined by

$$\|f\|_{H^m(0, T; \mathcal{X})} := \left(\sum_{l=0}^m \int_0^T \left\| \frac{\partial^l f(\cdot, t)}{\partial t^l} \right\|_{\mathcal{X}}^2 dt \right)^{1/2}.$$

The following stability and error estimates for finite element scheme (7) was proved in [43].

Theorem 2.1. Suppose that $\alpha, q \in C^1[0, T]$ with $0 < \alpha_m \leq \alpha(t) \leq 1$ for $t \in [0, T]$, $k_{i,j} \in C^1(\overline{\Omega})$ for $i, j = 1, \dots, d$, $u_0 \in \check{H}^4$ and $f \in H^1(0, T; \check{H}^2) \cap H^2(0, T; L_2)$. If $\alpha(0) = 1$, $\alpha'(0) = 0$ and $\lim_{t \rightarrow 0^+} \alpha'(t) \ln t$ exists, an optimal order error estimate holds

$$\begin{aligned} \|u_h - u\| &:= \max_{1 \leq n \leq N} \|u_h^n - u(\cdot, t_n)\|_{L_2(\Omega)} \\ &\leq Q (\|u_0\|_{\check{H}^4} + \|f\|_{H^1(0, T; \check{H}^2)} + \|f\|_{H^2(0, T; L_2)}) (N^{-1} + h^2). \end{aligned}$$

If $\alpha(0) < 1$, a suboptimal order error estimate holds

$$\|u_h - u\| \leq Q (\|u_0\|_{\check{H}^4} + \|f\|_{H^1(0, T; \check{H}^2)} + \|f\|_{H^2(0, T; L_2)}) (N^{-\alpha(0)} + h^2).$$

In both cases, the stability estimate holds

$$\|u_h\| \leq Q (\|u_0\|_{\check{H}^4} + \|f\|_{H^1(0, T; \check{H}^2)} + \|f\|_{H^2(0, T; L_2)}).$$

Here $Q = Q(T, \alpha_m, \|\alpha\|_{C^1[0, T]}, \|k\|_{C^1[0, T]})$.

3. An approximation to the all-at-once system

Like in the constant-order tFDE case, the numerical scheme (7) has a computational complexity of order $O(MN^2)$ as it requires the evaluation of the numerical solutions at all the previous time steps. Due to the impact of the variable order $\alpha(t)$, the scheme loses the Toeplitz-like structure. Hence, the fast methods developed for constant-order tFDEs do not apply.

3.1. Matrix form of the all-at-once system

To develop a fast solution method, we follow the previous practice to begin with the exploration of the matrix structure. In this section we prove the following theorem

Theorem 3.1. *The finite element scheme (7) can be expressed in the following space-time coupled all-at-once matrix form*

$$\mathbf{A}\mathbf{u} = \mathbf{f} \quad (8)$$

where $\mathbf{u}, \mathbf{f} \in \mathbb{R}^{NM}$ are block vectors of the form

$$\begin{aligned} \mathbf{u} &:= (\mathbf{u}^1, \mathbf{u}^2, \dots, \mathbf{u}^N)^T, \quad \mathbf{u}^n := (u_1^n, u_2^n, \dots, u_M^n)^T \\ \mathbf{f} &:= (\mathbf{f}^1, \mathbf{f}^2, \dots, \mathbf{f}^N)^T, \quad \mathbf{f}^n := (f_1^n, f_2^n, \dots, f_M^n)^T, \quad 1 \leq n \leq N. \end{aligned} \quad (9)$$

The stiffness matrix $\mathbf{A}$ can be decomposed in the tensor product form

$$\mathbf{A} = \mathbf{B}_N \otimes \mathbf{M}_0 + \tau \mathbf{I}_N \otimes \mathbf{M}_1. \quad (10)$$

Here $\mathbf{I}_N \in \mathbb{R}^{N \times N}$ is the identity matrix, $\mathbf{B}_N$ is a lower triangular matrix generated by the time dependent derivatives, $\mathbf{M}_0, \mathbf{M}_1 \in \mathbb{R}^{M \times M}$ are the sparse mass and stiffness matrices with $(\mathbf{M}_0)_{i,j} = (\phi_i, \phi_j)$ and $(\mathbf{M}_1)_{i,j} = (\mathbf{K}(\cdot) \nabla \phi_i, \nabla \phi_j)$, respectively.

Proof 3.1. We incorporate (6) into the finite element scheme (7) and use (5) to rewrite (7) as follows

$$\begin{aligned} & \sum_{k=1}^n b_{n,k} (u_h^k, \chi_h) + \tau (\mathbf{K}(\cdot) \nabla u_h^n, \nabla \chi_h) \\ &= \tau^{\alpha(t_n)} q(t_n) \left(\frac{n^{\alpha(t_n)} - (n-1)^{\alpha(t_n)}}{\Gamma(1 + \alpha(t_n))} - \frac{n^{\alpha(t_n)-1}}{\Gamma(\alpha(t_n))} \right) (u^0, \chi_h) \\ & \quad + \tau (f(\cdot, t_n), \chi_h), \end{aligned} \quad (11)$$

where the coefficients $b_{n,k}$ for $1 \leq k \leq n \leq N$ are given as follows

$$b_{n,k} = \begin{cases} 1 + \frac{\tau^{\alpha(t_n)} q(t_n)}{\Gamma(1 + \alpha(t_n))}, & k = n; \\ \frac{\tau^{\alpha(t_n)} q(t_n)}{\Gamma(1 + \alpha(t_n))} [2^{\alpha(t_n)} - 2] - 1, & k = n - 1; \\ \frac{\tau^{\alpha(t_n)} q(t_n)}{\Gamma(1 + \alpha(t_n))} [(n - k + 1)^{\alpha(t_n)} - 2(n - k)^{\alpha(t_n)} + (n - k - 1)^{\alpha(t_n)}], & 1 \leq k \leq n - 2. \end{cases} \quad (12)$$

The f_i^n in (9) is given by

$$\begin{aligned} f_i^n &:= \tau (f(\cdot, t_n), \phi_i) \\ & \quad + \tau^{\alpha(t_n)} q(t_n) \left(\frac{n^{\alpha(t_n)} - (n-1)^{\alpha(t_n)}}{\Gamma(1 + \alpha(t_n))} - \frac{n^{\alpha(t_n)-1}}{\Gamma(\alpha(t_n))} \right) (u^0, \phi_i). \end{aligned} \quad (13)$$

A careful examination of (11) and (12) reveals that the following decomposition holds.

$$\mathbf{A} = \mathbf{B}_N \otimes \mathbf{M}_0 + \tau \mathbf{I}_N \otimes \mathbf{M}_1. \quad (14)$$

Here $\mathbf{B}_N \in \mathbb{R}^{N \times N}$ is a lower triangular matrix defined by $(\mathbf{B}_N)_{n,k} := b_{n,k}$ for $1 \leq k \leq n \leq N$. We observe from (12) that the matrix $\mathbf{B}_N$ can be decomposed as

$$\mathbf{B}_N = \mathbf{C}_N + \text{diag}(\mathbf{\Lambda})\mathbf{T}_N, \quad (15)$$

where $\mathbf{C}_N$ is a bi-diagonal matrix, in which the all the entries on the main diagonal is 1 and all the entries on the subdigonal is -1 , $\mathbf{\Lambda} = (\Lambda_n)_{n=1}^N$ is given by $\Lambda_n := \tau^{\alpha(t_n)} q(t_n) / \Gamma(1 + \alpha(t_n))$, and the entries $(\mathbf{T}_N)_{n,k}$ of the lower triangular matrix $\mathbf{T}_N$ are given by

$$(\mathbf{T}_N)_{n,k} := \begin{cases} 1, & k = n, \\ 2^{\alpha(t_n)} - 2, & k = n - 1, \\ (n - k + 1)^{\alpha(t_n)} - 2(n - k)^{\alpha(t_n)} \\ \quad + (n - k - 1)^{\alpha(t_n)}, & 1 \leq k \leq n - 2. \end{cases} \quad (16)$$

3.2. Approximations of $\mathbf{B}_N$

Theorem 3.2. Let $\bar{\alpha} = \frac{1}{2}(\alpha_{\min} + \alpha_{\max})$. Then for $1 \leq k \leq n - 2$, $(\mathbf{T}_N)_{n,k}$ can be approximated by

$$(\mathbf{T}_N)_{n,k} \approx (\tilde{\mathbf{T}}_N)_{n,k} = \sum_{j=0}^s \frac{(\alpha(t_n) - \bar{\alpha})^j}{j!} \left[(n - k + 1)^{\bar{\alpha}} \ln^j(n - k + 1) \right. \\ \left. - 2(n - k)^{\bar{\alpha}} \ln^j(n - k) + (n - k - 1)^{\bar{\alpha}} \ln^j(n - k - 1) \right], \quad (17)$$

for some $s \in \mathbb{N}^+$ with the residue $R_{s+1}^{n,k}$ given by

$$R_{s+1}^{n,k} = \frac{(\alpha - \bar{\alpha})^{s+1}}{2(s+1)!} \left\{ \eta(\eta - 1) \left[\frac{\ln^{s+1}(n - k + \xi_1)}{(n - k + \xi_1)^{2-\eta}} + \frac{\ln^{s+1}(n - k - \xi_2)}{(n - k - \xi_2)^{2-\eta}} \right] \right. \\ \left. + (s+1)(2\eta - 1) \left[\frac{\ln^s(n - k + \xi_1)}{(n - k + \xi_1)^{2-\eta}} + \frac{\ln^s(n - k - \xi_2)}{(n - k - \xi_2)^{2-\eta}} \right] \right. \\ \left. + s(s+1) \left[\frac{\ln^{s-1}(n - k + \xi_1)}{(n - k + \xi_1)^{2-\eta}} + \frac{\ln^{s-1}(n - k - \xi_2)}{(n - k - \xi_2)^{2-\eta}} \right] \right\}, \quad (18)$$

here $\xi_1, \xi_2 \in (0, 1)$, $\eta = \bar{\alpha} + \theta(\alpha(t_n) - \bar{\alpha})$, $\theta \in (0, 1)$.

Proof 3.2. We decouple the nonlinear dependence of n and k in $(\mathbf{T}_N)_{n,k}$ for $1 \leq k \leq n - 3$ by applying the Taylor's expansion of power functions. Define $\psi(x) := (n - k + 1)^x - 2(n - k)^x + (n - k - 1)^x$. Then

$$\psi(\alpha) = \sum_{j=0}^s \frac{(\alpha - \bar{\alpha})^j}{j!} \psi^{(j)}(\bar{\alpha}) + \frac{(\alpha - \bar{\alpha})^{s+1}}{(s+1)!} \psi^{(s+1)}(\bar{\alpha} + \theta(\alpha - \bar{\alpha})), \quad \theta \in (0, 1),$$

here $\psi^{(j)}(\alpha)$ represent the j -th derivative. Replacing the $\psi(\alpha)$ part in the definition (16) of $(\mathbf{T}_N)_{n,k}$ yields

$$(\mathbf{T}_N)_{n,k} = \sum_{j=0}^s \frac{(\alpha(t_n) - \bar{\alpha})^j}{j!} \left[(n - k + 1)^{\bar{\alpha}} \ln^j(n - k + 1) \right. \\ \left. - 2(n - k)^{\bar{\alpha}} \ln^j(n - k) + (n - k - 1)^{\bar{\alpha}} \ln^j(n - k - 1) \right] + R_{s+1}^{n,k}, \quad (19)$$

with the local truncation error $R_{s+1}^{n,k}$ given by

$$R_{s+1}^{n,k} = \frac{(\alpha - \bar{\alpha})^{s+1}}{(s+1)!} \left[(n - k + 1)^{\bar{\alpha} + \theta(\alpha - \bar{\alpha})} \ln^{s+1}(n - k + 1) \right. \\ \left. - 2(n - k)^{\bar{\alpha} + \theta(\alpha - \bar{\alpha})} \ln^{s+1}(n - k) \right. \\ \left. + (n - k - 1)^{\bar{\alpha} + \theta(\alpha - \bar{\alpha})} \ln^{s+1}(n - k - 1) \right], \quad \theta \in (0, 1).$$

Dropping $R_{s+1}^{n,k}$ in (19) yields the approximation (17) of $(\mathbf{T}_N)_{n,k}$ for $1 \leq k \leq n - 2$.

To seek a more compact form of $R_{s+1}^{n,k}$, we define $g(x) = x^\eta \ln^{s+1} x$ with $\eta = \bar{\alpha} + \theta(\alpha - \bar{\alpha})$. Direct calculation yields

$$\begin{aligned} g'(x) &= \eta x^{\eta-1} \ln^{s+1} x + (s+1)x^{\eta-1} \ln^s x, \\ g''(x) &= \eta(\eta-1)x^{\eta-2} \ln^{s+1} x + \eta(s+1)x^{\eta-2} \ln^s x \\ &\quad + (s+1)(\eta-1)x^{\eta-2} \ln^s x + s(s+1)x^{\eta-2} \ln^{s-1} x \\ &= x^{\eta-2} \left[\eta(\eta-1) \ln^{s+1} x + (s+1)(2\eta-1) \ln^s x + s(s+1) \ln^{s-1} x \right]. \end{aligned}$$

We incorporate these with the following relation

$$g(x+1) - 2g(x) + g(x-1) = \frac{1}{2} \left[g''(x + \xi_1) + g''(x - \xi_2) \right], \quad \xi_1, \xi_2 \in (0, 1)$$

to simplify the local truncation error $R_{s+1}^{n,k}$ to (18), which finishes the proof.

According to Theorem 3.2, we obtain an approximation $\tilde{\mathbf{T}}_N$ of $\mathbf{T}_N$ by replacing $(\mathbf{T}_N)_{n,k}$ by $(\tilde{\mathbf{T}}_N)_{n,k}$ for $1 \leq k \leq n-3$. Thus the linear system (8) can be approximated by

$$\tilde{\mathbf{A}}\tilde{\mathbf{u}} = \mathbf{f}, \quad \tilde{\mathbf{A}} = \tilde{\mathbf{B}}_N \otimes \mathbf{M}_0 + \tau \mathbf{I}_N \otimes \mathbf{M}_1, \quad (20)$$

where $\tilde{\mathbf{B}}_N = \mathbf{C}_N + \text{diag}(\Lambda)\tilde{\mathbf{T}}_N$. To prove the asymptotically consistency of the approximated system to the original problem, we bound the local truncation error $R_{s+1}^{n,k}$ in the following theorem.

Theorem 3.3. For $1 \leq k \leq n-3$, the $R_{s+1}^{n,k}$ can be bounded by

$$|R_{s+1}^{n,k}| \leq \frac{Q}{2^{s+1}\sqrt{s}}.$$

Furthermore, for any $\epsilon > 0$ it holds for $s \geq \frac{1 - \alpha_{\min}}{\ln 2} \ln N - 1$

$$\max_{1 \leq k \leq n \leq N} |(\tilde{\mathbf{B}}_N)_{n,k} - (\mathbf{B}_N)_{n,k}| \leq \frac{Q}{N\sqrt{s}}.$$

Here Q is a constant independent of N .

Proof 3.3. We incorporate (18) with $|\bar{\alpha} - \alpha(t_n)| \leq 1/2$ and $(n-k-1) \geq (n-k+1)/2$ for $n-k \geq 3$ to obtain

$$\begin{aligned} |R_{s+1}^{n,k}| &\leq \frac{Q}{2^{s+1}} \left(\frac{1}{(s+1)!} \frac{\ln^{s+1}(n-k+1)}{(n-k+1)^{2-\eta}} + \frac{1}{s!} \frac{\ln^s(n-k+1)}{(n-k+1)^{2-\eta}} \right. \\ &\quad \left. + \frac{1}{(s-1)!} \frac{\ln^{s-1}(n-k+1)}{(n-k+1)^{2-\eta}} \right). \end{aligned} \quad (21)$$

For $\hat{s} \geq 1$, direct calculations show that the function $g_1(x) := \ln^{\hat{s}} x / x^{2-\eta}$ attains its maximum at $x = e^{\hat{s}/(2-\eta)}$ on $1 \leq x \leq N$. Combining this with (21) and the Stirling's formula $s! \approx \sqrt{2\pi s} (s/e)^s$ for $s \gg 1$ we get

$$\begin{aligned} |R_{s+1}^{n,k}| &\leq \frac{Q}{2^{s+1}} \left(\frac{1}{(s+1)!} \frac{\left(\frac{s+1}{2-\eta}\right)^{s+1}}{\left(e^{\frac{s+1}{2-\eta}}\right)^{2-\eta}} + \frac{1}{s!} \frac{\left(\frac{s}{2-\eta}\right)^s}{\left(e^{\frac{s}{2-\eta}}\right)^{2-\eta}} + \frac{1}{(s-1)!} \frac{\left(\frac{s-1}{2-\eta}\right)^{s-1}}{\left(e^{\frac{s-1}{2-\eta}}\right)^{2-\eta}} \right) \\ &\leq \frac{Q}{2^{s+1}} \left(\frac{e^{s+1}}{\sqrt{2\pi(s+1)}(s+1)^{s+1}} \frac{(s+1)^{s+1}}{(2-\eta)^{s+1}e^{s+1}} \right. \\ &\quad \left. + \frac{e^s}{\sqrt{2\pi s s^s} (2-\eta)^s e^s} + \frac{e^{s-1}}{\sqrt{2\pi(s-1)}(s-1)^{s-1}} \frac{(s-1)^{s-1}}{(2-\eta)^{s-1}e^{s-1}} \right) \\ &= \frac{Q}{2^{s+1}} \left(\frac{1}{\sqrt{2\pi(s+1)}(2-\eta)^{s+1}} + \frac{1}{\sqrt{2\pi s}(2-\eta)^s} \right. \\ &\quad \left. + \frac{1}{\sqrt{2\pi(s-1)}(2-\eta)^{s-1}} \right) \leq \frac{Q}{2^{s+1}\sqrt{s}}, \quad s \geq 2. \end{aligned} \quad (22)$$

Plugging (22) into (15) yields

$$\max_{1 \leq k \leq n \leq N} |(\tilde{\mathbf{B}}_N)_{n,k} - (\mathbf{B}_N)_{n,k}| \leq \frac{Q}{N^{\alpha_{\min}} \sqrt{s} 2^{s+1}}.$$

By setting $s + 1 \geq \frac{1 - \alpha_{\min}}{\ln 2} \ln N$ for any $\epsilon > 0$, i.e., $2^{s+1} \geq N^{1-\alpha_{\min}}$, we immediately obtain

$$\max_{1 \leq k \leq n \leq N} |(\tilde{\mathbf{B}}_N)_{n,k} - (\mathbf{B}_N)_{n,k}| \leq \frac{Q}{N\sqrt{s}},$$

which finishes the proof.

Remark 3.1. Theorem 3.3 implies that

$$\max_{1 \leq i, j \leq M, N} |(\mathbf{A})_{i,j} - (\tilde{\mathbf{A}})_{i,j}| \rightarrow 0, \quad s \rightarrow \infty,$$

which proves the asymptotically consistency of the approximated system to the original problem.

4. A preconditioned fast DAC solver

By the block lower-triangular structure of $\mathbf{A}$ and the sparsity of the mass and the stiffness matrices, the all-at-once system $\mathbf{A}\mathbf{u} = \mathbf{f}$ can be solved based on the Kronecker product, which requires $O(MN^2)$ storage and $O(MN^2)$ operations. To reduce the memory requirement and the computational complexity, we develop a preconditioned fast DAC method for the approximated system $\tilde{\mathbf{A}}\tilde{\mathbf{u}} = \mathbf{f}$, in which the Toeplitz-like structure of $\tilde{\mathbf{B}}$ is exploited for the efficient matrix-vector multiplications. For convenience, we slightly abuse the notations to replace $\tilde{\mathbf{u}}$ by $\mathbf{u}$ in the following statement.

4.1. A fast DAC method

It is clear that the matrix-vector multiplication $(\mathbf{I}_N \otimes \mathbf{M}_1)\mathbf{v}$ for any $\mathbf{v} \in \mathbb{R}^{NM}$ requires $O(NM)$ memories and $O(NM)$ operations. The remaining task is to efficiently implement the $(\tilde{\mathbf{B}}_N \otimes \mathbf{M}_0)\mathbf{v}$. Define the operator $\mathcal{P}$ by $\mathcal{P}\mathbf{v} := (\mathbf{v}^1, \dots, \mathbf{v}^N) \in \mathbb{R}^{M \times N}$ for any $\mathbf{v} = (\mathbf{v}^1, \dots, \mathbf{v}^N)^T \in \mathbb{R}^{NM}$ with $\mathbf{v}^1, \dots, \mathbf{v}^N \in \mathbb{R}^M$. By the relation

$$(\tilde{\mathbf{B}}_N \otimes \mathbf{M}_0)\mathbf{u} = \mathcal{P}^{-1}(\mathbf{M}_0(\tilde{\mathbf{B}}_N(\mathcal{P}\mathbf{u})^T)^T), \quad (23)$$

it suffices us to investigate the implementation of $\tilde{\mathbf{B}}_N\mathbf{v}_N$ for some $\mathbf{v}_N \in \mathbb{R}^N$.

Let $\tilde{\mathbf{B}}_N\mathbf{v}_N = \mathbf{p}_N$ for $N = 2^J$ with $J \in \mathbb{N}^+$ and $N' = N/2$. Then $\tilde{\mathbf{B}}_N$, $\mathbf{v}_N$ and $\mathbf{p}_N$ can be divided as

$$\tilde{\mathbf{B}}_N = \begin{bmatrix} \tilde{\mathbf{B}}_{N'}^* & \mathbf{0} \\ \tilde{\mathbf{F}}_{N'} & \tilde{\mathbf{B}}_{N'}^{**} \end{bmatrix}, \quad \mathbf{v}_N = \begin{bmatrix} \mathbf{v}_{N'}^* \\ \mathbf{v}_{N'}^{**} \end{bmatrix}, \quad \mathbf{p}_N = \begin{bmatrix} \mathbf{p}_{N'}^* \\ \mathbf{p}_{N'}^{**} \end{bmatrix}, \quad (24)$$

where, by the approximation (17), $\tilde{\mathbf{F}}_{N'}$ can be expressed as

$$\tilde{\mathbf{F}}_{N'} = \text{diag}(\mathbf{D}^0)\mathbf{T}^0 + \text{diag}(\mathbf{D}^1)\mathbf{T}^1 + \dots + \text{diag}(\mathbf{D}^s)\mathbf{T}^s$$

in which $\mathbf{D}^j := (D_i^j)_{i=1}^{N'}$ with

$$D_i^j = \frac{\tau^{\alpha(t_{i+N'})} q(t_{i+N'}) (\alpha(t_{i+N'}) - \bar{\alpha})^j}{j! \Gamma(1 + \alpha(t_{i+N'}))}, \quad 1 \leq i \leq N'$$

and $\mathbf{T}^j := \text{Toeplitz}(\mathbf{t}_c^j, \mathbf{t}_r^j)$ for $0 \leq j \leq s$ with $\mathbf{t}_c^j = (t_{ci}^j)_{i=1}^{N'}$ and $\mathbf{t}_r^j = (t_{ri}^j)_{i=1}^{N'}$ referring to the first column and the first row of $\mathbf{T}^j$, respectively

$$\begin{aligned} t_{ci}^j &= (N' + i)^{\bar{\alpha}} \ln^j(N' + i) - 2(N' + i - 1)^{\bar{\alpha}} \ln^j(N' + i - 1) \\ &\quad + (N' + i - 2)^{\bar{\alpha}} \ln^j(N' + i - 2), \quad 1 \leq i \leq N', \\ t_{ri}^j &= (N' - i + 2)^{\bar{\alpha}} \ln^j(N' - i + 2) - 2(N' - i + 1)^{\bar{\alpha}} \ln^j(N' - i + 1) \\ &\quad + (N' - i)^{\bar{\alpha}} \ln^j(N' - i), \quad 1 \leq i \leq N'. \end{aligned}$$

Then $\tilde{\mathbf{B}}_N\mathbf{v}_N = \mathbf{p}_N$ can be divided into the following two sub-systems

$$\mathbf{p}_{N'}^* = \tilde{\mathbf{B}}_{N'}^* \mathbf{v}_{N'}^*, \quad \mathbf{p}_{N'}^{**} = \tilde{\Gamma}_{N'} \mathbf{v}_{N'}^* + \tilde{\mathbf{B}}_{N'}^{**} \mathbf{v}_{N'}^{**}. \quad (25)$$

Indeed, the matrix-vector multiplication $\mathbf{T}^j \mathbf{v}_{N'}$ can be carried out efficiently via the discrete fast Fourier transform. However, as proved in Theorem 3.3, the local truncation error $R_{s+1}^{n,k}$ may not be sufficiently small for k near n , which in turn affects the accuracy of the approximations. For this reason we keep the true values in the original matrix $\mathbf{B}_{N'}$ of the entries on the right-up bands of $\Gamma_{N'}$ with width $c = O(\lfloor \log n \rfloor)$ as follows

$$\tilde{\Gamma}_{N'} = \hat{\Gamma}_{N'} + \Psi_{N'} \quad (26)$$

where

$$\hat{\Gamma}_{N'} = \begin{bmatrix} \tilde{\Gamma}_{1,1} & \cdots & \tilde{\Gamma}_{1,N'-c} & 0 & 0 & \cdots & 0 \\ \tilde{\Gamma}_{2,1} & \cdots & \tilde{\Gamma}_{2,N'-c} & \tilde{\Gamma}_{2,N'-c+1} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \tilde{\Gamma}_{c,1} & \cdots & \tilde{\Gamma}_{c,N'-c} & \tilde{\Gamma}_{c,N'-c+1} & \tilde{\Gamma}_{c,N'-c+2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \tilde{\Gamma}_{N',1} & \cdots & \tilde{\Gamma}_{N',N'-c} & \tilde{\Gamma}_{N',N'-c+1} & \tilde{\Gamma}_{N',N'-c+2} & \cdots & \tilde{\Gamma}_{N',N'} \end{bmatrix}$$

$$\Psi_{N'} = \begin{bmatrix} 0 & \cdots & 0 & (\mathbf{B}_N)_{1,N'-c+1} & (\mathbf{B}_N)_{1,N'-c+2} & \cdots & (\mathbf{B}_N)_{1,N'} \\ 0 & \cdots & 0 & 0 & (\mathbf{B}_N)_{2,N'-c+2} & \cdots & (\mathbf{B}_N)_{2,N'} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & \cdots & (\mathbf{B}_N)_{c,N'} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Therefore, the matrix $\hat{\Gamma}_{N'}$ reserves the Toeplitz-like structure like $\tilde{\Gamma}_{N'}$

$$\hat{\Gamma}_{N'} = \text{diag}(\mathbf{D}^0) \hat{\mathbf{T}}^0 + \text{diag}(\mathbf{D}^1) \hat{\mathbf{T}}^1 + \cdots + \text{diag}(\mathbf{D}^s) \hat{\mathbf{T}}^s$$

where $\hat{\mathbf{T}}^j = \text{Toeplitz}(\hat{\mathbf{t}}_c^j, \hat{\mathbf{t}}_r^j)$ ($0 \leq j \leq s$) with $\hat{\mathbf{t}}_c^j = \mathbf{t}_c^j$ and

$$\hat{\mathbf{t}}_r^j = [t_{r1}^j, \dots, t_{rN'-c}^j, 0, \dots, 0]^T \quad (27)$$

Repeating the subdivision procedure like (24)–(27) generates the following fast DAC algorithm for the matrix-vector multiplication $\tilde{\mathbf{B}}_N \mathbf{v}_N$.

Theorem 4.1. *The matrix-vector multiplication $\tilde{\mathbf{B}}_N \mathbf{v}_N$ requires $O(N \log^2 N)$ memory and $O(N \log^3 N)$ operations.*

Proof 4.1. It is well known that the matrix-vector multiplication $\mathbf{T}^j \mathbf{v}_{N'}$ can be performed in $O(N' \log N')$ operations via the discrete fast Fourier transform. Thus the implementation of $\hat{\Gamma}_{N'} \mathbf{v}_{N'}$ only requires $O(sN' \log N')$ operations and $O(sN')$ memory. As s takes the value of $O(\log N)$ in Theorem 3.3, the aforementioned computational cost and storage become $O(N' \log N' \log N)$ and $O(N' \log N)$, respectively. Let $\tilde{\Theta}_N$ and $\tilde{\Phi}_N$ be the total computational cost and the storage of the implementations of $\hat{\Gamma}_{N'} \mathbf{v}_{N'}$ in the fast DAC method. Then

Algorithm 1 Fast DAC algorithm for $\tilde{\mathbf{B}}_N \mathbf{v}_N$.

```

function  $\mathbf{p}_N = \text{fastDAC}(\tilde{\mathbf{B}}_N, \mathbf{v}_N)$ 
  if  $N \leq 2$ 
     $\mathbf{p}_N = \tilde{\mathbf{B}}_N \mathbf{v}_N$ 
  else
     $\mathbf{p}^* = \text{fastDAC}(\tilde{\mathbf{B}}^*, \mathbf{v}^*)$ 
     $\mathbf{p}^{**} = \text{fastDAC}(\tilde{\mathbf{B}}^{**}, \mathbf{v}^{**}) + \hat{\mathbf{I}}_N \mathbf{v}^*$ 
  end if
end function

```

$$\begin{aligned}
\tilde{\Theta}_N &= O(N' \log N' \log N) + 2\tilde{\Theta}_{N'} \\
&= O(N' \log N' \log N) + 2O\left(\frac{N'}{2} \log \frac{N'}{2} \log N\right) + 2^2 \tilde{\Theta}_{\frac{N'}{2}} \\
&= \dots = O\left(\log^2 N(N' + 2\frac{N'}{2} + 2^2 \frac{N'}{2^2} + \dots + 2^J \frac{N'}{2^J})\right) = O(N \log^3 N) \\
\tilde{\Phi}_N &= O(N' \log N) + 2\tilde{\Phi}_{N'} \\
&= O(N' \log N) + 2O\left(\frac{N'}{2} \log N\right) + 2^2 \tilde{\Phi}_{\frac{N'}{2}} \\
&= \dots = O\left(\log N(N' + 2\frac{N'}{2} + 2^2 \frac{N'}{2^2} + \dots + 2^J \frac{N'}{2^J})\right) = O(N \log^2 N).
\end{aligned} \tag{28}$$

The matrix $\Psi_{N'}$ has $\frac{c(c+1)}{2}$ nonzero entries totally, which implies that the storage and the computational work of the matrix-vector multiplication $\Psi_{N'} \mathbf{v}_{N'}$ are both $O(\log^2 N)$. Let ϵ_θ and ϵ_ϕ be the computational cost and the storage of $\Psi_n \mathbf{v}_n$ for $n = N/2, \dots, N/2^J$. Similar calculations like (28) yield

$$\begin{aligned}
\epsilon_\theta &= \frac{c(c+1)}{2} + 2\frac{c(c+1)}{2} + \dots + 2^J \frac{c(c+1)}{2} = \frac{2^J - 1}{2} \frac{c(c+1)}{2} = O(Nc^2), \\
\epsilon_\phi &= \frac{c(c+1)}{2} + 2\frac{c(c+1)}{2} + \dots + 2^J \frac{c(c+1)}{2} = \frac{2^J - 1}{2} \frac{c(c+1)}{2} = O(Nc^2).
\end{aligned}$$

In particular, setting $c = \log N$ yields $\epsilon_\theta = O(N \log^2 N)$ and $\epsilon_\phi = O(N \log^2 N)$. Consequently the total flops Θ_N and the total storage Φ_N of the fast DAC algorithm for $\tilde{\mathbf{B}}_N \mathbf{v}_N$ are approximately

$$\Theta_N = \tilde{\Theta}_N + \epsilon_\theta = O(N \log^3 N), \quad \Phi_N = \tilde{\Phi}_N + \epsilon_\phi = O(N \log^2 N),$$

which finishes the proof.

Remark 4.1. It is worth mentioning that it takes $O(N^2)$ operations to generate the entries of $\mathbf{B}_N$ while only $O(N \log N)$ operations are needed to generate the entries of $\tilde{\mathbf{B}}_N$ in the fast DAC method.

Remark 4.2. According to the discussions on this section, the memory requirement and the computational cost of carrying out the matrix-vector multiplication $\tilde{\mathbf{A}} \mathbf{v}$ are $O(MN \log^2 N)$ and $O(MN \log^3 N)$, respectively. Consequently the all-at-once system (20) can be solved by Krylov subspace methods with $O(MN \log^2 N)$ storage and $O(MN \log^3 N)$ operations per iteration.

4.2. A preconditioned fast DAC method

To reduce the number of iterations, we follow the criteria suggested in [21,22] to select the preconditioner $\mathbf{M}_p = \tilde{\mathbf{B}}_N \otimes \mathbf{M}_0$ for the approximated system, that is, we solve the following linear system instead of (20)

$$\mathbf{M}_p^{-1} \tilde{\mathbf{A}} \mathbf{u} = \mathbf{M}_p^{-1} \mathbf{f}. \tag{29}$$

As we have discussed the implementation of $\tilde{\mathbf{B}}_N \mathbf{v}_N$ (and thus $\tilde{\mathbf{A}} \mathbf{v}$) in details in Section 4.1, we remain to consider the fast solver of the system $\mathbf{M}_p \mathbf{v} = \mathbf{y}$ for any $\mathbf{y} \in \mathbb{R}^{NM}$.

By the definition of $\mathbf{M}_p$, we express $\mathbf{v} = \mathbf{M}_p^{-1} \mathbf{y}$ by

$$\mathbf{v} = (\tilde{\mathbf{B}}_N \otimes \mathbf{M}_0)^{-1} \mathbf{y} = \mathcal{P}^{-1} (\mathbf{M}_0^{-1} (\tilde{\mathbf{B}}^{-1} (\mathcal{P} \mathbf{y})^\top)^\top),$$

that is, it suffices to consider

$$\tilde{\mathbf{B}}_N \mathbf{v}_N = \mathbf{z}_N, \text{ for } \mathbf{z}_N \in \mathbb{R}^N. \quad (30)$$

Let $\mathbf{v}_N = [\mathbf{v}_{N'}, \mathbf{v}_{N'}^{**}]^\top$ and $\mathbf{z}_N = [\mathbf{z}_{N'}, \mathbf{z}_{N'}^{**}]^\top$. Then, similar to (25), $\mathbf{v}_N = \tilde{\mathbf{B}}^{-1} \mathbf{z}_N$ can be divided into the following two sub-systems

$$\mathbf{v}_{N'}^* = \tilde{\mathbf{B}}_{N'}^{*-1} \mathbf{z}_{N'}^*, \quad \mathbf{v}_{N'}^{**} = \tilde{\mathbf{B}}_{N'}^{**,-1} (\mathbf{z}_{N'}^{**} - \tilde{\Gamma}_{N'} \mathbf{v}_{N'}^*), \quad (31)$$

in which the matrix-vector multiplication $\tilde{\Gamma}_{N'} \mathbf{v}_{N'}^*$ can be efficiently performed as discussed in Section 4.1. Repeating the subdivision procedure like (30)–(31) generates the following fast DAC algorithm (named by InvDAC algorithm) for solving the system (30).

Algorithm 2 The InvDAC algorithm.

```

function  $\mathbf{v}_N = \text{InvDAC}(\tilde{\mathbf{B}}_N, \mathbf{z}_N)$ 
    if  $N \leq 2$ 
         $\mathbf{v}_N = \tilde{\mathbf{B}}_N^{-1} \mathbf{z}_N$ 
    else
         $\mathbf{v}^* = \text{InvDAC}(\tilde{\mathbf{B}}^*, \mathbf{z}^*)$ 
         $\mathbf{z}^{**} = \mathbf{z}^{**} - \tilde{\Gamma}_{N'} \mathbf{v}^*$ 
         $\mathbf{v}^{**} = \text{InvDAC}(\tilde{\mathbf{B}}^{**}, \mathbf{z}^{**})$ 
    end if
end function

```

Combining the Algorithm 2 with Theorem 4.1 yields the following theorem.

Theorem 4.2. *The computational cost and storage of solving (30) by Algorithm 2 are $O(N \log^3 N)$ and $O(N \log^2 N)$, respectively.*

5. Numerical experiments

5.1. Efficiency of the fast method

We carry out numerical experiments to investigate the performance of the fast DAC method (FDAC) and the preconditioned FDAC (PFDAC) by comparing them with the forward substitution method (FS). The generalized minimal residual (GMRES) iteration method is applied to solve the linear systems. The convergence rates, the numbers $Iter$, $Iter_f$ and $Iter_{pre}$ of iterations in FS, FDAC and PFDAC, respectively, the CPU times CPU_M and CPU_{Mf} of generating $\mathbf{B}_N$ and $\tilde{\mathbf{B}}_N$, respectively, and the CPU times CPU , CPU_f and CPU_{pre} of solving the linear systems (8) and (20), respectively are recorded. According to Theorem 3.3, we set $c = s = 2 \lfloor \ln N / \ln 2 \rfloor$ in our numerical experiments.

Let $T = 1$, $\Omega = [0, 1]^d$ for $d = 1, 2, 3$ and $\alpha(t) = \alpha_0 + (\alpha_1 - \alpha_0)t$ with $\alpha_0 = 1$ and $\alpha_1 = 0.8$. The exact solution and the corresponding right-hand side term are given by

$$u(\mathbf{x}, t) = t^2 \sin(2\pi x_1) \times \cdots \times \sin(2\pi x_d),$$

$$f(\mathbf{x}, t) = \left(2t + \frac{2q(t)t^{1+\alpha(t)}}{\Gamma(2+\alpha(t))} + 4d\pi^2 t^2 \right) \sin(2\pi x_1) \times \cdots \times \sin(2\pi x_d).$$

Experiment 1. Let $q(t) = 1$ and $\mathbf{K} = \text{diag}(0.01, \dots, 0.01) \in \mathbb{R}^{d \times d}$ with $d = 1, 2, 3$. Numerical results are listed in Tables 1–7, from which we observe that: (i) All three methods have almost the same accuracy and temporal convergence rates. (ii) It is more efficient to generate the coefficient matrix of the approximated system than that of the original problem. (iii) The FDAC is much more efficient than the FS while the number of iterations for both of them grow rapidly due to the bad condition number of the coefficient matrix. (iv) The PFDAC significantly reduces the number of iterations and

Table 1
Convergence for $d = 1$ in Experiment 1.

N	h	FS		FDAC		PFDAC	
		$\ u - u_h\ $	order	$\ u - u_h\ $	order	$\ u - u_h\ $	order
2^3	$1/2^5$	4.5547E-02	–	4.5548E-02	–	4.5548E-02	–
2^4	$1/2^5$	2.2935E-02	0.99	2.2935E-02	0.99	2.2935E-02	0.99
2^5	$1/2^5$	1.1430E-02	1.00	1.1430E-02	1.00	1.1430E-02	1.00
2^6	$1/2^5$	5.6305E-03	1.04	5.6305E-03	1.02	5.6305E-03	1.02
2^7	$1/2^5$	2.7202E-03	1.05	2.7201E-03	1.05	2.7201E-03	1.05

Table 2
Errors and convergence rates for $d = 2$ in Experiment 1.

N	h	FS		FDAC		PFDAC	
		$\ u - u_h\ $	order	$\ u - u_h\ $	order	$\ u - u_h\ $	order
2^3	$1/2^5$	2.7843E-02	–	2.7844E-02	–	2.7844E-02	–
2^4	$1/2^5$	1.3952E-02	1.00	1.3951E-02	1.00	1.3951E-02	1.00
2^5	$1/2^5$	6.8836E-03	1.02	6.8836E-03	1.02	6.8836E-03	1.02
2^6	$1/2^5$	3.3210E-03	1.05	3.3210E-03	1.05	3.3210E-03	1.05
2^7	$1/2^5$	1.5335E-03	1.11	1.5335E-03	1.11	1.5335E-03	1.11

Table 3
Errors and convergence rates for $d = 3$ in Experiment 1.

N	h	FS		FDAC		PFDAC	
		$\ u - u_h\ $	order	$\ u - u_h\ $	order	$\ u - u_h\ $	order
2^3	$1/2^5$	1.7220E-2	–	1.7221E-2	–	1.7221E-2	–
2^4	$1/2^5$	8.5739E-3	1.01	8.5738E-3	1.01	8.5738E-3	1.01
2^5	$1/2^5$	4.1799E-3	1.04	4.1799E-3	1.04	4.1799E-3	1.04
2^6	$1/2^5$	1.9669E-3	1.09	1.9670E-3	1.09	1.9669E-3	1.09
2^7	$1/2^5$	8.8281E-4	1.16	8.8281E-4	1.16	8.8281E-4	1.16

Table 4
CPU times for $d = 1$ in Experiment 1.

N	h	CPU	Iter	CPU _f	Iter _f	CPU _{pre}	Iter _{pre}
2^9	$1/2^3$	17.7 s	35	1 m 38 s	35	2.35 s	1
2^{10}	$1/2^3$	3 m 16 s	63	6 m 44 s	63	5.22 s	1
2^{11}	$1/2^3$	33 m 42 s	116	28 m 20 s	117	11.3 s	1
2^{12}	$1/2^3$	4 h 38 m	221	2 h 2 m	224	25.2 s	1
2^{13}	$1/2^3$	1 d 14 h	438	10 h 30 m	439	55.8 s	1
2^{14}	$1/2^3$	> 21 d	–	1 d 16 h	873	2 m 5 s	1

Table 5
CPU times for $d = 2$ in Experiment 1.

N	h	CPU	Iter	CPU _f	Iter _f	CPU _{pre}	Iter _{pre}
2^9	$1/2^3$	45 s	24	9 m 44 s	23	21 s	1
2^{10}	$1/2^3$	4 m 45 s	37	40 m	37	49 s	1
2^{11}	$1/2^3$	48 m 30 s	67	2 h 43 m	66	1 m 46 s	1
2^{12}	$1/2^3$	8 h 10 m	121	11 h 20 m	121	4 m 20 s	1
2^{13}	$1/2^3$	2 d 18 h	235	2 d 47 m	235	9 m 44 s	1
2^{14}	$1/2^3$	> 30 d	–	8 d 4 h	465	23 m 42 s	1

consequently saves much time compared with the FDAC, which demonstrates the effectiveness of the proposed precondition and thus provides a potential mean for large-scale and high-dimensional simulations.

Experiment 2. Let $q(t) = 1$, $K = 1$ and $d = 1$. Numerical results are presented in Tables 8–9. Similar conclusions can be obtained as Experiment 1. In addition, we observed that when K becomes larger, the numbers of iterations of PFDAC increase compared with Table 4.

5.2. Comparison with high-order scheme

We compare the performances between PFDAC and the high-order scheme (HOS) proposed in [42] by solving the following time-dependent variable-order fractional diffusion equation

Table 6
CPU times for $d = 3$ in Experiment 1.

N	h	CPU	Iter	CPU _f	Iter _f	CPU _{pre}	Iter _{pre}
2 ⁹	1/2 ²	1 m 43 s	51	12 m 22 s	52	26 s	1
2 ¹⁰	1/2 ²	12 m 2 s	94	53 m 20 s	93	59 s	1
2 ¹¹	1/2 ²	2 h 5 m	183	3 h 37 m	183	2 m 12 s	1
2 ¹²	1/2 ²	1 d 3 h	367	15 h 20 m	370	5 m 2 s	1
2 ¹³	1/2 ²	7 d 12 h	500	1 d 14 h	500	11 m	1
2 ¹⁴	1/2 ²	> 30 d	–	>5 d	–	24 m 22 s	1

Table 7
CPU times generating $\mathbf{B}_N$ and $\tilde{\mathbf{B}}_N$ in Experiment 1.

N	CPU _M	CPU _{Mf}
2 ⁹	0.22 s	0.05 s
2 ¹⁰	0.88 s	0.05 s
2 ¹¹	3.4 s	0.11 s
2 ¹²	13.8 s	0.23 s
2 ¹³	56 s	0.5 s
2 ¹⁴	3 m 4 s	1.1 s

Table 8
Errors and convergence rates of Experiment 2.

N	h	FS		FDAC		PFDAC	
		$\ u - u_h\ $	order	$\ u - u_h\ $	order	$\ u - u_h\ $	order
2 ³	1/2 ⁷	2.1724E-03	–	2.1724E-03	–	2.1724E-03	–
2 ⁴	1/2 ⁷	1.0803E-03	1.01	1.0803E-03	1.01	1.0803E-03	1.01
2 ⁵	1/2 ⁷	5.3868E-04	1.00	5.3868E-04	1.00	5.3868E-04	1.00
2 ⁶	1/2 ⁷	2.6840E-04	1.01	2.6840E-04	1.01	2.6840E-04	1.01
2 ⁷	1/2 ⁷	1.3343E-04	1.01	1.3343E-04	1.01	1.3343E-04	1.01

Table 9
CPU times of Experiment 2.

N	h	CPU	Iter	CPU _f	Iter _f	CPU _{pre}	Iter _{pre}
2 ⁹	1/2 ³	1.27 s	8	17 s	8	22 s	5
2 ¹⁰	1/2 ³	9.72 s	15	1 m 8 s	15	56 s	6
2 ¹¹	1/2 ³	2 m 9 s	29	5 m 18 s	29	2 m 23 s	6
2 ¹²	1/2 ³	29 m 2 s	59	14 m 39 s	59	5 m 1 s	6
2 ¹³	1/2 ³	10 h 3 m	118	1 h 7 m	118	12 m 25 s	7

Table 10
Comparisons of two methods in Experiment 3.

	N	$\alpha_0 = 0.2$		$\alpha_0 = 1$	
		$\ u - u_h\ $	CPU	$\ u - u_h\ $	CPU
PFDAC	2 ⁹	4.4915e-04	0.05 s	1.20100e-3	0.04 s
	2 ¹⁰	2.2629e-04	0.08 s	6.0050e-04	0.07 s
	2 ¹¹	1.1339e-04	0.12 s	3.0025e-04	0.11 s
	2 ¹²	5.5641e-05	0.26 s	1.5012e-04	0.19 s
	2 ¹³	2.6681e-05	0.44 s	7.5064e-05	0.30 s
HOS	2 ⁵	4.8884e-04	0.09 s	1.4557e-04	0.10 s
	2 ⁶	2.2512e-04	0.16 s	3.6364e-05	0.09 s
	2 ⁷	1.0245e-04	0.30 s	9.0831e-06	0.17 s
	2 ⁸	4.6396e-05	0.65 s	2.2651e-06	0.22 s
	2 ⁹	2.0949e-05	1.65 s	5.6089e-07	0.48 s

$$u'(t) + q(t)_0^C D_t^{1-\alpha(t)} u(t) = f(t) - \frac{u(0)t^{\alpha(t)-1}}{\Gamma(\alpha(t))}, \quad t \in (0, T], \quad u(0) = u_0.$$

Experiment 3. Let $q(t) = 1$, $u(t) = 1 + t^{1+\alpha_0}$ and $\alpha(t) = \alpha_0 + (\alpha_1 - \alpha_0)t$ with $\alpha_1 = 0.8$. If $\alpha_0 < 1$, the exact solution has singularity at the initial time, which is common in the context of constant-order fractional problems and can be eliminated by setting $\alpha_0 = 1$. We evaluate the CPU times (CPU) of PFDAC by adding CPU_M to CPU_{pre}. We intend to compare the CPU times of two methods under the same threshold of the error. Numerical results are presented in Table 10, from which we

observe that if the solution has singularity at the initial time, the accuracy of the HOS may be affected and the PFDAC is more efficient than the HOS. If the solution is smooth, HOS is faster than the PFDAC. Based on these motivations, we will carry out studies on fast implementations of HOS in the near future by the idea of the current work.

6. Concluding remarks

In this article we derived a fast approximation method for variable-order time-fractional diffusion PDEs in multiple space dimension. Since variable-order fractional differential operators lose the translation invariance or convolution structure, the fast solution methods developed for constant-order fractional PDEs, which depend heavily on these properties, fail to apply. In this article, we approximate the coefficient matrices by a finite sum of Toeplitz-like matrices. Consequently, we solve the discrete system in an all-at-once fashion by combining the solution technique developed for constant-order time-fractional PDEs [13] as well as the fast solution technique on the steady-state variable-order space-fractional PDEs in one space dimension [18,19]. Numerical experiments show the utility of the fast method.

Finally, we comment on some modeling issues of the two time-scale variable-order time-fractional diffusion PDE (3). For simplicity of exposition, we assume $f \equiv 0$ and the fractional order $0 < \alpha < 1$ is constant. It is well known that the classical Fickian diffusion PDE is not invariant under the time inversion $t \rightarrow -t$, which reflects the fact that the diffusion process is physically not revisable. Consequently, the two time-scale time-fractional diffusion PDE is not time invariant. In fact, since the fractional time derivative is defined as the convolution of the first-order differential operator with a power-law kernel, the fractional time differential operator is not invariant either under the time inversion. The reason is that the subdiffusive process modeled by the time-fractional derivative is still a diffusion process, although with a heavy tail, and hence is not physically revisable.

Acknowledgements

The authors would like to express their most sincere thanks to the referees for their very helpful comments and suggestions, which greatly improved the quality of this paper. This work was partially funded by the National Natural Science Foundation of China under Grants 11971272 and 12001337, by the Natural Science Foundation of Shandong Province under Grant ZR2019BA026, by the ARO MURI Grant W911NF-15-1-0562, by the National Science Foundation under Grants DMS-1620194 and DMS-2012291, and by a SPARC Graduate Research Grant from the Office of the Vice President for Research at the University of South Carolina. All data generated or analyzed during this study are included in this published article.

References

- [1] R.A. Adams, J.J.F. Fournier, *Sobolev Spaces*, Elsevier, San Diego, 2003.
- [2] B. Baeumer, M. Kovács, M.M. Meerschaert, H. Sankaranarayanan, Boundary conditions for fractional diffusion, *J. Comput. Appl. Math.* 339 (2018) 414–430.
- [3] J. Bear, *Dynamics of Fluids in Porous Media*, Elsevier, New York, 1972.
- [4] P. Becker-Kern, M. Meerschaert, H. Scheffler, Limit theorem for continuous-time random walks with two time scales, *J. Appl. Probab.* 41 (2004) 455–466.
- [5] D. Benson, R. Schumer, M.M. Meerschaert, S.W. Wheatcraft, Fractional dispersion, Lévy motions, and the MADE tracer tests, *Transp. Porous Media* 42 (2001) 211–240.
- [6] D. Bertaccini, F. Durastante, Block structured preconditioners in tensor form for the all-at-once solution of a finite volume fractional diffusion equation, *Appl. Math. Lett.* 95 (2019) 92–97.
- [7] J.X. Cao, C.P. Li, Y.Q. Chen, High-order approximation to Caputo derivatives and Caputo-type advection-diffusion equations (II), *Fract. Calc. Appl. Anal.* 18 (3) (2015) 735–761.
- [8] H. Chen, H. Wang, Numerical simulation for conservative fractional diffusion equations by an expanded mixed formulation, *J. Comput. Appl. Math.* 296 (2016) 480–498.
- [9] W. Deng, Finite element method for the space and time fractional Fokker-Planck equation, *SIAM J. Numer. Anal.* 47 (2008) 204–226.
- [10] K. Diethelm, N. Ford, A note on the well-posedness of terminal value problems for fractional differential equations, *J. Integral Equ. Appl.* 30 (2017) 371–376.
- [11] V.J. Ervin, N. Heuer, J.P. Roop, Regularity of the solution to 1-D fractional diffusion equations, *Math. Comput.* 87 (2018) 2273–2294.
- [12] H. Fu, M.K. Ng, H. Wang, A divide-and-conquer fast finite difference method for space-time fractional partial differential equation, *Comput. Math. Appl.* 73 (2017) 1233–1242.
- [13] H. Fu, H. Wang, A preconditioned fast finite difference method for space-time fractional partial differential equations, *Fract. Calc. Appl. Anal.* 20 (1) (2017) 88–116.
- [14] G. Gao, Z. Sun, The finite difference approximation for a class of fractional subdiffusion equations on a space unbounded domain, *J. Comput. Phys.* 236 (2013) 443–460.
- [15] R. Garrappa, I. Moret, M. Popolizio, Solving the time-fractional Schrödinger equation by Krylov projection methods, *J. Comput. Phys.* 293 (2015) 115–134.
- [16] J. Jia, H. Wang, A fast finite volume method for conservative space-time fractional diffusion equations discretized on space-time locally refined meshes, *Comput. Math. Appl.* 78 (5) (2019) 1345–1356.
- [17] J. Jia, H. Wang, A fast finite volume method on locally refined meshes for fractional diffusion equations, *East Asian J. Appl. Math.* 9 (2019) 755–779.
- [18] J. Jia, H. Wang, X. Zheng, A fast collocation approximation to a two-sided variable-order space-fractional diffusion equation and its analysis, available online, *J. Comput. Appl. Math.* (2020) 113234, <https://doi.org/10.1016/j.cam.2020.113234>, in press.
- [19] J. Jia, X. Zheng, H. Fu, P. Dai, H. Wang, A fast method for variable-order space-fractional diffusion equations, *Numer. Algorithms* 85 (2020) 1519–1540.
- [20] S. Jiang, J. Zhang, Q. Zhang, Z. Zhang, Fast evaluation of the Caputo fractional derivative and its applications to fractional diffusion equations, *Commun. Comput. Phys.* 21 (2017) 650–678.
- [21] X. Jin, F. Lin, Z. Zhao, Preconditioned iterative methods for two-dimensional space-fractional diffusion equations, *Commun. Comput. Phys.* 18 (2015) 469–488.

- [22] X. Jin, Y. Wei, Numerical Linear Algebra and Its Applications, Science Press, Beijing, 2004.
- [23] R. Ke, M.K. Ng, H. Sun, A fast direct method for block triangular Toeplitz-like with tri-diagonal block systems from time-fractional partial differential equations, *J. Comput. Phys.* 303 (2015) 203–211.
- [24] N. Kopteva, Error analysis of the L1 method on graded and uniform meshes for a fractional-derivative problem in two and three dimensions, *Math. Comput.* 88 (2019) 2135–2155.
- [25] C.F. Lorenzo, T.T. Hartley, Variable order and distributed order fractional operators, *Nonlinear Dyn.* 29 (2002) 57–98.
- [26] Z. Mao, J. Shen, Efficient spectral–Galerkin methods for fractional partial differential equations with variable coefficients, *J. Comput. Phys.* 307 (2016) 243–261.
- [27] M.M. Meerschaert, A. Sikorskii, Stochastic Models for Fractional Calculus, De Gruyter Studies in Mathematics, 2011.
- [28] R. Metzler, J. Klafter, The random walk's guide to anomalous diffusion: a fractional dynamics approach, *Phys. Rep.* 339 (2000) 1–77.
- [29] K. Sakamoto, M. Yamamoto, Initial value/boundary value problems for fractional diffusion-wave equations and applications to some inverse problems, *J. Math. Anal. Appl.* 382 (2011) 426–447.
- [30] R. Schumer, D. Benson, M.M. Meerschaert, B. Baeumer, Fractal mobile/immobile solute transport, *Water Resour. Res.* 39 (2003) 1–12.
- [31] S. Shen, F. Liu, J. Chen, I. Turner, V. Anh, Numerical techniques for the variable order time fractional diffusion equation, *J. Putian Univer.* 218 (22) (2012).
- [32] M. Stynes, E. O'Riordan, J.L. Gracia, Error analysis of a finite difference method on graded mesh for a time-fractional diffusion equation, *SIAM J. Numer. Anal.* 55 (2017) 1057–1079.
- [33] H. Sun, A. Chang, Y. Zhang, W. Chen, A review on variable-order fractional differential equations: mathematical foundations, physical models, numerical methods and applications, *Fract. Calc. Appl. Anal.* 22 (2019) 27–59.
- [34] H. Sun, W. Chen, C. Li, Y. Chen, Finite difference schemes for variable-order time fractional diffusion equation, *Inter. J. Bifurc. Chaos* 22 (4) (2012) 1250085.
- [35] V. Thomée, Galerkin Finite Element Methods for Parabolic Problems, Lecture Notes in Mathematics, vol. 1054, Springer-Verlag, New York, 1984.
- [36] F. Wang, H. Chen, H. Wang, Finite element simulation and efficient algorithm for fractional Cahn–Hilliard equation, *J. Comput. Appl. Math.* 356 (2019) 248–266.
- [37] H. Wang, K. Wang, T. Sircar, A direct $O(N \log^2 N)$ finite difference method for fractional diffusion equations, *J. Comput. Phys.* 229 (2010) 8095–8104.
- [38] H. Wang, X. Zheng, Wellposedness and regularity of the variable-order time-fractional diffusion equations, *J. Math. Anal. Appl.* 475 (2019) 1778–1802.
- [39] Y. Yan, Z. Sun, 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) 1028–1048.
- [40] M. Zayernouri, G.E. Karniadakis, Fractional spectral collocation method, *SIAM J. Sci. Comput.* 36 (2014) A40–A62.
- [41] F. Zeng, Z. Zhang, G. Karniadakis, A generalized spectral collocation method with tunable accuracy for variable-order fractional differential equations, *SIAM J. Sci. Comput.* 37 (2015) A2710–A2732.
- [42] X. Zhao, Z. Sun, G.E. Karniadakis, Second-order approximations for variable order fractional derivatives: algorithms and applications, *J. Comput. Phys.* 293 (2015) 184–200.
- [43] X. Zheng, H. Wang, Optimal-order error estimates of finite element approximations to variable-order time-fractional diffusion equations without regularity assumptions of the true solutions, *IMA J. Numer. Anal.* (2020), <https://doi.org/10.1093/imanum/draa013>.
- [44] P. Zhuang, F. Liu, V. Anh, I. Turner, Numerical methods for the variable-order fractional advection-diffusion equation with a nonlinear source term, *SIAM J. Numer. Anal.* 47 (2009) 1760–1781.