Wavelet Galerkin method for fractional elliptic differential equations

Weihua Deng†, Yuwei Lin, Zhijiang Zhang
School of Mathematics and Statistics, Gansu Key Laboratory of Applied Mathematics and Complex Systems, Lanzhou University, Lanzhou 730000, P.R. China

[Received on 28 May 2014]

Under the guidance of the general theory developed for classical partial differential equations (PDEs), we investigate the Riesz bases of wavelets in the spaces where fractional PDEs usually work, and their applications in numerically solving fractional elliptic differential equations (FEDEs). The technique issues are solved and the detailed algorithm descriptions are provided. Compared with the ordinary Galerkin methods, the wavelet Galerkin method we propose for FEDEs has the striking benefit of efficiency, since the condition numbers of the corresponding stiffness matrices are small and uniformly bounded; and the Toeplitz structure of the matrix still can be used to reduce cost. Numerical results and comparison with the ordinary Galerkin methods are presented to demonstrate the advantages of the wavelet Galerkin method we provide.

Keywords: fractional elliptic equation; Riesz bases; B-splines function; wavelet Galerkin method; condition number

AMS subject classifications: 35R11; 65T60; 65N30

1. Introduction

In recent decades, fractional operators have been playing more and more important roles in building the models [Diethelm (2010)], e.g., in statistical physics (subdiffusion and superdiffusion), mechanics (theory of viscoelasticity and viscoplasticity), (bio-)chemistry (modelling of polymers and proteins), electrical engineering (transmission of ultrasound waves), medicine (modelling of human tissue under mechanical loads), etc. And in most of the cases, the models are appeared in the form of the fractional partial differential equations (PDEs), including the time dependent fractional PDEs and steady state fractional PDEs. Efficiently solving these fractional PDEs naturally becomes an urgent topic. Because of the nonlocal properties of fractional operators, obtaining the analytical solutions of the fractional PDEs is more challenging or sometimes even impossible; or the obtained analytical solutions are less valuable (expressed by transcendental functions or infinite series). Luckily, some important progress has been made for numerically solving the fractional PDEs by finite difference methods, e.g., see [Deng & Chen (2014); Meerschaert & Tadjeran (2004); Sousa & Li (2011); Tian et al. (2014); Yuste (2006); Zhuang et al. (2009)], finite element methods [Deng (2008); Ervin & Roop (2006)], spectral methods [Li & Xu (2010); Zayernouri & Karniadakis (2013)], etc.

For the time dependent fractional PDEs, there are already some works to deal with the issue of computational efficiency, including the method of using the Toeplitz structure of the matrixes to reduce computational cost [Wang & Basu (2012)] and the multigrid method [Chen et al. (2014); Pang & Sun (2012)]. Heavy computational costs caused by the fast increasing of the condition numbers of the corresponding stiffness matrix with the mesh refinement and the inherent nonlocal properties of fractional

†Corresponding author. Email: dengwh@lzu.edu.cn
operators are the main challenges that numerically solving the fractional elliptic differential equations (FEDEs) faces. The condition number of the ordinary Galerkin equation is proportional to \( h^{-2} \) for classical second order elliptical differential equations, and is proportional to \( h^{-4} \) for fourth order ones [Jia & Zhao (2011)], where \( h \) is the mesh size. And the condition number of the ordinary Galerkin equation for FEDEs with \( \beta \)-th order fractional derivative is proportional to \( h^{-\beta} \) [Deng & Hesthaven (2013)]. So this problem is expected to be solved by the multiresolution methods including multigrid methods [Braess (1995); Bramble et al. (1990)] and wavelet methods [Christon & Roach (2000); Jia & Zhao (2011)]. It seems there are few works on the numerical methods for FEDEs [Ervin & Roop (2006); Wang & Yang (2013)], and almost no works for considering the efficiency of the numerical methods.

Wavelets have the strong multiresolution properties, and have been proven to be a powerful tool in signal and image processing such as image compression and denoising. In recent decades, the wavelet methods have also been well developed in solving the classical PDEs. For the numerical treatment of PDEs, the efficiency of the wavelet method is greatly impacted by the properties of the wavelet bases; and in the sense of controlling the condition number, one can choose the Riesz bases of spline wavelets [Jia & Liu (2006); Jia (2006)]. Spline wavelets with short support are investigated in [Jia et al. (2011)] and [Han & Shen (2006)]. The paper [Jia (2009)] constructs the Riesz bases of spline wavelets on the interval \([0,1]\) with homogeneous boundary conditions. One can also refer to [Jia & Zhao (2011)] for the general theory of the construction of Riesz bases of wavelets and their applications to the numerical solutions of elliptic differential equations. Under the guidance of the theory being well developed for classical PDEs, in this paper we discuss the Riesz bases in the spaces where fractional PDEs usually work, and their applications in effectively solving FDEEs. The central gain of using the wavelet Galerkin method to solve FDEEs is its efficiency since the condition numbers of the corresponding stiffness matrix are small and uniformly bounded. The concrete FDEEs we discuss are the following one and two dimensional steady state fractional equations:

\[
-D_{\alpha}^{\alpha}u(x) = \frac{1}{\Gamma(\alpha)} \int_a^x (x - \xi)^{\alpha - 1} u(\xi) d\xi, \quad (1.3)
\]

and

\[
-D_{\beta}^{\beta}u(x) = \frac{1}{\Gamma(\beta)} \int_x^b (\xi - x)^{\beta - 1} u(\xi) d\xi. \quad (1.4)
\]

The outline of this paper is as follows. In Section 2, the fractional Sobolev space and a class of B-spline functions are firstly introduced; based on these functions, we introduce the Riesz bases in one and two dimensional fractional Sobolev spaces. In Section 3, we discuss the wavelet Galerkin method for FDEEs, present its detailed algorithm description, and the extensive numerical experiments are also performed to show its powerfulness. We conclude the paper with some remarks in the last section.
2. Wavelet Riesz bases in fractional Sobolev space

The Riesz bases play a vital role in controlling the condition number of stiffness matrix when using wavelet Galerkin method to solve the FEDEs. We present the Riesz bases in one and two dimensional fractional Sobolev spaces. First, we introduce the spaces where FEDEs work and their relations to the fractional order Hilbert spaces [Ervin & Roop (2006)].

2.2 (right fractional derivative) Let $u$ be a function defined on $R$, $\mu > 0$, $n$ be the smallest integer greater than $\mu$ ($n - 1 \leqslant \mu < n$), and $\sigma = n - \mu$. Then the right fractional derivative of order $\mu$ is defined to be

$$D^\mu u := D_{-\infty}^\sigma u(x) = \frac{1}{\Gamma(\sigma)} \frac{d^n}{dx^n} \int_{-\infty}^{x} (x - \xi)^{\sigma-1} u(\xi) d\xi.$$  

2.3 (left fractional derivative) Let $u$ be a function defined on $R$, $\mu > 0$, $n$ be the smallest integer greater than $\mu$ ($n - 1 \leqslant \mu < n$), and $\sigma = n - \mu$. Then the left fractional derivative of order $\mu$ is defined to be

$$D^{\mu} u := \frac{(-1)^n}{\Gamma(\sigma)} \frac{d^n}{dx^n} \int_{x}^{\infty} (\xi - x)^{\sigma-1} u(\xi) d\xi.$$  

2.1 Fractional Sobolev space

We introduce the abstract setting for FEDEs, including the left, right, and symmetric fractional derivative spaces; and then show the equivalence of the fractional derivative spaces with fractional order Hilbert spaces [Ervin & Roop (2006)].

Definition 2.1 (left fractional derivative) Let $u$ be a function defined on $R$, $\mu > 0$, $n$ be the smallest integer greater than $\mu$ ($n - 1 \leqslant \mu < n$), and $\sigma = n - \mu$. Then the left fractional derivative of order $\mu$ is defined to be

$$D^\mu u := D_{-\infty}^\sigma u(x) = \frac{1}{\Gamma(\sigma)} \frac{d^n}{dx^n} \int_{-\infty}^{x} (x - \xi)^{\sigma-1} u(\xi) d\xi,$$

and

$$D^{\mu} u := \frac{(-1)^n}{\Gamma(\sigma)} \frac{d^n}{dx^n} \int_{x}^{\infty} (\xi - x)^{\sigma-1} u(\xi) d\xi.$$  

Note: If $\text{supp}(u) \subset (a, b)$, then $D^\mu u = \lambda D^\mu u$ and $D^{\mu} u = \lambda D^\mu u$, where $\lambda D^\mu u$ and $\lambda D^{\mu} u$ are the left and right Riemann-Liouville fractional derivative of order $\mu$ defined as

$$\lambda D^\mu u = \frac{1}{\Gamma(\sigma)} \frac{d^n}{dx^n} \int_{a}^{x} (x - \xi)^{\sigma-1} u(\xi) d\xi,$$

and

$$\lambda D^{\mu} u = \frac{(-1)^n}{\Gamma(\sigma)} \frac{d^n}{dx^n} \int_{x}^{b} (\xi - x)^{\sigma-1} u(\xi) d\xi.$$  

Definition 2.3 (left fractional derivative space) Let $\mu > 0$. Define the semi-norm

$$| u |_{L^2(R)} := || D^\mu u ||_{L^2(R)},$$

and norm

$$|| u ||_{L^2(R)} := (|| u ||_{L^2(R)}^2 + | u |_{L^2(R)}^2)^{1/2},$$

and let $L^2(R)$ denote the closure of $C_0^\infty(R)$ with respect to $|| \cdot ||_{L^2(R)}$.

Definition 2.4 (right fractional derivative space) Let $\mu > 0$. Define the semi-norm

$$| u |_{L^2(R)} := || D^{\mu} u ||_{L^2(R)},$$

and norm

$$|| u ||_{L^2(R)} := (|| u ||_{L^2(R)}^2 + | u |_{L^2(R)}^2)^{1/2},$$

and let $L^2(R)$ denote the closure of $C_0^\infty(R)$ with respect to $|| \cdot ||_{L^2(R)}$.  

The Fourier transform of a function \( f \in L^1(R) \) is defined by

\[
\hat{f}(\xi) = \frac{1}{\sqrt{2\pi}} \int_R f(x) e^{-ix \xi} \, dx, \quad \xi \in R.
\]

The Fourier transform can be naturally extended to functions in \( L^2(R) \). For \( \mu > 0 \), we denote by \( H^\mu(R) \) the Sobolev space of all functions \( f \in L^2(R) \) such that the seminorm

\[
|f|_{H^\mu(R)} := \left( \frac{1}{2\pi} \int_R |\hat{f}(\xi)|^2 |\xi|^{2\mu} \, d\xi \right)^{1/2}
\]

is finite. The space \( H^\mu(R) \) is a Hilbert space with the inner product given by

\[
\langle f, g \rangle_{H^\mu(R)} := \frac{1}{2\pi} \int_R \hat{f}(\xi) \overline{\hat{g}(\xi)} (1 + |\xi|^{2\mu}) \, d\xi, \quad f, g \in H^\mu(R).
\]

The corresponding norm in \( H^\mu(R) \) is given by

\[
\| f \|_{H^\mu(R)} := \sqrt{\| f \|_{L^2(R)}^2 + |f|_{H^\mu(R)}^2}.
\]

**Lemma 2.1 (Ervin & Roop 2006)** Let \( \mu > 0 \). The spaces \( J^\mu_L(R) \), \( J^\mu_R(R) \), and \( H^\mu(R) \) are equal with equivalent semi-norms and norms.

**Definition 2.5** (symmetric fractional derivative space) Let \( \mu > 0, \mu \neq n - 1/2, n \in N \). Define the semi-norm

\[
| u |_{J^\mu_S(R)} := |(D^\mu u, D^\mu u)|_{L^2(R)}^{1/2},
\]

and norm

\[
\| u \|_{J^\mu_S(R)} := (\| u \|^2_{L^2(R)} + | u |_{J^\mu_S(R)}^2)^{1/2},
\]

and let \( J^\mu_S(R) \) denote the closure of \( C_0^\infty(R) \) with respect to \( \| \cdot \|_{J^\mu_S(R)} \).

**Lemma 2.2 (Ervin & Roop 2006)** For \( \mu > 0, \mu \neq n - 1/2, n \in N \), the spaces \( J^\mu_L(R) \) and \( J^\mu_R(R) \) are equal, with equivalent semi-norms and norms.

**Definition 2.6** Define the spaces \( J^\mu_{L,0}(\Omega), J^\mu_{R,0}(\Omega), J^\mu_{S,0}(\Omega) \), and \( H^\mu_0(\Omega) \) as the closures of \( C_0^\infty(\Omega) \) under their respective norms.

We next turn to the equivalence of the fractional derivative spaces \( J^\mu_{L,0}(\Omega), J^\mu_{R,0}(\Omega), J^\mu_{S,0}(\Omega) \), and the fractional order Hilbert space \( H^\mu_0(\Omega) \).

**Lemma 2.3** (Ervin & Roop 2006) Let \( \mu > 0 \). Then the spaces \( J^\mu_{L,0}(\Omega), J^\mu_{R,0}(\Omega) \), and \( H^\mu_0(\Omega) \) are equal. Also, if \( \mu \neq n - 1/2, n \in N \), the spaces \( J^\mu_{L,0}(\Omega), J^\mu_{R,0}(\Omega) \), and \( H^\mu_0(\Omega) \) have equivalent semi-norms and norms.

### 2.2 Wavelet bases and the related lemmas

Let \( N \) denote the set of positive integers, \( J \) be a (finite or infinite) countable set. By \( \ell(J) \) we denote the linear space of all complex-valued sequences \( (u_j)_{j \in J} \); \( \ell_0(J) \) denotes the linear space of all sequences \( (u_j)_{j \in J} \) with only finite nonzero terms; and \( \ell^2(J) \) denotes the linear space of all sequences \( u = (u_j)_{j \in J} \) such that \( \| u \|_2 := (\sum_{j \in J} |u_j|^2)^{1/2} < \infty \).
Let $H$ be a Hilbert space. A sequence $\{v_j\}_{j \in J}$ in $H$ is said to be a **Riesz sequence** if there exist two positive constants $C_1$ and $C_2$ such that the inequalities

$$C_1 \left( \sum_{j \in J} |c_j|^2 \right)^{1/2} \leq \| \sum_{j \in J} c_j v_j \| \leq C_2 \left( \sum_{j \in J} |c_j|^2 \right)^{1/2}$$

(2.2)

hold true for every sequence $(c_j)_{j \in J}$ in $\ell_2(J)$. If this is the case, then the series $\sum_{j \in J} c_j v_j$ converges unconditionally for every $(c_j)_{j \in J}$ in $\ell_2(J)$, and the inequalities in (2.2) are valid for all $(c_j)_{j \in J}$ in $\ell^2(J)$. We call $C_1$ a Riesz lower bound and $C_2$ a Riesz upper bound. If $\{v_j\}_{j \in J}$ is a Riesz sequence in $H$, and the linear span of $\{v_j\}_{j \in J}$ is dense in $H$, then $\{v_j\}_{j \in J}$ is a **Riesz basis** of $H$.

In numerical simulations, spline wavelet bases are more popular, since they are relatively smooth, have a small support, and can be got in a closed form. The widely and effectively way to build the Riesz bases is based on multiresolution analysis. Following [Jia (2009)], we first introduce the Riesz bases in one dimension, then extend them to two dimensional case.

### 2.2.1 Riesz bases in $H^\mu_0((0, 1))$

For a positive integer $m$, let $M_m$ denote the $B$-spline of order $m$, which is the convolution of $m$ copies of the characteristic function of the interval $[0, 1]$: 

$$M_m(x) = \int_0^1 M_{m-1}(x-t) \, dt, \quad x \in \mathbb{R},$$

where $M_1 := \chi_{[0,1]}$. For $m = 2$ and $3$, the spline functions are given as follows,

$$M_2(x) = \begin{cases} x, & 0 \leq x \leq 1, \\ 2-x, & 1 < x \leq 2, \\ 0, & \text{else}; \end{cases}$$

$$M_3(x) = \begin{cases} \frac{1}{2}x^2, & 0 \leq x \leq 1, \\ -x^2 + 3x - \frac{3}{2}, & 1 \leq x < 2, \\ \frac{1}{2}x^2 - 3x + \frac{9}{2}, & 2 \leq x \leq 3, \\ 0, & \text{else}, \end{cases}$$

and they, respectively, satisfy the following refinement equations,

$$M_2(x) = \frac{1}{2} M_2(2x) + M_2(2x-1) + \frac{1}{2} M_2(2x-2);$$

(2.3)

$$M_3(x) = \frac{1}{4} M_3(2x) + \frac{3}{4} M_3(2x-1) + \frac{3}{4} M_3(2x-2) + \frac{1}{4} M_3(2x-3).$$

(2.4)

From the definition, it follows immediately that $M_m$ is supported on $[0, m]$, $M_m(x) > 0$ and $M_m(m-x) = M_m(x)$ for $0 < x < m$. Moreover, $M_m \in H^\mu_0([0, m])$ for $0 < \mu < m - 1/2$. Let 

$$\phi_{n,j}(x) := 2^{n/2} M_n(2^n x - j), \quad j \in I_n := \{0, 1, \ldots, 2^n - r\}.$$ 

Then there exists $n_0 \in N$, such that $n \geq n_0$, $V_n := \text{span}\{\phi_{n,j} : j \in I_n\}$ is a subspace of $H^\mu_0((0, 1))$ for $0 \leq \mu \leq r - 1/2$. Evidently, $V_n \subset V_{n+1}$, for $r \geq 2$, each function $f$ in $V_n$ satisfies the homogeneous boundary conditions 

$$f^{(k)}(0) = f^{(k)}(1) = 0, \quad k = 0, 1, \ldots, r-2.$$
And the following result on approximation accuracy holds:
\[ \inf_{v \in V_n} ||u - v||_{H^r(0,1)} \leq C(1/2^n)^{-\mu} ||u||_{H^r(0,1)}, \quad u \in H_{0}^{r-1}(0,1) \cap H^{r}(0,1). \] (2.5)

Its proof is similar to the one given in [Jia & Zhao (2011)].

For the construction of wavelet bases, suppose that \( t \in N, r \geq t, \) and \( r + t \) is an even integer. Let \( n_0 \) be the least integer such that \( 2^{n_0} \geq r + t, \) and define \( \phi_{n,j}(x) \)
\[ \hat{\phi}_{n,j}(x) := 2^{n/2} M_j(2^n x - j - (r - t)/2). \]

Let \( V_n := \text{span}\{ \phi_{n,j} : j \in I_n \} \), obviously, \( \phi_{n,j}(x) = 0 \) for \( x \in R \setminus [0, 1] \) when \( n \geq n_0, V_n \subset \hat{V}_{n+1} \). Then we find the direct sum decomposition of \( V_{n+1} (V_n \oplus W_n) \) and \( \hat{V}_{n+1} (\hat{V}_n \oplus \hat{W}_n) \) by demanding that \( W_n := V_{n+1} \cap \hat{V}_{n+1} \) and \( \hat{W}_n := \hat{V}_{n+1} \cap \hat{V}_n \), respectively. The desired wavelet bases for \( W_n \) and \( \hat{W}_n \) can be constructed by studying the slant matrices. Two important wavelet bases for \( r = 2 \) and \( 3 \) are given as follows:

(a) For \( r = 2 \) and \( t = 2 \), let
\[ \psi(x) = \frac{1}{24} M_2(2x) - \frac{1}{4} M_2(2x - 1) + \frac{5}{12} M_2(2x - 2) - \frac{1}{4} M_2(2x - 3) + \frac{1}{24} M_2(2x - 4), \] (2.6)
and
\[ \psi_1(x) = \frac{3}{8} M_2(2x) - \frac{1}{4} M_2(2x - 1) + \frac{1}{24} M_2(2x - 2). \] (2.7)

For \( n \geq 2 \) and \( x \in R \), we define

\[ \psi_{n,j}(x) := \begin{cases} 2^{n/2} \psi_j(2^n x) & j = 1, \\ 2^{n/2} \psi_j(2^n x - j + 2) & j = 2, \ldots, 2^n - 1, \\ 2^{n/2} \psi_{2^n - j + 1}(2^n (1 - x)) & j = 2^n. \end{cases} \]

(b) For \( r = 3 \) and \( t = 1 \), let
\[ \psi(x) = \frac{1}{12} M_3(2x) - \frac{5}{12} M_3(2x - 1) + \frac{5}{12} M_3(2x - 2) - \frac{1}{12} M_3(2x - 3), \] (2.8)
and
\[ \psi_1(x) = \frac{5}{12} M_3(2x) - \frac{1}{12} M_3(2x - 1). \] (2.9)

For \( n \geq 2 \) and \( x \in R \), we define

\[ \psi_{n,j}(x) := \begin{cases} 2^{n/2} \psi_j(2^n x) & j = 1, \\ 2^{n/2} \psi_j(2^n x - j + 2) & j = 2, \ldots, 2^n - 1, \\ 2^{n/2} \psi_{2^n - j + 1}(2^n (1 - x)) & j = 2^n. \end{cases} \]

Then we have the following important lemma.

**Lemma 2.4** [Jia (2009)] For \( n \geq n_0 \) and \( j \in J_n := \{1, 2, \ldots, 2^n\} \), let \( \psi_{n,j} \) be the functions as constructed above. Then the set
\[ \{ 2^{-n_0\mu} \phi_{n_0,j} : j \in I_{n_0} \} \cup \bigcup_{n=n_0}^{\infty} \{ 2^{-n\mu} \psi_{n,j} : n \geq n_0, j \in J_n \}, \]
forms a Riesz basis of \( H_{0}^{r}(0,1) \) for \( 0 < \mu < r - \frac{1}{2} \).
2.2.2 Riesz bases in $H_0^\mu((0,1)^2)$. In order to obtain a Riesz basis of $H_0^\mu((0,1)^2)$, we use the tensor product denoted by $\otimes$. For two functions $\nu$ and $\omega$ defined on $(0,1)$, we use $\nu \otimes \omega$ to denote the function on $(0,1)^2$ given by

$$v \otimes \omega(x,y) := v(x)\omega(y), \quad 0 \leq x, y \leq 1.$$ 

For $n \geq n_0, n \in \mathbb{N}$, let $I_n := \{ j = (j_1, j_2) \in \mathbb{Z}^2 : 0 \leq j_1 \leq 2^n - r, 0 \leq j_2 \leq 2^n - r \}$. We denote the approximate space of $H_0^\mu((0,1)^2)$ by $(V_n)_{n \geq n_0}$. Define

$$\tilde{\phi}_{n,j} := \phi_{n,j_1} \otimes \phi_{n,j_2}; \quad j \in I_n;$$

$$\tilde{\Phi}_n := \{ \tilde{\phi}_{n,j}, j = (j_1, j_2) \in I_n \};$$

$$V_n := \text{span}\{ \tilde{\Phi}_n \}.$$ 

Similarly we can define the corresponding $\tilde{\phi}_{n,j}$.

For the sequence of the subspaces $V_n$, we have the following properties

- $\widetilde{V}_{n_0} \subset \widetilde{V}_{n_0+1} \subset \widetilde{V}_{n_0+2} \subset \ldots$;
- $\bigcup_{n=n_0}^{\infty} \widetilde{V}_n$ is dense in $H_0^\mu((0,1)^2)$ for $0 < \mu < r - 1/2$;
- $\text{dim}(V_n) = (2^n - r + 1)^2$;

Furthermore, define

$$\Gamma_n := \{ \phi_{n,j_1} \otimes \psi_{n,j_2} : 0 \leq j_1 \leq 2^n - r, 1 \leq j_2 \leq 2^n \};$$

$$\Gamma_n' := \{ \psi_{n,j_1} \otimes \phi_{n,j_2} : 0 \leq j_2 \leq 2^n - r, 1 \leq j_1 \leq 2^n \};$$

$$\Gamma_n'' := \{ \phi_{n,j_1} \otimes \psi_{n,j_2} : 1 \leq j_1 \leq 2^n, 1 \leq j_2 \leq 2^n \}.$$ 

For $n \geq n_0$, let $\Gamma_n := \Gamma_n' \cup \Gamma_n'' \cup \Gamma_n'''$, and $\widetilde{W}_n := \text{span}\{ \Gamma_n \}$, then $\Gamma_n$ is a Riesz basis of $\widetilde{W}_n$ in the $L_2^2$ space. The dimensions satisfied the following relation:

$$\text{dim}(\widetilde{V}_{n+1}) = \text{dim}(\widetilde{V}_n) + \text{dim}(\widetilde{W}_n)$$

$$= (2^n - r + 1)^2 + (2^n - r + 1)2^n + 2^n(2^n - r + 1) + 2^n$$

$$= (2^{n+1} - r + 1)^2.$$ 

For every $f \in L^2(0,1)^2$ and $n \geq n_0$, let $P_n f$ be the unique element in $\widetilde{V}_n$ such that

$$< P_n f, \tilde{\phi}_{n,j} > = < f, \tilde{\phi}_{n,j} >, \quad \forall j \in I_n.$$ 

It is easy to check that $P_n$ is a projector from $L^2(0,1)^2$ onto $\widetilde{V}_n$, $\widetilde{W}_n$ is the kernel space of $P_n$, and $\widetilde{V}_{n+1}$ is the direct sum of $\widetilde{V}_n$ and $\widetilde{W}_n$. Using the similar way of the proof to the one dimensional case given in [Jia (2009)], we can prove the following lemma.

**Lemma 2.5** For $0 < \mu < r - 1/2$, the set

$$\{ 2^{-n_0 \mu} \tilde{\phi}_{n_0} \} \cup \bigcup_{n=n_0}^{\infty} \{ 2^{-n \mu} \Gamma_n \}$$

forms a Riesz basis of $H_0^\mu((0,1)^2)$. 
We have discussed the Riesz bases for the fractional Sobolev spaces in one and two dimensional cases. Lemma 2.6 will present the reason of constructing the Riesz bases and give the theoretical foundation of the following numerical analysis.

**Lemma 2.6** (Jia & Zhao (2011)) If \( \Psi^\text{norm} \) is a Riesz basis of \( H^\mu_0(\Omega) \) and the bilinear form \( a(\cdot, \cdot) \) corresponding to the equation is symmetric, continuous, and coercive, then the condition number of the stiffness matrix associated with \( \Psi^\text{norm} \), i.e., \( B_n := (a(\chi, \psi))_{\chi, \psi \in \Psi^\text{norm}} \), is uniformly bounded.

3. Wavelet Galerkin method for fractional elliptic differential equation

In this section, we apply the Riesz bases to FEDEs in one and two dimensional spaces and present the corresponding algorithms. The provided methods are compared with the ordinary Galerkin method from the condition number and computational time that show the efficiency of wavelet Galerkin method.

3.1 Wavelet Galerkin method for one dimensional FEDE

For the one dimensional space, we have provided the Riesz bases of the fractional Sobolev spaces in Lemma 2.4. And they can be used to solve the following equation

\[
-Da(p_0D_x^{-\beta} + q_1D_1^{-\beta})Du = f, \quad x \in \Omega = (0,1),
\]

where \( D \) represents the first spatial derivative, \( p_0D_x^{-\beta} \) and \( q_1D_1^{-\beta} \) are the left and right Riemann-Liouville fractional integral operators, respectively, with \( 0 \leq \beta < 1 \) and \( p + q = 1 \).

The variational formulation of this fractional differential equation is as follows

\[
B(u, v) = (f, v), \quad \forall v \in H^\mu_0(0,1),
\]

where \( B(u, v) = ap(\langle \cdot D_x^{-\beta}Du_n, \cdot Du_n \rangle + ap(\langle \cdot D_1^{-\beta}Du_n, \cdot Du_n \rangle), \mu = \frac{2-\beta}{2}, \) and \( \frac{1}{2} < \mu \leq 1 \). According to Ervin & Roop (2006), the above variational formulation has the unique solution in space \( H^\mu_0(0,1) \). In order to numerically solve the resulting variational formulation, we can use the subspace \( V_n \) to approximate the space \( H^\mu_0(0,1) \), i.e., find a \( u_n \in V_n \) such that

\[
ap(\langle \cdot D_x^{-\beta}Du_n, \cdot Du_n \rangle + ap(\langle \cdot D_1^{-\beta}Du_n, \cdot Du_n \rangle, f, v), \forall v \in V_n.
\]

Suppose that \( \Phi_n := \{ \phi_{n,j} : j \in I_n \} \) being a basis of \( V_n \) and \( u_n = \sum_{\phi \in \Phi_n} y_n \phi \). Let \( A_n \) be the matrix \( ap(\langle \cdot D_x^{-\beta}D\sigma, D\phi \rangle + ap(\langle \cdot D_1^{-\beta}D\sigma, D\phi \rangle), \sigma, \phi \in \Phi_n, \) and \( \xi_n \) the column vector \( (\langle f, \phi \rangle)_{\phi \in \Phi_n} \). Then the column vector \( y_n = (y_\phi)_{\phi \in \Phi_n} \) is the solution of the system of linear equations

\[
A_n y_n = \xi_n.
\]

We also employ the Riesz bases constructed in the previous section to solve the variational problem. For \( n \geq n_0 \), we have \( \Psi_n := \{ 2^{-n_0\mu} \phi_{n,j} : j \in I_n \} \cup \bigcup_{k=n_0}^{\mu - 1} \{ 2^{-k\mu} \psi_{k,j} : j \in J_k \} \). Similarly, find a column vector \( z_n = (z_\psi)_{\psi \in \Psi_n} \) to satisfy

\[
B_n z_n = \eta_n,
\]

where \( B_n = (ap(\langle \cdot D_x^{-\beta}D\chi, D\psi \rangle + ap(\langle \cdot D_1^{-\beta}D\chi, D\psi \rangle), \chi, \psi \in \Psi_n, \) and \( \eta_n \) denote the column vector \( (\langle f, \psi \rangle)_{\psi \in \Psi_n} \). Hence, \( u_n = \sum_{\psi \in \Psi_n} z_\psi \psi \) is the approximate solution of \( u \) in \( V_n \).
Since $\Phi_n$ and $\Psi_n$ are two different bases of $V_n$, there is a unique transformation $S_n$ between $\Phi_n$ and $\Psi_n$, such that $\Psi_n=S_n\Phi_n$, which is called wavelet transformation. So we have $B_n=S_nA_nS_n^T$, $\eta_n=S_n\xi_n$ and that (3.4) is equivalent to (3.5). If we set $y_n=S_n^Tz_n$, then (3.4) is preconditioned by the transformation $S_n$. And $S_n$ is called a preconditioner.

Now, we present the algorithm of generating the matrixes $A_n$ and $B_n$ in Algorithm 1 (for performing the numerical analysis, sometimes we need explicitly to get the matrixes). For the corresponding classical problems, the matrix $A_n$ is symmetric and sparse. But for the fractional problems, $A_n$ is dense and also nonsymmetric when $p \neq q$. Fortunately, because $\phi_{n,j}$ are the dilation and translation of one single function $M_r(x)$, the matrix $A_n$ has the Toeplitz (diagonal-constant) structure. Then we only need to produce the first row and column of $A_n$, which greatly reduces the computation and storage costs.

**Algorithm 1 Generating matrix $A_n$ and $B_n$ for 1D**

1. for $i = 0, 1, \ldots, r - 1$
2.   $a_1(i) = \langle 0D_x^{-\beta} \phi_{n,i}^r, \phi_{n,0} \rangle$
3. end for
4. for $j = 0, 1, \ldots, 2^n - r$
5.   $q_1(j) = \langle 0D_x^{-\beta} \phi_{n,0}^r, \phi_{n,j} \rangle$
6. end for
7. Initialize the unit matrix $P$: $P = \text{speye}(2^n - r + 1)$
8. for $k = n_0 + 1, \ldots, n$
9.   Let $P_k$ satisfy:
   
   $$(P_k^{-1})_{(2^{(k-1)-1}i, 2^{(k-1)-1}j)} = P_k \Phi_k$$
10. $P_k(1 : 2^{k-1} - r + 1, 1 : 2^{k-1} - r + 1) := P \times P_k(1 : 2^{k-1} - r + 1, 1 : 2^{k-1} - r + 1)$
    $P_k(1 : 2^{k-1} - r + 1, 2^{k-1} - r + 2 : \text{end}) := P \times P_k(1 : 2^{k-1} - r + 1, 2^{k-1} - r + 2 : \text{end})$
    $P := P_k$
11. end for
12. $P(1 : 2^n - r + 1, :) = 2^{-n_0}P(1 : 2^n - r + 1, :)$
13. Get the transformation matrix $S_n$: $S_n = P$
14. $A_n = a \times p \mathcal{T}(q_1, a_1) + a \times q \mathcal{T}(a_1, q_1)$
15. $B_n := S_n A_n S_n^T$

Note: $\Gamma_k := \{\psi_{k,j} : j \in J_k\}$, and $\mathcal{T}(\text{col}, \text{row})$ denote the Toeplitz matrix produced by its first column $\text{col}$ and its first row $\text{row}$, $\mu = 1 - \beta/2$.

When taking $p = q = 0.5$ in (3.1), the matrixes $A_n$ and $B_n$ are both symmetrical; in Table 1, it is shown that the increasing of the condition numbers of $A_n$ is as $O(h^{-1/2})$, where $h$ is the mesh size; and the condition numbers of the corresponding $B_n$ are uniformly bounded w.r.t $h$, which confirms Lemma 2.6. The observations also hold for the nonsymmetrical case with $p = 1$ and $q = 0$ in (3.1), see Table 2.

For further showing the powerfulness of the wavelet Galerkin method (solving the algebraic equation w.r.t. $B_n$), we use the Krylov subspace method to solve the algebraic equations w.r.t. $A_n$ and $B_n$, respectively. In fact, solving the algebraic system of $B_n$ is essentially to solve the preconditioned system of $A_n$. It is well known that the conjugate gradient method has the properties of short recursive and residuals minimality; a small condition number usually means a fast iterative speed; and it can only be used to the symmetric positive define systems. For the nonsymmetrical system, the Bi-CGSTAB method is pop-
Table 1. The condition numbers of the matrixes $A_n$ and $B_n$ with $p = 0.5$, $q = 0.5$, and $r = 2$

| $\beta$ | $n$  | size of $B_n$ | $\kappa(A_n)$ | $\kappa(B_n)$ |
|---------|-----|---------------|---------------|---------------|
|        | 3   | $7 \times 7$  | 10.0502       | 8.7751        |
|        | 4   | $15 \times 15$| 28.4815       | 10.0365       |
|        | 5   | $31 \times 31$| 80.6947       | 10.6426       |
| $\beta = 0.5$ | 6   | $63 \times 63$| 228.5288      | 11.0895       |
|        | 7   | $127 \times 127$| 646.8779     | 11.4778       |
|        | 8   | $255 \times 255$| 1.8304e+03  | 11.8235       |
|        | 9   | $511 \times 511$| 5.1784e+03  | 12.1302       |
|        | 10  | $1023 \times 1023$| 1.4648e+04 | 12.4007       |
|        | 3   | $7 \times 7$  | 6.2382        | 9.2001        |
|        | 4   | $15 \times 15$| 14.7486       | 10.1688       |
|        | 5   | $31 \times 31$| 35.0981       | 10.7312       |
| $\beta = 0.75$ | 6   | $63 \times 63$| 83.6018       | 11.2143       |
|        | 7   | $127 \times 127$| 199.0556     | 11.6654       |
|        | 8   | $255 \times 255$| 473.7381     | 12.0813       |
|        | 9   | $511 \times 511$| 1.1271e+03  | 12.4573       |
|        | 10  | $1023 \times 1023$| 2.6813e+03 | 12.7929       |

Table 2. The condition numbers of the matrixes $A_n$ and $B_n$ with $p = 1$, $q = 0$, and $r = 2$

| $\beta$ | $n$  | size of $B_n$ | $\kappa(A_n)$ | $\kappa(B_n)$ |
|---------|-----|---------------|---------------|---------------|
|        | 3   | $7 \times 7$  | 8.3362        | 6.6338        |
|        | 4   | $15 \times 15$| 23.2013       | 7.6409        |
|        | 5   | $31 \times 31$| 65.3566       | 8.7345        |
| $\beta = 0.5$ | 6   | $63 \times 63$| 184.6258      | 9.5570        |
|        | 7   | $127 \times 127$| 522.0054     | 10.2252       |
|        | 8   | $255 \times 255$| 1.4763e+03  | 10.7896       |
|        | 9   | $511 \times 511$| 4.1754e+03  | 11.2744       |
|        | 10  | $1023 \times 1023$| 1.1810e+04 | 11.6965       |
|        | 3   | $7 \times 7$  | 6.2421        | 6.8589        |
|        | 4   | $15 \times 15$| 14.2077       | 8.5584        |
|        | 5   | $31 \times 31$| 33.2186       | 9.8513        |
| $\beta = 0.75$ | 6   | $63 \times 63$| 78.5830       | 10.6949       |
|        | 7   | $127 \times 127$| 185.1202     | 11.9194       |
|        | 8   | $255 \times 255$| 441.7010     | 12.7937       |
|        | 9   | $511 \times 511$| 1.0501e+03  | 13.5884       |
|        | 10  | $1023 \times 1023$| 2.4971e+03 | 14.3144       |

ular, since it retains the property of short recurrence, usually have a fast convergence speed compared with the other Krylov subspace methods; but the interruption may occurs. The algorithm of wavelet preconditioned Bi-CGSTAB method is given in Algorithm 2. For computing $A_n x$, we use the method of Toeplitz fast matrix-vector multiplications proposed in Algorithm 3, with the computational complexity

just as $O(n2^n)$. And for $S_nx$ and $S_n^Tx$, the fast wavelet transformation (FWT) or the sparsity of $S_n$ can be applied, which just has the computational complexity as $O(2^n)$.

Algorithm 2 Wavelet preconditioned Bi-CGSTAB

1: Given the initial value of iteration $y_0^n$, compute $r_0^n = \xi_n - A_ny_0^n$
2: Choose $\hat{r} = r_0^n$
3: for $k = 1, 2, \ldots, $ do
4: $p^{k-1} = \hat{r}^T r^{k-1}$
5: if $k = 1$ then
6: $\hat{p} = r^{k-1}$
7: else
8: $\beta^{k-1} = \left(\frac{\rho^{k-1}}{\rho^{k-2}}\right) \left(\frac{\alpha^{k-2}}{\omega^{k-1}}\right)$
9: $p = r^{k-1} + \beta^{k-1} (p^{k-1} - \omega^{k-1} v^{k-1})$
10: end if
11: $\hat{p} = S_n p$, $\hat{p} = S_n^T \hat{p}$
12: $y^{k} = A_n \hat{p}$
13: $\alpha^{k} = \frac{\rho^{k-1}}{\rho^{k-2}}$
14: $s = r^{k-1} - \alpha^{k} v^{k}$
15: if $||s|| \leq \varepsilon$ then
16: $y_n = y_n^{k-1} + \alpha^{k} \hat{p}$
17: Stop
18: end if
19: $\hat{s} = S_n s$, $\hat{s} = S_n^T \hat{s}$
20: $t = A_n \hat{s}$
21: $\omega^{k} = \frac{\omega^{k-1}}{\omega^{k-2}}$
22: $y_n^{k} = y_n^{k-1} + \alpha^{k} \hat{p} + \omega^{k} \hat{s}$
23: $r^{k} = s - \omega^{k} t$
24: if $||r^{k}|| \leq \varepsilon$ then
25: Stop
26: end if
27: end for

Algorithm 3 Calculating $A_n y_n$ by FFT

1: Given initial vectors $a_1, q_1$
2: Set: $c = [q_1, 0, a_1, \text{zeros}(1, 2^n - 2r + 1)]^T$, $t = y_n + \sqrt{-1} \hat{y}_n$
3: Do: $z = \mathcal{F} (\mathcal{F}(t) \circ \mathcal{F}(c))$
4: Set: $A_n y_n := a p \times \text{real}(z(1 : 2^n - r + 1)) + a q \times \text{imag}(z(2^n - r + 1 : -1 : 1))$

Note: $\mathcal{F}$, $\mathcal{F}$ denote the FFT and inverse FFT, respectively, $\hat{y}_n = y_n(\text{end} : -1 : 1)$, $\circ$ denotes the Hadamard product of vector $a$ and $b$.

Now using the provided algorithms and taking $p = 1$, $q = 0$, and $a = 1$, we solve (1.1) with $f(x) = f_1$
and $f_2$ given in (3.6) and (3.7), respectively, i.e.,
\[
\begin{cases}
-D_0D_\alpha D^\beta u = f_i, \\
u(0) = u(1) = 0,
\end{cases}
\]
with $i = 1, 2$. When letting the exact solution be $u = x^2 - x^3$, the forcing function is
\[
f_1 = \frac{-2x^\beta}{\Gamma(\beta + 1)} + \frac{6x^{\beta+1}}{\Gamma(\beta + 2)};
\]
and when the exact solution being taken as $u = x^\lambda - x$, we have the forcing function
\[
f_2 = \frac{-\Gamma(\lambda + 1)-x^\lambda+\beta-2}{\Gamma(\lambda + 1-\beta)} + \frac{x^{\beta-1}}{\Gamma(\beta)}.
\]

The Bi-CGSTAB and the wavelet preconditioned Bi-CGSTAB methods are respectively used to solve the above equations. The numerical results are listed in Tables 3 and 4 for $f(x) = f_1$ and $f_2$, respectively. In performing the numerical computations, the initial value of iteration is taken as zero, and the stopping criterion $\varepsilon = 10^{-7}$. In fact, for making the comparisons, the Gaussian elimination based on the Doolittle LU decomposition (GE) is also used to solve the corresponding equations.

### Table 3. Numerical performances of the Bi-CGSTAB method, the GE method, and the preconditioned Bi-CGSTAB method, respectively, with the forcing function $f(x) = f_1, \beta = 0.5$, and $r = 2$

| $n$ | Bi-CGSTAB | GE | Pre-Bi-CGSTAB | $L_2$ error |
|-----|------------|----|---------------|-------------|
|     | iter | cpu(s) |     | iter | cpu(s) | iter | cpu(s) |          |
| 5   | 33.5 | 0.0152 | 0.0073 | 15.5 | 0.0132 | 2.3973e-04 |
| 6   | 70.5 | 0.0328 | 0.0277 | 18.5 | 0.0180 | 6.0006e-05 |
| 7   | 142.5 | 0.0792 | 0.1264 | 20.5 | 0.0241 | 1.5021e-05 |
| 8   | 312.5 | 0.1664 | 0.5030 | 23.5 | 0.0334 | 3.7596e-06 |
| 9   | 783.5 | 0.5962 | 2.5780 | 26.5 | 0.0583 | 9.4116e-07 |
| 10  | 1933.5 | 2.0709 | 14.9119 | 27.0 | 0.1015 | 2.3875e-07 |

From Tables 3 and 4, it can be noted that both the Bi-CGSTAB method and the preconditioned Bi-CGSTAB method have a stable convergence rate 2 or 1.6 (due to the limited smoothness of the exact solution), but the preconditioned Bi-CGSTAB method is much faster than the Bi-CGSTAB method; the iteration numbers of the Bi-CGSTAB method increases quickly, but the ones of the preconditioned Bi-CGSTAB method tends to be uniformly bounded. It can also be noted that compared with the GE method the computational time can be greatly reduced, while they have almost the same $L_2$ error.

For the sake of completeness, we also show the condition numbers of $A_n$ and $B_n$ with the spline basis of order 3 in Tables 5 and 6.

### 3.2 Wavelet Galerkin method for two dimensional FEDE

We know that the wavelet bases constructed in Lemma 2.5 are the Riesz bases of $H^1_0((0,1)^2)$ with $0 < \mu < r - 1/2$; and they are applied to solve the following FEDE:

\[-D_x^\alpha a_1(p_{10}D_x^{-\alpha} + q_{11}D_1^{-\alpha})D_xu - D_y^\beta a_2(p_{20}D_y^{-\beta} + q_{22}D_1^{-\beta})D_yu = f, \quad x,y \in \Omega = [0,1] \times [0,1],\]
Table 4. Numerical performances of the Bi-CGSTAB method, the GE method, and the preconditioned Bi-CGSTAB, respectively, with the forcing term \( f(x) = f_2, \beta = 0.75, \lambda = 1.1, \) and \( r = 2 \)

| \( n \) | Bi-CGSTAB | GE | Pre-Bi-CGSTAB | \( L_2 \) error |
|---|---|---|---|---|
| | iter | cpu(s) | iter | cpu(s) | |
| 5 | 54.5 | 0.0335 | 0.0135 | 31 | 0.0230 | 1.0539e-04 |
| 6 | 134.5 | 0.0558 | 0.0466 | 47 | 0.0370 | 3.4800e-05 |
| 7 | 246.5 | 0.1256 | 0.2065 | 49 | 0.0466 | 1.1484e-05 |
| 8 | 516.5 | 0.2672 | 0.2845 | 61 | 0.0648 | 3.7889e-06 |
| 9 | 1095.5 | 0.8060 | 4.3384 | 65 | 0.1014 | 1.2499e-06 |
| 10 | 2471.5 | 2.5550 | 24.0359 | 69 | 0.1736 | 4.1252e-07 |

Table 5. The condition numbers of the matrices \( A_n \) and \( B_n \) with \( p = q = 0.5, \) and \( r = 3 \)

| \( \beta \) | \( n \) | size of \( B_n \) | \( \kappa(A_n) \) | \( \kappa(B_n) \) |
|---|---|---|---|---|
| \( \beta = 0.5 \) | 3 | 6 \( \times \) 6 | 4.6180 | 6.0093 |
| | 4 | 14 \( \times \) 14 | 14.0369 | 7.7079 |
| | 5 | 30 \( \times \) 30 | 41.3358 | 9.2886 |
| | 6 | 62 \( \times \) 62 | 119.1580 | 10.6787 |
| | 7 | 126 \( \times \) 126 | 340.1521 | 11.8809 |
| | 8 | 254 \( \times \) 254 | 966.4690 | 12.9127 |
| | 9 | 511 \( \times \) 511 | 2.7397e+03 | 13.7796 |
| \( \beta = 0.75 \) | 3 | 6 \( \times \) 6 | 3.4303 | 6.2506 |
| | 4 | 14 \( \times \) 14 | 8.8072 | 7.6850 |
| | 5 | 30 \( \times \) 30 | 21.7487 | 9.0615 |
| | 6 | 62 \( \times \) 62 | 52.6590 | 10.2663 |
| | 7 | 126 \( \times \) 126 | 126.3300 | 11.3041 |
| | 8 | 254 \( \times \) 254 | 301.7399 | 12.1916 |
| | 9 | 510 \( \times \) 510 | 719.1686 | 12.9483 |

where \( D_x = \frac{\partial u(x,y)}{\partial x} \) and \( D_y = \frac{\partial u(x,y)}{\partial y} \), and \( D_x^s \) or \( D_y^s \) denotes \( s \)-th derivative; \( 0D_x^{-\alpha} \) (or \( 0D_x^{-\beta} \)) and \( 1D_x^{-\alpha} \) (or \( 1D_x^{-\beta} \)) represent the left and right Riemann-Liouville fractional integral operators, respectively, with \( 0 < \alpha, \beta < 1 \) satisfying \( p_1 + q_1 = 1 \) and \( p_2 + q_2 = 1 \). When \( s = 2 \) or \( 3 \), the Riesz bases constructed in Lemma 2.5 can be applied to solve this equation.

The variational formulation of this fractional differential equation is given as follows

\[
B(u,v) = a_1 p_1 \left( D_x^{-1} 0D_x^{-\alpha} D_x u, D_x v \right) + a_1 q_1 \left( D_x^{-1} 1D_x^{-\alpha} D_x u, D_x v \right) + a_2 p_2 \left( D_x^{-1} 0D_y^{-\beta} D_y u, D_y v \right) + a_2 q_2 \left( D_x^{-1} 1D_y^{-\beta} D_y u, D_y v \right).
\]

Consequently, in order to solve the variational formulation in \( \Omega = (0, 1)^2 \), we use \( \hat{V}_n \) to approximate the \( H^\mu_0(\Omega) \) space, since \( \Phi_n := \{ \hat{f}_{n,(j_1,j_2)} : f_1 = 0, 1, \ldots, 2^n - r; j_2 = 0, 1, \ldots, 2^n - r \} \) is a basis of \( \hat{V}_n \). We
investigate a $u_k \in \tilde{V}_n$ such that
\begin{equation}
\begin{aligned}
a_1p_1 \langle D_x^{r-1} - \beta D_x \psi, D_x v \rangle &+ a_1q_1 \langle D_x^{r-1} - \beta D_x \sigma, D_x v \rangle + a_3q_3 \langle D_x^{r-1} - \beta D_x \psi, D_x v \rangle \\
a_2p_2 \langle D_x^{r-1} - \beta D_x \psi, D_x v \rangle &+ a_2q_2 \langle D_x^{r-1} - \beta D_x \sigma, D_x v \rangle + a_3q_3 \langle D_x^{r-1} - \beta D_x \psi, D_x v \rangle = \langle f, v \rangle, \quad \forall v \in \tilde{V}_n.
\end{aligned}
\end{equation}

Suppose $u_n = \sum_{\tilde{\phi} \in \tilde{\Phi}_n} \tilde{\phi} \tilde{\phi}$. Let $C_n^\psi$ be the matrix $(a_1p_1 \langle D_x^{r-1} - \beta D_x \psi, D_x \tilde{\phi} \rangle + a_1q_1 \langle D_x^{r-1} - \beta D_x \tilde{\phi}, D_x \psi \rangle + a_3q_3 \langle D_x^{r-1} - \beta D_x \psi, D_x \tilde{\phi} \rangle)_{\tilde{\phi} \in \tilde{\Phi}_n}$ and let $\tilde{\eta}_n$ be the column vector $(\langle f, \tilde{\phi} \rangle)_{\tilde{\phi} \in \tilde{\Phi}_n}$. Then the column vector $y_n = (\langle \tilde{y}, \tilde{\phi} \rangle)_{\tilde{\phi} \in \tilde{\Phi}_n}$ is the solution of the linear system
\begin{equation}
C_n^\psi y_n = \tilde{\eta}_n.
\end{equation}

Similar to the one dimensional case, without preconditioning it would be difficult to solve the system when we increase the discrete level of $n$.

Now we employ the wavelet bases constructed above to solve the variational problem. For $n \geq n_0$, $\bar{\Phi}_n := \{2^{-n_0k} \phi_{0,j} : j \in J_{n_0} \} \cup \bigcup_{k = k_0}^{n-1} \{2^{-k} \omega : \omega \in \Gamma_k \}$. To find a column vector $z_n = (z_{\bar{\psi}})_{\bar{\psi} \in \bar{\Psi}_n}$ such that
\begin{equation}
D^\ast_n z_n = \bar{\eta}_n,
\end{equation}
where $D^\ast_n$ is matrix $(a_1p_1 \langle D_x^{r-1} - \beta D_x \tilde{\chi}, D_x \bar{\phi} \rangle + a_1q_1 \langle D_x^{r-1} - \beta D_x \bar{\phi}, D_x \tilde{\chi} \rangle + a_3q_3 \langle D_x^{r-1} - \beta D_x \tilde{\chi}, D_x \bar{\phi} \rangle)_{\bar{\phi} \in \bar{\Phi}_n} + (a_2p_2 \langle D_x^{r-1} - \beta D_x \bar{\psi}, D_x \tilde{\chi} \rangle + a_2q_2 \langle D_x^{r-1} - \beta D_x \tilde{\chi}, D_x \bar{\psi} \rangle)_{\bar{\phi} \in \bar{\Phi}_n}$ and $\bar{\eta}_n$ denotes the column vector $(\langle f, \bar{\psi} \rangle)_{\bar{\psi} \in \bar{\Psi}_n}$. Then the approximate solution in $\tilde{V}_n$ can be written as $u_n = \sum_{\bar{\psi} \in \bar{\Psi}_n} z_{\bar{\psi}} \bar{\psi}$.

Since $\bar{\Phi}_n$ and $\bar{\Psi}_n$ are two different bases of $\tilde{V}_n$, there is a unique transformation $\bar{S}_n$ between the two bases. Consequently, we have
\begin{equation}
D^\ast_n = \bar{S}_n C_n^\psi \bar{S}_n^\ast, \quad \bar{\eta}_n = \bar{S}_n \bar{\eta}_n,
\end{equation}
and (3.9) is equivalent to (3.10). If we set $y_n = \bar{S}_n z_n$, then (3.8) is preconditioned by the matrix $\bar{S}_n$. 

---

**Table 6. The condition numbers of the matrices $A_n$ and $B_n$ with $p = 1$, $q = 0$, and $r = 3$**

| $\beta$ | $n$   | size of $B_n$ | $\kappa(A_n)$ | $\kappa(B_n)$ |
|---------|-------|--------------|----------------|----------------|
| $\beta = 0.5$ | 3     | $6 \times 6$ | 4.9508         | 6.0966         |
|         | 4     | $14 \times 14$ | 15.0241        | 8.0872         |
|         | 5     | $30 \times 30$ | 43.9033        | 9.8905         |
|         | 6     | $62 \times 62$ | 126.2291       | 11.5143        |
|         | 7     | $126 \times 126$ | 359.9874       | 12.9501        |
|         | 8     | $254 \times 254$ | 1.0224e+003  | 14.2106        |
|         | 9     | $511 \times 511$ | 2.8980e+003  | 15.2991        |
| $\beta = 0.75$ | 3     | $6 \times 6$ | 4.4583         | 6.9203         |
|         | 4     | $14 \times 14$ | 11.3404        | 9.8872         |
|         | 5     | $30 \times 30$ | 26.7126        | 12.0692        |
|         | 6     | $62 \times 62$ | 63.2405        | 13.9770        |
|         | 7     | $126 \times 126$ | 150.4342       | 15.7375        |
|         | 8     | $254 \times 254$ | 358.1752       | 17.3488        |
|         | 9     | $510 \times 510$ | 852.6666       | 18.8178        |
Next, we provide the algorithm of generating $D_n^s$, where $s = 2$ or $s = 3$. For generating the matrix $D_n^s$, we need the transform matrix $S_n$, which is far more complex than one dimensional case; the one-level transform is given in Algorithm 4, and the others are like the steps proposed in Algorithm 1, which are omitted here.

### Algorithm 4 Generating the matrix $\hat{P}_k$ for 2D

1. Let $P_k$ be one-level transform matrix in 1D, satisfying: $(\Phi_{k-1}^T \Phi_{k-1}) = P_k \Phi_k$

2. Initialize the matrixes:
   \[
   t_1 = 2^{k-1} - r_1 + 1, \quad t_2 = 2^k - r_1 + 1
   \]
   \[
   L1 = a_1 \otimes P_k(1 : t_1, 1 : t_2)
   \]
   \[
   L2 = a_1 \otimes P_k(t_1 + 1 : t_2, 1 : t_2)
   \]
   \[
   L3 = a_2 \otimes P_k(1 : t_1, 1 : t_2)
   \]
   \[
   L4 = a_3 \otimes P_k(1 : t_1, 1 : t_2)
   \]
   \[
   L5 = a_2 \otimes P_k(t_1 + 1 : t_2, 1 : t_2)
   \]
   \[
   L6 = a_2 \otimes P_k(t_1 + 1 : t_2, 1 : t_2)
   \]
   \[
   L7 = a_3 \otimes P_k(t_1 + 1 : t_2, 1 : t_2)
   \]
   \[
   L8 = a_2 \otimes P_k(t_1 + 1 : t_2, 1 : t_2)
   \]

3. for $i = 1 : 2^{k-1} - r_1 + 1$

4. $A = [T_b A; T_f^{(i-1)} L_1]$, $C = [T_b C; T_f^{(i-1)} L_2]$

5. end for

6. $L_3 = \{L_3, \text{zeros}(2^{k-1} - r_1 + 1, (2^k - r_1 + 1)^2 - \text{len}_c(L_3))\}$

7. $L_6 = \{L_6, \text{zeros}(2^{k-1} - 1, (2^k - r_1 + 1)^2 - \text{len}_c(L_6))\}$

8. for $i = 1 : 2^{k-1} - 2$

9. $B = [T_b B, T_f^{(i-1)} L_4]$, $D = [T_b D, T_f^{(i-1)} L_7]$

10. end for

11. $L_5 = \text{zeros}(2^{k-1} - r_1 + 1, (2^k - r_1 + 1)^2 - \text{len}_c(L_5)) \cdot L_5$

12. $L_8 = \text{zeros}(2^{k-1} - 1, (2^k - r_1 + 1)^2 - \text{len}_c(L_8)) \cdot L_8$

13. $\hat{P}_k(1 : (2^{k-1} - r_1 + 1)^2, 1 : (2^k - r_1 + 1)^2) \cdot A$

14. $\hat{P}_k((2^{k-1} - r_1 + 1)(2^k - r_1 + 1), 1 : (2^k - r_1 + 1)^2) \cdot C$

15. $\hat{P}_k((2^{k-1} - r_1 + 1)^2, 1 : (2^k - r_1 + 1)^2, 1 : (2^k - r_1 + 1)^2) \cdot L_5 \cdot B \cdot L_5$

16. $\hat{P}_k((2^{k-1} - r_1 + 1)(3 \times 2^{k-1} - r_1 + 1), 1 : (2^k - r_1 + 1)^2, 1 : (2^k - r_1 + 1)^2) \cdot [L_6 \cdot D \cdot L_8$

Note: $a_1, a_2, a_3$ denote the refinement coefficient vectors.

For example, when $r = 3$, we have:

- $a_1 = \left[\frac{1}{4}, \frac{3}{4}, \frac{1}{4}\right]$, $a_2 = \left[\frac{5}{12}, -\frac{1}{12}\right]$,
- $a_3 = \left[\frac{5}{12}, -\frac{1}{12}, \frac{5}{12}, -\frac{1}{12}\right]$,
- $a_2 = a_2(\text{end} : -1 : 1)$.

$a \otimes b$ denotes the kronecker product of vector $a$ and $b$; and $\text{len}_c(L)$ denotes the column number of matrix $L$.

For $[T_b G; T_f^{(i-1)} L]$, $T_b$ denotes the zero padding operator for making the column of $T_b G$ equal to $T_f^{(i-1)} L$; for simplicity, $A, B, C, D$ are initialized with empty matrix. $T_f$ is the operator to extend the matrix $L$ by adding new columns with the value of zero at the left hand side of the matrix. More precisely, for $L = L_1$ and $L_4$, $T_f L$ adds $2^k - 2r + 2$ column zeros before $L$, but for $L = L_2$ and $L_7$, it adds $2^k$ column zeros before $L$. 
Taking \( u = x^2(1-x)^2y^2(1-y)^2 \), \( \alpha = \beta = 3/4 \), \( p_1 = p_2 = 1 \), \( q_1 = q_2 = 0 \), \( \alpha_1 = \alpha_2 = 0 \), then \( u \) is the exact solution of the following equation

\[
-D_n^3 Du - D_n^3 Du = f_s
\]

with the boundary conditions \( u(x,y)|_{\partial \Omega} = 0, (\partial u(x,y) / \partial x)|_{x=0,y \in [0,1]} = 0, \) and \((\partial u(x,y) / \partial y)|_{x \in [0,1], y=0} = 0 \) for \( s = 2 \), and \( u(x,y)|_{\partial \Omega} = 0, (\partial u(x,y) / \partial x)|_{x=0} \) and \( (\partial u(x,y) / \partial y)|_{x=1,y \in [0,1]} = 0, \) and \((\partial u(x,y) / \partial y)|_{x \in [0,1], y=0} \) and \( y=1 = 0 \) for \( s = 3 \), respectively; and the forcing function \( f_s \) is given as follows

\[
f_s = \left( \frac{-24}{\Gamma(\frac{19}{4} - s)} x^{\frac{13}{4} - s} + \frac{12}{\Gamma(\frac{15}{4} - s)} x^{\frac{11}{4} - s} + \frac{-2}{\Gamma(\frac{11}{4} - s)} y^{\frac{7}{4} - s} \right) y^2(1-y^2)
\]

\[
+ \left( \frac{-24}{\Gamma(\frac{19}{4} - s)} y^{\frac{13}{4} - s} + \frac{12}{\Gamma(\frac{15}{4} - s)} y^{\frac{11}{4} - s} + \frac{-2}{\Gamma(\frac{11}{4} - s)} x^{\frac{7}{4} - s} \right) x^2(1-x^2).
\]

We first calculate the condition numbers of the corresponding stiffness matrices, then use the algorithms presented in previous sections to compute the numerical solutions for different \( s \) with \( r = 3 \). For confirming the relation between the conditional numbers and \( \beta \), we also list the condition numbers of the matrix with \( \beta = 0.25 \).

| Table 7. When \( r = 3, \alpha = \beta = 0.75 \), the condition number of the matrix \( C_n^s \) and \( D_n^s \) |
|-------------|------------|------------------|-----------------|------------------|-----------------|
| \( s \) | \( n \) | size of \( D_n^s \) | \( \kappa(C_n^s) \) | ratio | \( \kappa(D_n^s) \) | ratio |
|-------------|------------|------------------|-----------------|-----------------|-----------------|
| 4          | 4          | 196 x 196        | 29.4463         | 54.8262         | 0.1206          |
| 5          | 5          | 900 x 900        | 144.4390        | 59.8263         | 0.0954          |
| 6          | 6          | 3844 x 3844      | 698.5068        | 63.6789         | 0.1077          |
|-------------|------------|------------------|-----------------|-----------------|-----------------|

| Table 8. When \( r = 3, \alpha = \beta = 0.25 \), the condition number of the matrix \( C_n^s \) and \( D_n^s \) |
|-------------|------------|------------------|-----------------|-----------------|-----------------|
| \( s \) | \( n \) | size of \( D_n^s \) | \( \kappa(C_n^s) \) | ratio | \( \kappa(D_n^s) \) | ratio |
|-------------|------------|------------------|-----------------|-----------------|-----------------|
| 4          | 4          | 196 x 196        | 7.4192e+01      | 69.8276         | 0.1206          |
| 5          | 5          | 900 x 900        | 5.1254e+02      | 89.9318         | 0.3650          |
| 6          | 6          | 3844 x 3844      | 3.4862e+03      | 94.9550         | 0.0784          |
|-------------|------------|------------------|-----------------|-----------------|-----------------|
| 4          | 4          | 196 x 196        | 5.8919e+02      | 159.0936        | 0.1468          |
| 5          | 5          | 900 x 900        | 8.0236e+03      | 176.1326        | 0.0662          |
| 6          | 6          | 3844 x 3844      | 1.0824e+05      | 184.4045        | 0.0662          |

From Tables 7 and 8, it can be noted that the condition numbers of the stiffness matrix corresponding to the ordinary Galerkin methods increase with a rate as \( O(h^{-(s+1-\beta)}) \), but the conditional numbers corresponding to the wavelet Galerkin methods tend to be uniformly bounded, as stated in Lemma 2.5. For the numerical iterative schemes, because of the tensor form of the matrices \( C_n^s \), we can still
make use of the Toeplitz structure of the matrix and FWT such that the computation complexity is as $O(N \log N)$, where $N$ denotes the number of bases. The numerical performances for $\beta = 0.75$ are presented in Table 9, 10, 11, and 12. It can be seen that for getting the same accuracy, compared with the Bi-CGSTAB method, the preconditioned Bi-CGSTAB method needs less computation time and the number of iterations when $s = 2$ and 3; and in fact when $s = 3$, the numerical errors for Bi-CGSTAB method increase early. For the GMRES method and the preconditioned GMRES method, they have almost the same $L^2$ errors, but the latter method converges much faster. From Tables 11 and 12, it can be noted that GMRES(50) is faster than GMRES(20), but preconditioned GMRES(20) is faster than preconditioned GMRES(50).

Table 9. Numerical performances for the Bi-CGSTAB method and the preconditioned Bi-CGSTAB method with $\alpha = \beta = 0.75$, and $r = 3$

| $s$ | $n$ | Bi-CGSTAB | Pre-Bi-CGSTAB |
|-----|-----|-----------|---------------|
|     |     | iter     | cpu(s) | $L^2$ error | iter     | cpu(s) | $L^2$ error |
| 5   | 109 | 43.0     | 0.2046 | 6.1592e-07 | 41.5     | 0.2315 | 6.1592e-07 |
| 6   | 277 | 42.0     | 0.2535 | 8.1592e-07 | 41.5     | 0.2315 | 6.1592e-07 |
| 7   | 347 | 42.0     | 0.2535 | 8.1592e-07 | 41.5     | 0.2315 | 6.1592e-07 |
| 5   | 135 | 109.5    | 1.3395 | 8.8556e-08 | 46.5     | 0.8530 | 8.8556e-08 |
| 6   | 309 | 109.5    | 1.3395 | 8.8556e-08 | 46.5     | 0.8530 | 8.8556e-08 |
| 7   | 347 | 109.5    | 1.3395 | 8.8556e-08 | 46.5     | 0.8530 | 8.8556e-08 |
| 5   | 347 | 277.5    | 8.3340 | 1.3326e-08 | 55.5     | 3.0553 | 1.3326e-08 |
| 6   | 996 | 277.5    | 8.3340 | 1.3326e-08 | 55.5     | 3.0553 | 1.3326e-08 |
| 7   | 2569| 277.5    | 8.3340 | 1.3326e-08 | 55.5     | 3.0553 | 1.3326e-08 |

Table 10. Numerical performances for the non-restarted GMRES method and the preconditioned GMRES method with $\alpha = \beta = 0.75$, and $r = 3$

| $s$ | $n$ | GMRES | Pre-GMRES | $L^2$ error |
|-----|-----|-------|-----------|-------------|
|     |     | iter  | cpu(s) | iter | cpu(s) | $L^2$ error |
| 5   | 109 | 55    | 0.3359 | 55   | 1.3109 | 6.1592e-07 |
| 6   | 309 | 55    | 0.3359 | 55   | 1.3109 | 6.1592e-07 |
| 7   | 309 | 55    | 0.3359 | 55   | 1.3109 | 6.1592e-07 |
| 5   | 135 | 61    | 3.3185 | 61   | 1.1250 | 8.8556e-08 |
| 6   | 309 | 61    | 3.3185 | 61   | 1.1250 | 8.8556e-08 |
| 7   | 309 | 61    | 3.3185 | 61   | 1.1250 | 8.8556e-08 |
| 5   | 231 | 185   | 3.0238 | 185  | 2.0874 | 5.8094e-06 |
| 6   | 721 | 260   | 70.1586| 260  | 7.8870 | 1.4475e-06 |
| 7   | 2670| 313   | 3299.2131| 313  | 53.7952 | 3.6123e-07 |

4. Conclusion

For improving the efficiency of solving FEDEs, three natural ways can be adopted: 1. reducing matrix vector multiplication from $O(N^2)$ to $O(N \log N)$; 2. keeping the condition numbers small and uniformly bounded; 3. increasing the convergence orders. For the general linear finite element methods, because of the potential Toeplitz structure of the stiffness matrix, the cost of the matrix vector multiplication can be kept as $O(N \log N)$. But for the high order elements, the potential Toeplitz structure is destroyed and the cost of matrix vector multiplication is $O(N^2)$. If taking the scale functions (generally used to generate wavelets) as the base functions of the Galerkin methods, the potential Toeplitz structure of the
Table 11. Numerical performances for the restarted GMRES(50) method and the preconditioned GMRES(50) method with $\alpha = \beta = 0.75$, and $r = 3$

| $s$ | $n$ | GMRES(50) | Pre-GMRES(50) | $L_2$ error |
|-----|-----|-----------|--------------|-------------|
|     |     | iter      | cpu(s)       | iter        | cpu(s)      | $L_2$ error  |
| 5   | 5 x 50+15 | 0.2746    | 1 x 50+7    | 0.2838      | 6.1592e-07 |
| 6   | 6 x 50+5  | 3.7541    | 1 x 50+14   | 0.9887      | 8.8556e-08 |
| 7   | 14 x 50+35| 24.8405   | 1 x 50+16   | 3.3791      | 1.3343e-08 |

Table 12. Numerical performance with the restarted GMRES(20) method and the preconditioned GMRES(20) method, respectively, $\alpha = \beta = 0.75$, $r = 3$.

| $s$ | $n$ | GMRES(20) | Pre-GMRES(20) | $L_2$ error |
|-----|-----|-----------|--------------|-------------|
|     |     | iter      | cpu(s)       | iter        | cpu(s)      | $L_2$ error  |
| 5   | 6 x 20+14 | 0.4269    | 13 x 20+14  | 0.2193      | 6.1592e-07 |
| 6   | 17 x 20+8 | 3.0920    | 17 x 20+6   | 0.7959      | 8.8556e-08 |
| 7   | 81 x 20+12| 40.0962   | 20 x 20+14  | 2.5415      | 1.3343e-08 |

stiffness matrix can be kept. Furthermore, based on the general wavelet theory, the Riesz bases of the space that the FEDE works are found and effectively used to solve the one and two dimensional FEDEs. The detailed algorithm descriptions are presented. The extensive numerical experiments are performed, and the numerical observables, including the condition numbers, iteration numbers, cpu time cost, are calculated; all demonstrate the striking benefits of the wavelet Galerkin methods in solving FEDEs.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant No. 11271173.

References

BRAESS, D. (1995) Towards algebraic multigrid for elliptic problems of second order. Computing, 55, 379–393.

BRAMBLE, J. H., PASCIAK, J. E. & XU, J. (1990) Parallel multilevel preconditioners. Math. Comp., 55, 1–22.

BRENNER, S. & SCOTT, L. R. (1994) The mathematical theory of finite element methods. Springer-Verlag.
REFERENCES

CHEN, M. H., WANG, Y. T., CHENG, X. & DENG, W. H. (2014) Second-order LOD multigrid method for multidimensional Riesz fractional diffusion equation. *BIT Numer. Math.*, doi:10.1007/s10543-014-0477-1.

CHRISTON, M. A. & ROACH D. W. (2000) The numerical performance of wavelets for PDEs: the multi-scale finite element. *Comput. Mech.*, 25, 230–244.

CHUI, C. K. & WANG, J. Z. (1992) On compactly supported spline wavelets and a duality principle. *Trans. Amer. Math. Soc.*, 330, 903–916.

DENG, W. H. & CHEN, M. H. (2014) Efficient numerical algorithms for three-dimensional fractional partial differential equations. *J. Comput. Math.*, doi:10.4208/jcm.1401-m3893.

DENG, W. H. & HESTHAVEN, J. (2013) Local discontinuous Galerkin methods for fractional differential equations. *ESAIM:M2AN*, 47, 1845–1864.

DENG, W. H. (2008) Finite element method for the space and time fractional Fokker-Planck equation. *SIAM J. Numer. Anal.*, 47, 204–226.

DIETHEL, K. (2010) *The Analysis of Fractional Differential Equations*. Springer-Verlag Berlin Heidelberg.

ERVIN, V. J. & ROOP, J. P. (2006) Variational formulation for the stationary fractional advection dispersion equation. *Numer. Methods Partial Differential Equations.*, 22, 558–576.

HAN, B. & SHEN, Z. (2006) Wavelets with short support. *SIAM J. Math. Anal.*, 38, 530–556.

JIA, R. Q. & ZHAO, W. (2011) Riesz bases of wavelets and applications to numerical solutions of elliptic equations. *Math. Comp.*, 25, 1525–1556.

JIA, R. Q., WANG, J. Z. & ZHOU, D. X. (2011) Compactly supported wavelet bases for Sobolev spaces. *Appl. Comput. Harmon. Anal.*, 15, 224–241.

JIA, R. Q. (2010) Approximation by quasi-projection operators in Besov sequences. *J. Approx. Theory.*, 162, 186–200.

JIA, R. Q. (2009) Spline wavelet on the interval with homogeneous boundary conditions. *Adv. Comput. Math.*, 30, 177–200.

JIA, R. Q. (2006) Bessel sequences in Sobolev spaces. *Appl. comput. Harmon. Anal.*, 20, 298–311.

JIA, R. Q. & LIU, S. T. (2006) Wavelet bases of Hermite cubic splines on the interval. *Adv. Comput. Math.*, 25, 23–39.

JIA, R. Q. (2004) Approximation with scaled shift-invariant spaces by means of quasi-projection operators. *J. Approx. Theory.*, 131, 30–46.

LI, X. J. & XU, C. J. (2010) Existence and uniqueness of the weak solution of the space-time fractional diffusion equation and a spectral method approximation. *Commun. Comput. Phys.*, 8, 1016–1051.

MEERSCHAERT, M. M. & TADJERAN, C. (2004) Finite difference approximations for fractional advection-dispersion flow equations. *J. Comput. Appl. Math.*, 172, 65–77.
PANG, H. K. & SUN, H. W. (2012) Multigrid method for fractional diffusion equations. *J. Comput. Phys.*, **231**, 693–703.

PODLUBNY, I. (1999) *Fractional Differential Equations*. New York: Academic Press.

QUARTERONI, A., SACCO, R. & SALERI, F. (2007) *Numerical Mathematics, 2nd ed.* Springer.

SOUSA, E. & LI, C. (2011) A weighted finite difference method for the fractional diffusion equation based on the Riemann-Liouville derivative. [arXiv:1109.2345v1 [math.NA]].

TIAN, W. Y., ZHOU, H. & DENG, W. H. (2014) A class of second order difference approximations for solving space fractional diffusion Equations. *Math. Comp.*, in press. [arXiv:1201.5949 [math.NA]].

WANG, H. & YANG, D. P. (2013) Wellposedness of variable-coefficient conservative fractional elliptic differential equations. *SIAM J. Numer. Anal.*, **51**, 1088–1107.

WANG, H. & BASU, T. S. (2012) A fast finite difference method for two-dimensional space-fractional diffusion equations. *SIAM J. Sci. Comput.*, **34**, A2444–A2458.

YUSTE, S. B. (2006) Weighted average finite difference methods for fractional diffusion euqations. *J. Comput. Phys.*, **216**, 264–274.

ZAYERNOURI, M. & KARNiadakis, G. E. (2013) Fractional Sturm-Liouville eigen-problems: Theory and numerical approximation. *J. Comput. Phys.*, **252**, 495–517.

ZHUANG, P., LIU, F., ANH, V. & TURNER, I. (2009) Numerical methods for the variable-order fractional advection-diffusion equation with a nonlinear source term. *SIAM J. Numer. Anal.*, **47**, 1760–1781.