East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.010122.210722

Vol. **13**, No. 1, pp. 95-118
February 2023

Neural Network Method for Integral Fractional Laplace Equations

Zhaopeng Hao*, Moongyu Park and Zhiqiang Cai

Department of Mathematics, Purdue University, West Lafayette, In 47906, USA.

Received 1 January 2022; Accepted (in revised version) 21 July 2022.

Abstract. A neural network method for fractional order diffusion equations with integral fractional Laplacian is studied. We employ the Ritz formulation for the corresponding fractional equation and then derive an approximate solution of an optimization problem in the function class of neural network sets. Connecting the neural network sets with weighted Sobolev spaces, we prove the convergence and establish error estimates of the neural network method in the energy norm. To verify the theoretical results, we carry out numerical experiments and report their outcome.

AMS subject classifications: 35B65, 65N35, 65N12, 41A25, 26B40
Key words: Deep Ritz method, neural network, fractional elliptic PDE, ReLU.

1. Introduction

Fractional Laplacian operator [8, 14, 34] has been employed in various nonlocal models, including turbulence [6, 21, 22], quantum mechanics [26, 33], finances [12], statistical physics [23], phase transitions [4, 5], material sciences [7], image processing [24], geophysics [13], acoustic wave propagation in heterogeneous media [45] and anomalous diffusion in porous media [35, 36]. Due to versatile applications and the ability to capture anomalous diffusion and model complex physical phenomena with long range interaction [16,39,42], many numerical methods have been developed — e.g. finite difference-quadrature methods [18,19,30,32,37], finite element methods (FEM) [1,3,15,41], spectral methods [28,29], mesh-free pseudospectral methods [10,40], the isogeometric collocation method [43], and deep learning method [38]. We refer the readers to [8, 14, 34] for a review of many definitions of fractional Laplacian and their numerical methods.

In this paper, we consider an n-dimensional fractional diffusion equation with Dirichlet boundary condition — viz.

$$(-\Delta)^{\frac{\alpha}{2}} u = f(x), \quad x \in \Omega, \quad \alpha \in (0,2), \tag{1.1}$$

$$u(x) = 0, \qquad x \in \Omega^c, \tag{1.2}$$

*Corresponding author. *Email addresses:* hao27@purdue.edu (Z. Hao), park633@purdue.edu (M. Park), caiz@purdue.edu (Z. Cai)

where Ω and Ω^c are a domain and its complement in $\mathbb{R}^n$, $n=1,2,3$, $f(x)$ is a given function, and the fractional Laplacian is defined by

$$(-\Delta)^{\frac{\alpha}{2}}u(x)=c_{n,\alpha}\int_{\mathbb{R}^n}\frac{u(x)-u(y)}{|x-y|^{n+\alpha}}\,dy,\quad c_{n,\alpha}=\frac{2^\alpha\Gamma((\alpha+n)/2)}{\pi^{\frac{n}{2}}|\Gamma(-\alpha/2)|}. \tag{1.3}$$

The essential difficulties for the fractional diffusion equations are twofold:

(1) The most of the numerical methods mentioned exhibit a low convergence order and accuracy because the solution has a singularity near the boundary inherited from the kernel.

(2) For $\alpha\in(1,2)$, the complexity caused by the singular integral makes the corresponding discretization challenging.

For special domains such as a disk in two-dimensional space or a ball in three-dimensional space, one can construct very accurate high-order methods based on pseudo-eigenfunctions of the fractional Laplacian operator [28]. For general domains any other higher order methods are not known. The FEM proposed in [1] on a graded mesh can capture the singularity near the boundary and recover the optimal convergence order for the piecewise linear polynomials. Nonetheless, the assembling the stiffness matrix is expensive, and it is still under investigation for high-order polynomials. Therefore, here we turn our attention to the current state-of-the-art neural network methods (NNMs). Neural network methods have gained increased attention recently in the science and engineering [20, 44]. The methods are powerful in approximation and have huge expressive power. Although many numerical methods use meshes, which are often constructed in prior or posterior in an adaptive way, they can be understood and classified into mesh-free methods. This make it easy to capture boundary or corner singularities and recover boundary and transition layers and the shocks encountered in hyperbolic problems. Their success for the integer-order problems [20, 44] naturally motives us to apply the methods to the model problem (1.1)-(1.2).

Compared to extensive research on integer-order local problems, there are limited research on neural networks for fractional counterparts incorporating nonlocalities. Pang *et al.* [38] applied the so-called FPINN method to a advection-diffusion equation with the fractional Laplacian. They showed numerical convergence of the method without theoretical analysis. Combining with the Monte Carlo numerical integration, Guo *et al.* [27] extended the FPINN method to high-dimensional forward and inverse problems. We remark that although their least-squares formulation works well for $\alpha\in(0,1)$, it experiences problems if $\alpha\in(1,2)$. Since the solution does not have enough regularity to use the least-squares form, numerical solution for $\alpha\in(1,2)$ was not reported in [38]. To resolve this issue we use the energy formulation of the model problem (1.1)-(1.2). Note that it is called the deep Ritz method in [20].

The main goal of this paper is the development of a more accurate neural network method for the model problem. The neural network method is set up for the energy form

of the model problem and transforms strong singularity in the energy form to a weaker singularity. We provide error estimates and the convergence order of the method. Numerical tests support our theoretical results. Comparing our work with [38], we note the following differences:

1. We study the energy formulation while [38] considers the least-squares formulation.
2. This work focuses on the great potential and the advantage of accuracy in terms of the total number of parameters for neural networks methods applying them to forward problems, while [38] puts emphasis on inverse problems.
3. The architecture of the neural networks here is different from that in [38].

We believe that the study in the current work provides a new insight into the application of neural networks methods to fractional differential equations. For high-dimension problems, we refer the reader to [25], where the deep Ritz method is considered for spectral fractional Laplacian equations using the Caffarelli-Silvestre extension. It should be noted that (1.3) is a different definition of fractional Laplacian.

The rest of the paper is organized as follows. In Section 2, we introduce mathematical settings used in this work. In Section 3, we present the neural network method for the model equation analyze its convergence and the errors. Section 4 is devoted to numerical quadratures used in the discretization of the Ritz formulation and the implementation of the method. Note that the derivation of the numerical quadrature is given in Appendix A. We also carry out numerical experiments and report the corresponding numerical results. Section 5 presents our conclusions and a discussion.

2. Preliminary

2.1. Function spaces

We use the standard notations for Sobolev spaces [2]. For a non-negative integer m let

$$W^{m,p}(\Omega) := \left\{ v \in L^p(\Omega) \,|\, D^\beta v \in L^p(\Omega),\ |\beta| \le m \right\},$$

where

$$\|v\|_{W^{m,p}(\Omega)} = \left(\sum_{k=0}^{m} \sum_{|\beta|=k} \|D^\beta v\|^p_{L^p(\Omega)} \right)^{\frac{1}{p}}, \quad 1 \le p < \infty.$$

If $p = 2$, we let $H^m(\Omega) = W^{m,2}(\Omega)$. According to [2], the fractional order Sobolev space $H^s(\Omega)$, $0 < s < 1$ is defined by

$$H^s(\Omega) := \left\{ v \in L^2(\Omega) : \|v\|_{H^s(\Omega)} < \infty \right\},$$

where

$$\|v\|_{H^s(\Omega)} = \left(\|v\|^2_{L^2(\Omega)} + |v|^2_{H^s(\Omega)} \right)^{\frac{1}{2}}, \tag{2.1}$$

$$|v|_{H^s(\Omega)} = \left(\iint_{\Omega\times\Omega} \frac{(v(x)-v(y))^2}{|x-y|^{n+2s}} dxdy) \right)^{\frac{1}{2}}.$$

The zero trace space $H_0^s(\Omega)$ can be defined as the closure of $C_c^\infty(\Omega)$ in the norm (2.1). It is well known that for $0 < s \leq 1/2$, the identity $H_0^s(\Omega) = H^s(\Omega)$ holds. For $s > 1$, we write $s = m + \sigma$, where m is the integer part of s and $\sigma \in [0, 1)$. Then

$$H^s(\Omega) := \left\{ v \in H^m(\Omega) : |D^\beta v| \in H^\sigma \text{ for any } \beta \text{ such that } \quad |\beta| = m \right\} \tag{2.2}$$

with the norm

$$\|v\|_{H^s(\Omega)} = \left(\|v\|^2_{H^m(\Omega)} + \sum_{|\beta|=m} |D^\beta v|^2_{H^\sigma(\Omega)} \right)^{\frac{1}{2}}. \tag{2.3}$$

In this work, we are mainly interested in the case $s < 2$.

Next, we introduce weighted Sobolev spaces which is a tool for taking care of singular solutions. Let $\delta(x) = \text{dist}(x, \partial\Omega)$ be the distance function whose value is the distance to the boundary of Ω. If $s \in (m, m+1)$ then following [1], we also consider the weighted fractional Sobolev space equipped with the norm

$$\|v\|_{H_r^s(\Omega)} = \left(\|v\|^2_{H^m(\Omega)} + \sum_{|\beta|=m} \iint_{\Omega\times\Omega} \frac{|D^\beta v(x) - D^\beta v(y)|^2}{|x-y|^{n+2\sigma}} \delta^{2r}(x,y)\, dydx \right)^{\frac{1}{2}}, \tag{2.4}$$

where $r \geq 0$, $\sigma = s - m$ and $\delta(x, y) = \min\{\delta(x), \delta(y)\}$.

Remark 2.1. It is evident that for $v \in H^s(\Omega)$, the following inequality holds:

$$\|v\|_{H_r^s(\Omega)} \leq C\|v\|_{H^s(\Omega)}$$

with a constant C. However, the norms (2.3) and (2.4) are not equivalent.

2.2. Well-posedness

We also consider the solution space

$$\mathbb{V} = \left\{ v \in H^{\frac{\alpha}{2}}(\mathbb{R}^n) : v = 0 \text{ in } \Omega^c \right\}$$

equipped with the $H^{\alpha/2}(\mathbb{R}^n)$ norm. For $v \in \mathbb{V}$, the fractional Laplacian operator is closely related to the with the regional the regional fractional Laplacian operator

$$(-\Delta)^{\frac{\alpha}{2}}_\Omega v(x) = c_{n,\alpha} \int_\Omega \frac{v(x) - v(y)}{|x-y|^{n+\alpha}}\, dy.$$

More exactly,

$$(-\Delta)^{\frac{\alpha}{2}} v = (-\Delta)^{\frac{\alpha}{2}}_\Omega v + \rho(x)v(x),$$

where

$$\rho(x) = C_{n,\alpha} \int_{\Omega^c} \frac{1}{|x-y|^{n+\alpha}}\, dy \approx \delta^{-\alpha}(x), \tag{2.5}$$

and $c_{n,\alpha}$ is the constant defined in (1.3). If u and v vanish outside of Ω, the symmetry of the power kernel function $|x-y|^{-n-\alpha}$ yields

$$\begin{aligned}\left((-\Delta)^{\frac{\alpha}{2}}_{\Omega} u, v\right) &= c_{n,\alpha} \iint_{\Omega\times\Omega} \frac{(u(x)-u(y))v(x)}{|x-y|^{n+\alpha}}\, dydx \\ &= c_{n,\alpha} \iint_{\Omega\times\Omega} \frac{(u(y)-u(x))v(y)}{|x-y|^{n+\alpha}}\, dxdy \\ &= \frac{1}{2} c_{n,\alpha} \iint_{\Omega\times\Omega} \frac{(v(x)-v(y))(u(x)-u(y))}{|x-y|^{n+\alpha}}\, dydx.\end{aligned}$$

The variational formulation of the problem (1.1)-(1.2) is: Find $u \in \mathbb{V}$ such that

$$a(u,v) = (f,v) \quad \text{for all} \quad v \in \mathbb{V}, \tag{2.6}$$

where

$$\begin{aligned}a(u,v) = \frac{c_{n,\alpha}}{2} &\iint_{\Omega\times\Omega} \frac{(u(x)-u(y))(v(x)-v(y))}{|x-y|^{n+\alpha}}\, dydx \\ &+ c_{n,\alpha} \int_{\Omega} \rho(x)u(x)v(x)dx.\end{aligned} \tag{2.7}$$

The bilinear form $a(\cdot,\cdot)$ is coercive and continuous [1]. Therefore, we can define the energy norm

$$\|v\|_{\mathbb{V}} := a(v,v)^{\frac{1}{2}}.$$

The well posedness of the variational form is the direct consequence of the Lax-Milgram theorem — i.e. if f belongs to the dual space $\mathbb{V}^*$ of $\mathbb{V}$, then there exists a unique solution $u \in \mathbb{V}$ of the problem (2.6).

The equivalent Ritz formulation is to find $u \in \mathbb{V}$ such that

$$J(u) = \min_{v\in\mathbb{V}} J(v), \tag{2.8}$$

where

$$J(v) = \frac{1}{2} a(v,v) - (f,v). \tag{2.9}$$

Remark 2.2. The discussion throughout the paper can be readily extended to the case of non-homogeneous boundary conditions by standard boundary homogenization techniques. Let $u(x) = g(x)$, $x \in \Omega^c$ be given. Considering the zero extension

$$\tilde{g}(x) = \begin{cases} 0, & x \in \Omega, \\ g(x), & x \in \Omega^c \end{cases}$$

of g into interior domain Ω and setting $\tilde{u} := u - \tilde{g}(x)$, we obtain

$$(-\Delta)^{\frac{\alpha}{2}} u = (-\Delta)^{\frac{\alpha}{2}} \tilde{u} + (-\Delta)^{\frac{\alpha}{2}} \tilde{g} = (-\Delta)^{\frac{\alpha}{2}} \tilde{u} + c_{n,\alpha} \int_{\Omega^c} \frac{g(y)}{|x-y|^{n+\alpha}} dy.$$

Thus $u = \tilde{u}$ in Ω and $\tilde{u}$ satisfies the boundary value problem with the homogeneous boundary condition

$$\begin{aligned} &(-\Delta)^{\frac{\alpha}{2}} \tilde{u} = \tilde{f} := f - c_{n,\alpha} \int_{\Omega^c} \frac{g(y)}{|x-y|^{n+\alpha}} dy, \quad x \in \Omega, \\ &\tilde{u} = 0, \quad x \in \Omega^c. \end{aligned}$$

3. Neural Networks Method and Error Estimates

We consider neural networks with one hidden layer in one-dimensional space and two hidden layers in two- or three-dimensional space. Using the rectified linear unit activation function

$$\mathrm{ReLU}(x) = \max(x, 0), \quad x \in \mathbb{R},$$

we introduce a set of neural networks $\mathscr{M}_N^{(n)}$. The superscript n shows the dependence on the spatial dimension. We write simply $\mathscr{M}_N$ if it causes no confusion. In the one-dimensional space let

$$\mathscr{M}_N^{(1)} = \left\{ \sum_{i=1}^{N_1} a_i^{(2)} \mathrm{ReLU}\left(x - b_i^{(1)}\right) + b^{(2)} \,:\, a_i^{(2)}, b_i^{(1)}, b^{(2)} \in \mathbb{R} \right\}.$$

Its total degrees of freedom are $N = 2N_1 + 1$. For $n = 2$ or 3 let

$$\mathscr{M}_N^{(n)} = \left\{ \sum_{j=1}^{N_2} w_j^{(3)} \mathrm{ReLU}\left(\sum_{i=1}^{N_1} w_{j,i}^{(2)} \mathrm{ReLU}\left(\mathbf{w}_i^{(1)} \cdot \mathbf{x} + b_i^{(1)}\right) + b_j^{(2)} \right) + b^{(3)} \right\},$$

where $\mathbf{x}, \mathbf{w}_i^{(1)} \in \mathbb{R}^n$ and $w_{j,i}^{(2)}, w_j^{(3)}, b_i^{(1)}, b_j^{(2)}, b^{(2)}, b^{(3)}$ are real numbers. It is clear that the total degrees of freedom are $N = (n+1) * N_1 + N_1 * N_2 + 2N_2 + 1$. We remark that the architecture of the neural networks is motivated by [31].

Now we are ready to present the neural network method for fractional diffusion equations. The approximate minimization problem is to find $u_N \in \mathscr{M}_N$ such that

$$J(u_N) = \min_{v \in \mathscr{M}_N} J(v). \tag{3.1}$$

Note that the problem (3.1) has a solution since $\mathscr{M}_N \subset \mathbb{V}$. In order to evaluate the error of approximate solution we need the following result.

Theorem 3.1. *If u and u_N are respectively the solutions of the problems* (2.8) *and* (3.1)*, then*

$$\|u - u_N\|_{\mathbb{V}} = \min_{v \in \mathscr{M}_N} \|u - v\|.$$

Proof. Using (2.9) gives

$$\begin{aligned}
J(u_N)-J(u) &= \frac{1}{2}a(u_N,u_N)-\frac{1}{2}a(u,u)+(f,u-u_N)\\
&= \frac{1}{2}a(u_N,u_N)-\frac{1}{2}a(u,u)+a(u,u-u_N)\\
&= \frac{1}{2}a(u_N,u_N)+a\left(u,\frac{1}{2}u-u_N\right)\\
&= \frac{1}{2}a(u_N,u_N-u)+\frac{1}{2}a(u,u-u_N)\\
&= \frac{1}{2}a(u_N-u,u_N-u).
\end{aligned}$$

Since the above relations holds on $\mathscr{M}_N$, we get

$$\begin{aligned}
\|u-u_N\|_{\mathbb{V}}^2 &= 2\big(J(u_N)-J(u)\big) = 2\big(\min_{v\in\mathscr{M}_N} J(v)-J(u)\big)\\
&= \min_{v\in\mathscr{M}_N} 2\big(J(v)-J(u)\big) = \min_{v\in\mathscr{M}_N}\|u-v\|_{\mathbb{V}}^2.
\end{aligned}$$

The proof is complete. □

Unlike the standard piecewise polynomial linear spaces in finite element methods or orthogonal polynomial spaces in spectral methods, the approximation properties of neural networks are not yet fully studied. It is worth mentioning that neural networks represent nonlinear approximations and the relevant function sets are not always linear spaces. Nevertheless, they contain very rich sets or subspaces of functions. In order to show the convergence of the method, we use the fact that the standard finite element spaces constitute a subclass of neural networks. This helps to explain why the neural networks can be applied to fractional order problems.

We assume that $\bar{\Omega}=\bigcup_{T\in\mathscr{T}_h} T$, where $\mathscr{T}_h$ is an admissible triangulation of Ω. The diameter of the elements T is denoted by h_T. Let ρ_T be the diameter of the largest ball contained in T. Here we consider a graded mesh with a family of triangulations satisfying the standard mesh regularity — i.e. there is a positive number σ such that

$$h_T \le \sigma\rho_T$$

for all $T\in\mathscr{T}_h$. Let $h=\max_{T\in\mathscr{T}_h} h_T$. If $T\bigcap\Omega\neq\emptyset$, then $h_T\le Ch^{\mu}$, otherwise

$$h_T \le Ch\,\mathrm{dist}(T,\partial\Omega)^{\frac{\mu-1}{\mu}},$$

where $\mu=2/(1+2\epsilon)\in[1,2)$ for a positive number $\epsilon\in(0,1/2]$.

Next, we consider a discrete space of piecewise linear polynomials and present the error estimate of the neural network method for the fractional Laplace equation. Let

$$\mathbb{V}_N := \big\{v\in\mathbb{V} : v|_T\in\mathscr{P}_1 \text{ for any } T\in\mathscr{T}_h\big\},$$

where N is the total number of degrees of freedom of the space which is approximately h^{-n}. Therefore $h \approx N^{-1/n}$ and there exist constants C_0 and C_1 such that $C_0 N^{-1/n} \leq h \leq C_1 N^{-1/n}$. The approximation property for graded finite element method in weighted Sobolev spaces was proved in [1, 9].

Lemma 3.1 (cf. Acosta & Borthagaray [1] and Borthagaray [9]). *If $\alpha \in (0,2)$, $\epsilon \in (0,1/2)$ and $v \in \mathbb{V} \cap H^{1+\alpha/2}_{\alpha/2}(\Omega)$, then*

$$\min_{v_N \in \mathbb{V}_N} \|v - v_N\|_{\mathbb{V}} \leq C N^{-\frac{1}{n}} \sqrt{\log N} \|v\|_{H^{1+\alpha/2}_{1/2-\epsilon}(\Omega)}.$$

The next lemma states that any finite element function can be approximated by a neural network.

Lemma 3.2 (cf. He *et al.* [31, Corollary 3.1]). *Given a locally convex and shape regular finite element grid $\mathcal{T}_h$, any linear finite element function with N degrees of freedom can be written as an ReLU-DNN with at most $\mathcal{O}(d)$ hidden layers. The number of neurons is at most $\mathcal{O}(d\kappa N)$ with a constant $\kappa > 2$ depending on the shape-regularity of $\mathcal{T}_h$. The number of the non-zero parameters is at most $\mathcal{O}(d\kappa^d N)$, where d refers to the dimension.*

We are ready to establish an error estimate.

Theorem 3.2. *Let u and u_N be the solutions of the problems* (2.8) *and* (3.1)*, respectively. If $u \in \mathbb{V} \cap H^{1+\alpha/2}_{\alpha/2}(\Omega)$, then for $\epsilon \in (0,1/2)$ we have*

$$\|u - u_N\|_{\mathbb{V}} \leq C N^{-\frac{1}{n}} \sqrt{\log N} \|u\|_{H^{1+\alpha/2}_{1/2-\epsilon}(\Omega)}.$$

Proof. By Lemma 3.2, $\mathbb{V}_N$ is a subset of $\mathscr{M}_{N_1}$, where N_1 is bounded by CN with C independent of N. Theorem 3.1 and Lemma 3.1 give

$$\|u - u_N\|_{\mathbb{V}} \leq \min_{v \in \mathscr{M}_{N_1}} \|v - u\|_{\mathbb{V}} \leq \min_{v \in \mathbb{V}_{N_1}} \|v - u\|_{\mathbb{V}} \leq C N_1^{-\frac{1}{n}} \sqrt{\log N_1} \|v\|_{H^{1+\alpha/2}_{1/2-\epsilon}(\Omega)}.$$

This completes the proof. □

Remark 3.1. Note that the above convergence order depends on the dimension n, which looks not so interesting compared to well-established graded finite element methods. However, such dependency is not necessary and we hope it can be removed.

The above results with continuous norm are used for a purely theoretical purpose. In practice however we implement simulations using discretized formulations. Therefore we analyze the effect of the discrete energy functional

$$J_h(v) = \frac{1}{2} a_h(v,v) - f_h(v),$$

where a_h and f_h are approximations of a and f by a numerical integration.

Let u_N^h be the solution of the discrete optimization problem

$$J_h\left(u_N^h\right) = \inf_{v\in\mathbb{V}_N} J_h(v). \tag{3.2}$$

Using the equivalence of the energy norm and the energy functional difference along with (3.2), we have

$$\begin{aligned}
&\left\|u-u_N^h\right\|_{\mathbb{V}}^2 = 2\left(J\left(u_N^h\right)-J(u)\right)\\
&= 2\left(J\left(u_N^h\right)-J_h\left(u_N^h\right)+J_h\left(u_N^h\right)-J_h(u_N)+J_h(u_N)-J(u_N)+J(u_N)-J(u)\right)\\
&\le 2\left(J\left(u_N^h\right)-J_h\left(u_N^h\right)+J_h(u_N)-J(u_N)+J(u_N)-J(u)\right)\\
&\le 2\left|J_h(u_N)-J(u_N)\right|+2\left|J_h\left(u_N^h\right)-J\left(u_N^h\right)\right|+\|u-u_N\|_{\mathbb{V}}^2.
\end{aligned} \tag{3.3}$$

It follows from (3.3) that in order to exploit the great potential of the neural networks, we should use proper numerical quadratures to balance the approximation error and numerical differentiation/integration error. This fact will be taken into consideration when we carry out numerical experiments — cf. Subsection 4.1.3.

4. Numerical Experiments

To illustrate the accuracy of the neural networks methods, we mainly test the one-dimensional cases for simplicity. The corresponding numerical results are presented in Subsection 4.1. To study the influence of the solution layers, we numerically test different hidden layers and provide the comparison of their empirical performance in Subsection 4.2.

In the training, we use the popular Adam optimizer with 20 000 epochs and set the learning rate 0.001 for $\alpha = 0.4$ and $\alpha = 0.8$, and 0.0001 for $\alpha = 1.8$. In one-dimensional cases with one-hidden layer, the biases b_i play the role of grid-points. Therefore, we take uniform grid-points for the biases as the initial setup. For the weights a_i based on the relation between the ReLU bases and the finite element bases [31], we use FEMs on a coarse mesh with the half number of the neurons. In two-dimensional cases, we directly use the random initialization of the Adam optimizer package. All experiments are run on Pytorch platforms and are replicated three times to reduce the variability of numerical tests.

4.1. Numerical results in 1D

For the domain $\Omega = (0,1)$ and the right hand side function $f \equiv 1$ the exact solution is

$$u(x) = \frac{(1-x^2)^{\frac{\alpha}{2}}}{\Gamma(\alpha+1)},$$

cf. [29]. Unlike the integer-order differential operator, for fractional operators the numerical differentiation/integration are not trivial. In this subsection, we describe the numerical integration of a continuous energy functional.

To make the numerical integration error negligible, we use the mesh with $M = 200$ of grid-points. Set $h = 1/M$ and $x_i = ih$ and $x_{i-1/2} = x_i - h/2$, $i = 0, 1, 2, \dots M$ and discretize the double integral by a second-order quadrature rule — viz.

$$\begin{aligned}
&\frac{1}{2}\int_0^1\int_0^1 \frac{(v(x)-v(y))^2}{|x-y|^{1+\alpha}}dydx = \int_0^1\left(\int_0^x \frac{(v(x)-v(y))^2}{|x-y|^{1+\alpha}}dy\right)dx \\
&\approx \frac{1}{2}h^{1-\alpha}\sum_{i=2}^{M-1}\sum_{j=1}^{i-1} b_{i-j}^{(\alpha)}(v_i - v_{j-1})^2 + \frac{1}{2}h^{1-\alpha}\sum_{i=2}^{M-1}\sum_{j=1}^{i-1} c_{i-j}^{(\alpha)}(v_i - v_j)^2 \\
&\quad + \frac{1}{4}h^{1-\alpha}\sum_{j=1}^{M-1} b_{M-j}^{(\alpha)}(v_M - v_{j-1})^2 + \frac{1}{4}h^{1-\alpha}\sum_{j=1}^{M-1} c_{M-j}^{(\alpha)}(v_M - v_j)^2 \\
&\quad + 4a_0^{(\alpha)}h^{1-\alpha}\sum_{i=1}^{M-1}(v_i - v_{i-\frac{1}{2}})^2 + 2a_0^{(\alpha)}h^{1-\alpha}(v_M - v_{M-\frac{1}{2}})^2, \qquad (4.1)
\end{aligned}$$

where

$$a_k^{(\alpha)} = \frac{(k+1)^{2-\alpha} - k^{2-\alpha}}{(2-\alpha)}, \quad b_k^{(\alpha)} = \frac{a_k^{(\alpha)}}{(k+1)^2}, \quad c_k^{(\alpha)} = \frac{a_k^{(\alpha)}}{k^2},$$

see Appendix for a more detail.

In the one dimensional case, the term $\rho(x)$ of (2.5) has the form $\rho(x) = (x^{-\alpha} + (1-x)^{-\alpha})/\alpha$. Since the second integral in (2.7) contains $\rho(x)$ an has a stronger singularity, we use the standard Gauss Jacobi quadrature. For $\alpha < 1$ we have

$$\begin{aligned}
\int_0^1 \rho(x)v^2(x)dx &= \frac{1}{\alpha}\int_0^1 \left(x^{-\alpha} + (1-x)^{-\alpha}\right)v^2(x)dx \\
&\approx \frac{1}{\alpha}\sum_i^M \left(v^2\left(X_i^{0,-\alpha}\right)W_i^{0,-\alpha} + v^2\left(X_i^{-\alpha,0}\right)W_i^{-\alpha,0}\right),
\end{aligned}$$

where $X^{\beta,\gamma}$ and $W^{\beta,\gamma}$ are the standard Gauss-Jacobi nodes and weights related to the Jacobi pair (β,γ). However, if $1 \le \alpha < 2$, Gauss-Jacobi quadrature does not work since it only applies to the Jacobi index greater than -1. To deal with this issue, we split the singular term and apply the quadrature rule

$$\begin{aligned}
\int_0^1 \rho(x)v^2(x)dx &= \frac{1}{\alpha}\left(\int_0^1 x^{1-\alpha}\frac{v^2(x)}{x} + \int_0^1 (1-x)^{1-\alpha}\frac{v^2(x)}{1-x}dx\right) \\
&\approx \frac{1}{\alpha}\sum_i^M\left(\frac{v^2(X_i^{0,1-\alpha})}{X_i^{0,1-\alpha}}W_i^{0,1-\alpha} + \frac{v^2(X_i^{1-\alpha,0})}{1-X_i^{1-\alpha,0}}W_i^{1-\alpha,0}\right). \qquad (4.2)
\end{aligned}$$

We apply the mid-point rule to the linear term $\int_0^1 f(x)v(x)dx$ in the energy functional.

4.1.1. Test 1. Numerical solutions with different number of neurons

We first test the accuracy of the neural network methods for fractional orders $\alpha \in (0,1)$ and $\alpha \in (1,2)$. Numerical results are reported through the relative error in the L^2 norm and the energy norm or $H^{\alpha/2}$ seminorm — cf. Table 1. Besides, numerical approximations versus exact solution are shown in Figs. 1 and 2 for $\alpha = 0.8$ and $\alpha = 1.8$, respectively. Note that by using only 5 neurons, we achieve at most four percentages of L^2 relative error — cf. Fig. 1. If the number of neurons is doubled, the error drops dramatically in the interior of the domain, but the error near the boundary dominates. By further doubling the neuron numbers the error decreases only slightly, and we did not observe predicted first-order convergence. Nevertheless, comparing the method with the finite difference method [30] with the same degrees of freedom, we note a better accuracy — cf. Table 2. Note that the continuous L^2 norm and the energy functional $J(u)-J(u_N)$ are computed by the above described numerical quadrature. For the energy norm, we use the equivalence $\|u-u_N\|_{\mathbb{V}}^2 = 2(J(u)-J(u_N))$, while $J(u)$ and $\|u\|_{\mathbb{V}}$ are calculated analytically [29].

For $\alpha = 1.8$, we note the first order convergence in the energy norm, which agrees with the theoretical results (see Fig. 2 and Table 1). The error is very small near the boundary in the energy norm. It may come from the numerical integration (4.2) which requires the vanishing boundary conditions for the integrand.

Table 1: Test 1. Relative error of NNM.

DOF	$\alpha = 0.8$			$\alpha = 1.8$		
	$\frac{\|u-u_n\|_{L^2(\Omega)}}{\|u\|_{L^2(\Omega)}}$	$\frac{\|u-u_n\|_{\mathbb{V}}}{\|u\|_{\mathbb{V}}}$	$\frac{\|J(u)-J(u_N)\|}{\|J(u)\|}$	$\frac{\|u-u_n\|_{L^2(\Omega)}}{\|u\|_{L^2(\Omega)}}$	$\frac{\|u-u_n\|_{\mathbb{V}}}{\|u\|_{\mathbb{V}}}$	$\frac{\|J(u)-J(u_N)\|}{\|J(u)\|}$
10	0.035315	0.128593	0.016536	0.046187	0.201595	0.040641
20	0.016857	0.085550	0.007319	0.012615	0.090913	0.008265
40	0.012649	0.075684	0.005728	0.004022	0.042591	0.001814

Table 2: Test 1. Relative errors of NNM and FDM, $\alpha = 0.8$.

DOF	NNM			FDM		
	$\frac{\|u-u_n\|_{L^2(\Omega)}}{\|u\|_{L^2(\Omega)}}$	$\frac{\|u-u_n\|_{\mathbb{V}}}{\|u\|_{\mathbb{V}}}$	$\frac{\|J(u)-J(u_N)\|}{\|J(u)\|}$	$\frac{\|u-u_n\|_{L^2(\Omega)}}{\|u\|_{L^2(\Omega)}}$	$\frac{\|u-u_n\|_{\mathbb{V}}}{\|u\|_{\mathbb{V}}}$	$\frac{\|J(u)-J(u_N)\|}{\|J(u)\|}$
10	0.035315	0.128593	0.016536	0.086952	0.233671	0.054602
20	0.016857	0.085550	0.007319	0.046912	0.164795	0.027157
40	0.012649	0.075684	0.005728	0.025335	0.117519	0.013811

4.1.2. Test 2. Accuracy of least-squares and energy formulations

To formulate a least-squares loss functional, we replace the continuous fractional operator by the second-order fractional centered difference operator [30] and consider the following

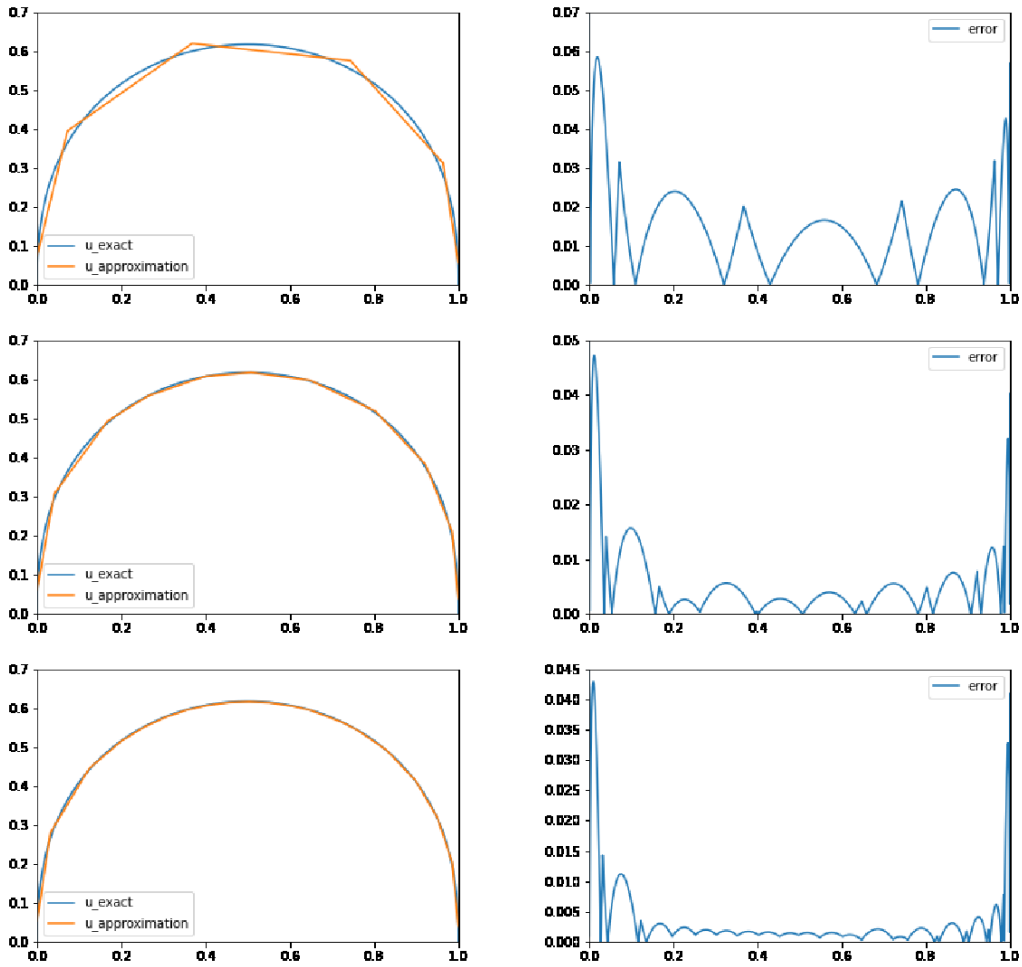

Figure 1: Left: Exact and numerical solutions of fractional Poisson equation, $\alpha = 0.8$. Right: Error := $|u_{ext} - u_{apr}|$. Top row: $N = 5$. Middle row: $N = 10$. Bottom row: $N = 20$.

functional:

$$J_h(v) = h \sum_{j=1}^{M-1} \left[(-\Delta)_h^{\alpha/2} v(x_j) - f(x_j) \right]^2,$$

where

$$(-\Delta_h)^{\frac{\alpha}{2}} u(x) =: \frac{1}{h^\alpha} \sum_{j \in \mathbb{Z}} a_j^{(\alpha)} u(x + jh), \quad a_j^{(\alpha)} = \frac{(-1)^j \Gamma(\alpha+1)}{\Gamma(\alpha/2 - j + 1)\Gamma(\alpha/2 + j + 1)}.$$

To control the numerical integration and differentiation error, we use the same number $M = 200$ of grid-points, the same optimizer, and the same number of iterations. Two formulations with different fractional orders are displayed in Figs. 3 and 4. Note that the least-squares formulation demonstrates a slightly better performance than the Ritz formulation for $\alpha = 0.8$, but it does not perform well for $\alpha = 1.8$, cf. Fig. 4.

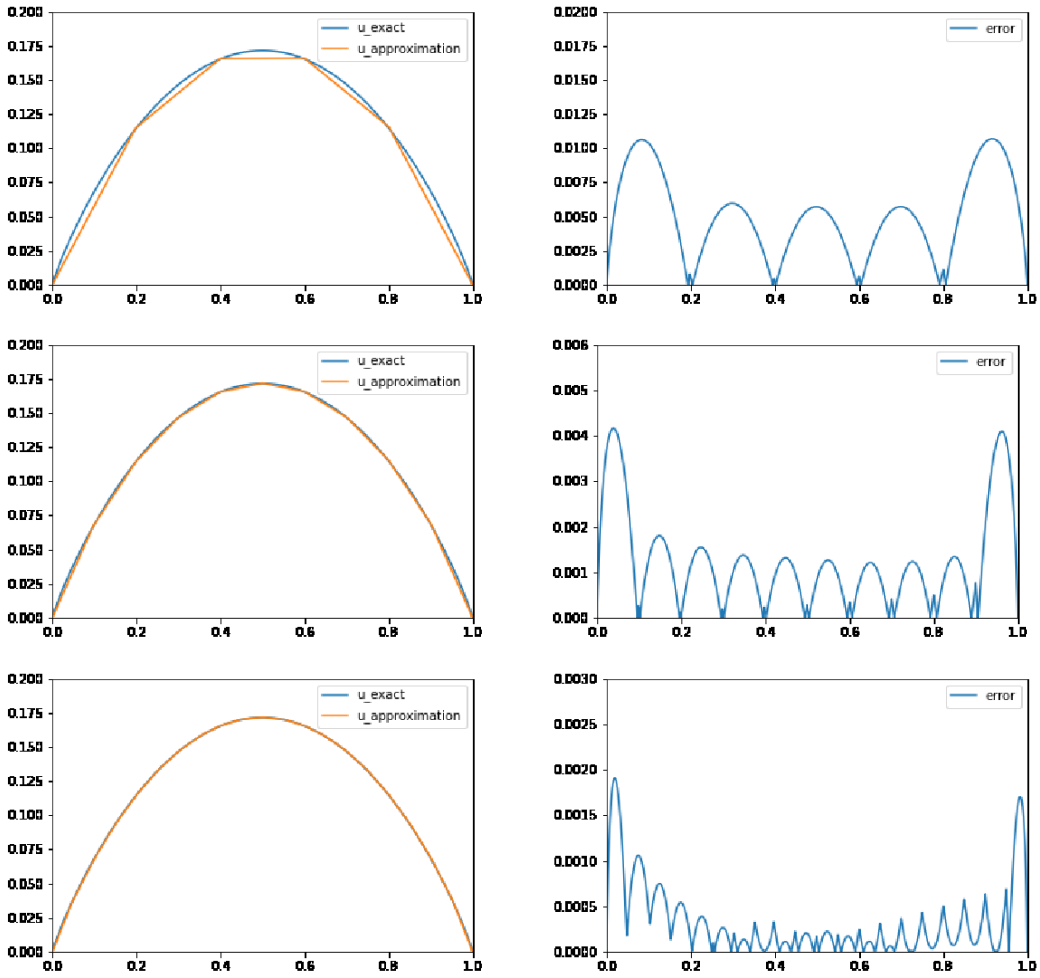

Figure 2: Left: Exact and numerical solutions of fractional Poisson equation, $\alpha = 1.8$. Right: Error := $|u_{ext} - u_{apr}|$. Top row: $N = 5$. Middle row: $N = 10$. Bottom row: $N = 20$.

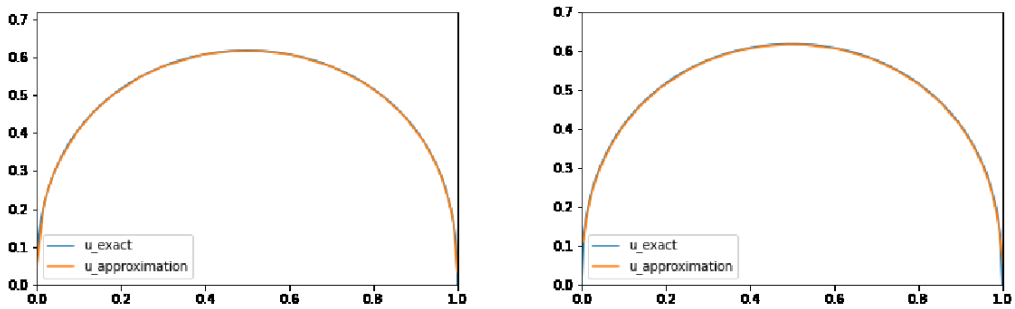

Figure 3: Left: Ritz formulation. Right: Least squares formulation. $\alpha = 0.8$, $N = 40$.

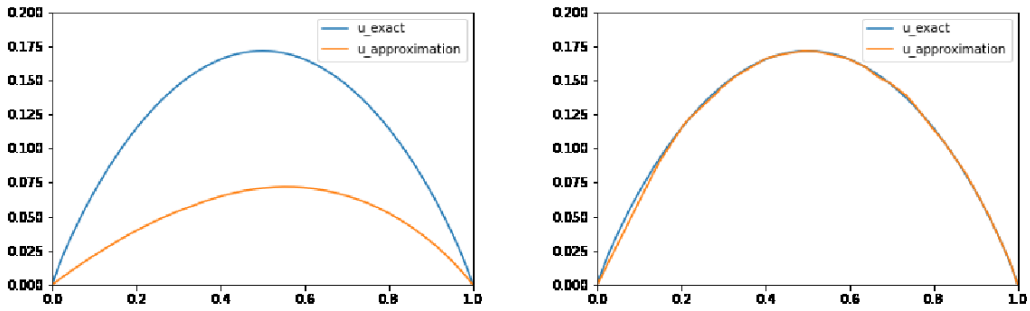

Figure 4: Left: Least squares formulation. Right: Ritz formulation. $\alpha = 1.8$, $N = 40$.

4.1.3. Test 3: Influence of training and evaluation points.

We examine the influence of numerical quadratures on the training performance. Note that in order to reduce a stronger singularity at the boundary into weaker, one usually adopts the standard Duffy transform technique [17] and applies a nonuniform Gauss-Jacobi quadrature to the resultant integrals, thus obtaining the discrete functional

$$\begin{aligned} J_M(u_N) = &\frac{c_{1,\alpha}}{2}\sum_{i,j=1}^{M}\frac{\left(u_N(X_i^{0,1-\alpha})-u_N(X_i^{0,1-\alpha}Z_j^{1-\alpha,0})\right)^2}{X_i^{0,1-\alpha}(1-Z_j^{1-\alpha,0})^2}W_i^{0,1-\alpha}W_j^{1-\alpha,0} \\ &+\frac{c_{1,\alpha}}{2\alpha}\sum_{i}^{M}\left(\frac{u_N^2(X_i^{0,1-\alpha})}{X_i^{0,1-\alpha}}W_i^{0,1-\alpha}+\frac{u_N^2(X_i^{1-\alpha,0})}{1-X_i^{1-\alpha,0}}W_i^{1-\alpha,0}\right) \\ &-\sum_{i}^{M} f\left(X_i^{0,0}\right)u_N\left(X_i^{0,0}\right)W_i^{0,0}. \end{aligned} \tag{4.3}$$

The only difference between (4.1) and (4.3) is the treatment of the leading term in the fractional semi-norm.

Here we use the same optimizer and maximum iteration number. The corresponding numerical results are shown in Fig. 5. It is worth noting that the functional (4.1) performs

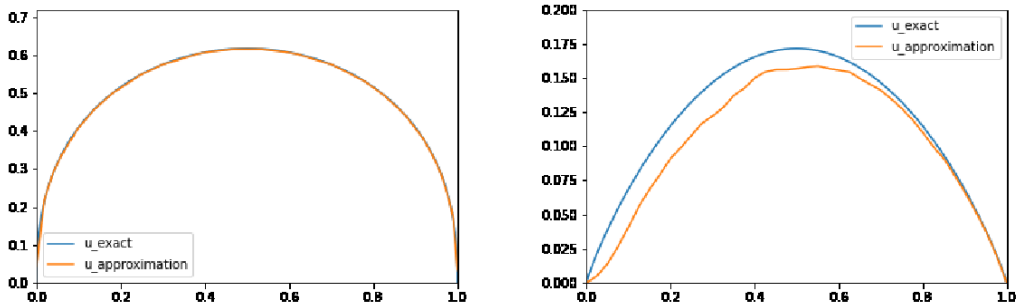

Figure 5: Performance of numerical quadratures. Left: Eq. 4.1. Right: Eq. 4.3.

much better than (4.3). One of the possible reasons is that the nonuniform quadrature (4.3) makes the non-convex functional even worse and leads to highly non-convex one which is more difficult to train. Therefore, we must be careful when we use quadrature rules for continuous energy functionals.

4.1.4. Test 4. Reaction-diffusion equation with sharp interior layer

Consider now the reaction-diffusion equation

$$\epsilon^{\alpha}(-\Delta)^{\frac{\alpha}{2}}u + G(u) = f, \quad x \in \Omega.$$

It is well known that for small ϵ the nonlinear equation can have sharp interior transition layers and their locals are unknown. In order to test problem, we choose the linear reaction term $G(u) = u$ and the Gaussian function $u = \exp(-|x-0.5|^2/\epsilon^2)$ as fabricated solution on the interval $\Omega = [0,1]$. The corresponding right hand side has the form

$$f(x) = \epsilon^{\alpha}(-\Delta)^{\frac{\alpha}{2}}u + u$$

where

$$(-\Delta)^{\frac{\alpha}{2}}u = \mu_{1,\alpha}\frac{1}{\epsilon^{\alpha}}\,{}_1F_1\left(\frac{1+\alpha}{2};\frac{1}{2};-\frac{|x-0.5|^2}{\epsilon^2}\right), \quad \mu_{1,\alpha} = \frac{2^{\alpha}\Gamma((1+\alpha)/2)}{\Gamma(1/2)}$$

and ${}_1F_1$ is two-parameters hypergeometric function directly evaluated by the `scipy.special` package.

In this case, the reaction term has to be added into the optimization functional — i.e.

$$J(u) = \frac{c_{1,\alpha}\epsilon^{\alpha}}{2}\int_0^1\int_0^1 \frac{|u(x)-u(y)|^2}{|x-y|^{1+\alpha}}dxdy + \epsilon^{\alpha}c_{1,\alpha}\int_0^1 \rho(x)u^2(x)dx + \frac{1}{2}\int_0^1 u^2(x)dx - \int_0^1 fudx.$$

In the numerical simulations, we take $\alpha = 0.8$, $\epsilon = 0.03$ and test $N = 40$ and $N = 80$ neurons. In the former case, the relative errors in L^2 and energy norms are about 0.075833 and 0.17818, respectively. On the other hand, in the latter case the respective are 0.036144 and 0.135395. The corresponding numerical and exact solutions shown in Fig. 6. Note that the numerical solution obtained by NNM with proper number of neurons can accurately capture the sharp interior transition layer although its location was not known beforehand.

4.2. Numerical results in 2D

We now consider the two-dimensional equation

$$(-\Delta)^{\frac{\alpha}{2}}u = f$$

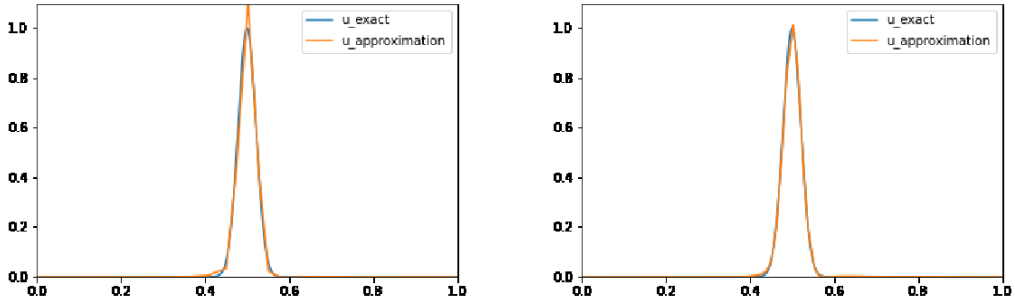

Figure 6: Left: 40 Neurons. Right: 80 Neurons.

on the circular domain $\Omega := \{x = (x_1, x_2) : x_1^2 + x_2^2 < 1\}$ with the right-hand side $f(x) = 1$, $x \in \Omega$. According to [28], the solution of this problem is

$$u(x_1, x_2) = \frac{(1 - x_1^2 - x_2^2)^{\frac{\alpha}{2}}}{2^\alpha [\Gamma(\alpha/2 + 1)]^2}, \quad \Omega = \left\{(x_1, x_2) : x_1^2 + x_2^2 < 1\right\}.$$

To simplify the computation of integrals in the energy functional, we embed the disk into the rectangular domain $[0, 1]^2$, i.e.

$$\begin{aligned} J(u) &= \frac{1}{2} \iint_\Omega u(-\Delta)^{\frac{\alpha}{2}} u \, dx - \iint_\Omega uf \, dx \\ &= \frac{1}{2} \int_{-1}^{1} \int_{-1}^{1} \tilde{u}(-\Delta)^{\frac{\alpha}{2}} \tilde{u} \, dx - \int_{-1}^{1} \int_{-1}^{1} \tilde{u} f \, dx, \end{aligned}$$

where $\tilde{u} = 1_\Omega u$ is the zero extension of u outside of Ω.

We show the numerical results for $\alpha = 0.4$ and $\alpha = 1.4$. We test and compare the performance for two different hidden layers but under the condition of the same degrees of freedom (DOF=501). More precisely, consider the following situations:

Case 1. Two hidden layers with 20 neurons for each layers. The corresponding architecture is referred to as 2-20-20-1.

Case 2. A one hidden layer with 125 neurons. The corresponding architecture is referred to as 2-125-1.

Case 3. Three hidden layers with 10 neurons for the first and second hidden layers, and 30 neurons for the third hidden layer. The corresponding architecture is referred to as 2-10-10-30-1.

For the description of deep neural networks the reader can consult [11].

In this example, we use the fractional centered scheme [30] to discretize the high-dimensional fractional derivatives. For the integrand v vanishing on the boundary, recall the

trapezoidal quadrature rule with the relatively large numerical differentiation/integration number N=200 in each direction,

$$\int_{-1}^{1}\int_{-1}^{1} v(x_1,x_2)dx_1dx_2 \approx \sum_{i,j=1}^{N-1} v(x_{1,i},x_{2,j})h_1h_2, \quad x_{1,i}=ih_1, \quad x_{2,j}=jh_2.$$

For $\alpha = 0.4$, numerical and exact solutions for Cases 1 and 2 are shown in Figs. 7 and 8, respectively. Note that the numerical solution mimics true solution quite well. To have a closer look at the magnitude of the errors, we display the point-wise picture of them in Fig. 9. It is worth mentioning that for the same number of parameters or the DOF, and the same setup of the optimization training, the shallow neural networks perform much better than the deep ones. Actually, we test more hidden layers and found that the conclusion is still true — cf. Fig. 10, where the error for two-hidden layers is smaller than that for three layers. When $\alpha = 1.4$, the similar behavior has been observed in Figs. 11-14.

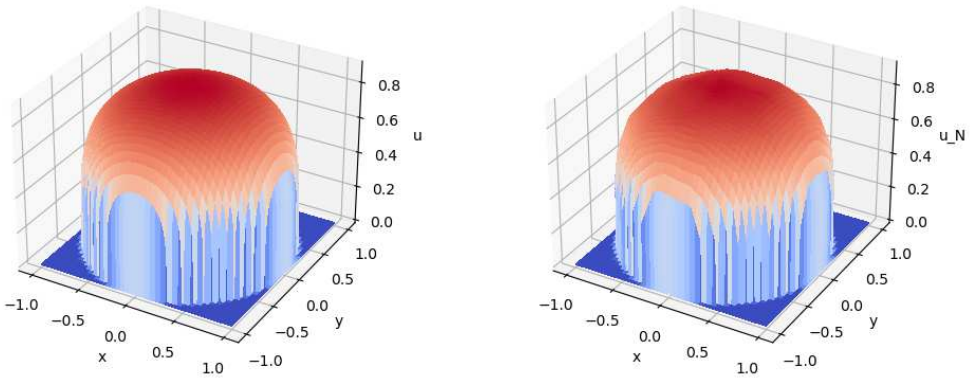

Figure 7: Left: Exact solution. Right: Numerical solution by two-hidden layers, Case 2-20-20-1, DOF= 501, $\alpha = 0.4$.

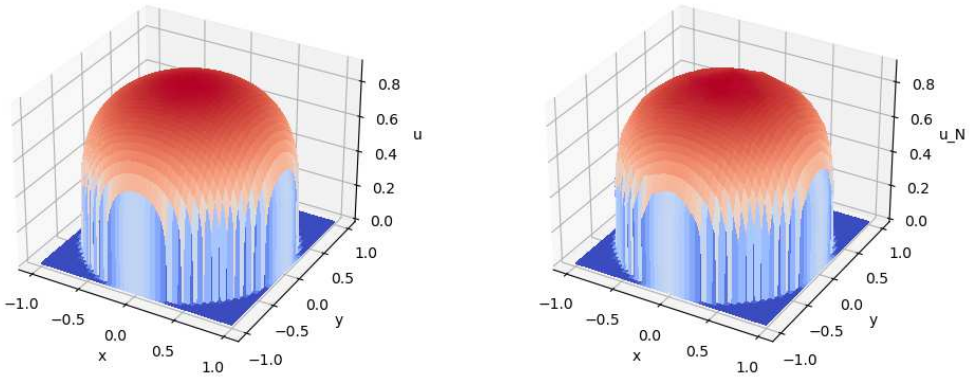

Figure 8: Left: Exact solution. Right: Numerical solution by one-hidden-layer, Case 2-125-1, DOF=501, $\alpha = 0.4$.

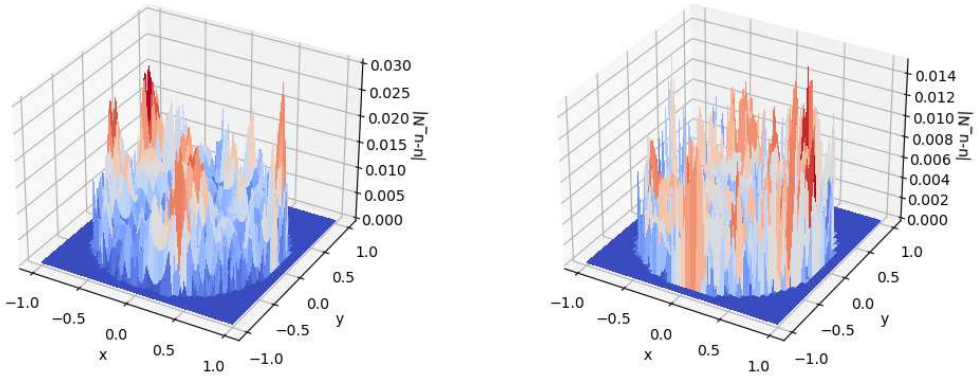

Figure 9: Numerical errors, DOF=501, $\alpha = 0.4$. Left: two-hidden layers, Case 2-20-20-1. Right: One-hidden layer, Case: 2-125-1.

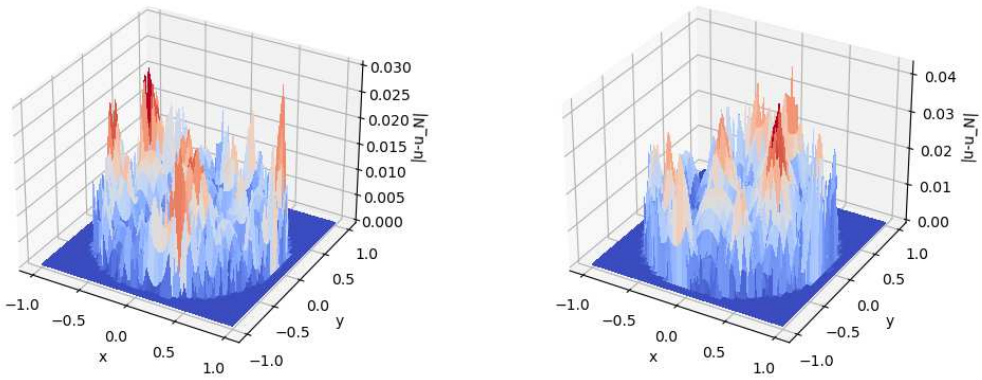

Figure 10: Numerical errors, DOF=501, $\alpha = 0.4$. Left: two-hidden layers, Case 2-20-20-1. Right: three-hidden layers, Case 2-10-10-30-1.

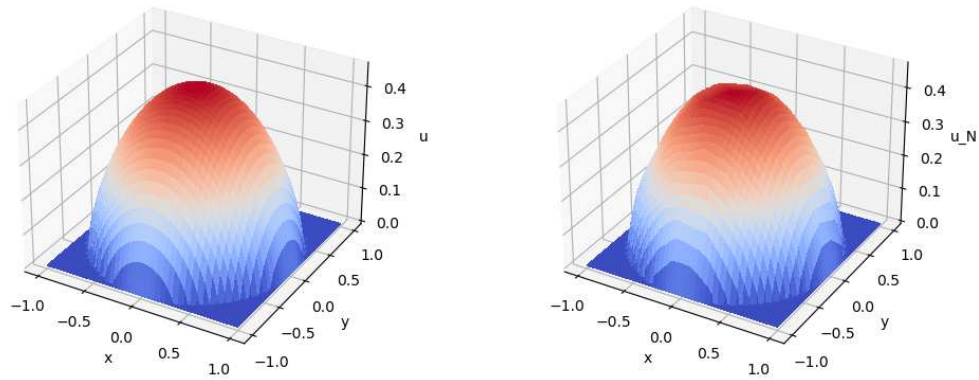

Figure 11: Left: Exact solution. Right: Numerical solution by two-hidden layers, Case 2-20-20-1, DOF= 501, $\alpha = 1.4$.

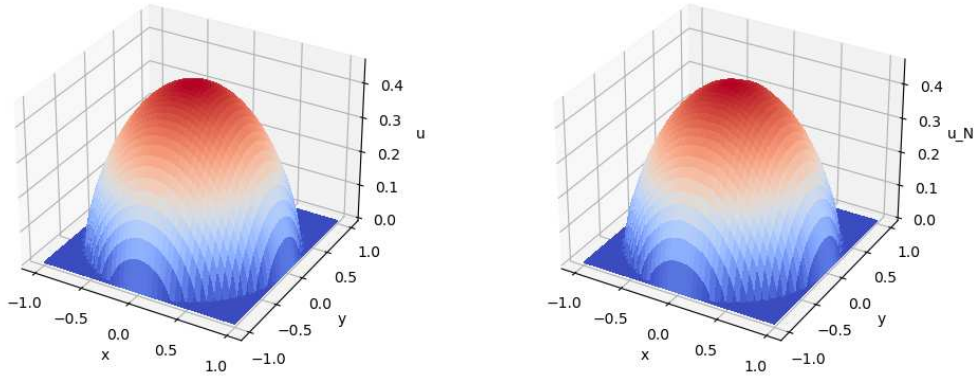

Figure 12: Left: Exact solution. Right: Numerical solution by one-hidden-layer, Case 2-125-1, DOF= 501, $\alpha = 1.4$.

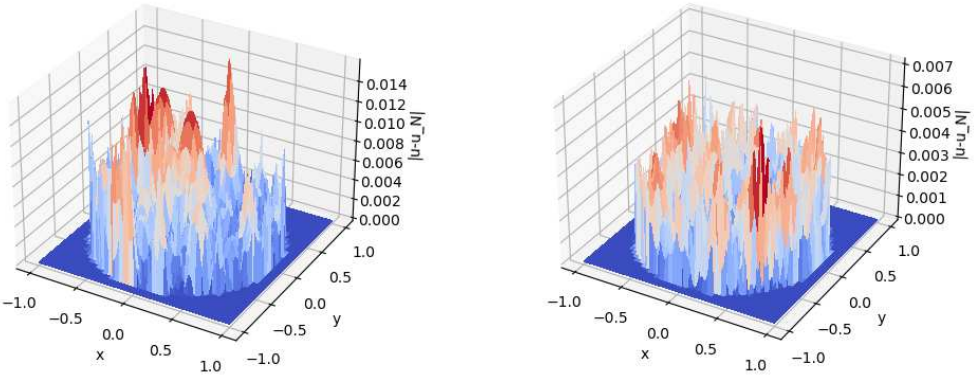

Figure 13: Numerical errors, DOF=501, $\alpha = 1.4$. Left: Two-hidden layers, Case 2-20-20-1. Right: One-hidden layer, Case 2-125-1.

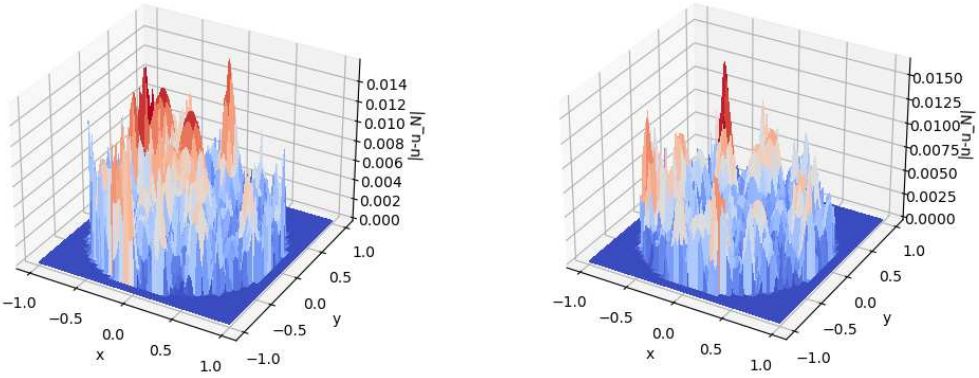

Figure 14: Numerical errors, DOF=501, $\alpha = 1.4$. Left: Two-hidden layers, Case 2-20-20-1. Right: Three-hidden layers, Case 2-10-10-30-1.

There is also another interesting phenomenon. For the shallow neural networks with a one hidden layer, the larger point-wise errors seem to have clusters at the boundary. In contrast, for more layers, the point-wise errors distribute mildly uniform near the boundary except for few spikes. However, one would expect the same performance under the same DOF. One possible explanation of the performance difference is that it may be caused by a difficult and inadequate training of deep ones.

5. Conclusion and Discussions

We consider the neural network method based on the Ritz formulation for the fractional diffusion equation. Connecting neural networks with piecewise linear polynomials, we analyze its convergence and errors. Numerical experiments are used to verify the accuracy of the method.

In this work, we only explore the rectified linear unit (ReLU) activation function. It is possible to extend the theoretical results to other smooth activation functions such as the power ReLU or radial basis functions. We remark that the current work mainly focuses on the approximation error of the neural network method. Here, the error source consists of three parts: discretization error, numerical integration error and optimization error.

Regarding the numerical quadrature of the fractional operator, for simplicity, we only use the traditional 2D FDM operator in [30] for the numerical integration. Other efficient and accurate numerical quadrature can also be explored, for example, the very recent work by the kernel-based numerical quadrature [10] for the fractional Laplacian operator on the general domain.

We did not compare the computational complexity of NNMs with traditional methods but rather focus on the high order convergence and potential of method. In particular, we showed that NNMs are more accurate than other traditional methods such as finite difference methods. Nevertheless, using the ADAM method as an optimizer for the neural network training is still very expensive. Therefore, more efficient optimizer and initialization method are needed.

When the model equation contains asymmetrical fractional operators or a convection term, the symmetric energy formulation considered in this work is not applicable. To address this issue, one may introduce the artificial least-squares formulation. However, the construction of efficient leas-squares formulations which can uncover the potential benefits of the NNMs it still an open problem.

Appendix A

A.1 Numerical integration

Denote $\Phi(x,y)=(u(x)-u(y))^2/|x-y|^2$. By the standard trapezoidal rule, we have

$$\int_0^1\left(\int_0^x \frac{(v(x)-v(y))^2}{|x-y|^{1+\alpha}}dy\right)dx=\int_0^1\left(\int_0^x \Phi(x,y)(x-y)^{1-\alpha}dy\right)dx$$

$$= h\sum_{i=1}^{M-1}\int_0^{x_i}\Phi(x_i,y)(x_i-y)^{1-\alpha}dy + \frac{h}{2}\int_0^{x_M}\Phi(x_M,y)(x_M-y)^{1-\alpha}dy + \mathcal{O}\left(h^2\right), \quad \text{(A.1)}$$

where $x_i = ih$ and $h = 1/M$. Splitting each weighted integral into two parts and respectively using weighted trapezoidal and weighted mid-point rules yields

$$\begin{aligned}
&\int_0^{x_i}\Phi(x_i,y)(x_i-y)^{1-\alpha}dy\\
&=\sum_{j=1}^{i-1}\int_{x_{j-1}}^{x_j}\Phi(x_i,y)(x_i-y)^{1-\alpha}dy + \int_{x_{i-1}}^{x_i}\Phi(x_i,y)(x_i-y)^{1-\alpha}dy\\
&=\frac{1}{2}h^{2-\alpha}\sum_{j=1}^{i-1}a_{i-j}^{(\alpha)}\left[\Phi(x_i,x_{j-1})+\Phi(x_i,x_j)\right] + a_0^{(\alpha)}h^{2-\alpha}\Phi(x_i,x_{i-\frac{1}{2}}) + \mathcal{O}\left(h^2\right), \qquad \text{(A.2)}
\end{aligned}$$

where

$$a_{i-j}^{(\alpha)} = h^{\alpha-2}\int_{x_{j-1}}^{x_j}(x_i-y)^{1-\alpha}dy = \frac{(i-j+1)^{2-\alpha}-(i-j)^{2-\alpha}}{2-\alpha}.$$

Substituting (A.2) into (A.1) and regrouping the terms gives the result.

A.2 Quasi-linear complexity of double summation

Writing the representation

$$\begin{aligned}
\sum_{i=2}^{M-1}\sum_{j=1}^{i-1}(u_i-u_{j-1})^2 b_{i-j}^{(\alpha)} &= \sum_{i=2}^{M-1}\sum_{j=1}^{i-1}(u_i)^2 b_{i-j}^{(\alpha)} - 2\sum_{i=2}^{M-1}\sum_{j=1}^{i-1}u_i u_{j-1} b_{i-j}^{(\alpha)}\\
&\quad + \sum_{i=2}^{M-1}\sum_{j=1}^{i-1}(u_{j-1})^2 b_{i-j}^{(\alpha)},
\end{aligned}$$

we can use the product of a Toeplitz matrix vector product and the standard fast Fourier transform to accelerate the computation with quasi-linear complexity. For example, the first term can be written as

$$\sum_{i=2}^{M-1}\sum_{j=1}^{i-1}u_i^2 b_{i-j}^{(\alpha)} = \left[u_2^2, u_3^2, \ldots, u_{M-1}^2\right]\begin{bmatrix} b_1^{(\alpha)} & 0 & \cdots & \cdots & 0\\ b_2^{(\alpha)} & b_1^{(\alpha)} & 0 & \ddots & \vdots\\ \vdots & b_2^{(\alpha)} & \ddots & \ddots & \vdots\\ \vdots & & \ddots & \ddots & \\ b_{M-2}^{(\alpha)} & \cdots & \cdots & \cdots & b_1^{(\alpha)}\end{bmatrix}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix}.$$

Acknowledgments

The authors thank Dr. Jingshuang Chen who helped us with coding and proofread the earlier version of the manuscript. We also thank the anonymous referees for their insightful and constructive comments.

This work was supported in part by the National Science Foundation under Grant DMS-2110571.

References

[1] G. Acosta and J.P. Borthagaray, *A fractional Laplace equation: regularity of solutions and finite element approximations*, SIAM J. Numer. Anal. **55**, 472–495 (2017).
[2] R.A. Adams and J.F. Fournier, *Sobolev Spaces*, Academic Press (2003).
[3] M. Ainsworth and C. Glusa, *Towards an efficient finite element method for the integral fractional Laplacian on polygonal domains*, in: *Contemporary Computational Mathematics – A Celebration of the 80th Birthday of Ian Sloan*, pp.17–57, Springer (2018).
[4] G. Alberti and G. Bellettini, *A nonlocal anisotropic model for phase transitions: The optimal profile problem*, Math. Ann. **310**, 527–560 (1998).
[5] G. Alberti, G. Bouchitté and P. Seppecher, *Phase transition with the line-tension effect*, Arch. Rational Mech. Anal. **144**, 1–46 (1998).
[6] O.G. Bakunin, *Turbulence and diffusion*, Springer Series in Synergetics, Springer-Verlag (2008).
[7] P.W. Bates, *On some nonlocal evolution equations arising in materials science*, Fields Inst. Commun. **48**, 13–52 (2006).
[8] A. Bonito, J.P. Borthagaray, R.H. Nochetto, E. Otárola and A.J. Salgado, *Numerical methods for fractional diffusion*, Comput. Vis. Sci. **19**, 19–46 (2018).
[9] J.P. Borthagaray Peradotto, *Laplaciano Fraccionario: Regularidad de Soluciones y Aproximaciones for Elementos Finitos*, PhD Thesis, Universidad de Buenos Aires (2017).
[10] J. Burkardt, Y. Wu and Y. Zhang, *A unified meshfree pseudospectral method for solving both classical and fractional pdes*, SIAM J. Sci. Comput. **43**, A1389–A1411 (2021).
[11] Z. Cai, J. Chen, M. Liu and X. Liu, *Deep least-squares methods: An unsupervised learning-based numerical method for solving elliptic PDEs*, J. Comput. Phys. **420**, 109707 (2020).
[12] R. Cont and P. Tankov, *Financial Modelling with Jump Processes*, Chapman & Hall/CRC Financial Mathematics Series, Chapman & Hall/CRC (2004).
[13] A. Córdoba and D. Córdoba, *A maximum principle applied to quasi-geostrophic equations*, Comm. Math. Phys. **249**, 511–528 (2004).
[14] M. D'Elia, Q. Du, C. Glusa, M. Gunzburger, X. Tian and Z. Zhou. *Numerical methods for nonlocal and fractional models*, Acta Numer. **29**, 1–124 (2020).
[15] M. D'Elia and M. Gunzburger, *The fractional Laplacian operator on bounded domains as a special case of the nonlocal diffusion operator*, Comput. Math. Appl. **66**, 1245–1260 (2013).
[16] Q. Du, *Nonlocal Modeling, Analysis, and Computation*, CBMS-NSF Regional Conference Series in Applied Mathematics **94**, SIAM (2019).
[17] M.G. Duffy, *Quadrature over a pyramid or cube of integrands with a singularity at a vertex*, SIAM J. Numer. Anal. **19**, 1260–1262 (1982).
[18] S. Duo, H.W. Wyk and Y. Zhang, *A novel and accurate finite difference method for the fractional Laplacian and the fractional Poisson problem*, J. Comput. Phys. **355**, 233–252 (2018).

[19] S. Duo and Y. Zhang, *Accurate numerical methods for two and three dimensional integral fractional Laplacian with applications*, Comput. Methods Appl. Mech. Engrg. **355**, 639–662 (2019).
[20] W. E and B. Yu, *The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems*, Commun. Math. Stat. **6**, 1–12 (2018).
[21] P.W. Egolf and K. Hutter, *Nonlinear, Nonlocal and Fractional Turbulence: Alternative Recipes for the Modeling of Turbulence*, Springer (2020).
[22] B.P. Epps and B. Cushman-Roisin, *Turbulence modeling via the fractional Laplacian*, ArXiv, 2018.
[23] P. Garbaczewski and V. Stephanovich, *Fractional Laplacians in bounded domains: Killed, reflected, censored, and taboo Lévy flights*, Phys. Rev. E, **99** (2019).
[24] G. Gilboa and S. Osher, *Nonlocal operators with applications to image processing*, Multiscale Model. Simul. **7**, 1005–1028 (2008).
[25] Y. Gu and M.K. Ng, *Deep Ritz method for the spectral fractional Laplacian equations using the Caffarelli-Silvestre extension*, SIAM J. Sci. Comput. **44**, 2018–2036 (2022).
[26] B. Guo, X. Pu and F. Huang, *Fractional Partial Differential Equations and Their Numerical Solutions*, World Scientific Publishing (2015).
[27] L. Guo, H. Wu, X. Yu and T. Zhou, *Monte Carlo pinns: deep learning approach for forward and inverse problems involving high dimensional fractional partial differential equations*, ArXiv, 2022.
[28] Z. Hao, H. Li, Z. Zhang and Z. Zhang, *Sharp error estimates of a spectral Galerkin method for a diffusion-reaction equation with integral fractional Laplacian on a disk*, Math. Comp. **90**, 2107–2135 (2021).
[29] Z. Hao and Z. Zhang, *Optimal regularity and error estimates of a spectral Galerkin method for fractional advection-diffusion-reaction equations*, SIAM J. Numer. Anal. **58**, 211–233 (2020).
[30] Z. Hao, Z. Zhang and R. Du, *Fractional centered difference scheme for high-dimensional integral fractional Laplacian*, J. Comput. Phys. **424**, 109851, 2021.
[31] J. He, L. Li, J. Xu and C. Zheng, *Relu deep neural networks and linear finite elements*, J. Comput. Math. **38**, 502–527 (2020).
[32] Y. Huang and A. Oberman, *Numerical methods for the fractional Laplacian: A finite difference–quadrature approach*, SIAM J. Numer. Anal. **52**, 3056–3084 (2014).
[33] N. Laskin, *Fractional Quantum Mechanics*, World Scientific Publishing (2018).
[34] A. Lischke *et al.*, *What is the fractional Laplacian? A comparative review with new results*, J. Comput. Phys. **404**, 109009 (2020).
[35] S. Lisini, E. Mainini and A. Segatti, *A gradient flow approach to the porous medium equation with fractional pressure*, Arch. Ration. Mech. Anal. **227**, 567–606 (2018).
[36] K. Logvinova and M.-C. Néel, *A fractional equation for anomalous diffusion in a randomly heterogeneous porous medium*, Chaos **14**, 982–987 (2004).
[37] V. Minden and L. Ying, *A simple solver for the fractional Laplacian in multiple dimensions*, SIAM J. Sci. Comput. **42**, A878–A900 (2020).
[38] G. Pang, L. Lu and G.E. Karniadakis, *FPINNs: Fractional physics-informed neural networks*, SIAM J. Sci. Comput. **41**, A2603–A2626 (2019).
[39] C. Pozrikidis, *The Fractional Laplacian*, CRC Press (2016).
[40] J.A. Rosenfeld, S.A. Rosenfeld and W.E. Dixon, *A mesh-free pseudospectral approach to estimating the fractional Laplacian via radial basis functions*, J. Comput. Phys. **390**, 306–322 (2019).
[41] C. Vollmann and V. Schulz, *Exploiting multilevel Toeplitz structures in high dimensional nonlocal diffusion*, Comput. Vis. Sci. **20**, 29–46 (2019).
[42] B.J. West, *Fractional Calculus View of Complexity*, CRC Press (2016).
[43] K. Xu and E. Darve, *Isogeometric collocation method for the fractional Laplacian in the 2D*

bounded domain, Comput. Methods Appl. Mech. Engrg. **364**, 112936 (2020).
[44] N. Yadav, A. Yadav and M. Kumar, *An Introduction to Neural Network Methods for Differential Equations*, Springer Briefs in Applied Sciences and Technology, Springer (2015).
[45] T. Zhu and J.M. Harris, *Modeling acoustic wave propagation in heterogeneous attenuating media using decoupled fractional Laplacians*, Geosciences **79** (2014).