Contents lists available at ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

An open-source parallel code for computing the spectral fractional Laplacian on 3D complex geometry domains[☆,☆☆]

Max Carlson[b,1], Xiaoning Zheng[a,1], Hari Sundar[b], George Em Karniadakis[a], Robert M. Kirby[b,*]

[a] Division of Applied Mathematics, Brown University, United States of America
[b] School of Computing, University of Utah, United States of America

A R T I C L E I N F O

Article history:
Received 21 February 2020
Received in revised form 7 October 2020
Accepted 15 October 2020
Available online 26 October 2020

Keywords:
Fractional Laplacian
Anomalous diffusion
Eigen-decomposition
Surface quasi-geostrophic equation
Open-source code

A B S T R A C T

We present a spectral element algorithm and open-source code for computing the fractional Laplacian defined by the eigenfunction expansion on finite 2D/3D complex domains with both homogeneous and nonhomogeneous boundaries. We demonstrate the scalability of the spectral element algorithm on large clusters by constructing the fractional Laplacian based on computed eigenvalues and eigenfunctions using up to thousands of CPUs. To demonstrate the accuracy of this eigen-based approach for computing the factional Laplacian, we approximate the solutions of the fractional diffusion equation using the computed eigenvalues and eigenfunctions on a 2D quadrilateral, and on a 3D cubic and cylindrical domain, and compare the results with the contrived solutions to demonstrate fast convergence. Subsequently, we present simulation results for a fractional diffusion equation on a hand-shaped domain discretized with 3D hexahedra, as well as on a domain constructed from the Hanford site geometry corresponding to nonzero Dirichlet boundary conditions. Finally, we apply the algorithm to solve the surface quasi-geostrophic (SQG) equation on a 2D square with periodic boundaries. Simulation results demonstrate the accuracy, efficiency, and geometric flexibility of our algorithm and that our algorithm can capture the subtle dynamics of anomalous diffusion modeled by the fractional Laplacian on complex geometry domains. The included open-source code is the first of its kind.

Program summary
Program title: Nektarpp_EigenMM
CPC Library link to program files: https://doi.org/10.17632/whtc75rj55.1
Developer's repository link: https://github.com/paralab/Nektarpp_EigenMM
Licensing provisions: MIT License
Programming language: C/C++, MPI
Nature of problem: An open-source parallel code for computing the spectral fractional Laplacian on 3D complex geometry domains.
Solution method: A distributed, sparse, iterative algorithm is developed to solve an associated integer-order Laplace eigenvalue problem for use in computing approximate solutions to the fractional diffusion equation.
Additional comments including restrictions and unusual features: The code is implemented on CPUs with super-linear parallel efficiency at extreme scale.

1. Introduction

During the past few decades, the fractional Laplacian has found extensive applications in modeling long-range interaction occurring in various disciplines of science and engineering, such as energy dissipation of acoustic propagation [1–3], turbulence diffusion [4–7], contaminant transport in groundwater [8], nonlocal heat conduction [9], and electromagnetic fields on fractals [10].

☆ The review of this paper was arranged by Prof. N.S. Scott.
☆☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).
* Corresponding author.
E-mail addresses: mcarlson@cs.utah.edu (M. Carlson), xiaoning_zheng@brown.edu (X. Zheng), hari@cs.utah.edu (H. Sundar), george_karniadakis@brown.edu (G.E. Karniadakis), kirby@cs.utah.edu (R.M. Kirby).
1 Both authors contributed equally to this manuscript.

Many numerical schemes have been developed to compute the fractional Laplacian based on different definitions, such as the pseudo-spectral method using the spectral definition [11], the singular boundary method using implicit definition [12], the adaptive finite element method, and the walk-on-sphere method using Riesz definition [13]. However, most of these schemes are for structured domains in low dimensions and become very expensive to implement for unstructured domains in high dimensions [14]. Efficient and accurate numerical solvers for the higher dimensional fractional Laplacian are still scarce, especially on a geometrically complex higher dimensional domain, due to the complicated integral expression and nonlocal property of most existing definitions of the fractional Laplacian [12]. Unlike the Laplacian of integer order, the nonlocal property of the fractional Laplacian leads to the dense discretization matrix, even with a local discretization technique such as the finite difference method [15–17], the finite element method [18,19], and the spectral or spectral element method. The nonlocality increases the computational cost in forming the discretized matrix of the operator. For example, with the finite element method, the stiffness matrix assembly requires two elemental loops, but it needs only one loop for the integer order case [18]. Fast solvers for fractional Laplacian using numerical techniques with finite difference methods, i.e., the alternating-direction implicit (ADI) [20–22] are effective for structured domains but difficult to apply to general irregular domains [23]. Consequently, the fractional Laplacian solver development in higher dimensions is left far behind of what is needed for realistic problems of anomalous transport across disciplines, i.e. from geophysical to biomedical problems. The purpose of this paper is to develop a scalable, accurate, and efficient 3D parallel solver for computing the spectral fractional Laplacian on geometrically complex domains.

To overcome the numerical challenges of the fractional Laplacian computation, we used the definition of the fractional Laplacian $(-\triangle)^{\frac{\alpha}{2}}$ of order α through the eigenfunction expansion on a finite domain as proposed by [24–26] instead of using the integral expression. In this way, we avoid the two elemental loops for the stiffness matrix assembly. In particular, the eigenfunction definition can be easily generalized from one dimension to higher dimensions. In space, we employ a spectral/hp finite element method [27] to discretize the fractional Laplacian, which combines the high accuracy of spectral methods and the geometric flexibility of finite element methods. We also implement nonhomogeneous boundary conditions [28] into our solver. A comprehensive comparison of different definitions of the fractional Laplacian in [13] shows that the spectral definition can handle general boundary conditions.

The rest of the paper is organized as follows. Section 2 discusses the definition of the fractional Laplacian by the eigenfunction expansion, the method we use to solve for the associated eigenvalue problem, the implementation of the algorithm, and the scalability of our solver. Section 3 demonstrates the accuracy of our algorithm by solving the fractional diffusion equation and comparing the computational results with the contrived solutions on 2D/3D structured and unstructured domains. Furthermore, we compare the performance of C_0 and C_1 bases for the associated eigenvalue problem. Section 4 demonstrates the geometric flexibility of our method by solving the fractional diffusion equation on a hand-shaped domain discretized with 3D hexahedra and homogeneous boundary conditions and a 2D domain constructed from the Hanford site discretized with quadrilaterals and nonhomogeneous boundary conditions. We conclude in Section 5 with a short summary. Appendix A includes a discussion about the accuracy of the associated eigenvalue problem based on the spectral element method, and Appendix B includes the simulation results of the surface quasi-geostrophic equation. Appendix C includes a brief description of our solver Nektarpp_EigenMM.

2. Definition and computation of the fractional Laplacian through eigenfunction expansion

We use the definition of the fractional Laplacian through its eigenfunction expansion, or equivalently, the spectral decomposition. The idea was first proposed by Ilić [25], who approximated the fractional Laplacian by raising the matrix representation of the Laplacian (-$\triangle$) to the fractional index $\alpha/2$, where $\alpha \in (0,2]$. We start with one dimension and then make generalizations to higher dimensions; see more details in [29].

2.1. The fractional Laplacian with zero Dirichlet boundary conditions

We consider the 1D Laplacian eigenvalue problem,

$$-\triangle u-\lambda u=0, x\in\Omega, \tag{1}$$

where $u|_{\partial\Omega}=0$, or $\frac{\partial u}{\partial n}|_{\partial\Omega}=0$. $\Omega\in R^{1,2,3}$ is bounded.

We start from 1D and use the spectral element method to discretize Eq. (1). Let x denote global coordinates and ξ denote local coordinates in one standard element $\Omega_{st}=\{-1\le\xi\le1\}$. Let N1 be the expansion order in one standard local element; N the total number of global modes, or the number of degrees of freedom for the system; and α the order of the fractional Laplacian. Consider the polynomial space of the interior modes on Ω_{st} defined as $V_I=\{f(\xi)\in P_{N_1}: f(-1)=f(1)=0\}$, where P_{N_1} represents the set of polynomials of degree at most N1. We start with an arbitrary basis $\phi_p(\xi)=(1-\xi^2)\xi^{p-1}(p=1,\dots,N1-1)$ in V_I and form the elemental interior mass and stiffness matrix, denoted by $M^e=[M^e_{pq}]_{(N1-1)\times(N1-1)}$ with $M^e_{pq}=\int_{-1}^{1}\psi_p\psi_q d\xi$ and $K^e=[K^e_{pq}]_{(N1-1)\times(N1-1)}$ with $K^e_{pq}=\int_{-1}^{1}\frac{d\psi_p}{d\xi}\frac{d\psi_q}{d\xi}d\xi$, respectively. The global mass matrix M and global stiffness matrix K can be assembled from local M^e and K^e. The unknown u(x) can be expressed as $u(x)=\sum_{i=1}^{i=N}\hat{u_g}\phi_i(x)$, where $\phi_i(x)s$ are the global modes. Thus, Eq. (1) can be discretized as $K\hat{u_g}-\lambda M\hat{u_g}=0$, from which we can solve for the approximations of the eigenpairs (λ_i, ϕ_i) of the continuous eigenvalue problem Eq. (1). After getting the eigenmodes ϕ_i, the weighted Gram–Schmidt method is applied to make the eigenvectors orthogonal to each other, i.e., $\tilde{\phi_i}(\xi)=\phi_i(\xi)-\frac{(\phi_i,\phi_i)}{(\phi_p,\phi_p)}\phi_p(\xi)$, where $1\le p\le i$ and $(,)$ is the L_2 inner product. The resulting new global modes $\tilde{\phi_i}$ are orthogonalized to each other such that $\int_\Omega\phi_p(x)\phi_q(x)dx=1$ if $(p=q)$ and 0 if $(p\neq q)$. The fractional Laplacian operator can be constructed as $(-\triangle)^{\frac{\alpha}{2}}u(x)=\sum_{i=1}^{i=N}c_i\lambda_i^{\frac{\alpha}{2}}\phi_i(x)$ [25,29,30], if we approximate the standard Laplacian operator by $(-\triangle)u(x)=\sum_{i=1}^{i=N}\lambda_i c_i\phi_i(x)$. In 2D and 3D, we solve the associated eigenvalue problems, get the corresponding eigenvalues and eigenvectors, and construct the fractional Laplacian. Note that the mass and stiffness matrices computed from the global eigenfunctions are both diagonal, with the mass matrix an identity matrix and with the diagonal elements of the stiffness matrix the eigenvalues.

2.2. The fractional Laplacian with nonzero Dirichlet boundary conditions

We use the following definition of the fractional Laplacian with a nonzero Dirichlet boundary in [28], i.e., $(-\triangle_D)^{\frac{\alpha}{2}}u:=\sum_{k=1}^{\infty}(\lambda_k^{\alpha/2}\int_\Omega u\phi_k+\lambda_k^{\alpha/2-1}\int_{\partial\Omega}u\partial_\nu\phi_k)\phi_k$. The fractional diffusion equation $(-\triangle)^{\frac{\alpha}{2}}u=f$, given $u|_{\partial\Omega}=g$, is solved in two parts as $u=w+v$ with w and v satisfying following equations:

$$(-\triangle)^{\frac{\alpha}{2}}w=f, x\in\Omega, w|_{\partial\Omega}=0, \tag{2}$$
$$(-\triangle)^{\frac{\alpha}{2}}v=0, x\in\Omega, v|_{\partial\Omega}=g.$$

2.3. Implementation of the algorithm

The idea behind our custom algorithm is that each Krylov–Schur solve for the associated discrete Laplace eigenvalue problem $K\hat{u_g} - \lambda M\hat{u_g} = 0$ can be done independently. We developed a partitioning scheme so that we can divide the spectrum into P sub-intervals, each containing an equivalent number of eigenpairs. In order to reduce the overall communication costs, we chose P to be the number of physical machines. Each physical machine (with p MPI tasks each) solves for $\frac{N}{P}$ eigenpairs independently by repeatedly solving systems of the form $K\hat{u_g} - \lambda M\hat{u_g} = 0$ exactly using parallel Cholesky factorization. Our custom scalable eigenvalue solver built on top of PETSc and SLEPc is designed specifically to solve for all eigenpairs. PETSc [31] is a scalable linear algebra library that handles the storage of the discrete system in memory and that efficiently applies matrix operations such as matrix–vector multiplications. SLEPc [32] is a library for solving eigenvalue problems that is built around PETSc. From SLEPc, we use the Krylov–Schur iterative algorithm for solving for a given number of eigenpairs in specifically targeted portions of the spectrum. More detail about this algorithm and more performance results can be found in [33].

The dominating time complexity for our approach is $\tau \approx \frac{N}{P}T(k,N,p) + n_a(T(k,N,p)+P)$, where N is the number of degrees of freedom for the system, k is a measure of sparsity for K and M, P is the number of evaluators, and p is the number of processors per evaluator. Then $T(k,N,p)$ is the time it takes to do a single parallel Cholesky factorization of a matrix $K + \lambda M$, which is $O(N)$. The first term $\frac{N}{P}T(k,N,p)$ is the time it takes for a single evaluator to solve for $\frac{N}{P}$ eigenpairs, and the second term $n_a(T(k,N,p)+P)$ is the time it takes to partition the total interval into P equally loaded subintervals, where the constant n_a, which can be specified by users, is the number of iterations it takes to partition the interval.

For the scalability test, we ran scaling experiments on the University of Utah Center for High Performance Computing's (CHPC) Kingspeak cluster and the Texas Advanced Computing Center's (TACC) Frontera cluster; the configurations of the computing nodes of these clusters can be found on the websites [34] and [35], respectively. Fig. 1 shows the complex geometry we used for these tests, (a) for the Hanford site and (b) for an aorta. The Hanford site is a decommissioned nuclear production complex on the Columbia River in Benton County in the U.S. state of Washington, which was home to the first full-scale plutonium production reaction in the world in 1943 [36]. The aorta is the main artery that carries blood away from the heart to the rest of the body [37]. Fig. 2 shows the parallel efficiency of the solve phase with respect to the number of processes. We ran scaling experiments using different basis function orders to determine if we had achieved ideal scalability in all cases. For 2D, we used the Hanford site mesh with linear and quadratic basis functions; for 3D, we used a simple cube with linear and quadratic basis functions and the complex aorta mesh with quadratic basis functions. In all of the cases for which we ran the experiment, the speedup was greater than the ideal linear speedup due to memory efficiency of solving smaller independent problems. As the number of processors increased, the speedup eventually started to dip below the ideal but this did not occur until the total absolute run-time was on the order of 10 s.

We also tested our code on the complex 3D mesh of an aorta (see Fig. 1) for a variety of problem sizes to see the least required computing resources to compute the full set of eigenpairs in a reasonable amount of time. Specifically, we fixed the problem size and then increased the number of machines P until we got a total run-time less than 10 min. Table 1 shows that we achieved very good weak scaling results.

Table 1
Weak scaling for 3D aorta mesh using order $p = 1, 2, 3$ basis functions.

Number of DoFs	24k	165k	528k
Number of nodes	1	64	512
Number of evaluators	2	128	512
Threads per evaluator	28	28	56
Total processes	56	3584	28,672
Total elapsed time	4 m 59 s	4 m 23 s	9 m 33 s
(Compute spectral radius)	3 s	6 s	20 s
(Partition interval)	3 s	25 s	1 m 40 s
(Solve)	4 m 44 s	3 m 44 s	7 m 14 s
(Post-processing)	8 s	8 s	18 s

3. Code validation and benchmarks

In this section, we constructed the numerical solution for the fractional diffusion equation in 2D and 3D with eigenmodes being computed from the algorithm in Section 2 and compare the numerical results with the solutions based on analytical eigenvalues. The geometries we tested are a 2D square, a 3D cube, and a 3D cylinder. We also investigated the performance of using C_1 basis instead of C_0 basis used in the spectral element or finite element method. The goal was to study and demonstrate the accuracy of the proposed fractional Laplacian computation.

The fractional diffusion equation we solve is

$$\begin{aligned} \partial_t u = -\mu(-\triangle)^{\frac{\alpha}{2}} u, x \in \Omega, \\ u(x,t)|_{\partial\Omega} = 0\ , \\ u(x,0) = u_0(x), x \in \Omega, \end{aligned} \tag{3}$$

where $\Omega = R^d$ is a bounded domain with d = 2, 3. Using the eigenmodes ϕ_i, we can expand $u(x,t)$ to $u(x,t) = \sum_i^\infty c_i(t)\phi_i$, where $c_i = (\mathrm{u}, \phi_i)$. Then from Eq. (1), we have $\sum_{i=0}^\infty \partial_t c_i(t)\phi_i = -\mu \sum_{i=0}^\infty c_i(t)(\lambda_i)^{\frac{\alpha}{2}}\phi_i$. If we express the initial condition as $u_0(x) = \sum_{i=1}^N c_i(0)\phi_i$, where $c_i(0) = (u_0, \phi_i)$, the numerical solution for Eq. (3) can be given by $u_N(x,t) = \sum_{i=1}^N e^{-\mu\lambda_i^{\frac{\alpha}{2}} t} c_i(0)\phi_i$.

3.1. A 2D square with homogeneous boundary conditions

On a 2D square, the analytical eigenpairs (λ_i, ϕ_i) for Eq. (1) are

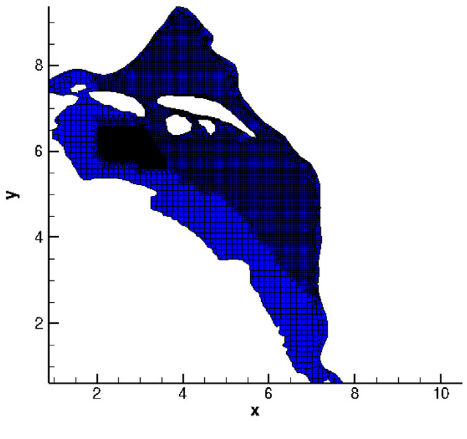
$$\begin{aligned} \lambda_i &= n^2\pi^2 + m^2\pi^2 \\ \phi_i &= \sin(n\pi x)\sin(m\pi y), \end{aligned} \tag{4}$$

where n, m = 1, 2, …N.

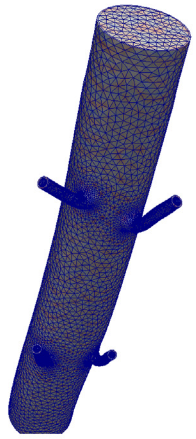
The test we consider is to solve Eq. (3) with the initial condition $u(x,y) = \sin(\pi x)\sin(\pi y)$ and $\mu = 1$. The analytical solution for this problem is $u = \sum_{i,j=1}^\infty \exp(-\mu\lambda_i t)\sin(i\pi x)\sin(j\pi y)$. The numerical solution to this problem is obtained on a domain $\Omega = \{(x,y) : -1 \le x \le 1, -1 \le y \le 1\}$ discretized with two quadrilaterals, and the computation time is T = 0.4. Fig. 3(a) shows the exponential convergence results for fractional orders $\alpha = 1.2, 1.5, 1.8$, and 2.0 with a p-type refinement. The solutions from our computation converge exponentially in terms of the element order. Moreover, the convergence rate decreases as the order of fractional order α decreases. However, for all α, machine precision is achieved at the element order around 20, which demonstrates the accuracy of our algorithm in a 2D square.

3.2. A 3D cube and cylinder with homogeneous boundary conditions

The test we consider is to solve Eq. (3) on a cube $\Omega = \{(x,y) : -1 \le x \le 1, -1 \le y \le 1, -1 \le z \le 1\}$ discretized with two hexahedrons and on a cylinder $\Omega = \{(r,z) : 0 \le r \le 1, -1 \le$

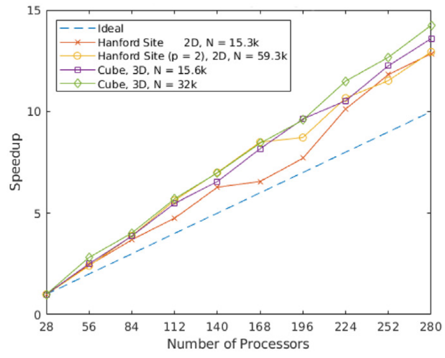


(a)

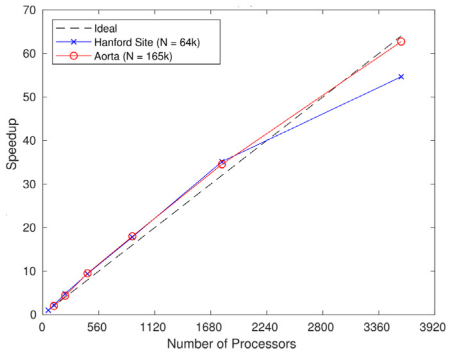


(b)

Fig. 1. Complex geometry for scalability tests. (a) 2D Hanford site; (b) 3D aorta.



(a) Kinspeak cluster



(b) Frontera cluster

Fig. 2. Parallel speed-up of solver Nektarpp_EigenMM's full eigenbasis solve as a function of the number of nodes, with a variety of matrix sizes. Results are from (a) CHPC's Kingspeak cluster and (b) TACC's Frontera cluster, respectively. N is the number of degrees of freedom, and p is the order of the polynomial basis. Speed-up is calculated as T_{28}/Tn where n is the number of processors used. Load balancing is achieved by evenly dividing the spectrum using our spectrum slicing approach detailed in [33].

$z \leq 1\}$ discretized with five hexahedrons using curvilinear edges. On a 3D cube, the analytical eigenpairs (λ_i, ϕ_i) for Eq. (1) are

$$\lambda_i = n^2\pi^2 + m^2\pi^2 + k^2\pi^2 \qquad (5)$$
$$\phi_i = \sin(n\pi x)\sin(m\pi y)\sin(k\pi z),$$

where n, m, k = 1, 2, …N. On a 3D cylinder, we assume the initial condition is given as $u = 20.0\sum_{i=1}^{100}\frac{J_0(\lambda_i r)}{(\lambda_i J_1(\lambda_i R))}$, where R = 1.0 is the radius of the cylinder, J_0, and J_1 are the Bessel functions of the first kind of order zero and one. The cylinder has homogeneous boundaries on the cylinder wall and periodic boundaries in the z direction. The analytical solution for this problem is $u = 20.0\sum_{i=1}^{\infty}\exp(-\mu\lambda_i^2 t)\frac{J_0(\lambda_i r)}{(\lambda_i J_1(\lambda_i R))}$, where eigenvalues λ_i are the roots of the equation $J_0(\lambda R) = 0$ [38].

Figs. 3(b)–(c) show the exponential convergence results for the numerical solutions as a function of the element order in a 3D cube and a 3D cylinder, respectively. Similar to the previous 2D square case, smaller α leads to lower solution accuracy. The time dependent diffusion problem Eq. (3) can be used to model a sudden cooling of a hot cylinder if we take u as the temperature variable. Figs. 3(d) and (e) show the field plot of temperature u on a 3D cylinder taken at computational time T = 0.2 with the fractional order 1.1 and 2.0. The smaller the fractional order is, the slower the cooling process. Fig. 3(f) shows that the solution curve declines as the fractional order α increases, and the variation of the solutions against the fractional order α is continuous. The fractional diffusion equation, as pointed out in [12], can also be used in modeling the transient heat conduction in nanosystems [9], e.g., in bodies where discontinuities such as cracks may emerge, interact and evolve [39], and is more adequate in describing the long-range thermal energy transfer.

3.3. Fractional Laplacian with nonzero Dirichlet boundary conditions

In this subsection, we compute the fractional Laplacian $(-\triangle)^{\frac{\alpha}{2}}$ of $u = \cos(\pi x)\sin(\pi y)$ using the method in [28] on one quadrilateral element [0 1]×[0 1] in 2D with expansion order 20. Fig. 4 shows the field plots for the fractional Laplacian solution $u = \cos(\pi x)\sin(\pi y)$ and the line plots along the line y = 0.46 at different fractional orders α, see also [13].

3.4. Regularity of basis functions across element boundaries

A well known phenomenon, described in [40] is that when using the finite element or spectral element method to compute eigenvalues and eigenvectors, the eigenvalues at the low end of the spectrum are represented accurately, but at the high end of the spectrum, the error becomes excessive. For the C_0 discretization, we use the hp-element framework Nektar++ [41]. Fig. 5 (a) shows the eigenvalue errors for this C_0 basis discretization throughout the spectrum for varying the basis function order for a typical spectral element method in a square domain. By increasing p, the error of the eigenvalues at the high end of the spectrum

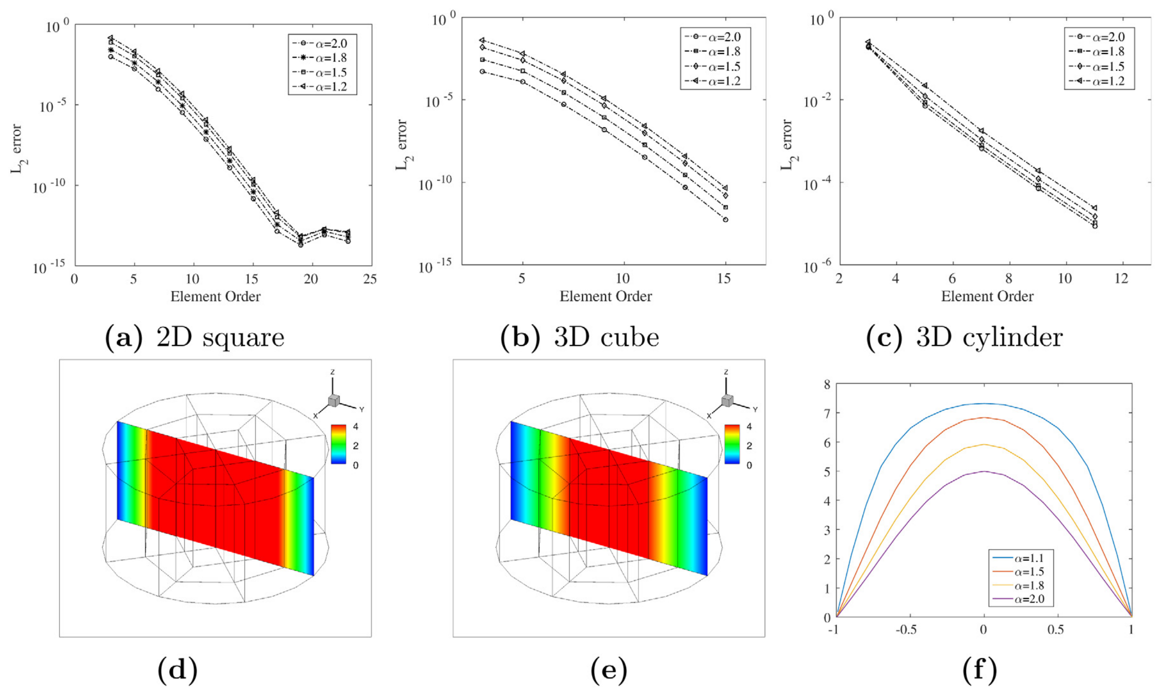


Fig. 3. Results of approximating the solution for the fractional diffusion equation using numerical eigenvalues and eigenvectors. 1st row: L_2 error of constructed solution from numerical eigenvalues and eigenvectors as a function of element order in (a) 2D square; (b) 3D cube; (c) 3D cylinder. 2nd row: snapshots of solution field (u) in a 3D cylinder at T = 0.2 (d) $\alpha = 1.1$; (e) $\alpha = 2.0$; (f) u-field profile along y line at the slice x = 0 and z = 0.5.

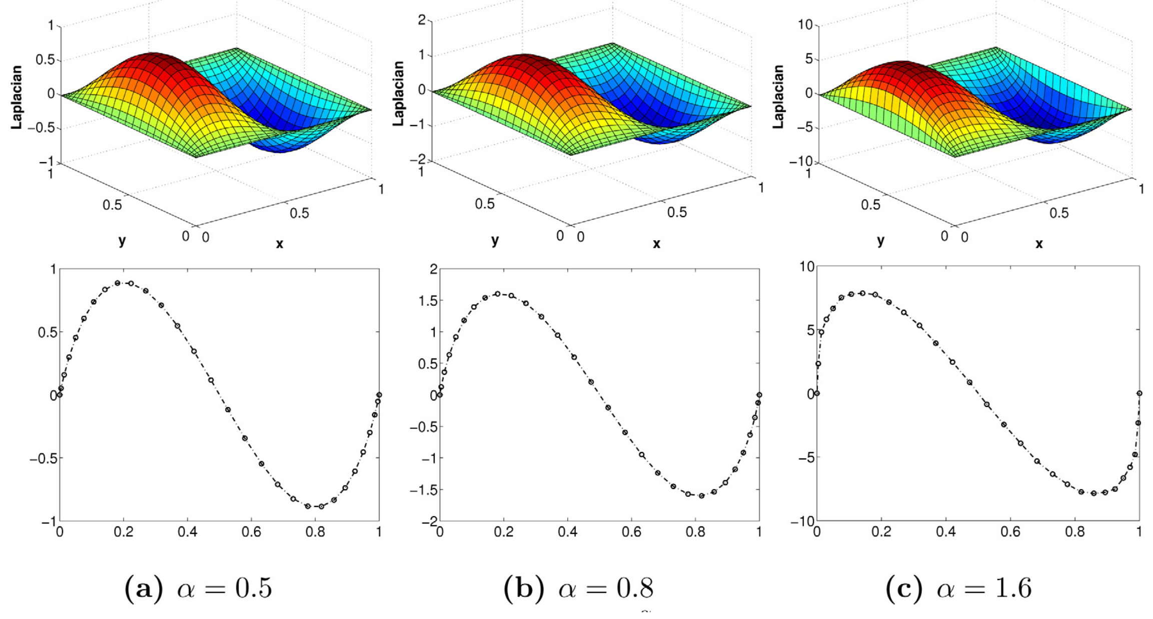


Fig. 4. Computed fractional Laplacian $(-\triangle)^{\frac{\alpha}{2}}$ for function $\mathbf{u} = \cos(\pi\mathbf{x})\sin(\pi\mathbf{y})$. 1st row: field plot; 2nd row: line plot at y = 0.46.

starts to blow up, resulting in a collection of eigenvectors that are not accurate since their corresponding eigenvalues cannot accurately reflect the true eigenvalues. One of the ways to improve the accuracy of eigenpairs is to use an alternative discretization that enforces C_1 regularity at element boundaries. We use a Matlab library geoPDEs [42] to construct the C_1 discrete system, which is then fed into our eigenvalue solver. Fig. 5(b) shows that with the C_1 discretization, the eigenvalue errors stay bounded throughout the entire spectrum with increasing p. Fig. 5(c) shows the eigenvalue errors for a C_0 and C_1 discretization with a roughly equivalent number of degrees of freedom on a log scale. The C_1 basis with higher regularity results in better eigenvalue accuracy at both the high and the low ends of the spectrum. Fig. 6 shows the residual error resulting from projecting the computed eigenvectors into the space of the exact eigenvectors and vice versa. We denote the computed and exact eigenpairs by (θ_k, ϕ_k) and (λ_k, ψ_k), respectively. The projected computed eigenvectors are $\phi_k^{(\psi)} = \sum_{i=1}^{N} \frac{(\psi_i, \psi_i)}{(\phi_k, \psi_i)} \psi_i$ and the projected exact eigenvectors are

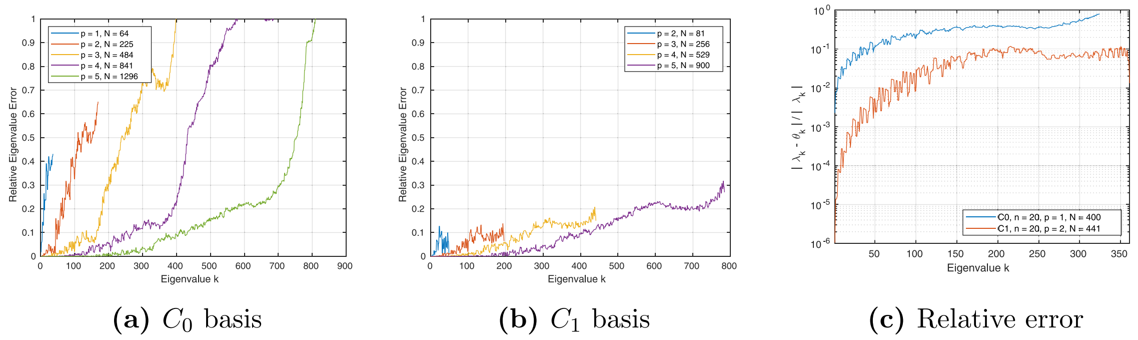


Fig. 5. Computed vs analytical eigenvalue error for a square with homogeneous Dirichlet boundary conditions computed using C_0 (Nektar++) discretizations and C_1 (geopdes). (a) C_0 basis (b) C_1 basis (c) relative error. N is the number of degrees of freedom and p the order of the polynomial basis.

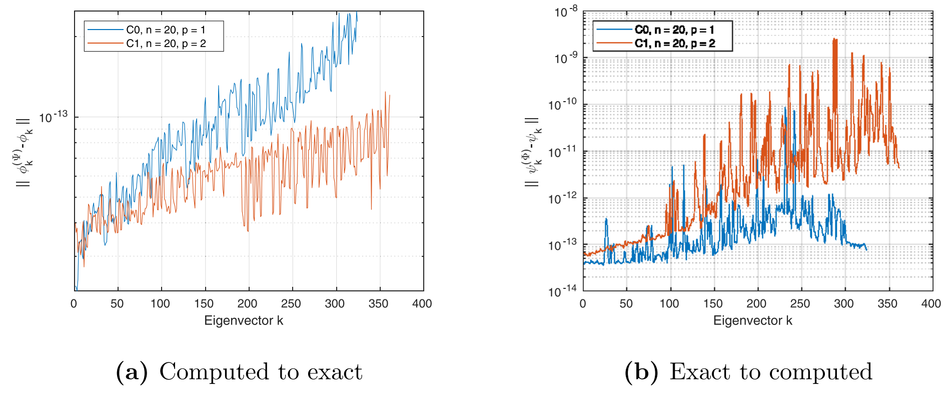


Fig. 6. Residual error resulting from projecting the computed eigenvectors into the space of exact eigenvectors (a) and vice versa (b). n is the number of elements and p the order of the polynomial basis.

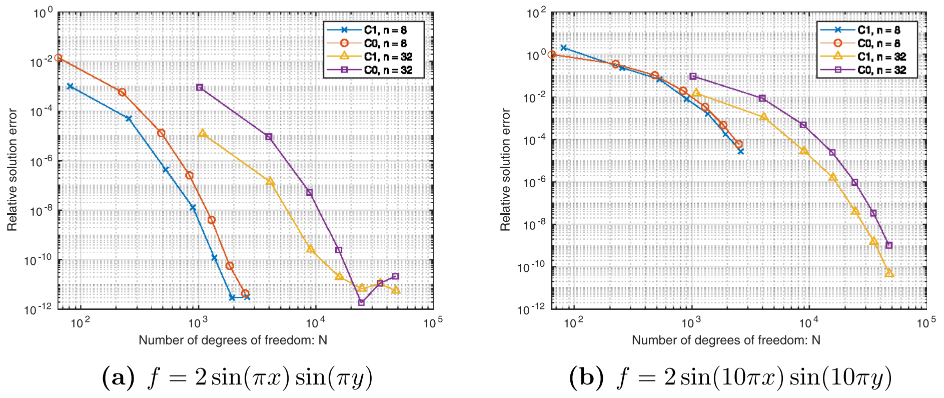


Fig. 7. L_2 error using C_0 and C_1 basis for solving the fractional Poisson equation on a square with homogeneous Dirichlet boundary conditions at fractional order $\alpha = 1$. (a) $f = 2\sin(\pi x)\sin(\pi y)$ (b) $f = 2\sin(10\pi x)\sin(10\pi y)$. f is the right-hand side of the Poisson equation and n the number of elements.

$\psi_k^{(\psi)} = \sum_{i=1}^{N} \frac{(\phi_i,\phi_i)}{(\psi_k,\phi_i)}\phi_i$. The residual errors are defined as $\|\phi_k^{\psi} - \phi_k\|$ and $\|\psi_k^{\phi} - \psi_k\|$, respectively.

In order to measure the performance of the C_0 and C_1 eigenvectors in solving the fractional diffusion equation, we ran the solver using both the C_1 and C_0 discretizations and compared the solution accuracy in Fig. 7. For both discretizations, the fractional Poisson problem is solved for forcing functions $f = 2\sin(\pi x)\sin(\pi y)$ and $f = 2\sin(10\pi x)\sin(\pi y)$. These forcing functions are eigenfunctions for a unit square with homogeneous Dirichlet boundary conditions and have exact solutions. Fig. 7 shows that the C_1 discretization is better. The overall convergence rate with respect to the number of degrees of freedom (and by extension, the order of the basis elements) remains the same, but the C_1 solution error has essentially been shifted by a constant. These results suggest that by increasing the regularity of the element boundaries during discretization, the computed eigenbasis attains more accuracy for the same number of degrees of freedom. A discretization with an arbitrarily high-order element boundary regularity has been proposed in [43]. Importantly, if a C_0 code and a C_1 code are given with the same number of mesh elements and the same basis order p, they result in a different total number of degrees of freedom as below:

$$N(C_0) = (1 + p(n-1))^d,$$
$$N(C_1) = (2 + (p-1)(n-1))^d,$$

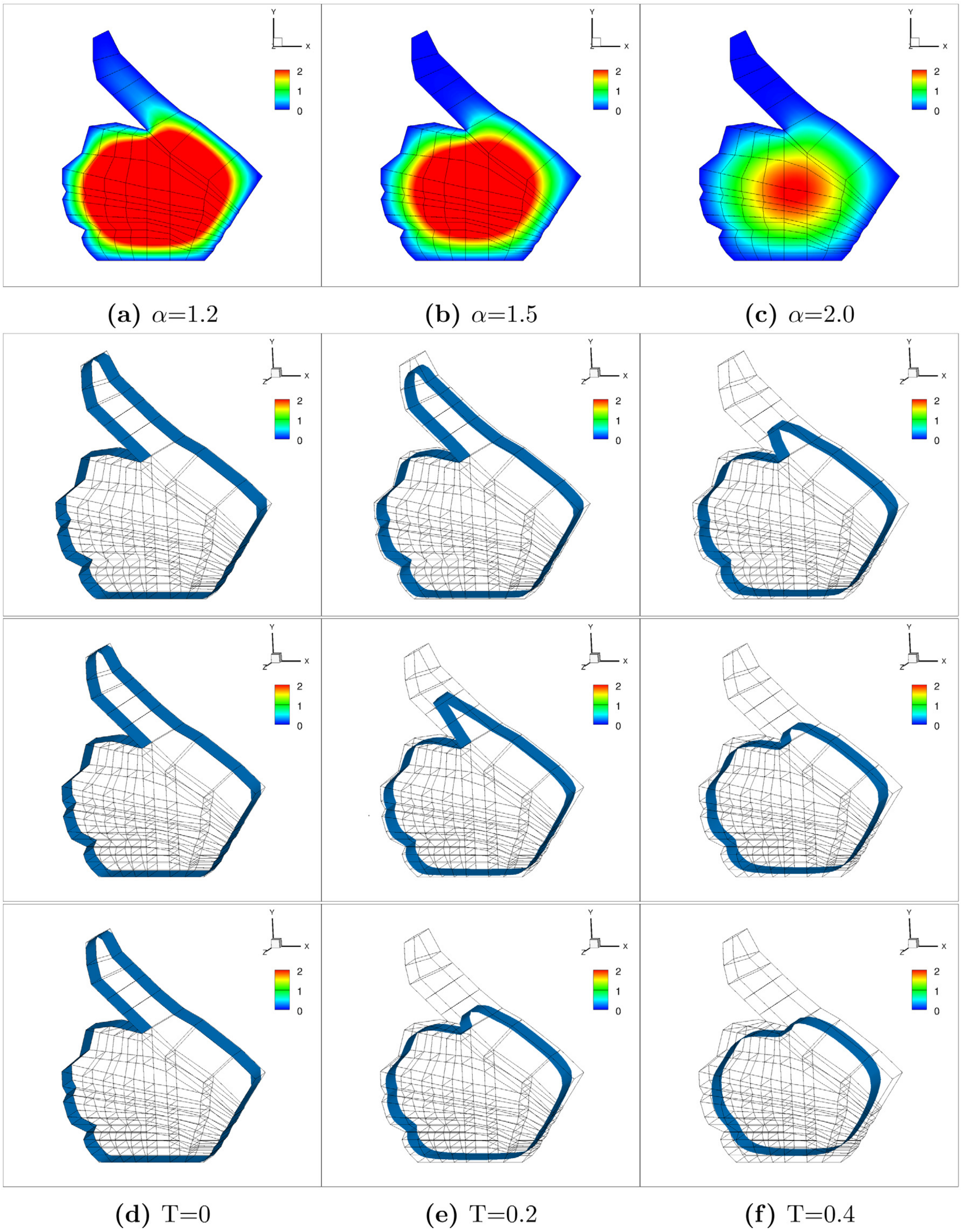


Fig. 8. 3D hand domain with zero Dirichlet boundary. 1st row: snapshots of fractional heat conduction at T = 0.4 with (a) $\alpha = 1.2$ (b) $\alpha = 1.5$ (c) $\alpha = 2.0$. Temperature isocontour 0.3 at T = 0, 0.2, 0.4. 2nd row: $\alpha = 1.2$; 3rd row $\alpha = 1.5$; 4th row $\alpha = 2.0$.

where p, n, d are the expansion order, the number of elements, and spatial dimensions, respectively. p = 1, 2 indicate linear and quadratic elements, respectively. Since the expansion order for these two methods have different meanings, we match the total degrees of freedom N to compare errors instead of matching the basis function order p. Since the amount of work that is required to compute the full eigenbasis depends directly on N, this comparison captures the work–precision tradeoff. Also note here the number of quadrature points used in the tests is different for C_0 and C_1 basis.

4. Capability demonstration: numerical simulation of fractional diffusion equation on 2D/3D complex geometries

In this section, we solve the fractional heat conduction equation on complex geometry domains to show the geometric flexibility of our algorithm for computing the fractional Laplacian.

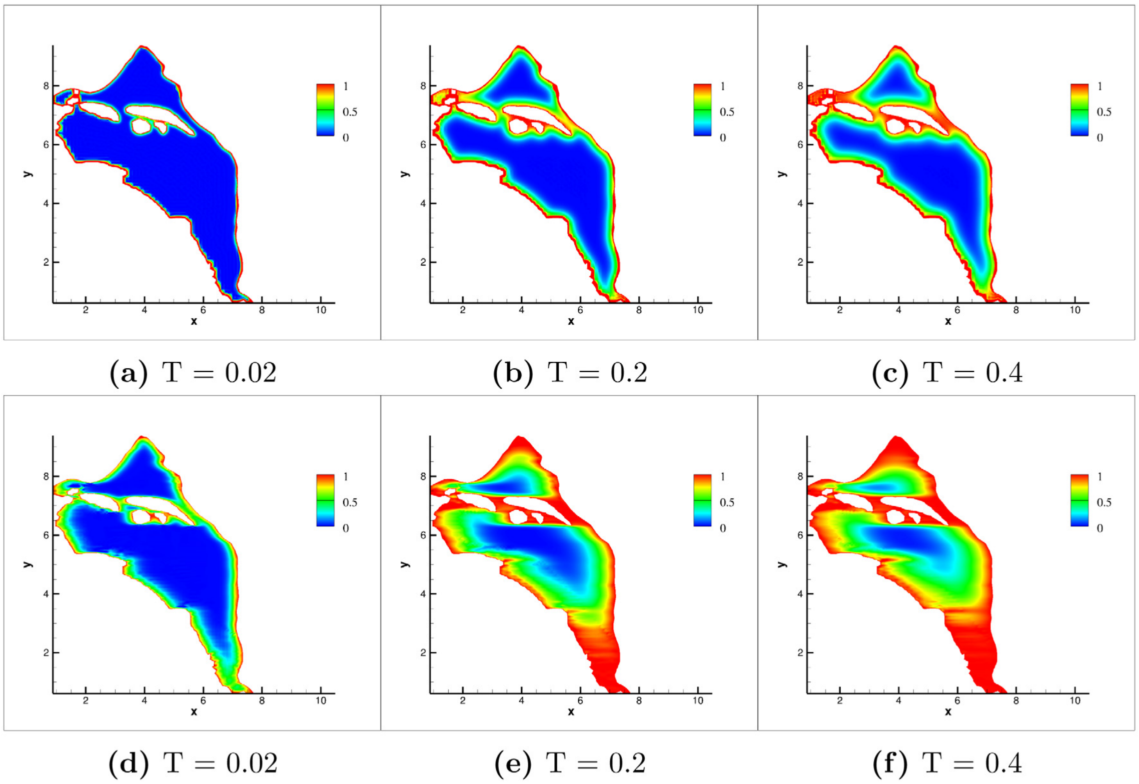


Fig. 9. 2D Hanford site domain with nonzero boundaries. Snapshots of solutions for fractional diffusion problem at time T = 0.02, 0.2, and 0.4. 1st row $\alpha = 1.0$; 2nd row $\alpha = 2.0$.

4.1. A 3D complex domain with zero Dirichlet boundary conditions

We solve the fractional heat conduction equation $\frac{\partial u}{\partial t} = -\mu(-\triangle)^{\alpha/2}u$ by 99 hexahedron elements with element order 5 and time step size 1e-3. The initial and boundary conditions are the same as for the fractional heat conduction equation on a 3D cylinder in Section 3. The resulting matrix of the associated eigenvalue problem is 7405 × 7405. The first row of Fig. 8 shows the contour plot for temperature at computation time T = 0.4 with different fractional order α. This row demonstrates that the smaller α, the slower the cooling. The temperature isocontour plots in the 2nd to 4th rows of the Fig. 8 show that the cooling is faster in the thumb area with the fractional order 2.0, than the fractional order 1.2 and 1.5. For the fractional order $\alpha < 2.0$, the heat penetrates the thumb area faster than the classical diffusion corresponding to $\alpha = 2.0$, which indicates the elongated tails of the temperature distribution, a striking characteristic of the anomalous diffusion [39]. Another interesting characteristic is the accelerating fronts. With the decrease of the fractional order, the movement of the front of the isocontour is accelerates.

4.2. Hanford site geometry with nonzero Dirichlet boundary

We solve the fractional diffusion equation $\partial_t u = -\mu(-\triangle)^{\frac{\alpha}{2}}u$ with $\mu = 0.01$ with nonhomogeneous Dirichlet boundary conditions on a complex geometry domain constructed from the Hanford site discretized with 14,814 quadrilateral elements. For the associated eigenvalue problem, 57,849 eigenvalues are computed, and the time cost is 1191.2s with 1024 CPUs. Fig. 9 shows the snapshots of the concentration u-field at time T = 0.02, 0.2, and 0.4 with fractional order $\alpha = 1.0$ and $\alpha = 2.0$.

5. Summary

An efficient and accurate 3D parallel solver Nektarpp_EigenMM for computing the spectral fractional Laplacian on complex geometries with homogeneous and nonhomogeneous boundaries through eigenfunction expansion is developed and implemented with the spectral element method. We demonstrate the scalability by testing the associated eigenvalue problem solver on different numbers of CPUs and geometries. We demonstrate the accuracy of this algorithm by solving the fractional diffusion equation and comparing the results with the analytic solutions. We discuss the possibilities of improving the solver by using higher order regularity at element boundaries during discretization. We apply this solver to the fractional diffusion equation on complex geometry domains, which shows the capability of our solver in handling anomalous diffusion problems modeled by the fractional Laplacian. Nektarpp_EigenMM is highly customizable and freely available at Github https://github.com/paralab/Nektarpp_EigenMM.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We gratefully acknowledge the support of the OSD/ARO/MURI on "Fractional PDEs for Conservation Laws and Beyond: Theory, Numerics and Applications" (W911NF-15-1-0562). High Performance Computing resources were provided by the Center for Computation and Visualization at Brown University, the Center of High Performance Computing at the University of Utah, and the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation, United States grant number ACI-1548562 [44].

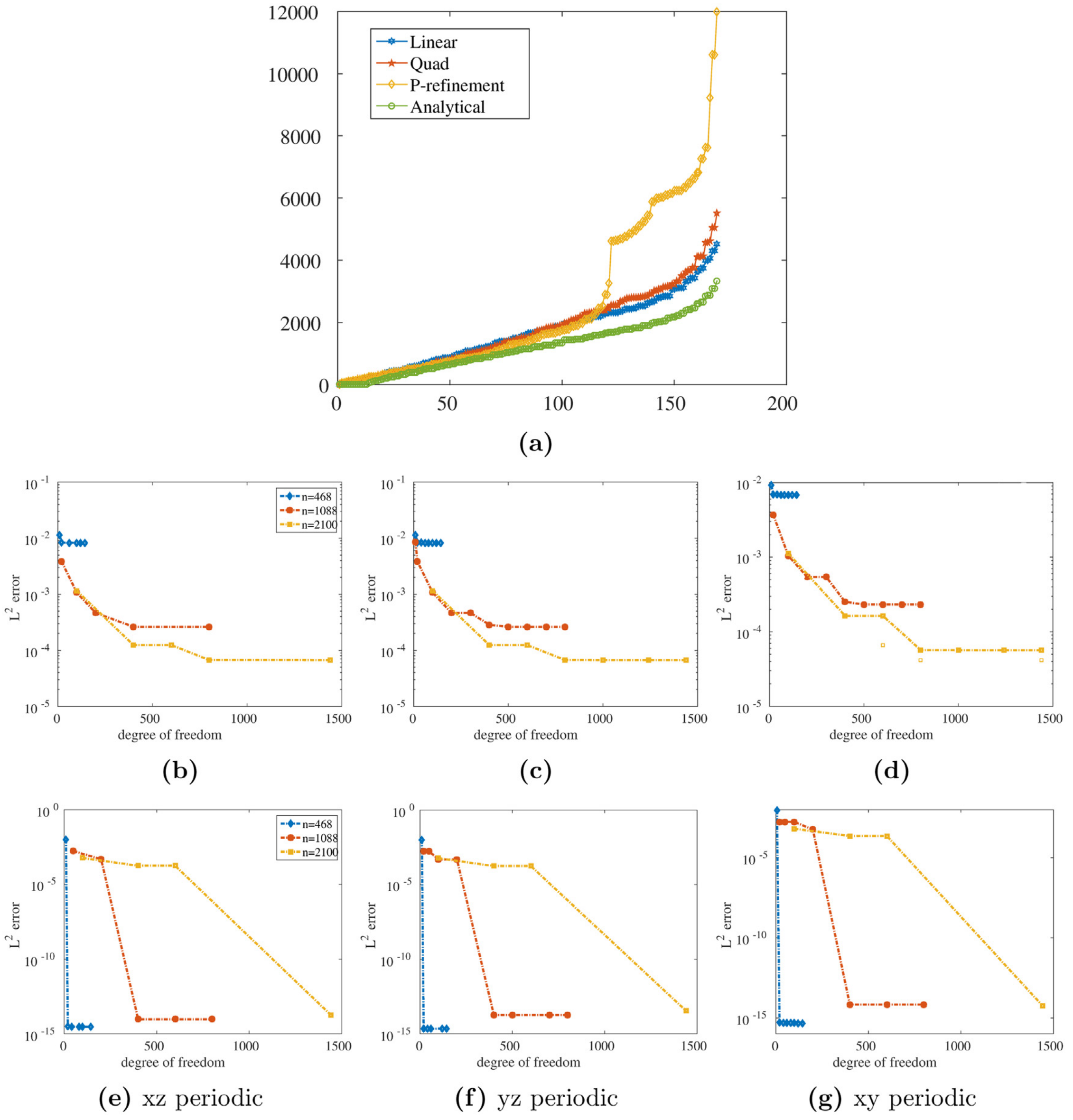


Fig. 10. Eigenvalue distributions: (a) a 2D square domain discretized with quadrilateral elements, using C_0 basis with hp refinement, and L^2-error as a function of degrees of freedom n when solving a Helmholtz equation. (b) and (e) periodic in xz-direction; (c) and (f) periodic in yz-direction; (d) and (g) periodic in xy-direction. n is the number of degrees of freedom.

Appendix A. Numerical issues: accuracy of the numerical eigenvalues and eigenfunctions and the completeness of the eigenspace with C_0 basis

As mentioned in Section 3.4, the computed numerical eigenvalues and eigenvectors from finite element method and spectral element method (C_0 basis) have been long known to deviate from the analytical ones [45,46], as shown in Fig. 10, if ordered in an increasing fashion. C_1 basis is an alternative to improve the accuracy of solving the fractional diffusion equation, see Section 3.4. In this section, we show that although some of the eigenvalues and eigenvectors are not accurate, we still need all of them to ensure the completeness of the space of C_0 basis functions and the convergence when solving fractional PDEs.

Completeness of the eigenspace

To demonstrate that all the eigenvalues and eigenvectors are needed to compute the fractional Laplacian to keep the high accuracy when solving a Helmholtz equation. Here, we solve $u - \triangle u = f$ in $\Omega = \{(x, y, z) : -1 \le x \le 1, -1 \le y \le 1, -1 \le z \le 1\}$ with Dirichlet and mixed boundary conditions (Dirichlet and periodic boundaries) using part of the eigenvectors in the full set in the solution space and compare the computation results with the contrived solutions.

We use mixed boundary conditions (Dirichlet and periodic boundaries). Three numerical experiments are conducted. The first set of tests are $u(x, y, z) = (5/(4 + \sin(\pi x)) - 5/4)$ and $u(x, y, z) = (1 - x^2)$ with xz periodic, the second set are $u(x, y, z) = (5/(4 + \sin(\pi y)) - 5/4)$ and $u(x, y, z) = (1 - y^2)$ with yz periodic, and the third are $u(x, y, z) = (5/(4 + \sin(\pi z)) - 5/4)$ and $u(x, y, z) = (1 - z^2)$ with xy periodic. Fig. 10(b) and (e) show the results for the first test, (c) and (f) for the second test, and (d) and (g) for the third test, respectively. We can see that if we use only part of the eigenvectors from the full set to solve the equation, we cannot get good accuracy. These results show that all the eigenvectors, even the erroneous ones, are needed to obtain exponential accuracy in the fractional Laplacian computation.

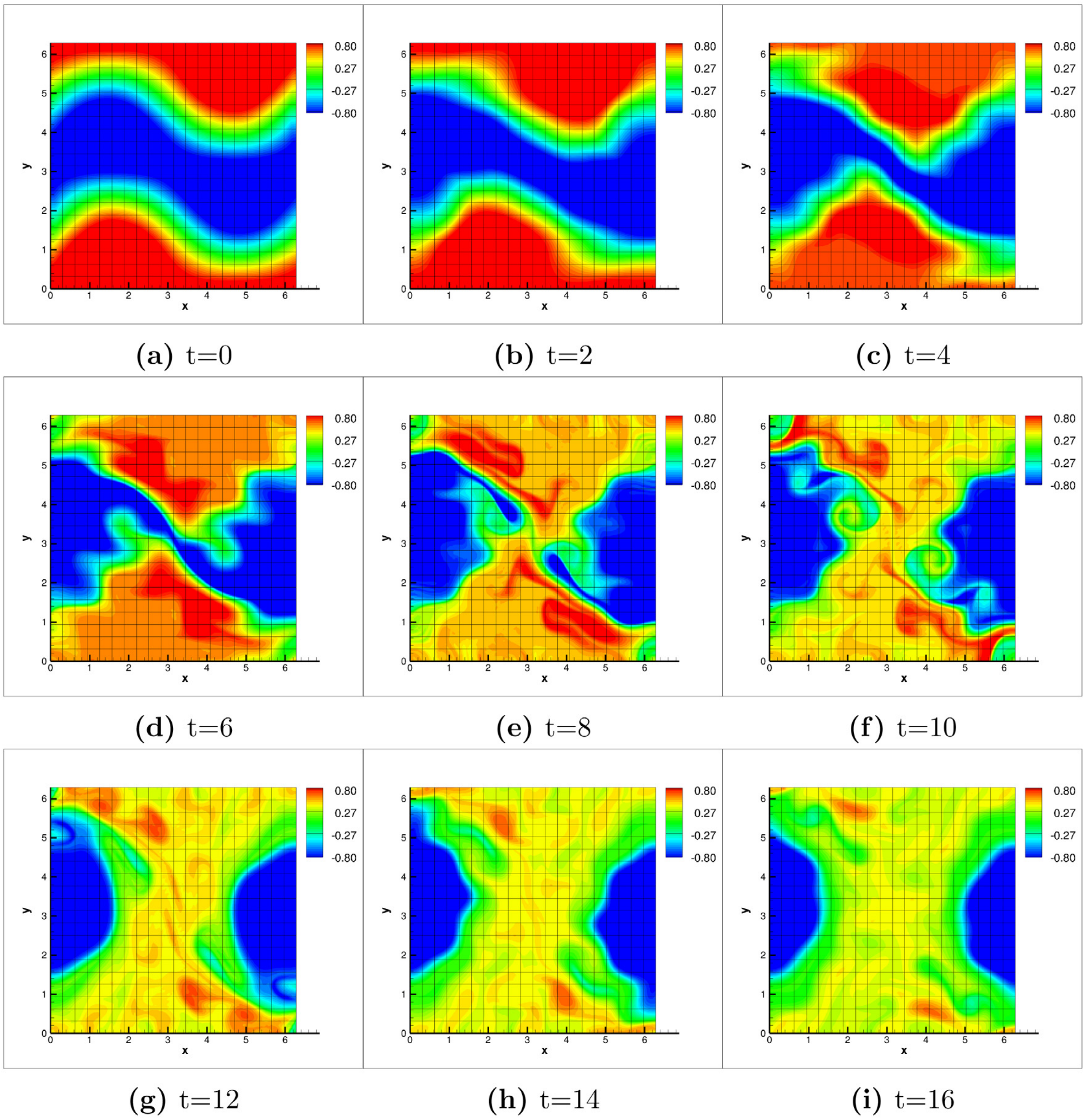


Fig. 11. Temperature field θ solved from the SQG equation on a structured domain. The initial data is given by Eq. (5). Background lines are the gridlines showing the 20 by 20 discretization of the square computational domain.

Appendix B. Numerical simulation of surface quasi-geostrophic equation (SQG)

We demonstrate the capability of our method to solve the SQG equation corresponding to the initial data constructed from the first few eigenmodes proposed in [47]. The SQG equation solves a quantity driven by the gradients of a uniform potential vorticity, which can be modeled by the fractional Laplacian. There is an analogy between the level sets of solutions of the SQG equation and the vortex lines of the 3D Euler equation [47]. The study of the singular front formation of SQG shed lights on the behavior of the 3D Euler equation, which is a prototype equation for studying turbulence behavior and potential meteorological and oceanic applications [48]. A popular existing numerical method for solving SQG is the pseudo-spectral method with the high wavenumber Fourier modes filtered out to avoid the heavy oscillations created by the singularity of the solution of the SQG equations [49]. However, the pseudo-spectral scheme cannot handle an irregular domain or nonperiodic initial condition. We show that our method can solve the SQG equation involving the fractional Laplacian calculation. For computations that simulate large timescales, we adopt a fractional spectral viscosity vanishing technique (fSVV) [50] to stabilize the method and minimize the oscillations. We conduct the test with periodic boundary conditions on a 2D square $[0,2\pi]\times[0,2\pi]$. The initial condition is $\theta = \sin(x)\sin(y) + \cos(y)$.

The SQG equation is given as follows [49]:

$$\begin{aligned} &\theta_t + u\cdot\nabla\theta + \kappa(-\triangle)^{\frac{\alpha}{2}}\theta = 0 \\ &\quad (u_1, u_2) = (-\partial_{x_2}\psi, \partial_{x_1}\psi) \\ &\qquad (-\triangle)^{1/2}\psi = -\theta\ , \end{aligned} \tag{6}$$

where $\kappa \gg 0$ and $\alpha > 0$ are parameters, $\theta = \theta(x_1, x_2, t)$ is a scalar representing the potential temperature, and $u = (u_1, u_2)$ is the velocity field determined from θ by the stream function via the auxiliary relations expressed in Eq. (6).

The SQG equations with fractional SVV technique [50] are as follows:

$$\begin{aligned} &\frac{\partial\theta_N}{\partial t} + (u_N\cdot\nabla\theta_N) + \kappa(-\triangle)^{\frac{\alpha}{2}}\theta_N + \xi_N S_N^{\beta}(\theta_N) + \xi_{N_1} S_N^{\beta_1}(\theta_N) = 0 \\ &\qquad (u_1, u_2) = (-\partial_{x_2}\psi, \partial_{x_1}\psi) \\ &\qquad\quad (-\triangle)^{1/2}\psi = -\theta\ . \end{aligned} \tag{7}$$

Table 2
Parameters used for SQG.

Parameters	κ	α	N	β	MN	ξ_N	β_1	MN1	ξ_{N_1}
	0	0.4	28	0.75	40	0.04	1.0	1540	0.04

Free parameter selection for SQG in Eq. (7) with an artificial viscosity

MN, MN1, ξ_N, ξ_{N_1}, β and β_1 are free parameters to be determined as long as they satisfy the spectral accuracy restriction. MN and MN1 are the wavenumbers we choose to add an artificial viscosity term, MN2 are the total number of modes and ξ_N and ξ_{N_1} are a fixed power of the vanishing grid size in order to yield an accurate approximation. In [51], for SVV, MN $\sim 5\sqrt{MN2}$ and $\xi_N \sim \frac{1}{MN2}$ when using the standard Laplacian. If we use the fractional Laplacian and fractional SVV [50], we can estimate the appropriate choices of the parameters MN, MN1, ξ_N, ξ_{N_1}, β and β_1, following [51].

Table 2 presents the parameters used in numerical simulations. Fig. 11 shows the results for the inviscid equation with $\kappa = 0$. Computation for the viscous equation with $\kappa \neq 0$ has numerical concerns beyond the scope of this paper and is omitted. Notice that with $\kappa = 0$, a fractional Laplacian still appears in the third equation of Eq. (6). Fig. 11(a)–(e) show snapshots from time 0∼8s, where a sharp front is developing along the hyperbolic saddle, and Fig. 11(f)–(i) show that from 8∼16 s, a chaotic mixing appears.

Appendix C

Example simulations are included in the program source code package, which can be downloaded from CPC or the Github repository that can be found in the program summary. Included are 2D and 3D examples of solving fractional diffusion and Poisson equations and some samples meshes. The sample meshes for 2D are a square with 1681 elements and the Hanford site mesh with 14,272 elements. The sample 3D meshes are a cube with 4913 elements and the aorta mesh with 24,095 elements. Boundary conditions can be set to homogeneous Dirichlet or Neumann inside the mesh files themselves.

The instructions for compilation and execution are detailed in the top-level README.md file. The validations of solving fractional diffusion equation described in Section 3 are also included in the repository on CPC and Github. Example simulations, meshes, and scripts can all be found in the 'exp' folder within the Nektarpp_EigenMM root directory.

References

[1] W. Chen, S. Holm, J. Acoust. Soc. Am. 115 (4) (2004) 1424–1430.
[2] B.E. Treeby, B. Cox, J. Acoust. Soc. Am. 127 (5) (2010) 2741–2748.
[3] T. Zhu, J.M. Harris, Geophysics 79 (3) (2014) T105–T116.
[4] D. Kusnezov, A. Bulgac, G. Do Dang, Phys. Rev. Lett. 82 (6) (1999) 1136.
[5] D. del Castillo-Negrete, B. Carreras, V. Lynch, Phys. Plasmas 11 (8) (2004) 3854–3864.
[6] D. del Castillo-Negrete, B. Carreras, V. Lynch, Phys. Rev. Lett. 94 (6) (2005) 065003.
[7] W. Chen, Chaos 16 (2) (2006) 023126.
[8] B. Berkowitz, H. Scher, S.E. Silliman, Water Resour. Res. 36 (1) (2000) 149–158.
[9] M.S. Mongioví, M. Zingales, Int. J. Heat Mass Transfer 67 (2013) 593–601.
[10] V.E. Tarasov, Modern Phys. Lett. A 21 (20) (2006) 1587–1600.
[11] S. Esmaeili, M. Shamsi, Commun. Nonlinear Sci. Numer. Simul. 16 (9) (2011) 3646–3654.
[12] W. Chen, G. Pang, J. Comput. Phys. 309 (2016) 350–367.
[13] A. Lischke, G. Pang, M. Gulian, F. Song, C. Glusa, X. Zheng, Z. Mao, W. Cai, M.M. Meerschaert, M. Ainsworth, et al., J. Comput. Phys. 404 (2020) 109009.
[14] L.A. Caffarelli, J.-M. Roquejoffre, Y. Sire, J. Eur. Math. Soc. 12 (5) (2010) 1151–1179.
[15] M.M. Meerschaert, H.P. Scheffler, C. Tadjeran, J. Comput. Phys. 211 (1) (2006) 249–261.
[16] Y. Gu, W. Chen, X. He, Comput. Struct. 135 (2014) 73–82.
[17] W. Chen, Z.-J. Fu, C.-S. Chen, Briefs in Applied Sciences and Technology, Springer, New York, NY, 2014, pp. 5–50.
[18] J.P. Roop, J. Comput. Appl. Math. 193 (1) (2006) 243–268.
[19] G. Pang, W. Chen, Z. Fu, J. Comput. Phys. 293 (2015) 280–296.
[20] H. Wang, K. Wang, J. Comput. Phys. 230 (21) (2011) 7830–7839.
[21] H. Wang, N. Du, J. Comput. Phys. 253 (2013) 50–63.
[22] H. Wang, N. Du, J. Comput. Phys. 258 (2014) 305–318.
[23] F. Song, C. Xu, J. Comput. Phys. 299 (2015) 196–214.
[24] M. Ilić, F. Liu, I. Turner, V. Anh, Fract. Calc. Appl. Anal. 8 (3) (2005) 323–341.
[25] M. Ilić, F. Liu, I. Turner, V. Anh, Fract. Calc. Appl. Anal. 9 (4) (2006) 333–349.
[26] Q. Yang, I. Turner, F. Liu, M. Ilić, SIAM J. Sci. Comput. 33 (3) (2011) 1159–1180.
[27] G.E. Karniadakis, S.J. Sherwin, Spectral/hp Element Methods for Computational Fluid Dynamics, Oxford University Press, 2013.
[28] H. Antil, J. Pfefferer, S. Rogovs, Fractional operators with inhomogeneous boundary conditions: Analysis, Control, and Discretization, 2017, arXiv preprint arXiv:1703.05256.
[29] F. Song, C. Xu, G.E. Karniadakis, SIAM J. Sci. Comput. 39 (4) (2017) A1320–A1344.
[30] Q. Yang, Novel Analytical and Numerical Methods for Solving Fractional Dynamical Systems, Queensland University of Technology, 2010.
[31] S. Balay, S. Abhyankar, M.F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, A. Dener, V. Eijkhout, W.D. Gropp, D. Karpeyev, D. Kaushik, M.G. Knepley, D.A. May, L.C. McInnes, R.T. Mills, T. Munson, K. Rupp, P. Sanan, B.F. Smith, S. Zampini, H. Zhang, H. Zhang, PETSc Users Manual, Tech. Rep. ANL-95/11 - Revision 3.12, Argonne National Laboratory, 2019, URL https://www.mcs.anl.gov/petsc.
[32] V. Hernandez, J.E. Roman, V. Vidal, ACM Trans. Math. Software 31 (3) (2005) 351–362, http://dx.doi.org/10.1145/1089014.1089019, URL http://doi.acm.org/10.1145/1089014.1089019.
[33] M. Carlson, R.M. Kirby, H. Sundar, Proceedings of the 34th ACM International Conference on Supercomputing, ICS '20, Association for Computing Machinery, New York, NY, USA, 2020, http://dx.doi.org/10.1145/3392717.3392769.
[34] Kingspeak, URL https://www.chpc.utah.edu/documentation/guides/kingspeak.php.
[35] Frontera, URL https://www.overleaf.com/project/5cbb457f1f07a5705c5be796.
[36] Hanford Site, URL https://en.wikipedia.org/wiki/Hanford5FSite.
[37] Aorta, URL https://www.heart.org/en/health-topics/aortic-aneurysm/.
[38] MS Windows NT Kernel Description, URL http://www.ewp.rpi.edu/hartford/7Eernesto/C5FSu2003/MMHCD/Notes/.
[39] B. Baeumer, M. Kovács, M.M. Meerschaert, Comput. Math. Appl. 55 (10) (2008) 2212–2226.
[40] J. Cottrell, T. Hughes, A. Reali, Comput. Methods Appl. Mech. Engrg. 196 (41) (2007) 4160–4183, URL http://www.sciencedirect.com/science/article/pii/S0045782507001703.
[41] D. Moxey, C.D. Cantwell, Y. Bao, A. Cassinelli, G. Castiglioni, S. Chun, E. Juda, E. Kazemi, K. Lackhove, J. Marcon, G. Mengaldo, D. Serson, M. Turner, H. Xu, J. Peiró, R.M. Kirby, S.J. Sherwin, Comput. Phys. Comm. 249 (2020) 107110, http://dx.doi.org/10.1016/j.cpc.2019.107110, URL http://www.sciencedirect.com/science/article/pii/S0010465519304175.
[42] C. de Falco, A. Reali, R. Vázquez, Adv. Eng. Softw. 42 (12) (2011) 1020–1034.
[43] O.A.G. de Suarez, R. Rossi, C.R. Altafini, C.R.A. da Silva, Appl. Math. Model. 39 (1) (2015) 396–413, URL http://www.sciencedirect.com/science/article/pii/S0307904X14002959.
[44] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G.D. Peterson, R. Roskies, J.R. Scott, N. Wilkins-Diehr, Comput. Sci. Eng. 16 (2014) 62–74, http://dx.doi.org/10.1109/MCSE.2014.80.
[45] I. Babuška, J.E. Osborn, Math. Comp. 52 (186) (1989) 275–297.
[46] J. Xu, SIAM J. Sci. Comput. 15 (1) (1994) 231–237.
[47] P. Constantin, A.J. Majda, E. Tabak, Nonlinearity 7 (6) (1994) 1495.
[48] I.M. Held, R.T. Pierrehumbert, S.T. Garner, K.L. Swanson, J. Fluid Mech. 282 (1995) 1–20.
[49] P. Constantin, M.C. Lai, R. Sharma, Y.-H. Tseng, J. Wu, J. Sci. Comput. 50 (1) (2012) 1–28.
[50] F. Song, G.E. Karniadakis, Chaos Solitons Fractals 102 (2017) 327–332.
[51] E. Tadmor, SIAM J. Numer. Anal. 26 (1) (1989) 30–44.