

BAYESIAN DEEP OPERATOR LEARNING FOR HOMOGENIZED TO FINE-SCALE MAPS FOR MULTISCALE PDE*

ZECHENG ZHANG[†], CHRISTIAN MOYA[‡], WING TAT LEUNG[§],
GUANG LIN[¶], AND HAYDEN SCHAEFFER^{||}

Abstract. We present a new framework for computing fine-scale solutions of multiscale partial differential equations (PDEs) using operator learning tools. Obtaining fine-scale solutions of multiscale PDEs can be challenging, but there are many inexpensive computational methods for obtaining coarse-scale solutions. Additionally, in many real-world applications, fine-scale solutions can only be observed at a limited number of locations. In order to obtain approximations or predictions of fine-scale solutions over general regions of interest, we propose to learn the operator mapping from coarse-scale solutions to fine-scale solutions using observations of a limited number of (possible noisy) fine-scale solutions. The approach is to train multi-fidelity homogenization maps using mathematically motivated neural operators. The operator learning framework can efficiently obtain the solution of multiscale PDEs at any arbitrary point, making our proposed framework a mesh-free solver. We verify our results on multiple numerical examples showing that our approach is an efficient mesh-free solver for multiscale PDEs.

Key words. neural operator, neural homogenization, multiscale finite element method, discretization invariant, multi-fidelity

MSC codes. 35B27, 68Q32, 68T05, 78M10, 65K10

DOI. 10.1137/23M160342X

1. Introduction. Obtaining fine-scale solutions for a multiscale partial differential equation (PDE) problem can be costly, often requiring extensive large-scale computations to fully resolve. In addition, while coarse-scale solutions are easier to observe, we typically only have access to a limited number of fine-scale solutions in real-world multiscale applications. This limitation becomes particularly significant when fine-scale information is obtained by only a few samples or from specific subregions within the domain. Therefore, there is a need for methods that effectively utilize coarse-scale solutions (via simulations) alongside scarce fine-scale measurements. By addressing this one could compute accurate fine-scale solutions throughout the domain of interest.

Solving multiscale problems presents significant challenges due to the need for computationally expensive fine-scale solvers to accurately capture multiscale features. To address this issue, various multiscale methods have been proposed with the aim of

*Received by the editors September 20, 2023; accepted for publication (in revised form) May 17, 2024; published electronically July 17, 2024.

<https://doi.org/10.1137/23M160342X>

Funding: The first author was supported in part by AFOSR MURI FA9550-21-1-0084. The fifth author was supported in part by AFOSR MURI FA9550-21-1-0084, NSF DMS-2208339, and an NSF CAREER Award DMS-2331100. The fourth author gratefully acknowledges the support of the National Science Foundation (DMS-2053746, DMS-2134209, ECCS-2328241, and OAC-2311848) and U.S. Department of Energy (DOE) Office of Science Advanced Scientific Computing Research program DE-SC0023161 and DOE – Fusion Energy Science, under grant DE-SC0024583.

[†]Department of Mathematics, Florida State University (zecheng.zhang.math@gmail.com).

[‡]Department of Mathematics, Purdue University (cmoyacal@purdue.edu).

[§]Department of Mathematics, City University Hong Kong (wtleung27@cityu.edu.hk).

[¶]Department of Mathematics and Mechanical Engineering, Purdue University (guanglin@purdue.edu).

^{||}Department of Mathematics, UCLA (hayden@math.ucla.edu).

resolving these complex phenomena. Among these methods, multiscale finite element methods (MsFEMs) have emerged as particularly successful techniques [11, 6, 9, 8, 7]. An MsFEM operates by first solving for the multiscale basis, which effectively captures the local multiscale features on a coarse mesh. Subsequently, the problem is solved using this constructed basis, enabling a computationally effective approach for tackling multiscale problems. Although using an MsFEM provides an effective framework for accurately representing and resolving the multiscale behavior inherent in these challenging problems, it can still be costly [5, 8, 18, 14, 13].

One technique for handling multiscale problems is through homogenization [12, 1, 18]. A standard approach is to employ the asymptotic expansion to represent the solution. One can construct a homogenized solution without fully resolving the various scales, which still leads to a valid approximation of the exact solution. In particular, one can compute the solution to a homogenized PDE which only contains information at the coarse scale. An alternative formulation is to establish a connection, in the form of an operator, between a solution at a finer scale and the homogenized solution. The mathematical formulation and numerical computation of the operator for this task remain an open problem. However, we propose addressing this problem through operator learning by creating an efficient deep neural network-based approximation of the operator that represents the fine-scale solver of the corresponding multiscale PDE.

In *operator learning* [28, 21, 42, 43] one trains deep neural networks to approximate nonlinear operators, which are mappings between infinite-dimensional spaces. These operator-learning frameworks have been successful in scientific computing due to their versatility and efficiency for various problems arising from physical systems, including modeling dependencies on initial conditions or parameters. The first operator learning framework for PDEs, the deep operator network (DON), was developed in [28]. DON is built on the universal approximation theorem for operators from [4] and can effectively learn operators with relatively small datasets. Compared to more traditional neural networks that learn mappings between vector spaces, DON exhibits improved generalization behavior on more complex tasks, as demonstrated in various applications. These include acting as surrogate solvers of PDEs (such as bubble dynamics [22]), and approximating operators arising in tasks for control systems [24], power grids [30], and multiphysics problems [2]. In [42], a discretization-invariant extension and analysis of DON was proposed, which allows the network to handle input functions with different discretizations. Some other extensions of DON have enabled incorporating physical information, leading to physics-informed DONs [40], handling noisy data [30, 25], quantifying uncertainty [36, 30, 25], or performing inverse design for complex applications [29, 38].

In a parallel effort, the Fourier neural operator (FNO) was presented in [21]. FNO approximates nonlinear operators by directly parametrizing integral kernels in the Fourier domain. FNO's effectiveness has been demonstrated in numerous applications across various domains, including but not limited to global weather prediction [33], multiphase flow [41], and solving PDEs with complex geometry [21]. In [17], the authors generalized FNO and proposed neural operators that can effectively learn operators. Specifically, they formulated the neural operator as a composition of linear integral operators and nonlinear activation functions. Furthermore, the authors supported the neural operators by providing a universal approximation theorem, which demonstrates the existence of a neural operator that can approximate a given nonlinear continuous operator.

Training effective operator networks for multiscale problems can be expensive due to the requirement of large amounts of high-fidelity (fine-scale) data [39, 38, 27]. However, in practice, high-fidelity data may be limited. Fortunately, we may have access to mathematical models that can generate abundant low-fidelity (coarse-scale) data. In such cases, our objective is to accurately calculate the fine-scale solution by constructing an operator learning framework that maps a function (i.e., the solution of the partially known PDE) to a new function that represents the exact solution of the target PDE. We then use the available data to train this framework, mixing scarce high-accuracy observation with abundant coarse simulations [35, 34].

Therefore, it is crucial to design operator learning frameworks that can be trained using high-fidelity data, mathematical models, and low-fidelity data. Several studies [16, 29, 10] have developed multi-fidelity neural operators to address this need, with applications to complex tasks such as fluid or materials science. However, they do not consider the uncertainty caused by using various fidelities or models during training and thus may not be effective in the presence of noisy data. To bridge this gap, this paper proposes a novel Bayesian deep operator learning architecture for developing surrogates of multiscale PDE solvers. The architecture can incorporate noisy high-fidelity data and efficient solvers developed based on homogenization, which is a PDE-based technique for handling multiscale PDEs [12, 1, 37, 15]. The main contributions of this paper are summarized below.

- We propose a data-driven neural operator approach that downscales a given coarse model. This approach maps coarse-scale solutions to fine-scale solutions directly from data, and the trained neural operator is a mesh-free fine-scale solver of the multiscale problems.
- We propose an “oversampling” approach to capture the input function to this operator through patches. In our numerical experiments, we observed that enlarging the patch leads to improvement in the prediction error.
- Furthermore, we have designed the first Bayesian, multiscale operator learning framework that is trained with noisy data. This framework can provide robust and mesh-free solutions, even from coarse-scale solutions.
- Finally, we demonstrate through multiple numerical experiments that the proposed framework represents an efficient mesh-free solver for multiscale PDEs.

The paper is organized as follows. Section 2 formulates the problem of learning the operator from coarse-scale solutions to fine-scale solutions. In section 3, we review the DON framework. Section 4 provides detailed information about the proposed operator learning frameworks, which are trained with coarse-scale solutions and a limited number of observations of fine-scale solutions to approximate fine-scale solutions. Section 5 develops a Bayesian, multiscale operator learning framework that enables reliable prediction of fine-scale solutions, even when trained with noisy observations. We demonstrate the effectiveness of the proposed framework with a series of numerical examples in section 6. Finally, section 7 concludes the paper.

2. Problem formulation. The main goal is to improve the accuracy of a low-scale/low-accuracy model or physical simulation by using real observation data related to a specific physical multiscale process. To address this, the proposed framework involves first obtaining a coarse-scale solution with lower accuracy via a numerical solver. Then, an operator learning approach is used to refine the coarse-scale solution by incorporating available, possibly noisy, observed data. Obtaining the coarse-scale solution is often a more feasible task in terms of the computational cost or algorithmic

complexity. Then, using the coarse-scale solution, the designed operator learning approach will act as a downscaled multiscale model, providing a fine-scale multiscale solver simultaneously.

We motivate and justify the descaling operator method as follows. Consider an example of homogenization of a multiscale elliptic problem:

$$(2.1) \quad -\frac{\partial}{\partial x_i} \left(a_{ij}(x/\epsilon) \frac{\partial}{\partial x_j} u_\epsilon(x) \right) = f(x), \quad x \in \Omega,$$

with $u_\epsilon(x) = 0$ at the boundary $\partial\Omega$. In the above equation, we have used the Einstein notation, u_ϵ is the PDE solution, a_{ij} is the multiscale permeability, $f(x)$ is the forcing, and Ω is the domain. We seek $u_\epsilon(x)$ with an asymptotic expansion of the form

$$(2.2) \quad u_\epsilon(x) = u_0(x, x/\epsilon) + \epsilon u_1(x, x/\epsilon) + \epsilon^2 u_2(x, x/\epsilon) + \mathcal{O}(\epsilon^3),$$

where $u_j(x, y)$ for $j = 0, 1, 2, \dots$ is periodic in $y = x/\epsilon$ with period 1. Denote

$$A^\epsilon = -\frac{\partial}{\partial x_i} \left(a_{ij}(x/\epsilon) \frac{\partial}{\partial x_j} \right).$$

Then, it follows that

$$A^\epsilon = \epsilon^{-2} A_1 + \epsilon^{-1} A_2 + \epsilon^0 A_3,$$

where

$$\begin{aligned} A_1 &= -\frac{\partial}{\partial y_i} \left(a_{ij}(y) \frac{\partial}{\partial y_j} \right), \\ A_2 &= -\frac{\partial}{\partial x_i} \left(a_{ij}(y) \frac{\partial}{\partial y_j} \right) - \frac{\partial}{\partial y_i} \left(a_{ij}(y) \frac{\partial}{\partial x_j} \right), \\ A_3 &= -\frac{\partial}{\partial x_i} \left(a_{ij}(y) \frac{\partial}{\partial x_j} \right). \end{aligned}$$

Thus, we have the decomposition $\epsilon^{-2} A_1 u_\epsilon + \epsilon^{-1} A_2 u_\epsilon + \epsilon A_3 u_\epsilon = f$. By equating the terms with the same power of ϵ , we obtain

$$(2.3) \quad A_1 u_0 = 0,$$

$$(2.4) \quad A_1 u_1 + A_2 u_0 = 0,$$

$$(2.5) \quad A_1 u_2 + A_2 u_1 + A_3 u_0 = f.$$

Substituting A_1 into (2.3) leads to

$$-\frac{\partial}{\partial y_i} \left(a_{ij}(y) \frac{\partial}{\partial y_j} \right) u_0 = 0.$$

According to the theory of second-order ordinary differential equations, u_0 is independent of y . This simplifies (2.4), and we have

$$-\frac{\partial}{\partial y_i} \left(a_{ij}(y) \frac{\partial}{\partial y_j} \right) u_1 = \left(\frac{\partial}{\partial y_i} a_{ij}(y) \right) \frac{\partial}{\partial x_j} u_0.$$

Thus, $u_1(x, y)$ can be solved by introducing $\chi_j(y)$, which is the solution to the following problem:

$$(2.6) \quad -\frac{\partial}{\partial y_i} \left(a_{ij}(y) \frac{\partial}{\partial y_j} \right) \chi_j = \frac{\partial}{\partial y_i} a_{ij}(y),$$

χ_j is periodic in y with mean 0.

Equation (2.6) is referred to as a cell problem that needs to be solved within one period of y , or, in the unit cell $Y = [0, 1]^d$, where d is the dimension of the problem. Then, u_1 can be expressed as follows:

$$u_1(x, y) = \chi_j \frac{\partial u_0}{\partial x_j}(x).$$

Substituting the above into (2.2), we have

$$(2.7) \quad u_{\epsilon,1}(x) = u_0 + \epsilon \chi_j \frac{\partial u_0}{\partial x_j}(x),$$

which is a higher-order approximation of u compared to u_0 .

Finally, we can express (2.7) in operator form as follows:

$$(2.8) \quad u_{\epsilon,1}(x) = G(u_0)(x).$$

Here the operator G maps the homogenized solution (i.e., the *coarse-scale solution*) u_0 to a *finer-scale solution* $u_{\epsilon,1}$. Our goal is to approximate the operator G using a deep operator network (DON).

3. A brief review of deep operator networks. This section provides a review of the DON proposed in [28]. DON is a neural network architecture that approximates mappings between infinite-dimensional spaces. It is built on the universal approximation theorem of continuous operators, which was introduced in the seminal works [4, 3]. In particular, DON satisfies the following approximation theorem. Suppose X is a Banach space and $K_1 \subset X$ and $K_2 \subset \mathbb{R}$ are compact sets. If $V \subset C(K_1)$ is compact, then the continuous operator $G : V \rightarrow C(K_2)$ can be effectively approximated by a parameterized function. Specifically, for any $\epsilon > 0$, there exist positive integers M , N , and K , constants c_i^k , ζ_k , θ_i^k , and $\varepsilon_{ij}^k \in \mathbb{R}$, and points $\omega_k \in \mathbb{R}^d$, $y_j \in K_1$, $i = 1, \dots, M$, $k = 1, \dots, K$, and $j = 1, \dots, N$, such that

$$(3.1) \quad \left| G(u)(x) - \sum_{k=1}^K \sum_{i=1}^M c_i^k g \left(\sum_{j=1}^N \varepsilon_{ij}^k u(y_j) + \theta_i^k \right) g(\omega_k \cdot x + \zeta_k) \right| < \epsilon$$

holds for all $u \in V$ and $x \in K_2$. Here g is the activation function in the Tauber–Wiener class; specifically, if $g : \mathbb{R} \rightarrow \mathbb{R}$ satisfies that all the linear combinations $\sum_{i=1}^N c_i g(\lambda_i x + \theta_i)$, $\lambda_i \in \mathbb{R}$, $\theta_i \in \mathbb{R}$, $c_i \in \mathbb{R}$, $i = 1, \dots, N$, and are dense in $C[a, b]$, then g is called a Tauber–Wiener function.

The above approximation theorem suggests a neural network architecture for DON illustrated in Figure 1. The architecture comprises two subnetworks: a branch net and a trunk net. The *branch net* is composed of a stacked collection of K networks, which take the function u discretized using N sensors as input, and output the vector $(b_1, \dots, b_K)^\top$. On the other hand, the *trunk net* takes the location x within the output function domain as input and outputs the vector $(t_1, \dots, t_K)^\top$. The final output of the DON is obtained through the inner product between the output vectors of the branch and trunk nets.

It is important to note that the DON inputs contain the independent variable, x , which denotes the location of the output target function. This means that a well-trained DON can predict the output function value at any arbitrary point in its domain. We will use this property to construct an operator learning-based mesh-free solver for multiscale PDEs.

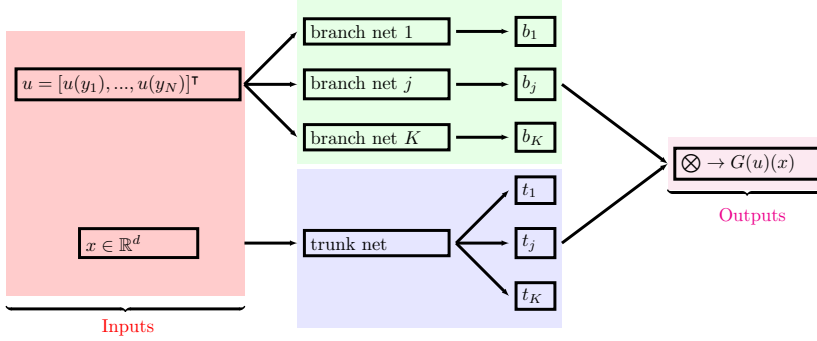


FIG. 1. Stacked version of the Deep Operator Network (DON). $\otimes$ denotes the inner product in $\mathbb{R}^K$. Specifically, branch nets are two-layer neural networks with weights ε_{ij}^k , c_i^k and bias θ_i^k in (3.1); trunk net has weights ω_k and bias ζ_k .

4. Proposed methodology. We use DON to design two operator learning-based algorithms that can approximate the operator mapping coarse-scale solutions to fine-scale solutions.

The Vanilla Operator Learning Algorithm (without Patch). Here, we design and train a deep neural network G_θ , with a vector of trainable parameters θ to approximate the *true* operator $G(u_0(x))(x)$, which maps the coarse-scale solution $u_0(x)$ to the fine-scale solution at any given location x within the domain Ω . We summarize this vanilla algorithm in Algorithm 4.1. We also note that we refer to this algorithm as “without patch” to emphasize that we only use the coarse solution at a point x , and not any neighboring locations within Ω .

To train the proposed DON $G_\theta : u_0 \mapsto u$, we minimize the loss function,

$$\mathcal{L}(\theta) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|G_\theta(u_0(x_i))(x_i) - u(x_i)\|^2,$$

using the dataset of N_p triplets $\{u_0(x_i), x_i, u(x_i)\}_{i=1}^{N_p}$, where $u_0(x_i)$ is the low-accuracy (coarse-scale) solution, $x_i \in \Omega$ the given location within the domain Ω , and $u(x_i)$ the fine-scale (observation) solution.

Remark 4.1. We have two remarks regarding Algorithm 4.1.

1. The coarse-scale solution u_0 can be obtained easily and with low computational cost. One approach is to use multiscale finite element methods (MsFEMs) [11, 6, 5]. Another option is to use data-free machine learning methods, such as neural homogenization physics-informed neural networks (NH-PINN) [18]. PINN is a mesh-free solver, which means it can calculate the solution at any point in the PDE’s domain, allowing u_0 to be evaluated at any point in the domain.
2. The resulting trained DON, $G_{\theta^*}(u_0(x))(x)$, with optimal parameters θ^* , is a fine-scale, mesh-free solver.

The Operator Learning Algorithm with Patch. In the vanilla algorithm, the input function u_0 (i.e., the coarse-scale solution) is evaluated at a single point x_i . However, in (2.7), $\frac{\partial u_0}{\partial x}(x_i)$ requires the consideration of the derivative at that point. To approximate this derivative, finite difference schemes use neighboring points, leading to the idea of sampling a local neighborhood centered around x_i , i.e., the *patch*.

To illustrate this concept, we provide an example (see Figure 2) involving a 5×5 patch for a two-dimensional (2D) problem. Subsequently, we propose a novel

algorithm, the operator learning with patch algorithm detailed in Algorithm 4.2 and illustrated in Figure 3, which uses the patch sampling approach to estimate the desired derivative.

Formally, we approximate the fine-scale solution $u(x_i)$ at any arbitrary point x_i within the domain using the DON $G_\theta(\hat{u}_i)(x_i)$. Here, $\hat{u}_i$ refers to the collection of coarse-scale solutions $\{u(x)\}_{x \in P_i}$ evaluated at the patch P_i (i.e., the neighborhood of locations around x_i), which serves as the new input to the branch net.

To train the proposed DON $G_\theta : \hat{u}_i \mapsto u(x_i)$, we minimize the loss function,

$$\mathcal{L}(\theta) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|G_\theta(\hat{u}_i)(x_i) - u(x_i)\|^2,$$

using the dataset of N_p triplets $\{\hat{u}_i, x_i, u(x_i)\}_{i=1}^{N_p}$.

Algorithm 4.1. Vanilla Operator Learning without Patch.

Requires: dataset: $\{u_0(x_i), x_i, u(x_i)\}_{i=1}^{N_p}$, where $u_0(x_i)$ is the low-accuracy (coarse-scale) solution, x_i the location within the domain Ω , and $u(x_i)$ the fine-scale (observation) solution.

Use a coarse solver to solve the multiscale equation and obtain $u_0(x_i)$, where $x_i \in \Omega$. Train the DON $G_\theta : u_0 \mapsto u$ by minimizing the loss function

$$(4.1) \quad \mathcal{L}(\theta) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|G_\theta(u_0(x_i))(x_i) - u(x_i)\|^2.$$

Predict fine-scale solutions at arbitrary locations $x \in \Omega$ using the trained DON $G_{\theta^*}(u_0)(x)$, where θ^* denotes the optimal network parameters.

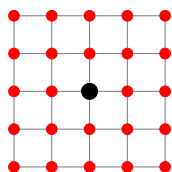


FIG. 2. A 5×5 patch (red dots) centered at an observation point (black dot). All red dots represent the sensors used to sample the local input function (coarse-scale solutions) centered at the black dot (observation). (Color available online.)

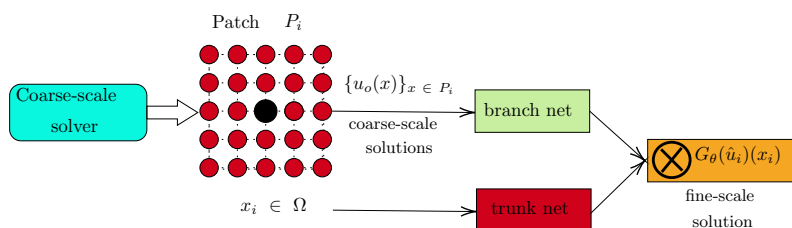


FIG. 3. A pictorial description of the operator learning algorithm with a patch. First, a coarse solver generates a collection of coarse-scale solutions $\hat{u}_i = \{u_o(x)\}_{x \in P_i}$ on the patch P_i centered at $x_i \in \Omega$. The collection of coarse-scale solutions $\hat{u}_i$ is then input to the branch net, while the target location $x_i \in \Omega$ is input to the trunk net. The proposed operator learning algorithm with patch outputs the fine-scale solution $G_\theta(\hat{u}_i)(x_i) = u(x_i)$ at the arbitrary target location $x_i \in \Omega$.

Algorithm 4.2. Operator Learning with Patch.

Require: dataset: $\{\hat{u}_i, x_i, u(x_i)\}_{i=1}^{N_p}$. Here $u(x_i)$ is the fine-scale (observation) solution and $\hat{u}_i = \{u(x)\}_{x \in P_i}$ is the collection of coarse-scale solutions $\{u(x)\}_{x \in P_i}$ evaluated at the patch P_i (i.e., the neighborhood of locations around x_i). For example, a patch of size three around x_i of the 1D case is $P_i = \{x_{i-1}, x_i, x_{i+1}\}$. A 5×5 2D patch demonstration is presented in Figure 2. Use a coarse solver to solve the multiscale equation and obtain $u_0(x)$, where $x \in P_i$. Train the DON $G_\theta : \hat{u}_i \mapsto u(x)$ by minimizing the loss function

$$\mathcal{L}(\theta) = \frac{1}{N_p} \sum_{i=1}^{N_p} \|G_\theta(\hat{u}_i)(x_i) - u(x_i)\|^2.$$

Predict fine-scale solutions at $x_i \in \Omega$ using the trained DON $G_{\theta^*}(\hat{u}_i)(x_i)$, where θ^* denotes the optimal network parameters.

Remark 4.2. We conclude this section with two remarks concerning the patch.

1. Our numerical experiments have revealed a notable trend: as the size of the patch or neighborhood increases, the relative error decreases.
2. We use DON to learn the operator. However, since DON is not invariant to the discretization of the input function, all patches must share the same discretization.

5. Bayesian, multiscale operator learning. The conventional optimization framework used to train DON does not accurately quantify uncertainty and provide robust predictions, which are crucial for creating credible intervals for scientific and engineering applications. This can lead to unreliable DON predictions and limit the reliability of DON. However, quantifying the uncertainty associated with limited and noisy training data, along with the fact that the network is over-parameterized, makes this a challenging task. This challenge is even greater in operator learning, as it involves mappings between infinite-dimensional spaces, such as the multiscale operator G . In this section, we address this challenge by developing a Bayesian DON (B-DON) [25]. B-DON can create estimators and credible intervals for the operator that maps the homogenized solution u_o to the fine-scale solution $G(u_o)(x)$ for any given $x \in \Omega$.

In this B-DON framework, our goal, given the training noisy dataset $\mathcal{D}$, is to construct a distribution $p(G|(u_o, x), \mathcal{D})$ that can predict the operator value of G (the fine-scale solution) based on the input homogenized solution u_o and at any new location x . To achieve this, we first assume the following factorized Gaussian likelihood function for the data:

$$(5.1) \quad p(G|(u_o, x), \theta) = \mathcal{N}(G|G_\theta(u_o)(x), \text{diag}(\Sigma^2)) = \prod_{j=1}^N \mathcal{N}(G_j|G_\theta(u_{o,j})(x_j), \sigma),$$

where the output $G_\theta(u_o)(x)$ is the mean of the Gaussian distribution assumed for G , given the homogenized input u_o at location x , and $\text{diag}(\Sigma^2)$ is a diagonal covariance matrix with $\Sigma^2 = (\sigma^2, \dots, \sigma^2)$ on the diagonal [36]. Note that σ can be assumed or estimated from the noisy data.

The fine-scale solution G for a given homogenized solution u_o at a location x , given the noisy training data $\mathcal{D}$, is the random variable $(G|(u_o, x), \mathcal{D})$. To obtain the density of this random variable, we need to integrate out the model parameters as follows:

$$p(G|(u_o, x), \mathcal{D}) = \int p(G|(u_o, x), \theta) p(\theta|\mathcal{D}) d\theta.$$

Here, $p(\theta|\mathcal{D})$ represents the posterior distribution of the trainable parameters. This distribution enables us to quantify the *epistemic uncertainty*, which refers to the uncertainty related to the trainable parameters θ [36, 25].

To obtain this posterior, we use Bayes's rule:

$$p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta),$$

where $p(\theta)$ is the *prior* distribution of the parameters and $p(\mathcal{D}|\theta)$ the *data likelihood*, i.e., $p(\mathcal{D}|\theta) = \prod_{j=1}^N p(G_j|(u_{o,j}, x_j), \theta)$, which we calculate using the DON forward pass and the independent and identically distributed (i.i.d.) noisy training dataset $\mathcal{D}$. Acquiring the posterior distribution using Bayes's rule is computationally and analytically intractable [36]. Therefore, in previous work [25], we approximated the posterior distribution through samples obtained from it. Specifically, we obtained an M_e -ensemble of θ samples, denoted as $\{\theta_k\}_{k=1}^M$, as described below.

5.1. Sampling the M_e -ensemble $\{\theta_k\}_{k=1}^M$. To obtain the M_e -ensemble of parameters $\{\theta_k\}_{k=1}^M$, B-DON uses the stochastic gradient replica exchange Langevin diffusion (SG-reLD), which we developed and studied in [25, 26, 23, 31, 19]. As demonstrated in our previous work, SG-reLD enjoys theoretical guarantees beyond convex scenarios, effectively handles large datasets, and accelerates convergence to the posterior distribution $p(\theta|\mathcal{D})$.

Specifically, SG-reLD uses two Langevin diffusions to describe the stochastic dynamics of θ , along with a stochastic process that allows the diffusions to swap simultaneously. The high-temperature diffusion enables exploration of the parameter space, facilitating convergence to the flattened distribution of θ . The low-temperature diffusion exploits the same parameter space, enabling faster convergence to local minima θ^* . By swapping the diffusions, SG-reLD effectively escapes local minima and allows θ_k to converge faster to the desired posterior $p(\theta|\mathcal{D})$. For more details about employing SG-reLD with B-DONs, please refer to our previous paper [25].

In practice, we can use the M_e -ensemble $\{\theta_k\}_{k=1}^M$ obtained using SG-reLD to fit a parametric distribution, such as the Gaussian distribution

$$\mathcal{N}(\bar{\mu}(u_o)(X_{\text{test}}), \bar{\sigma}_e^2(u_o)(X_{\text{test}}))$$

for an arbitrary mesh X_{test} of locations. The parameters of this distribution are given by

$$(5.2) \quad \bar{\mu}(u_o)(X_{\text{test}}) = \frac{1}{M} \sum_{k=1}^M G_{\theta_k}(u_o)(X_{\text{test}}),$$

$$(5.3) \quad \bar{\sigma}_e^2(u_o)(X_{\text{test}}) = \frac{1}{M} \sum_{k=1}^M (G_{\theta_k}(u_o)(X_{\text{test}}) - \bar{\mu}(u_o)(X_{\text{test}}))^2.$$

Sampling from the aforementioned distribution allows for estimating credible sets for the fine solution. This also enables the sampling of more reliable predictions, which can help reduce the average relative error of test trajectories in the presence of noise, as demonstrated in our numerical experiments section.

6. Numerical experiments. This section presents several numerical experiments to demonstrate the effectiveness of the proposed framework. For all examples, we conducted 100 independent experiments and present the average relative error. We will analyze the error decay with respect to the patch size, the error decay with respect to the number of observation points, and the error decay in the presence of noise.

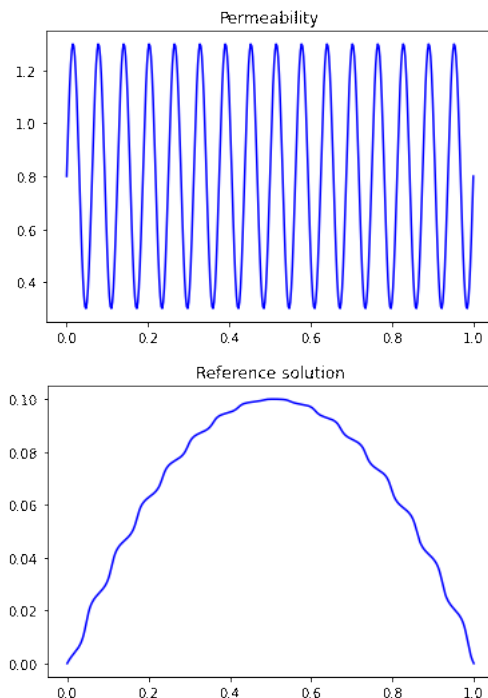


FIG. 4. 1D elliptic. Top: Permeability $\kappa(x)$. Bottom: Reference solution.

6.1. 1D elliptic. In our first example, we will study a 1D problem for which we can obtain an exact homogenized solution. Specifically, we will consider the following elliptic equation:

$$\begin{aligned} -\frac{d}{dx} \left(a(x/\epsilon) \frac{du}{dx} \right) &= f, \quad x \in [0, 1], \\ u(0) &= u(1) = 0, \end{aligned}$$

where $a(x) = 0.5 \sin(2\pi \frac{x}{\epsilon}) + 0.8$ and $f(x) = 0.5$. Figure 4 illustrates the multiscale permeability $\kappa(x)$ and the reference solution.

We set $\epsilon = 1/16$. Then, the coarse-scale solution is obtained using classical homogenization theory, and the relative error of the homogenized solution is 0.07%. To further improve the homogenized solution, we employ data from exact solutions at 16 uniformly distributed points in the domain, i.e., $N_p = 16$ in (4.1). We evaluate the performance of our approach as the patch enlarges using the oversampling trick, and we present the results in Figure 5.

6.2. 2D elliptic equation with 1 fast variable. In this experiment, we consider the following 2D elliptic equation:

$$(6.1) \quad -\nabla \cdot (\kappa(x/\epsilon) \nabla u) = f, \quad x \in \Omega = [0, 1]^2,$$

$$(6.2) \quad u(x) = 0, \quad x \in \partial\Omega,$$

where $\kappa(x/\epsilon) = 2 + \sin(2\pi x/\epsilon) \cos(2\pi y/\epsilon)$ and $\epsilon = \frac{1}{8}$. Figure 6 shows the permeability κ as a function of x .

For the proposed framework, we assume prior knowledge of the exact solution of the given equation at a limited number of points. To compute the coarse-scale

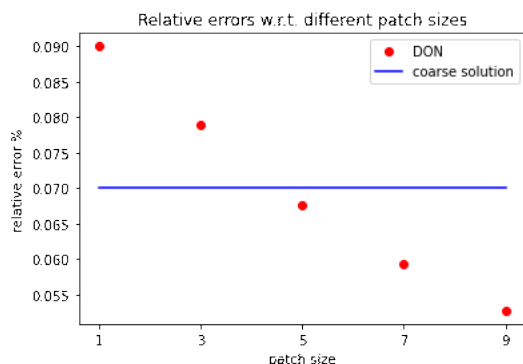


FIG. 5. Relative errors for 1D elliptic equations are shown with respect to different patch sizes. The patch size ranges from 1 (using only the observation point) to 9 (using 9 points with the observation point in the patch center). We trained 100 independent models and presented their average relative errors.

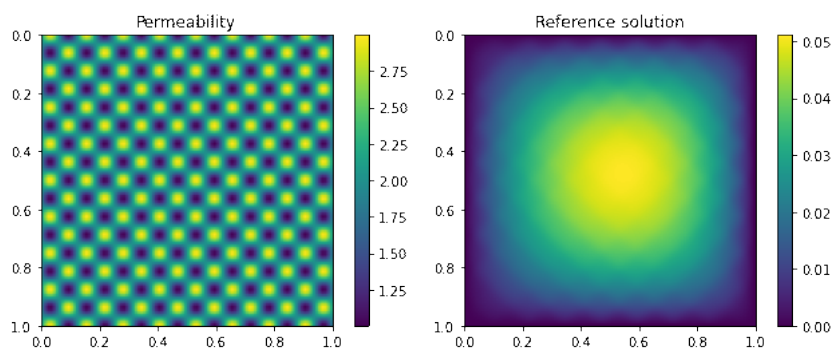


FIG. 6. 2D elliptic with 1 fast variable. Left: Permeability $\kappa(x)$. Right: Reference solution.

solution, we use the methodology described in [18]. Specifically, we use a neural-homogenized physics-informed neural network (NH-PINN) to derive the homogenized solution, denoted as u_0 . The NH-PINN method employed in this study is a mesh-free solver, which allows for the generation of coarse-scale solutions at any point within the domain. As a result, we can always obtain a patch consisting of coarse-scale solutions centered around the observed fine-scale solutions.

Our goal for this experiment is to use operator learning to approximate the mapping from the NH-PINN-based coarse-scale solution u_0 to the corresponding fine-scale solution u_f . By learning this mapping, we aim to improve the accuracy of the coarse-scale solution by leveraging the information from the fine-scale solution.

We demonstrate that the operator can be constructed (trained) more effectively as the patch size increases. We conducted five sets of experiments with patch sizes of 1×1 , 3×3 , 5×5 , 7×7 , and 9×9 . For each set of experiments, we trained 100 models and computed the average relative errors of the last 100 epochs for each model. The results are shown in Figure 7.

Remarkably, our analysis shows that as the patch size increases, there is a reduction in relative errors. This observation highlights the benefits of enlarging the patch size in order to achieve improved accuracy in our models.

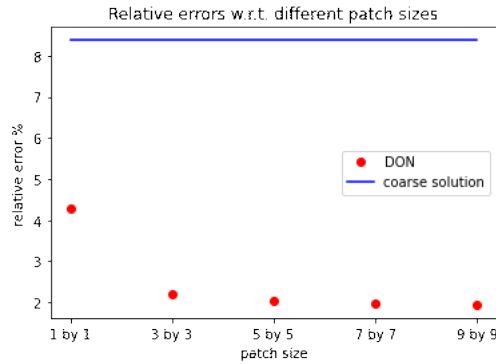


FIG. 7. We investigate the relative errors associated with different patch sizes, ranging from 1×1 (using only the observation point) to 9×9 (consisting of a total of 81 points, with the observation point located at the center of the patch). To obtain a thorough evaluation, we train a total of 100 independent models and report the average relative errors.

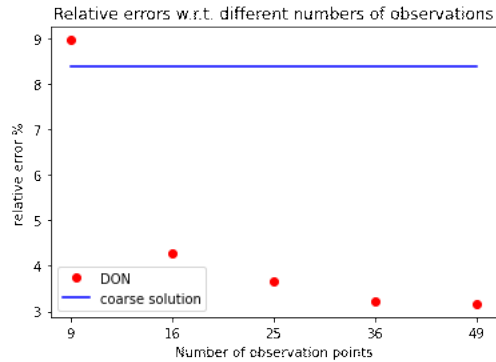


FIG. 8. We use various exact solutions, uniformly distributed in the domain, as the training labels (observations). We set the patch size to be 1, which means that only the observation point coordinate is included as the branch input. By increasing the number of observations from 9 to 49, we evaluate the predictions on a 100×100 mesh to compare the errors. For each observation set, we train a total of 100 independent models, which capture the inherent variability in the training process. To assess the performance, we calculate the average relative error across the ensemble of models. This approach provides a comprehensive evaluation of the predictive accuracy for different numbers of observations.

In the second part of this numerical experiment, we investigate the relationship between the relative error and the number of observation (training) points, which represent the exact solution. To achieve this, we use a fixed patch size of 1, which means that we only include one point from the neighborhood for each observation sample.

To investigate the influence of the number of observation points, we vary the number of observation points and present the results in Figure 8. This analysis helps us understand the impact of the number of observation points on the relative error, providing valuable insights into the behavior of our model.

To conclude this experiment, we train the proposed multiscale B-DON model using noisy observations. We again conduct experiments by gradually increasing the number of observations from 9 to 49, while constructing an M_e -ensemble of models.

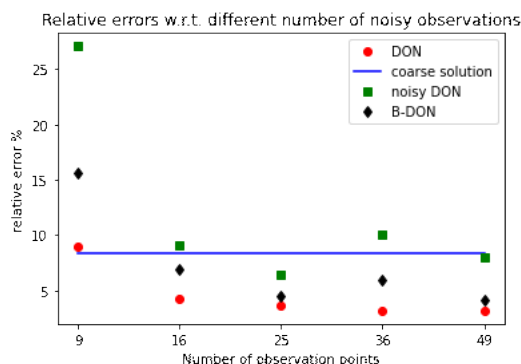


FIG. 9. We employ different numbers of exact solutions as the training labels (observations). We add Gaussian noise with variance $\sigma^2 = 0.005^2$ to the training labels. We fix the patch size to be 1, i.e., include only the observation point coordinate as the brunch input. The number of observations runs from 9 to 49, while we test the prediction on a 100×100 -resolution mesh. For each set of observations, we use as reference the results from the previous section that trains 100 models with the true labels (DON). We train another 100 model solutions with noisy targets (noisy DON). We also use the proposed Bayesian framework (B-DON) to sample from the predictive distribution constructed using an ensemble of $M_e = 100$ sets of parameters. For all the cases, we compute the average relative error.

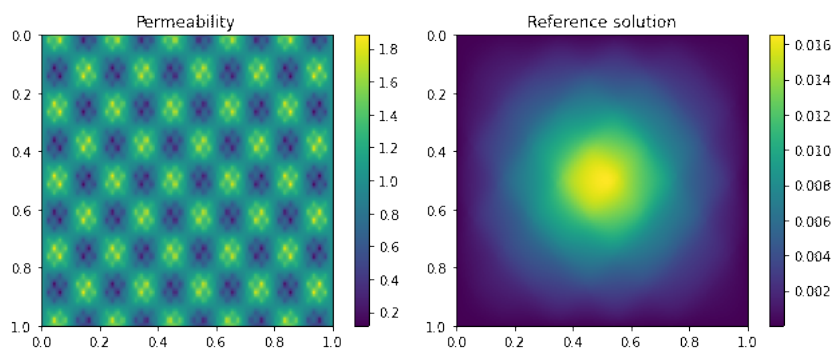


FIG. 10. 2D elliptic with multiple scales. Left: Permeability κ . Right: Reference solution.

Then, by sampling the fitted distribution whose parameters are given in (5.2) and (5.3) and using a 100×100 X_{test} mesh, we predict the values of the fine-scale solutions. Figure 9 depicts the results of such experiments, which illustrate that the proposed Bayesian multi-fidelity operator learning framework can provide robust predictions even in the presence of noisy observations.

6.3. 2D elliptic with multiple scales. This experiment considers the same equation (6.2) as before but with a different permeability κ . Specifically, we let κ be

$$\kappa(x, y) = 1 + \frac{\sin(2\pi \frac{x}{\epsilon_0}) \cos(2\pi \frac{y}{\epsilon_1})}{2 + \cos(2\pi \frac{x}{\epsilon_2}) \sin(2\pi \frac{y}{\epsilon_3})} + \frac{\sin(2\pi \frac{x}{\epsilon_4}) \cos(2\pi \frac{y}{\epsilon_5})}{2 + \cos(2\pi \frac{x}{\epsilon_6}) \sin(2\pi \frac{y}{\epsilon_7})},$$

where $\epsilon_0 = \frac{1}{5}$, $\epsilon_1 = \frac{1}{4}$, $\epsilon_2 = \frac{1}{25}$, $\epsilon_3 = \frac{1}{16}$, $\epsilon_4 = \frac{1}{16}$, $\epsilon_5 = \frac{1}{32}$, $\epsilon_6 = \frac{1}{3}$, $\epsilon_7 = \frac{1}{9}$. Figure 10 illustrates the permeability and reference solution.

We obtain the coarse-scale solution using multiscale finite element methods with one local basis [11, 6, 9, 5]. Then, we demonstrate that the approximation to the

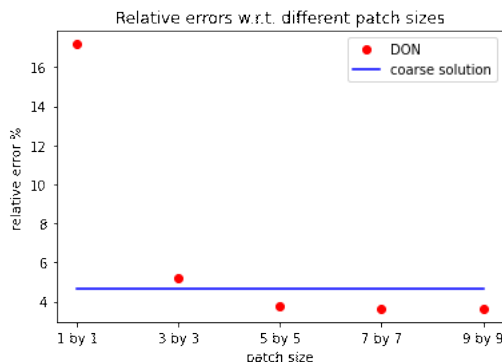


FIG. 11. Relative errors of the 2D elliptic example with respect to different patch sizes. The patch size ranges from 1×1 (using only the observation point) to 9×9 (using 81 points with the observation point at the patch center). We trained 100 independent models and present the average relative errors.

true solution operator can be better constructed (learned) as the patch size increases. To illustrate this, we conduct three sets of experiments with patch sizes of 1×1 , 3×3 , 5×5 , 9×9 , and 16×16 . For each set of experiments, we trained 100 models and computed the average relative errors of the last 100 epochs of all 100 models. Figure 11 shows the obtained results.

6.4. Radiative transfer equation. In this final experiment, we consider the radiative transfer equation (RTE) [32, 20, 7] with a high-contrast scattering coefficient $\sigma(x, \omega)$ (see Figure 12). The term “high-contrast” refers to the strong scattering in the channels:

$$s \cdot \nabla I(x, s) = \frac{\sigma(x, \omega)}{\epsilon} \left(\int_{\mathcal{S}^{n-1}} I(x, s') ds' - I(x, s) \right) \quad \forall x \in D, s \in \mathcal{S}^{n-1}.$$

Here, s is a vector on the unit sphere, and n is the dimension of the problem. In our experiments, we considered $n = 2$, and thus $\mathcal{S}^{n-1} = \mathcal{S}^1$ is the unit circle. Additionally, we set $\epsilon = 0.001$ and $D = [0, 1]^2$. We also introduced the Dirichlet boundary conditions $I(x, s) = I_{\text{in}}$ for entrant directions $s \cdot \mathbf{n} < 0$, i.e., on $\Gamma^- := \{(x, s) \in \partial D \times \mathcal{S}^{n-1} : s \cdot \mathbf{n} < 0\}$. Here, $\mathbf{n}$ is the unit outward normal vector field at $x \in \partial D$. The condition can be written as

$$I = I_{\text{in}}(x, s) \quad \text{for all } (x, s) \in \Gamma^-.$$

In our examples, the top, bottom, and right boundaries have zero incoming boundary conditions. We also assume that the left boundary has nonzero flow injected into the domain.

We chose the multiscale RTE with high contrast channels for its numerical complexity [7] and its challenge for learning-based approaches. However, as ϵ approaches zero, the elliptic solution converges to the RTE [32, 20]. We use the elliptic solution as a low-accuracy solution and incorporate observed real RTE solutions to learn the downscaling of the model. Specifically, the RTE solution is used to correct errors in the elliptic solution. To understand how observations improve downscaling, we

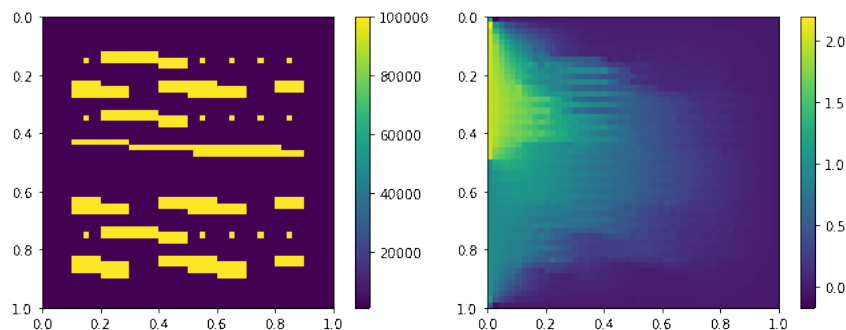


FIG. 12. Left: Demonstration of multiscale scattering $\sigma(x)/\epsilon$, where $\epsilon = 0.001$. Right: The solution of the RTE.

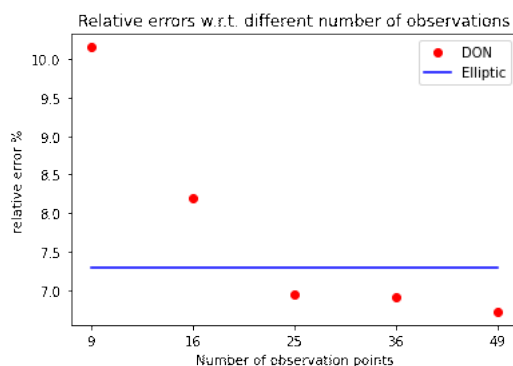


FIG. 13. We use varying numbers of RTE solutions, uniformly distributed in the domain, as training labels (observations). The number of observations ranges from 9 to 49. We test the prediction on a 51×51 mesh. For each set of observations, we train 50 independent models and calculate the average relative error.

conducted a series of experiments with different numbers of observation points. We present the results in Figure 13.

7. Conclusion. This paper introduces a mesh-free operator learning framework for computing the fine-scale solution of multiscale PDEs. The proposed framework is trained using (i) coarse-scale solutions, which are inexpensive to obtain, and (ii) a limited number of observations of the fine-scale solution, to approximate the fine-scale solution at any desired location within the domain. Additionally, when the observations are noisy, we designed a Bayesian, multiscale operator learning approach that can reliably predict fine-scale solutions. Finally, we demonstrated the effectiveness and reliability of the proposed framework using a 1D elliptic equation, 2D elliptic equations with one fast variable and multiple scales, and the radiative transfer equation. The results confirmed that the proposed framework can work as a multiscale, mesh-free solver. In future work, we plan to design a DON that is invariant to input discretization. This will enable more effective capturing of derivatives by having patches with different discretizations. Additionally, we will study the convergence with respect to the parameters of the proposed frameworks.

REFERENCES

- [1] G. ALLAIRE, *Homogenization and two-scale convergence*, SIAM J. Math. Anal., 23 (1992), pp. 1482–1518, <https://doi.org/10.1137/0523084>.
- [2] S. CAI, Z. WANG, L. LU, T. A. ZAKI, AND G. E. KARNIADAKIS, *DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks*, J. Comput. Phys., 436 (2021), 110296.
- [3] T. CHEN AND H. CHEN, *Approximations of continuous functionals by neural networks with application to dynamic systems*, IEEE Trans. Neural Networks, 4 (1993), pp. 910–918.
- [4] T. CHEN AND H. CHEN, *Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems*, IEEE Trans. Neural Networks, 6 (1995), pp. 911–917.
- [5] B. CHETVERUSHKIN, E. CHUNG, Y. EFENDIEV, S.-M. PUN, AND Z. ZHANG, *Computational multiscale methods for quasi-gas dynamic equations*, J. Comput. Phys., 440 (2021), 110352.
- [6] E. CHUNG, Y. EFENDIEV, AND T. Y. HOU, *Adaptive multiscale model reduction with generalized multiscale finite element methods*, J. Comput. Phys., 320 (2016), pp. 69–95.
- [7] E. CHUNG, Y. EFENDIEV, Y. LI, AND Q. LI, *Generalized multiscale finite element method for the steady state linear Boltzmann equation*, Multiscale Model. Simul., 18 (2020), pp. 475–501, <https://doi.org/10.1137/19M1256282>.
- [8] E. CHUNG, Y. EFENDIEV, S.-M. PUN, AND Z. ZHANG, *Computational multiscale method for parabolic wave approximations in heterogeneous media*, Appl. Math. Comput., 425 (2022), 127044.
- [9] E. T. CHUNG, Y. EFENDIEV, AND W. T. LEUNG, *Constraint energy minimizing generalized multiscale finite element method*, Comput. Methods Appl. Mech. Engrg., 339 (2018), pp. 298–319.
- [10] S. DE, M. REYNOLDS, M. HASSANALY, R. N. KING, AND A. DOOSTAN, *Bi-fidelity modeling of uncertain and partially unknown systems using DeepONets*, Comput. Mech., 71 (2023), pp. 1251–1267.
- [11] Y. EFENDIEV, J. GALVIS, AND T. Y. HOU, *Generalized multiscale finite element methods (GMS-FEM)*, J. Comput. Phys., 251 (2013), pp. 116–135.
- [12] Y. EFENDIEV AND T. Y. HOU, *Multiscale Finite Element Methods: Theory and Applications*, Surv. Tutor. Appl. Math. Sci. 4, Springer, 2009.
- [13] Y. EFENDIEV, W. T. LEUNG, W. LI, AND Z. ZHANG, *Hybrid explicit-implicit learning for multiscale problems with time dependent source*, Commun. Nonlinear Sci., 120 (2023), 107081.
- [14] Y. EFENDIEV, W. T. LEUNG, G. LIN, AND Z. ZHANG, *Efficient hybrid explicit-implicit learning for multiscale problems*, J. Comput. Phys., 467 (2022), 111326.
- [15] T. Y. HOU, Q. LI, AND H. SCHAEFFER, *Sparse+ low-energy decomposition for viscous conservation laws*, J. Comput. Phys., 288 (2015), pp. 150–166.
- [16] A. A. HOWARD, M. PEREGO, G. E. KARNIADAKIS, AND P. STINIS, *Multifidelity Deep Operator Networks*, preprint, arXiv:2204.09157, 2022.
- [17] N. KOVACHKI, Z. LI, B. LIU, K. AZIZZADENESHELI, K. BHATTACHARYA, A. STUART, AND A. ANANDKUMAR, *Neural Operator: Learning Maps between Function Spaces*, preprint, arXiv:2108.08481, 2021.
- [18] W. T. LEUNG, G. LIN, AND Z. ZHANG, *NH-PINN: Neural homogenization-based physics-informed neural network for multiscale problems*, J. Comput. Phys., 470 (2022), 111539.
- [19] G. LI, G. LIN, Z. ZHANG, AND Q. ZHOU, *Fast Replica Exchange Stochastic Gradient Langevin Dynamics*, preprint, arXiv:2301.01898, 2023.
- [20] Q. LI AND K. NEWTON, *Diffusion equation-assisted Markov chain Monte Carlo methods for the inverse radiative transfer equation*, Entropy, 21 (2019), 291.
- [21] Z. LI, N. KOVACHKI, K. AZIZZADENESHELI, B. LIU, K. BHATTACHARYA, A. STUART, AND A. ANANDKUMAR, *Fourier Neural Operator for Parametric Partial Differential Equations*, preprint, arXiv:2010.08895, 2020.
- [22] C. LIN, Z. LI, L. LU, S. CAI, M. MAXEY, AND G. E. KARNIADAKIS, *Operator learning for predicting multiscale bubble growth dynamics*, J. Chem. Phys., 154 (2021), 104118.
- [23] G. LIN, C. MOYA, AND Z. ZHANG, *Accelerated Replica Exchange Stochastic Gradient Langevin Diffusion Enhanced Bayesian DeepONet for Solving Noisy Parametric PDEs*, preprint, arXiv:2111.02484, 2021.
- [24] G. LIN, C. MOYA, AND Z. ZHANG, *On Learning the Dynamical Response of Nonlinear Control Systems with Deep Operator Networks*, preprint, arXiv:2206.06536, 2022.
- [25] G. LIN, C. MOYA, AND Z. ZHANG, *B-DeepONet: An enhanced Bayesian DeepONet for solving noisy parametric PDEs using accelerated replica exchange SGLD*, J. Comput. Phys., 473 (2023), 111713.

- [26] G. LIN, Y. WANG, AND Z. ZHANG, *Multi-variance replica exchange SGMCMC for inverse and forward problems via Bayesian PINN*, J. Comput. Phys., 460 (2022), 111173.
- [27] G. LIN, Z. ZHANG, AND Z. ZHANG, *Theoretical and numerical studies of inverse source problem for the linear parabolic equation with sparse boundary measurements*, Inverse Problems, 38 (2022), 125007.
- [28] L. LU, P. JIN, G. PANG, Z. ZHANG, AND G. E. KARNIADAKIS, *Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators*, Nat. Mach. Intell., 3 (2021), pp. 218–229.
- [29] L. LU, R. PESTOURIE, S. G. JOHNSON, AND G. ROMANO, *Multifidelity deep neural operators for efficient learning of partial differential equations with application to fast inverse design of nanoscale heat transport*, Phys. Rev. Res., 4 (2022), 023210.
- [30] C. MOYA, S. ZHANG, G. LIN, AND M. YUE, *DeepONet-grid-UQ: A trustworthy deep operator framework for predicting the power grid's post-fault trajectories*, Neurocomputing, 535 (2023), pp. 166–182.
- [31] O. NA, Z. ZHANG, AND G. LIN, *A Replica Exchange Preconditioned Crank-Nicolson Langevin Dynamic MCMC Method for Bayesian Inverse Problems*, preprint, arXiv:2210.17048, 2022.
- [32] K. NEWTON, Q. LI, AND A. M. STUART, *Diffusive optical tomography in the Bayesian framework*, Multiscale Model. Simul., 18 (2020), pp. 589–611, <https://doi.org/10.1137/19M1247346>.
- [33] J. PATHAK, S. SUBRAMANIAN, P. HARRINGTON, S. RAJA, A. CHATTOPADHYAY, M. MARDANI, T. KURTH, D. HALL, Z. LI, K. AZIZZADENESHELI, ET AL., *FourCastNet: A Global Data-Driven High-Resolution Weather Model Using Adaptive Fourier Neural Operators*, preprint, arXiv:2202.11214, 2022.
- [34] B. PEHERSTORFER, K. WILLCOX, AND M. GUNZBURGER, *Optimal model management for multifidelity Monte Carlo estimation*, SIAM J. Sci. Comput., 38 (2016), pp. A3163–A3194, <https://doi.org/10.1137/15M1046472>.
- [35] B. PEHERSTORFER, K. WILLCOX, AND M. GUNZBURGER, *Survey of multifidelity methods in uncertainty propagation, inference, and optimization*, SIAM Rev., 60 (2018), pp. 550–591, <https://doi.org/10.1137/16M1082469>.
- [36] A. F. PSAROS, X. MENG, Z. ZOU, L. GUO, AND G. E. KARNIADAKIS, *Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons*, J. Comput. Phys., 477 (2023), 111902.
- [37] T. ROBINSON, M. S. ELDERED, K. E. WILLCOX, AND R. HAIMES, *Surrogate-based optimization using multifidelity models with variable parameterization and corrected space mapping*, AIAA J., 46 (2008), pp. 2814–2822.
- [38] H. SCHAEFFER, *Learning partial differential equations via data discovery and sparse optimization*, Proc. Roy. Soc. A Math. Phys. Eng. Sci., 473 (2017), 20160446.
- [39] H. SCHAEFFER, R. CAFLISCH, C. D. HAUCK, AND S. OSHER, *Sparse dynamics for partial differential equations*, Proc. Natl. Acad. Sci. USA, 110 (2013), pp. 6634–6639.
- [40] S. WANG, H. WANG, AND P. PERDIKARIS, *Learning the solution operator of parametric partial differential equations with physics-informed DeepONets*, Sci. Adv., 7 (2021), eabi8605.
- [41] G. WEN, Z. LI, K. AZIZZADENESHELI, A. ANANDKUMAR, AND S. M. BENSON, *U-FNO—an enhanced Fourier neural operator-based deep-learning model for multiphase flow*, Adv. Water Resources, 163 (2022), 104180.
- [42] Z. ZHANG, W. T. LEUNG, AND H. SCHAEFFER, *BelNet: Basis Enhanced Learning, a Mesh-Free Neural Operator*, preprint, arXiv:2212.07336, 2022.
- [43] Z. ZHANG, W. T. LEUNG, AND H. SCHAEFFER, *A Discretization-Invariant Extension and Analysis of Some Deep Operator Networks*, preprint, arXiv:2307.09738, 2023.