ESAIM: PROCEEDINGS AND SURVEYS, **June 2025, Vol. 81, p. 16-32**
Didier Auroux, Martin Campos Pinto, Bruno Després, Victorita Dolean, Stéphane Lanteri, Victor Michel-Dansac

LEARNING LOCAL DIRICHLET-TO-NEUMANN MAPS OF NONLINEAR ELLIPTIC PDES WITH ROUGH COEFFICIENTS *

Miranda Boutilier, [1], Konstantin Brenner[1] and Larissa Miguez[2]

Abstract. Partial Differential Equations (PDEs) involving high-contrast and oscillating coefficients are common in scientific and industrial applications. Numerically approximating these PDEs is a challenging task, which can be addressed, for example, through Multi-scale Finite Element analysis. For linear problems, the Multi-scale Finite Element Method (MsFEM) is well-established, and several viable extensions to nonlinear PDEs have been proposed. However, some aspects of the method seem to be inherently linear. In particular, traditional MsFEM heavily relies on reusing computations. For example, the multi-scale basis and the stiffness matrix can be calculated once and used for multiple right-hand sides or as part of a multi-level iterative algorithm. In this contribution, we present preliminary numerical results of combining MsFEM with Machine Learning tools. The extension of MsFEM to nonlinear problems is achieved by learning local nonlinear approximate Dirichlet-to-Neumann maps. The resulting learning-based multi-scale method is tested on a set of model nonlinear PDEs involving the $p-$Laplacian and degenerate nonlinear diffusion.

Introduction

A variety of real-world phenomena are described by Partial Differential Equations (PDEs). When solutions to PDE models present a high variability in small spatial regions or short periods of time, this behavior is considered multiscale. Standard numerical methods, such as Finite Element, Finite Difference, and Finite Volume methods, struggle to approximate multiscale behavior unless all relevant scales are fully resolved by the discretization, which is often computationally infeasible. To address the limitations of traditional discretization schemes, various multiscale numerical methods have been proposed, including the Localized Orthogonal Decomposition method [16], the Heterogeneous Multiscale Methods [19], the Generalized Finite Element Method [1], the Multiscale Spectral Generalized Finite Element Method [2, 15], and the Multiscale Finite Element Method (MsFEM) [8]. MsFEM is a two-scale finite element approach that relies on two levels of discretization. Unlike traditional finite element methods, which use a predefined approximation space, typically consisting of piecewise polynomials, MsFEM constructs its basis functions numerically. The latter are tailored to the specific PDE and are computed using a high-resolution numerical scheme, allowing MsFEM to capture fine-scale features more accurately. Because the computation of the approximation basis is local to the cells of the coarse mesh, MsFEM is inherently parallel, which allows it to benefit from the ever-growing computing facilities. Notably, MsFEM enhances numerical efficiency by reusing calculations. For instance, the stiffness matrix can be computed once and reused with multiple right-hand sides or within a multi-level iterative algorithm [7].

---

* *The authors are presented in alphabetical order.*

[1] Université Côte d'Azur, Inria, CNRS, Laboratoire J.A. Dieudonné, team GALETS, Nice, France

[2] LNCC - National Laboratory for Scientific Computing, Petrópolis RJ, Brazil

Article published online by EDP Sciences and available at https://www.esaim-proc.org or https://doi.org/10.1051/proc/202581016

On the other hand, with the renewed interest in Machine Learning (ML) within the scientific computing community, numerous techniques have been proposed for applying ML to the solution of PDEs. These include, for example, the DeepRitz method [5], several variants of Physics-Informed Neural Networks [10–12, 17], the DeepONet approach [14], and the Finite Element Interpolated Neural Networks [3].

While multiscale methods for linear elliptic, parabolic, and certain transport problems are well understood, their extension to nonlinear problems remains a challenge [4, 6]. In this contribution, we propose to revisit the traditional MsFEM using machine learning tools. Similar to the classical MsFEM, our approach relies on two levels of discretization. In particular, given a coarse, non-overlapping partitioning of the domain, the solution of the model PDE will be approximated using the space of functions that are piecewise affine over the boundaries of the partitioning. Unlike MsFEM, however, we do not compute the multiscale basis. Instead, using data generated by the local high-resolution Finite Element method, we train a Deep Neural Network to replicate the action of the local nonlinear Dirichlet-to-Neumann (DtN) map on the former approximation space. For training, we adopt an approach similar to Differential Machine Learning [9]; that is, we incorporate into the loss function the terms penalizing the mismatch between the true DtN operator and the surrogate, both in terms of their output values and their partial derivatives with respect to the input variables. Additionally, the monotonicity of the surrogate model is enforced, based on the known monotonicity properties of the true DtN maps. Once the training is completed, the surrogate DtN operators are used to construct the global coarse substructured problem, which, in turn, is approximately solved using Newton's method.

This paper is organized as follows. In Section 1, we introduce the model problem and the associated substructured formulation based on local "continuous" DtN maps. We then discretize the substructured formulation, resulting in an approximate substructured problem expressed as a system of nonlinear algebraic equations involving the "coarse DtN maps". Section 2 provides a description of the Machine Learning process aimed to approximate those coarse DtN functions using surrogates that are significantly cheaper to evaluate. Section 3 contains numerical experiments in 1D and 2D, demonstrating the accuracy of the surrogate model and its ability to supplement the baseline coarse DtN operators in the approximate substructured formulation. Section 4 concludes with a summary and discussion for future work.

1. Model Problem and Approximate Substructured Problem

Let Ω be an open connected polygonal domain in $\mathbb{R}^2$. The adaptation of the method to the one-dimensional case is straightforward, in which case Ω is assumed to be an open bounded interval. In this work, we focus on model problems of the form

$$\left\{ \begin{array}{rcll} au + \operatorname{div}\left(K_\epsilon(\mathbf{x})\mathbf{F}(u, \nabla u)\right) & = & 0 & \text{in} \quad \Omega, \\ u & = & u_D & \text{on} \quad \partial\Omega, \end{array} \right. \tag{1}$$

where a is a positive real coefficient and $K_\epsilon(\mathbf{x})$ is a bounded real-valued function, representing the heterogeneity of the problem. It is assumed that $K_\epsilon(\mathbf{x}) \geq k_{min} > 0$ for all $\mathbf{x} \in \Omega$. For the "flux functions" $F(u, \nabla u)$, we consider either

$$F(u, \nabla u) = -\nabla\left(u^p\right), \quad p > 1,$$

leading to a porous medium problem, or the $p-$Laplace model with

$$F(u, \nabla u) = -|\nabla u|^p \nabla u, \quad p > 0.$$

For the analysis of the porous medium and $p-$Laplace models we refer to [18] and [13] respectively.

In this contribution, we focus on problem (1) with a zero right-hand side. This choice is made to simplify both the presentation and implementation of our machine learning-based model reduction approach. However, we note that extending the method to handle a non-zero right-hand side is, in principle, feasible.

Consider a finite nonoverlapping polygonal partitioning of Ω denoted by $(\Omega_i)_{i=1}^N$. We will refer to $(\Omega_i)_{i=1}^N$ as the coarse mesh over Ω. Additionally, we denote by Γ its skeleton, that is $\Gamma = \bigcup_{i=1}^N \partial\Omega_i$. Let $\mathcal{E}_i$ denote the

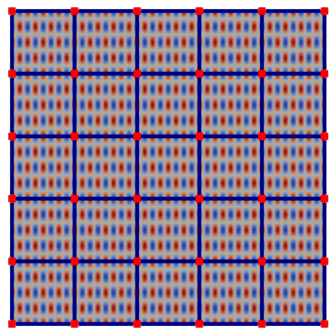

FIGURE 1. Model domain Ω with $N = 5\times 5$ coarse cells. Nonoverlapping skeleton Γ is depicted by thick dark blue lines, coarse nodes are shown in red. skeleton Γ is depicted by thick dark blue lines, with the coarse nodes shown in red. Typical example of the diffusion coefficient $K_\epsilon(\mathbf{x})$ is illustrated by the color shading.

set of edges of a polygonal coarse cell Ω_i, and let $\mathcal{E} = \bigcup_{i=1}^N \mathcal{E}_i$. The elements of $\mathcal{E}$ are considered to be open segments in $\mathbb{R}^2$, such that for any $e \in \mathcal{E}$, either $e \subset \partial\Omega$ or there exists a unique pair of subdomains Ω_e^+ and Ω_e^- from $(\Omega_i)_{i=1}^N$ such that $e \subset \Omega_e^+ \cap \Omega_e^-$. In the latter case, the edge is referred to as an interior edge.

The set of coarse grid nodes is defined by $(\mathfrak{s}_\alpha)_{\alpha=1}^{\mathscr{N}_\Gamma} = \bigcup_{e\in\mathcal{E}} \partial e$. An example of the coarse partitioning and corresponding coarse grid nodes is shown in Figure 1. For every subdomain i, let $\{\alpha_i^1, \alpha_i^2, \dots, \alpha_i^{\mathscr{N}_{\partial\Omega_i}}\}$ be the set of the indices of its nodes.

The main idea behind the substructured formulation that we present below is to replace the original problem (1) by one that only involves the unknown Dirichlet data g defined in Γ. This new unknown function is determined by imposing a matching flux condition along the interior edges of the skeleton Γ. For this end, we introduce the Dirichlet-to-Neumann (DtN) maps associated to elements of the domain partitioning. Let us consider a local Dirichlet problem

$$\left\{ \begin{array}{rcll} au_i + \operatorname{div}\left(K_\epsilon(\mathbf{x})\mathbf{F}(u_i, \nabla u_i)\right) &=& 0 & \text{in} \quad \Omega_i, \\ u_i &=& g_i & \text{on} \quad \partial\Omega_i. \end{array} \right. \tag{2}$$

For a regular enough function g_i we formally denote by $\mathrm{DtN}_i(g_i)$ the Neumann traces of u_i on $\partial\Omega_i$, that is

$$\mathrm{DtN}_i(g_i) = K_\epsilon(\mathbf{x})\mathbf{F}(u_i, \nabla u_i)|_{\partial\Omega_i} \cdot \mathbf{n}_i \qquad \text{on} \quad \partial\Omega_i,$$

where $\mathbf{n}_i$ denotes the unit outward normal on $\partial\Omega_i$. Further, for a regular enough g defined on Γ and any interior coarse edge e we formally introduce the local jump operator

$$[\![]\!]_e : g \mapsto \mathrm{DtN}_i(g|_{\partial\Omega_i})|_e + \mathrm{DtN}_j(g|_{\partial\Omega_j})|_e,$$

where $(\Omega_i, \Omega_j) = (\Omega_e^+, \Omega_e^-)$. Similarly, for any exterior edge e, we set

$$[\![]\!]_e : g \mapsto \mathrm{DtN}_i(g|_{\partial\Omega_i})|_e,$$

where Ω_i denotes the unique subdomain such that $e \subset \partial\Omega_i$. The global jump operator $[\![]\!]$ is defined by $[\![g]\!]|_e = [\![g]\!]_e$ for all $e \in \mathcal{E}$. Enforcing zero flux jump at every interior coarse edge, leads to a globally coupled problem that can be written in a weak form as: find g satisfying $g|_{\partial\Omega} = u_D$ such that for all regular enough w

vanishing on $\partial\Omega$ it holds that

$$\int_\Gamma w[\![g]\!]\, d\gamma(\mathbf{x}) = 0. \tag{3}$$

Let us assume, for the moment, that we are in possession of a procedure allowing one to evaluate the DtN operators. Then, the problem (3) can be discretized by taking the test and trial function from a finite dimensional space. Specifically, we consider the space $V_H(\Gamma)$ made of functions that are continuous on Γ and affine on every coarse edge of Γ that is

$$V_H(\Gamma) = \{v \in C^0(\Gamma) \quad \text{s.t.} \quad v|_e \in \mathbb{P}_1(e) \quad \text{for all } \ e \in \mathcal{E}\}. \tag{4}$$

Let us recall that Γ contains both the union of subdomain interfaces and $\partial\Omega$. Let $V_{H,0}(\Gamma) = \{v \in V_H(\Gamma) \quad \text{s.t.} \quad v|_{\partial\Omega} = 0\}$ and let $u_{D,H} \in V_H(\Gamma)$ denote some approximation of u_D, e.g. obtained by interpolation. To form the approximate substructured problem in $V_H(\Gamma)$, one replaces (3) by: find g_H satisfying $g_H|_{\partial\Omega} = u_{H,D}$ and

$$\int_\Gamma w_H[\![g_H]\!]\, d\gamma(\mathbf{x}) = 0 \qquad \forall w_H \in V_{H,0}(\Gamma). \tag{5}$$

Let us reformulate (5) in a way that highlights its local and global features. Let $(\phi^\alpha)_{\alpha=1}^{\mathscr{N}_\Gamma}$ denote the nodal basis of $V_H(\Gamma)$ made of "hat" functions associated to the set of the coarse grid nodes $(\mathfrak{s}_\alpha)_{\alpha=1}^{\mathscr{N}_\Gamma}$ (see Figure 1 for visual). More precisely, the basis function $\phi^\alpha \in V_H(\Gamma)$ is assumed to satisfy

$$\phi^\alpha(\mathfrak{s}_\beta) = \begin{cases} 1, & \alpha = \beta, \\ 0, & \alpha \neq \beta, \end{cases}$$

for all coarse grid nodes $\mathfrak{s}_\beta$. Given this nodal basis in $V_H(\Gamma)$ we define the function reconstruction operator π_H from $\mathbb{R}^{\mathscr{N}_\Gamma}$ to $V_H(\Gamma)$ by $\pi_H(\mathbf{v}) = \sum_{\alpha=1}^{\mathscr{N}_\Gamma} \mathbf{v}^\alpha \phi^\alpha$. We note that π_H is bijective, and thus admits an inverse denoted by π_H^{-1}.

Setting $w_H(\mathbf{x}) = \sum_{\alpha=1}^{\mathscr{N}_\Gamma} \mathbf{w}^\alpha \phi^\alpha(\mathbf{x})$ and $g_H(\mathbf{x}) = \sum_{\beta=1}^{\mathscr{N}_\Gamma} \mathbf{g}^\beta \phi^\beta(\mathbf{x})$, the left-hand side of (5) can be expressed as

$$\begin{aligned} \int_\Gamma w_H[\![g_H]\!]\, d\gamma(\mathbf{x}) &= \sum_{i=1}^N \int_{\partial\Omega_i} w_H|_{\partial\Omega_i} \mathrm{DtN}_i\left(g_H|_{\partial\Omega_i}\right) d\gamma(\mathbf{x}) \\ &= \sum_{i=1}^N \sum_{\alpha=1}^{\mathscr{N}_\Gamma} \mathbf{w}^\alpha \int_{\partial\Omega_i} \phi^\alpha|_{\partial\Omega_i} \mathrm{DtN}_i\left(\sum_{\beta=1}^{\mathscr{N}_\Gamma} \mathbf{g}^\beta \phi^\beta|_{\partial\Omega_i}\right) d\gamma(\mathbf{x}). \end{aligned}$$

With the local ordering of the nodal degrees of freedom and using the fact the basis functions are locally supported, we further obtain

$$\int_\Gamma w_H[\![g_H]\!]\, d\gamma(\mathbf{x}) = \sum_{i=1}^N \sum_{n=1}^{\mathscr{N}_{\partial\Omega_i}} \mathbf{w}^{\alpha_i^n} \int_{\partial\Omega_i} \phi^{\alpha_i^n}|_{\partial\Omega_i} \mathrm{DtN}_i\left(\sum_{m=1}^{\mathscr{N}_{\partial\Omega_i}} \mathbf{g}^{\alpha_i^m} \phi^{\alpha_i^m}|_{\partial\Omega_i}\right) d\gamma(\mathbf{x}). \tag{6}$$

Let us denote by $\mathrm{DtN}_{H,i}$ the mapping from $\mathbb{R}^{\mathscr{N}_{\partial\Omega_i}}$ to itself such that the l-th component of $\mathrm{DtN}_{H,i}$ is given by

$$\mathrm{DtN}^l_{H,i}(\mathbf{g}) = \int_{\partial\Omega_i} \phi^{\alpha_i^l}|_{\partial\Omega_i} \mathrm{DtN}_i\left(\sum_{m=1}^{\mathscr{N}_{\partial\Omega_i}} \mathbf{g}^m \phi^{\alpha_i^m}|_{\partial\Omega_i}\right) d\gamma(\mathbf{x}). \tag{7}$$

The mapping $\mathrm{DtN}_{H,i}$ combines the action of the local DtN operator with the projection on the coarse space. In view of (6) and (7) we have

$$\int_\Gamma w_H [\![g_H]\!] \, d\gamma(\mathbf{x}) = \sum_{i=1}^{N} \sum_{l=1}^{\mathscr{N}_{\partial\Omega_i}} \mathbf{w}^{\alpha_i^l} \mathrm{DtN}_{H,i}^{l} \left(\left(\mathbf{g}^{\alpha_i^m} \right)_{m=1,\ldots,\mathscr{N}_{\partial\Omega_i}} \right). \tag{8}$$

The reformulation (8) shows that the left-hand-side of (5) can be assembled by adding local contributions from the "coarse DtN maps".

Further, we denote by $\mathscr{R}_\Gamma^{\partial\Omega_i}$ be a boolean restriction matrix "from Γ to $\partial\Omega_i$" and by $\mathscr{R}_{\partial\Omega_i}^{\Gamma}$ its transpose. Specifically, $\mathscr{R}_\Gamma^{\partial\Omega_i}$ is $\mathscr{N}_{\partial\Omega_i}$ by $\mathscr{N}_\Gamma$ matrix with components satisfying

$$\left(\mathscr{R}_\Gamma^{\partial\Omega_i} \right)_{n\alpha} = \begin{cases} 1 & \text{if} \quad \alpha = \alpha_i^n, \\ 0 & \text{else.} \end{cases}$$

Setting $\boldsymbol{w} = \pi_H^{-1}(w_H)$, $\boldsymbol{g} = \pi_H^{-1}(g_H) \in \mathbb{R}^{\mathscr{N}_\Gamma}$, we express (8) as

$$\int_\Gamma [\![g_H]\!] w_H \, d\gamma(\mathbf{x}) = \sum_{i}^{N} \left(\mathscr{R}_\Gamma^{\partial\Omega_i} \boldsymbol{w} \right)^T \mathrm{DtN}_{H,i} \left(\mathscr{R}_\Gamma^{\partial\Omega_i} \boldsymbol{g} \right).$$

Let $\mathscr{R}_\Gamma^{\Gamma\cap\Omega}$ denote a boolean restriction matrix "from Γ to $\Gamma \cap \Omega$"; more precisely, let $\mathscr{N}_{\Gamma\cup\Omega}$ denote the total number of coarse nodes lying in Ω (but not on $\partial\Omega$), then $\mathscr{R}_\Gamma^{\Gamma\cap\Omega}$ is $\mathscr{N}_{\Gamma\cup\Omega}$ by $\mathscr{N}_\Gamma$ matrix with all zero entries except for its diagonal. Setting $\mathscr{R}_{\partial\Omega_i}^{\Gamma\cap\Omega} = \mathscr{R}_\Gamma^{\Gamma\cap\Omega} \mathscr{R}_{\partial\Omega_i}^{\Gamma}$, we reformulate the approximate substructuring problem (5) as follows:

$$\text{Find } \mathbf{g} \in \mathbb{R}^{\mathscr{N}_\Gamma} \text{ satisfying } \pi_H(\boldsymbol{g})|_{\partial\Omega} = u_{H,D} \text{ and } \sum_{i=1}^{N} \mathscr{R}_{\partial\Omega_i}^{\Gamma\cap\Omega} \mathrm{DtN}_{H,i} \left(\mathscr{R}_\Gamma^{\partial\Omega_i} \boldsymbol{g} \right) = 0. \tag{9}$$

2. Learning coarse Dirichlet-to-Neumann maps

By discretizing the variational equality (3), we have derived the system of nonlinear algebraic equations (9), which can be solved using a fixed-point algorithm. In particular, we have in mind Newton's method, which requires repeated evaluations of $\mathrm{DtN}_{H,i}$ and its partial derivatives.

Since the definition of $\mathrm{DtN}_{H,i}$ involves the "continuous" operator DtN_i, the former generally cannot be evaluated exactly. However, $\mathrm{DtN}_{H,i}$ can be approximated by solving (2) numerically, for example, using a Finite Element method. We remark that the numerical solution of (2) would be required at every evaluation of $\mathrm{DtN}_{H,i}$. In addition, because of the oscillatory diffusion coefficient $K_\epsilon(\mathbf{x})$ the discretization method for solving the local problems (2) requires a high resolution, making the approximation of $\mathrm{DtN}_{H,i}$ computationally demanding. This motivates us to replace the operator $\mathrm{DtN}_{H,i}$ with a faster to evaluate surrogate model based on the artificial neural network.

Let $\widetilde{\mathrm{DtN}}_{H,i}$ denote an approximation of $\mathrm{DtN}_{H,i}$ to which we will refer to as a surrogate model, and which will be introduced in more details below. By replacing in (9) the original operator $\mathrm{DtN}_{H,i}$ by $\widetilde{\mathrm{DtN}}_{H,i}$ we obtain the following surrogate problem:

$$\text{Find } \mathbf{g} \in \mathbb{R}^{\mathscr{N}_\Gamma} \text{ satisfying } \pi_H(\boldsymbol{g})|_{\partial\Omega} = u_{H,D} \text{ and } \sum_{i=1}^{N} \mathscr{R}_{\partial\Omega_i}^{\Gamma\cap\Omega} \widetilde{\mathrm{DtN}}_{H,i} \left(\mathscr{R}_\Gamma^{\partial\Omega_i} \boldsymbol{g} \right) = 0. \tag{10}$$

We construct the mapping $\widetilde{\mathrm{DtN}}_{H,i}$ by learning the individual components of the original map. This is achieved by training a fully-connected feed-forward neural network. Classically, each layer performs an affine transformation followed by a nonlinear activation function $\sigma : \mathbb{R} \mapsto \mathbb{R}$, so that each component $\mathrm{DtN}^l_{H,i}$, $l = 1, \dots, \mathscr{N}_{\partial\Omega_i}$, is approximated by a composition

$$\widetilde{\mathrm{DtN}}^l_{H,i}(\theta; \mathbf{u}) = \varphi^k_\theta \circ \dots \circ \varphi^1_\theta(\mathbf{u}),$$

where $\varphi^p_\theta(\mathbf{u}) = \sigma\left(W_p(\theta)\mathbf{u} + \mathbf{b}_p(\theta)\right)$ for all $p = 1, \dots, k$ and θ denotes the collection of neural network parameters $\{W_p, \mathbf{b}_p, 1 \le p \le k\}$.

The weights matrices W_p and the biases $\mathbf{b}_p$ are determined through supervised learning based on numerically generated values of $\mathrm{DtN}_{H,i}$ and its derivatives. Let $(\mathbf{u}^s)_{s=1}^{n_s}$ be a set of sampling points from a training domain $[u_{\min}, u_{\max}]^{\mathscr{N}_{\partial\Omega_i}}$. The loss function is defined as

$$\mathcal{L}^l_i(\theta, (\mathbf{u}^s)_{s=1}^{n_s}) = c_0 \mathcal{L}^l_{i,0}(\theta, (\mathbf{u}^s)_{s=1}^{n_s}) + c_1 \mathcal{L}^l_{i,1}(\theta, (\mathbf{u}^s)_{s=1}^{n_s}) + c_{\mathrm{mon}} \mathcal{L}^l_{i,\mathrm{mon}}(\theta), \tag{11}$$

where c_0, c_1 and c_{mon} are some positive weights, while $\mathcal{L}^l_{i,0}$, $\mathcal{L}^l_{i,1}$ and $\mathcal{L}^l_{i,\mathrm{mon}}$ are defined by (12), (13), (14) and (15) below. Specifically, the model error over the sampling set is measured by

$$\mathcal{L}^l_{i,0}(\theta, (\mathbf{u}^s)_{s=1}^{n_s}) = \frac{1}{n_s} \sum_{s=1}^{n_s} \left(\widetilde{\mathrm{DtN}}^l_{H,i}(\theta; \mathbf{u}^s) - \mathrm{DtN}^l_{H,i}(\mathbf{u}^s) \right)^2. \tag{12}$$

In addition, we measure the model's error in terms of its partial derivatives with respect to the input:

$$\mathcal{L}^l_{i,1}(\theta, (\mathbf{u}^s)_{s=1}^{n_s}) = \frac{1}{n_s} \sum_{s=1}^{n_s} \sum_{k=1}^{\mathscr{N}_{\partial\Omega_i}} \left(\partial_{u_k} \widetilde{\mathrm{DtN}}^l_{H,i}(\theta; \mathbf{u}^s) - \partial_{u_k} \mathrm{DtN}^l_{H,i}(\mathbf{u}^s) \right)^2. \tag{13}$$

Finally, we introduce the monotonously loss terms defined either by

$$\mathcal{L}^l_{i,\mathrm{mon}}(\theta) = \int_{(u_{\min}, u_{\max})^{\mathscr{N}_{\partial\Omega_i}}} \left(\left(\partial_{u_l} \widetilde{\mathrm{DtN}}^l_{H,i}(\theta; \mathbf{u}) \right)^- \right)^2 + \sum_{k=1 \,|\, k \neq l}^{\mathscr{N}_{\partial\Omega_i}} \left(\left(\partial_{u_k} \widetilde{\mathrm{DtN}}^l_{H,i}(\theta; \mathbf{u}) \right)^+ \right)^2 \mathrm{d}\mathbf{u}, \tag{14}$$

or simply by

$$\mathcal{L}^l_{i,\mathrm{mon}}(\theta) = \int_{(u_{\min}, u_{\max})^{\mathscr{N}_{\partial\Omega_i}}} \left(\left(\partial_{u_l} \widetilde{\mathrm{DtN}}^l_{H,i}(\theta; \mathbf{u}) \right)^- \right)^2 \mathrm{d}\mathbf{u}, \tag{15}$$

with $(x)^+ = \max(x, 0)$ and $(x)^- = \min(x, 0)$ for any real x.

We note that in contrast to $\mathcal{L}_{i,0}$ and $\mathcal{L}_{i,1}$, the monotonicity loss term $\mathcal{L}_{i,\mathrm{mon}}$ does not require any training data. The term $\mathcal{L}^l_{i,\mathrm{mon}}$ is added to the loss function to enforce certain monotonicity properties expected from the original $\mathrm{DtN}_{H,i}$ maps by penalize incorrect Jacobian signs. Specifically, in the one-dimensional case, we anticipate that the Jacobian of $\mathrm{DtN}_{H,i}$ should have positive diagonal entries and non-positive off-diagonal entries. For two-dimensional problems, however, the off-diagonal entries of $\mathrm{DtN}'_{H,i}$ do not have a specific sign in general. Which is why, for problems in 2D, we will be using (15) instead of (14). By incorporating the monotonicity constraint through (14) or (15), we improve the robustness of Newton's method for solving (10), which would otherwise be likely to fail. The integrals in (14) and (15) over the training domain (which is an $\mathscr{N}_{\partial\Omega_i}$ dimensional cube) will be approximated numerically.

For every sampling point $\mathbf{u}^s$ from $(\mathbf{u}^s)_{s=1}^{n_s}$ we generate the approximate values of $\mathrm{DtN}_{\mathrm{H,i}}(\mathbf{u}^s)$ and its Jacobian $\mathrm{DtN}'_{\mathrm{H,i}}(\mathbf{u}^s)$ by solving a nonlinear algebraic system resulting from the fine-scale finite element discretization of (2). The training points are sampled uniformly from $[u_{min}, u_{max}]^{\mathscr{N}_{\partial\Omega_i}}$. Specifically, given a sampling range $[u_{min}, u_{max}]$, each component of the input vector is sampled with m entries, equally spaced between u_{min} and u_{max} leading to $n_s = m^{\mathscr{N}_{\partial\Omega_i}}$ sampling points in total. Occasionally, the training set may be enriched with few additional points.

3. Numerical experiments

3.1. 1D degenerate elliptic problem

We first consider the following nonlinear equation

$$au - \left(K_\epsilon(x)\left(u^4\right)'\right)' = 0 \tag{16}$$

posed in the domain $(0, 1)$ with some non-negative Dirichlet boundary conditions. The diffusion coefficient is given by

$$K_\epsilon(x) = 10^{-2} + \frac{1}{2}\left(1 + \sin\left(50\pi x + \frac{\pi}{4}\right)\right),$$

and $a = 20$. We note that the case $a = 0$ is much simpler, as a change of variable $v = u^4$ would allow to rephrase (16) as a linear equation. Equation (16) can be interpreted as a stationary variant of the porous medium equation [18]. It can be shown that u takes values in $[0, \max(u(0), u(1)]$. We note that, since the diffusion term in (16) degenerates at $u = 0$, the solutions of (16) may vanish over some portion of the domain. The domain $(0, 1)$ is partitioned in five subdomains of equal size. We refer to Figure 2 for the illustration of the partitioning and the coefficient K_ϵ. Because the partitioning of the domain is matching periodicity

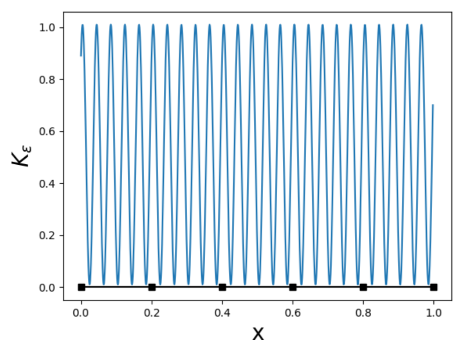

Figure 2. Partitioning of the model domain and the plot of $K_\epsilon(x)$

of $K_\epsilon(x)$, the Dirichlet-to-Neumann operators coincide for all of the subdomains. We also note that in the one-dimensional case the mapping DtN_i, acting from $\mathbb{R}^2$ to itself, coincides with the coarse operator $\mathrm{DtN}_{H,i}$, since there is no approximation of the subdomain's Dirichlet data. As the analytical expression of the local Dirichlet-to-Neumann operator is not available, the latter will be replaced by the approximate one computed by a finite element method using 200 grid points. To be more specific, we use $\mathbb{P}_1$ finite element method with mass lumping. For the surrogate model $\widetilde{\mathrm{DtN}}_{H,i}$ we use fully connected neural network with two hidden layers of 64 neurons each, leading to the total of 4417 parameters. The activation function used in all the numerical experiments is given by $\sigma(u) = \max(u, 0)^2$. This choice of activation function ensures the necessary regularity when higher-order loss terms (13) and (14) are active. The model is trained over the domain $[0, u_{max}]^2$ with $u_{max} = 4$. Training points are sampled over a regular grid in $[0, u_{max}]^2$ consisting of $n_s = 2^2, 3^2, 4^2$ and 5^2 points. The loss function involves both values and derivatives of $\mathrm{DtN}_{H,i}$ at the sampling points, as well as the monotonicity component (14). Unless stated otherwise, we set $c_0 = 1$, $c_1 = 0.1$, and $c_{mon} = 4$. The integral in (14) is approximated numerically based on integrand values over a regular 40×40 grid.

Let us begin with a qualitative analysis of the surrogate $\widetilde{\mathrm{DtN}}_{H,i}$ model. Recall that, in 1D, $\mathrm{DtN}_{H,i}$ maps $\mathbb{R}^2$ to itself, which can be interpreted as mapping left and right Dirichlet data to a pair of left and right fluxes. We report on Figure 3 the left flux and its surrogates as the function of the input Dirichlet values. The surface $z = \mathrm{DtN}_{H,i}((u_{\text{left}}, u_{\text{right}}))^1$ shown as solid is the same for all sub-figures. Colored wireframe surfaces show $z = \widetilde{\mathrm{DtN}}_{H,i}((u_{\text{left}}, u_{\text{right}}))^1$ for various number of sampling points (4, 5 and 9 for the first, second and third rows respectively), and different values of coefficients c_1 and c_{mon} (same for each column). The training points are depicted as black dots. The results obtained using DtN values alone ($c_1 = c_{mon} = 0$) are shown in the first column, those obtained using both values and derivatives ($c_{mon} = 0$) are reported in the middle column. Finally, the right column shows the outputs of the model using the additional monotonicity loss term (14). For the sake of clearer visualization, here, the parameter a is set to 1. Unsurprisingly, fitting not only the values, but also the tangential plains has a drastic benefit on the quality of the surrogate model (compare left and middle columns). Models trained with $c_1 > 0$ manage to capture the shape of the response surface reasonably well for very few training points. We also observe that adding the monotonicity loss does not seem to pollute the interpolation quality (middle versus right columns).

We report on Figure 4 the relative mismatch between $\widetilde{\mathrm{DtN}}_{H,i}$ and $\mathrm{DtN}_{H,i}$, measured in the $L^2((0, u_{max})^2)$ norm, as a function of the number of sampling points. The error is evaluated over a regular 20×20 grid in $[0, u_{max}]^2$. Here, in addition to the training points sampled over the regular grid, we include supplementary points taken at the centers of the sampling grid cells. The decrease of error deteriorates after approximately 25 sampling points. This saturation effect (around the relative error value of $10^{-3.5}$) is even more pronounced in the second test case (see Figure 8 below). The failure in achieving higher accuracy should probably be attributed to our current first-order optimization algorithm, combined with the fact that the structure of the surrogate model is fixed through the whole experiment.

We further evaluate the performance of the surrogate models in approximating the solution to (16) using the substructured formulation. We set $u(1) = 0$, and we let $u(0)$ taking values in $\{1, 2, 3, 4\}$. Once the coarse nodal values are computed from (9) or (10) we reconstruct the solution within the subdomains by numerically solving the local problems (2) on each Ω_i. Figure 5 shows the reconstructed solution using both the original and the surrogate DtN models for different boundary conditions and varying numbers of sampling points. Starting from $n_s = 3^2$, the surrogate model produced solutions that visually appear accurate. The relative $L^2(0, 1)$ error as a function of the number of sampling points is reported in Figure 4.

The numerical solution of (9) and (10) is obtained using Newton's method. In Figure 6, we present the typical convergence history of Newton's method for both the original model (9) and the surrogate model (10). For this example, we set the boundary condition to $u(0) = 4$ and used $n_s = 4^2$ sampling points. Here, the use of the surrogate model leads to convergence of Newton's method with slightly fewer iterations. Most importantly, the inference time of $\widetilde{\mathrm{DtN}}_{H,i}$ is much lower than the time required to evaluate $\mathrm{DtN}_{H,i}$.

3.2. 1D $p-$Laplace problem

Next, we consider the following equation

$$au - \left(K_\epsilon(x)|u'|^2 u'\right)' = 0 \tag{17}$$

over the interval $(0, 1)$. All the numerical parameters (including c_0, c_1 and c_{mon}) are set up as in the previous test case, except for the coefficient a set to 5. Again, we compare the solution obtained using either the original or the surrogate substructured problems, (9) and (10), respectively. The solution comparison for different boundary conditions and varying sampling point sizes is reported in Figure 7. Starting from $n_s = 3^2$ the solution obtained using the surrogate model appears to be reasonably accurate. Figure 8 shows the convergence of the relative interpolation and solution errors in the $L^2((0, u_{max})^2)$ and $L^2(0, 1)$ norms, respectively. As before, we observe the saturation of the interpolation error as the function of n_s; this phenomenon is even more pronounced than in the previous test case. The typical convergence history of Newton's method is shown in Figure 9, with $u(0) = 4$ and $n_s = 4^2$. Although Newton's method requires twice as many iterations for the surrogate model compared to the original one, the former leads to a faster algorithm because of a much lower inference time.

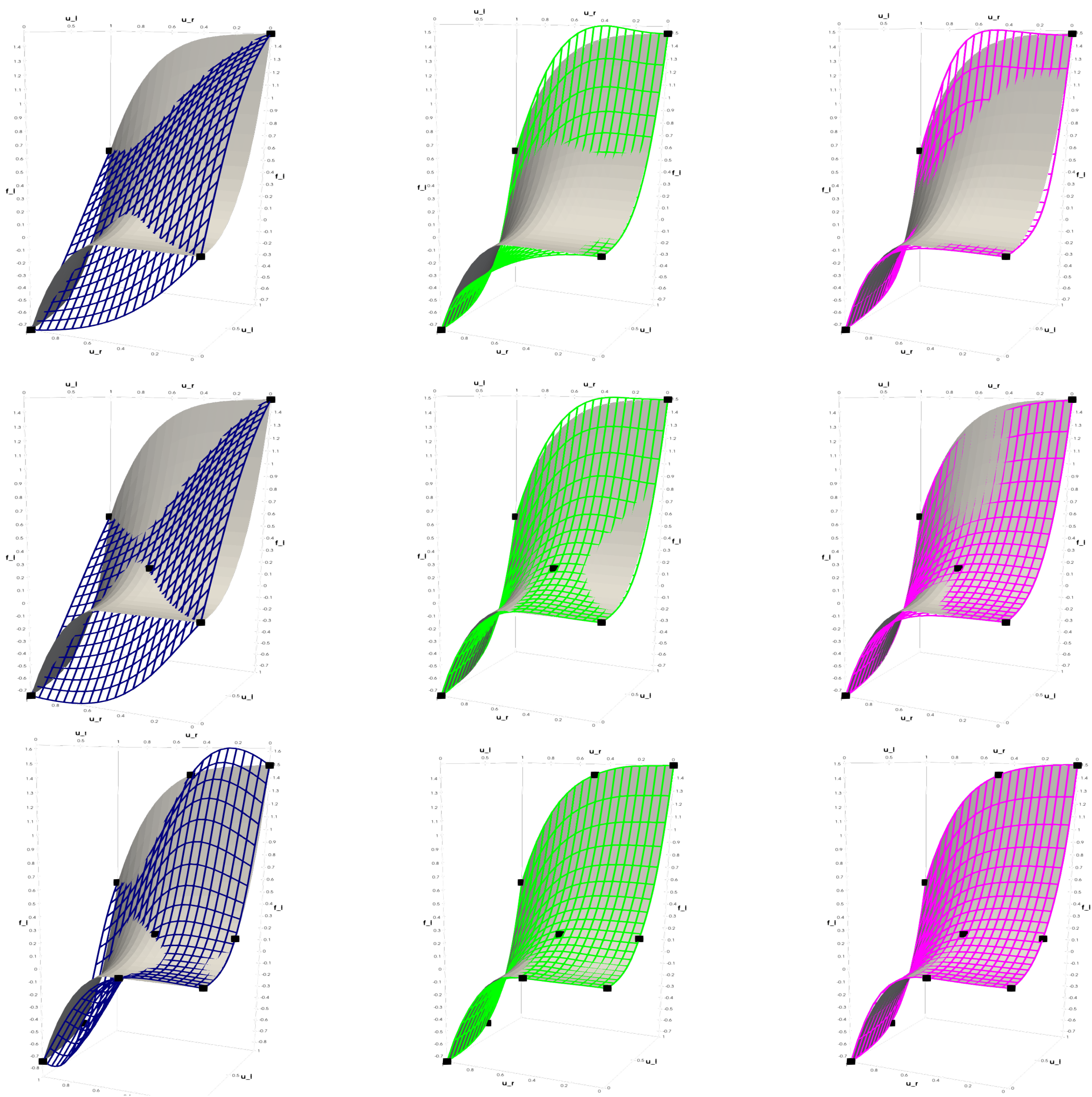

FIGURE 3. Qualitative comparison of the surfaces $z = \mathrm{DtN}^1_{H,i}((u_{\mathrm{left}}, u_{\mathrm{right}}))$ (solid) and $z = \widetilde{\mathrm{DtN}}^1_{H,i}((u_{\mathrm{left}}, u_{\mathrm{right}}))$ (wireframe). Left column (blue): only DtN values included. Middle column (green): DtN and its derivatives included. Right column (purple): DtN values, derivatives, and monotonicity loss included. Top to bottom: $n_s = 2^2$ training points, $n_s = 2^2 + 1$ training points, $n_s = 3^2$ training points. The training points are shown as black dots.

3.3. **2D degenerate elliptic problem**

Finally, we consider a bi-dimensional version of (16) given by

$$u - \operatorname{div}\left(K_\epsilon(x, y)\nabla(u^4)\right) = 0 \quad \text{in} \quad \Omega = (0,1)^2, \tag{18}$$

FIGURE 4. Left: Relative mismatch between $\widetilde{\mathrm{DtN}}_{H,i}$ and $\mathrm{DtN}_{H,i}$ measured in $L^2((0, u_{max})^2)$. The error is presented on a logarithmic scale. Right: The relative error of the solution obtained using the surrogate model; the blue curve corresponds to the mismatch of the coarse nodal values, while the red curve shows the error of the reconstructed fine-scale solution in the $L^2((0, 1))$.

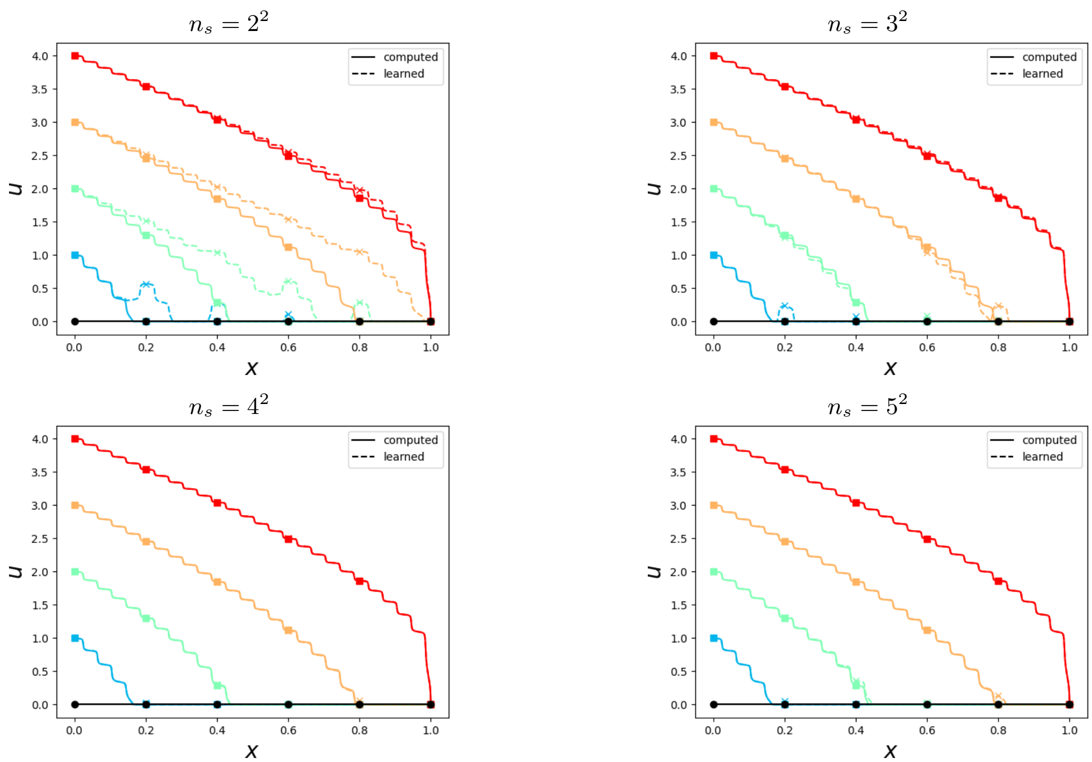

FIGURE 5. Approximate solution of (16) for various left boundary conditions using $\mathrm{DtN}_{H,i}$ (solid) and $\widetilde{\mathrm{DtN}}_{H,i}$ (dashed) operators for $n_s = 2^2, 3^2, 4^2$ and 5^2.

combined with the Dirichlet boundary conditions

$$u(x, y) = \max(u_{max}(x + y - 1), 0) \qquad \text{on} \qquad \partial\Omega$$

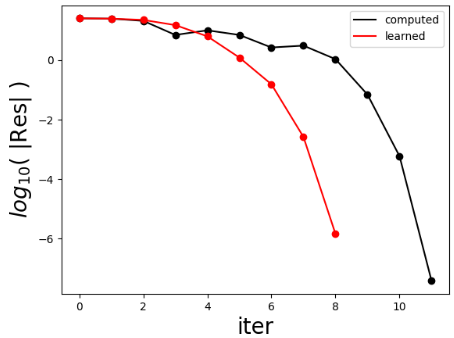

FIGURE 6. Convergence of Newton's method using $\mathrm{DtN}_{H,i}$ (black) and $\widetilde{\mathrm{DtN}}_{H,i}$ (red) operators for 1D degenerate elliptic problem.

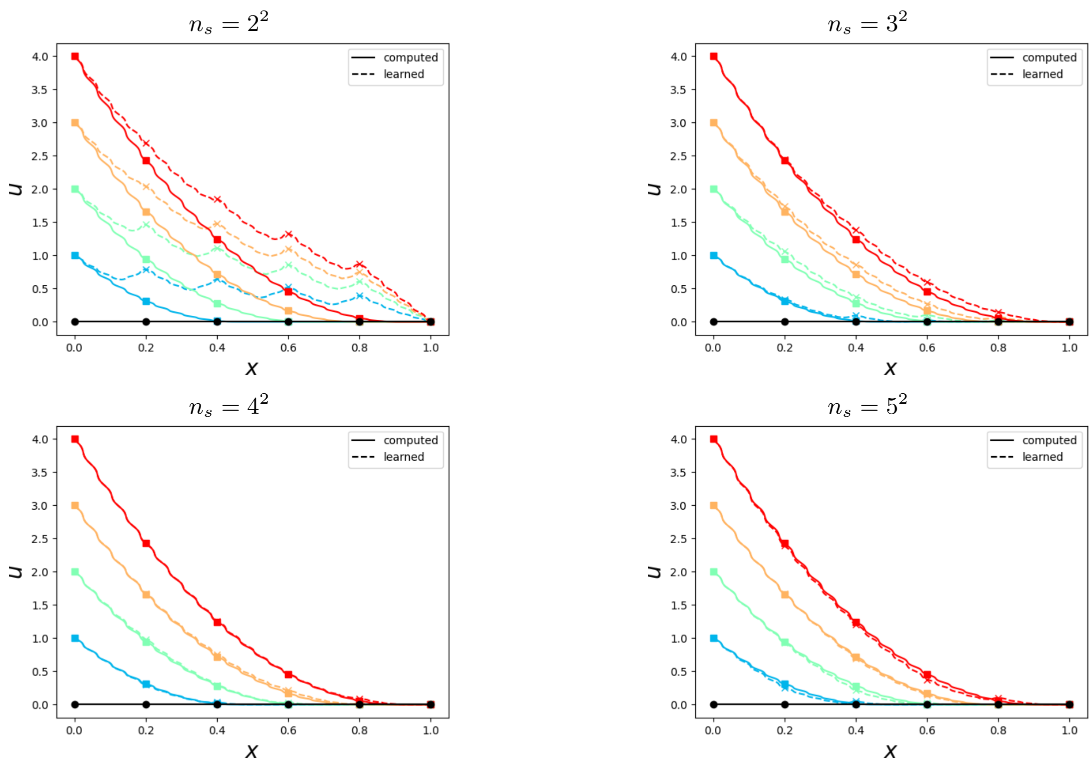

FIGURE 7. Approximate solution of (17) for various left boundary conditions using $\mathrm{DtN}_{H,i}$ (solid) and $\widetilde{\mathrm{DtN}}_{H,i}$ (dashed) operators for $n_s = 2^2, 3^2, 4^2$ and 5^2

with $u_{max} = 1.2$. That is, $u(x, y)$ is set to zero on the lower and the left boundaries, and to $u_{max}(x + y - 1)$ on the right and the top boundaries. We set

$$K_\epsilon(x, y) = 10^{-2} + \frac{1}{2}\left(1 + \sin\left(10x + \frac{\pi}{2}\right)\sin\left(5y + \frac{\pi}{2}\right)\right),$$

and consider a regular partitioning of Ω into 5×5 subdomains (see Figure 11 for illustration). Here again, since the partitioning of the domain matches the periodicity of the coefficient K_ϵ, one only needs to build a

FIGURE 8. Left: Relative mismatch between $\widetilde{\mathrm{DtN}}_{H,i}$ and $\mathrm{DtN}_{H,i}$ measured in $L^2((0,u_{max})^2)$. The error is presented on a logarithmic scale. Right: The relative error of the solution obtained using the surrogate model; the blue curve corresponds to the mismatch of the coarse nodal values, while the red curve shows the error of the reconstructed fine-scale solution in the $L^2((0,1))$.

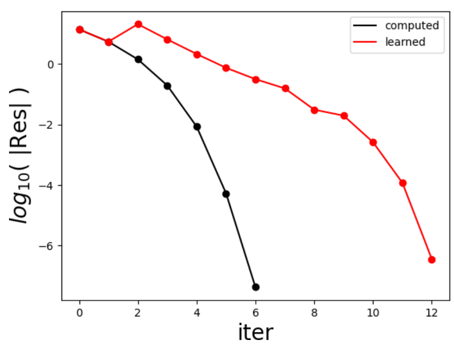

FIGURE 9. Convergence of Newton's method using DtN (black) and $\widetilde{\mathrm{DtN}}$ (red) operators for 1D $p-$Laplace problem.

single surrogate DtN model. More precisely, for every component of $\mathrm{DtN}^l_{H,i}$, we train a specific model $\widetilde{\mathrm{DtN}}^l_{H,i}$ using a fully connected neural network with 2 hidden layers of 20 neurons each and the activation function $\sigma(u) = \max(u,0)^2$.

Figure 10 illustrates the qualitative behavior of $\mathrm{DtN}_{H,i}$ and $\widetilde{\mathrm{DtN}}_{H,i}$ as a function of the input values associated with the four corners of the coarse cell. Specifically, the figure shows the first component of both DtN maps corresponding to the lower left corner of the coarse cell. The input values at the lower left and the upper right corners are varied, while the values at the remaining two corners are set to zero. We observe a good qualitative match between $\mathrm{DtN}_{H,i}$ and the surrogate. Table 1 reports the relative mismatch between $\mathrm{DtN}_{H,i}$ and $\widetilde{\mathrm{DtN}}_{H,i}$ measured in $L^2((0,u_{max})^4)^4$ norm. The error is evaluated numerically based on a regular grid over $(0,u_{max})^4$ with 10 points by dimension.

n_s	2^4	3^4	4^4	5^4
Rel. error	6.2%	0.72%	0.51%	0.40%

TABLE 1. The relative mismatch between $\mathrm{DtN}_{H,i}$ and $\widetilde{\mathrm{DtN}}_{H,i}$ measured in $L^2((0,u_{max})^4)^4$ norm.

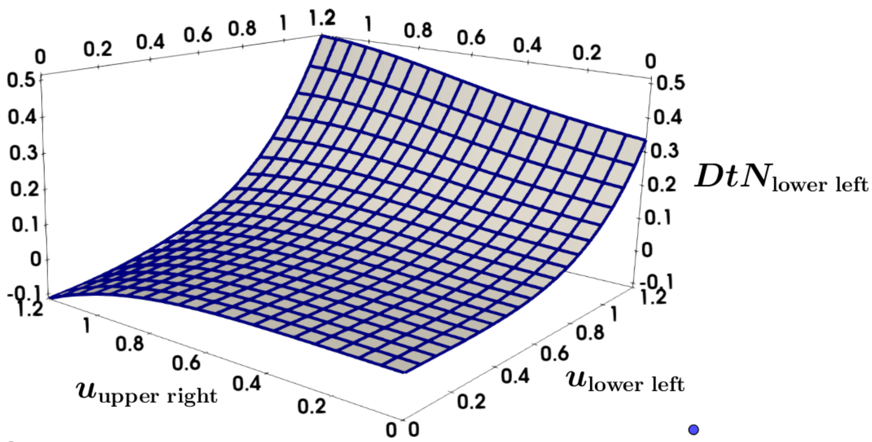

FIGURE 10. Qualitative comparison of the surfaces $z = \mathrm{DtN}^1_{H,i}((u_{\text{lower left}}, 0, u_{\text{upper right}}, 0))$ (solid) and $z = \widetilde{\mathrm{DtN}}^1_{H,i}((u_{\text{lower left}}, 0, u_{\text{upper right}}, 0))$ with 5^4 training points (wireframe).

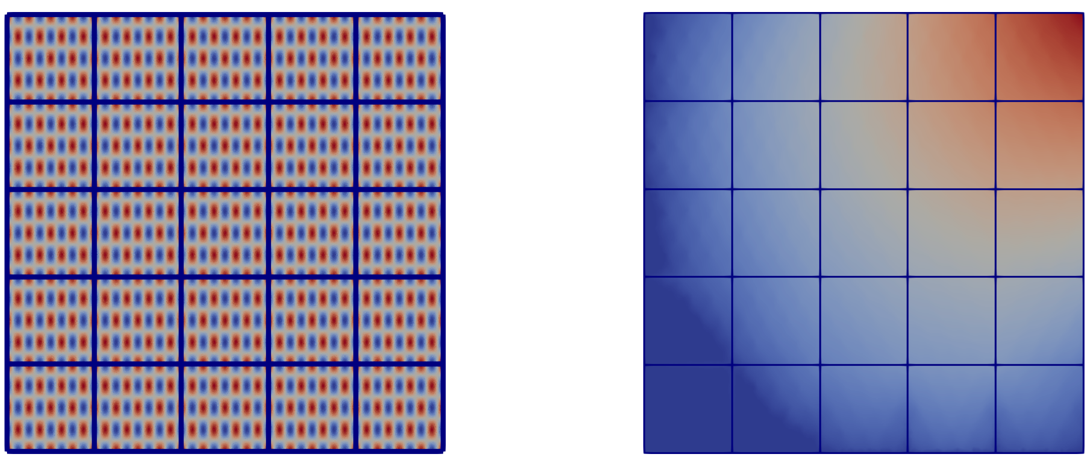

FIGURE 11. Left: Diffusion coefficient $K_\epsilon(x)$. Right: Approximate solution of the model equation (18) using the original $\mathrm{DtN}_{H,i}$ operator.

The reference $\mathrm{DtN}_{H,i}$ operator is computed using the mass lumped $\mathbb{P}_1$ finite element method on a fine grid with approximately 660 triangles and 340 nodal degrees of freedom by subdomain. We report on Figure 11 the approximate solution obtained using this reference map. The training data is sampled on a regular grid over $[0, u_{max}]^4$. The loss function uses both values and derivatives of $\mathrm{DtN}^l_{H,i}$ at the sampling points, with $c_0 = 1$ and $c_1 = 0.1$. The monotonicity of $\widetilde{\mathrm{DtN}}^l_{H,i}$ is enhanced by adding the loss term (15) with $c_{mon} = 10$. The integral in (15) is approximated by the Monte Carlo method, using 200 random points sampled in $[0, u_{max}]^4$ at every optimization step.

The approximate solution obtained using the surrogate models trained with $n_s = 2^4, 3^4, 4^4$ and 5^4 are reported on Figure 12, together with solution profiles along the line $x = y$ shown on Figure 13. Table 2 reports the relative $L^2(\Omega)$ error between the solution obtained using the reference $\mathrm{DtN}_{H,i}$ operator and the surrogate

$\widetilde{\mathrm{DtN}}_{H,i}$. We observe that the error decreases as the number of sampling points increases, with the relative $L^2(\Omega)$ error falling below 10% starting from $n_s = 3^4$.

n_s	2^4	3^4	4^4	5^4
Rel. error	33%	8%	4%	3%

Table 2. The relative $L^2(\Omega)$ error between the solution obtained using the surrogate model and the true model.

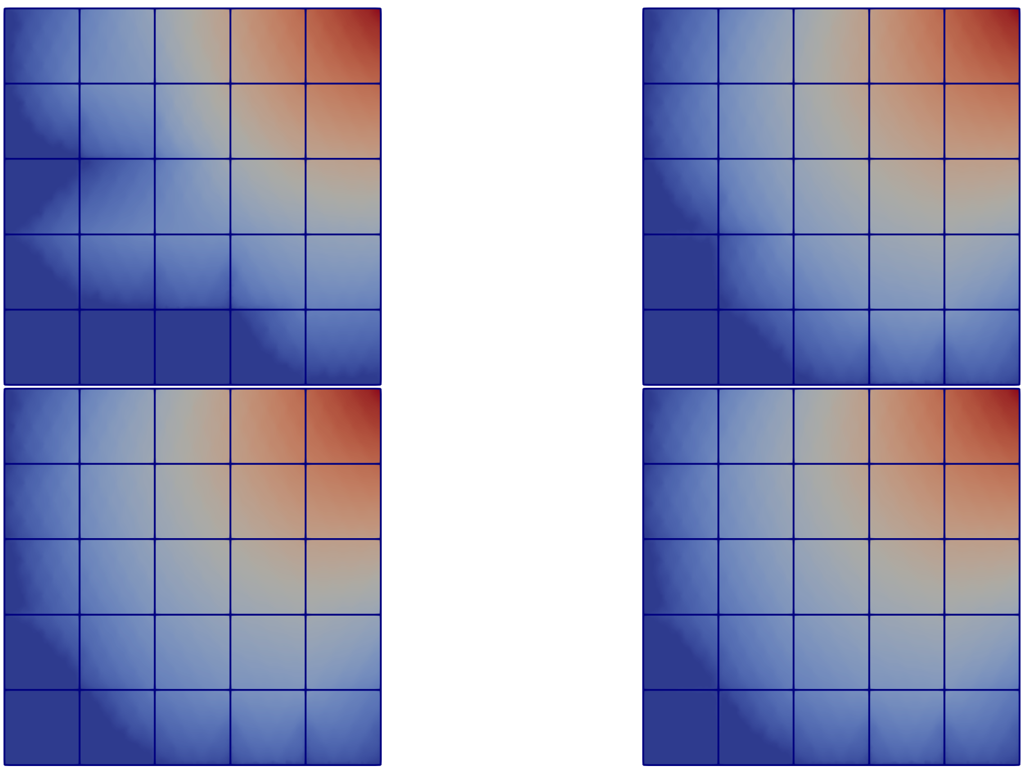

Figure 12. Approximate solution of the model equation (18) using the surrogate $\widetilde{\mathrm{DtN}}$ map for $n_s = 2^4$ (top left), $n_s = 3^4$ (top right), $n_s = 4^4$ (bottom left), and $n_s = 5^4$ (bottom right).

Let us recall that the evaluation of the residual function based on the original coarse $\mathrm{DtN}_{H,i}$ operator requires the solution of the family of local nonlinear finite element systems, which is computationally demanding. In contrast, the inference time of $\widetilde{\mathrm{DtN}}_{H,i}$ is very low, allowing for fast computation of the solution to the surrogate system (10). Specifically, the average computational time required to evaluate the reference $\mathrm{DtN}_{H,i}$ was observed to be 2.4 seconds, whereas the surrogate model's average inference time was approximately 0.07 seconds.

Although the surrogate solution may not be sufficiently accurate, it can serve to initialize the nonlinear solver for the original coarse problem (9). The performance of the algorithm based on this idea is illustrated in Figure 14. Figure 14 shows the convergence of the relative $L^2(\Omega)$ error between the reference numerical solution to (9) and the current Newton's method iterate, obtained using either $\widetilde{\mathrm{DtN}}_{H,i}$ or $\mathrm{DtN}_{H,i}$ with different initializations. The black curve shows convergence of Newton's method for the original problem (9) starting from zero initial guess and is the same for all three sub-figures. Similarly, the red curves depict the convergence

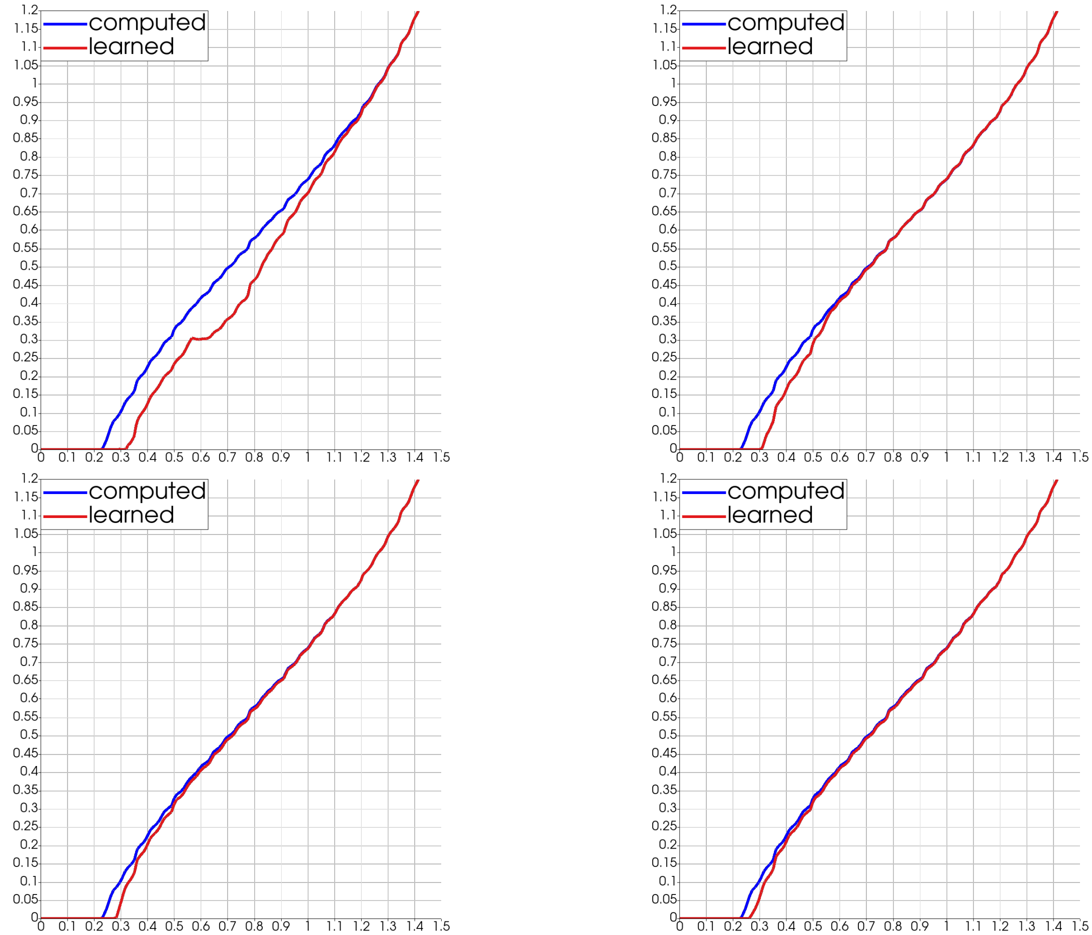

FIGURE 13. Original (blue) and surrogate (red) reconstructed solution along the line $y = x$ for $n_s = 2^4$ (top left), $n_s = 3^4$ (top right), $n_s = 4^4$ (bottom left) and $n_s = 5^4$ (bottom right)

of Newton's method for the surrogate problem (10) for $n_s = 2^4, 3^4$ and 4^4, again starting from zero initial guess. The stagnation of the red curve results from the approximation error associated to $\widetilde{\mathrm{DtN}}_{H,i}$. Finally, the blue curve shows convergence of Newton's method for (9) starting from the surrogate solution. For the original coarse problem, convergence up to the tolerance of 10^{-5} is obtained within 17 iterations. In contrast, the same Newton's method (applied to the original problem (9)) initialized with the surrogate solution, requires up to four times fewer iterations to achieve similar accuracy. Specifically, only 7, 5 or 4 iterations are needed for $n_s = 2^4, 3^4$ and 4^4, respectively. Note that even when using a very crude surrogate model with $n_s = 2$, a significant reduction in the number of Newton's steps is still achieved.

4. Conclusion and outlook

We have presented a framework that combines the Multi-scale Finite Element (MsFEM) method with techniques from Machine Learning, providing an extension of the former to the case of nonlinear PDEs with highly oscillatory coefficients. Our approach relies on learning Dirichlet-to-Neumann maps acting between low-dimensional (coarse) spaces. The surrogate DtN operators are further combined within a global substructured formulation solved by Newton's method. The optimization process involves fitting values and derivatives

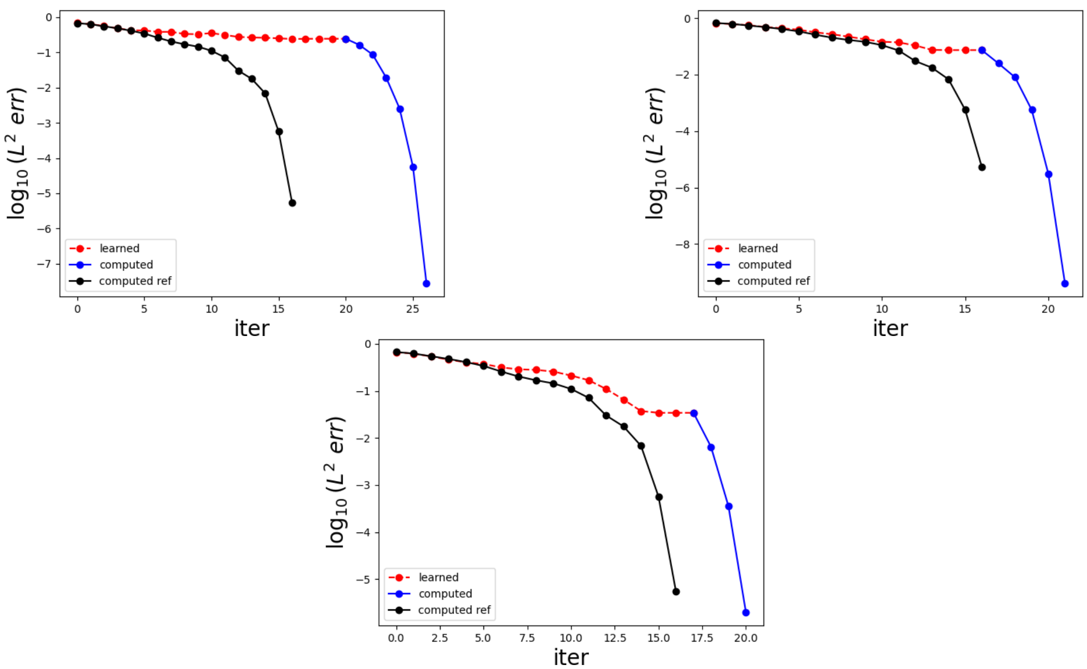

FIGURE 14. The L^2 error between the reference solution and the current iterate of Newton's method for (10) using a zero initial guess (red), and for (9) with the initial guess either set to zero (black) or provided by the approximate solution of (10) (blue). Top row from left to right, $n_s = 2^4, 3^4$, bottom row $n_s = 4^4$

of the true coarse DtN operators, where incorporating derivative information into the loss function plays a pivotal role in the accuracy of the learned operator. In addition, we weakly enforce some monotonicity properties of the original DtN maps, which improves the performance of Newton's method.

Numerical experiments performed in 1D and 2D, involving challenging p-Laplace and degenerate diffusion equations, have shown promising results. With just a few training points per dimension of the DtN map's domain, the surrogate model can approximate the solution with an accuracy of a few percent. Further improvements in the accuracy can be achieved by using the surrogate solution as an initial guess for Newton's method applied to the original substructured formulation, allowing it to enter directly into the region of quadratic convergence and obtain a more accurate final solution. Although our numerical experiments have so far been limited to the case of periodically distributed coefficients, extending to the non-periodic case is trivial and will be carried out in the future. In addition, we believe that our approach can be naturally generalized to handle a nonzero right-hand side. This would first require a low-dimensional representation of the right-hand side, such as using piecewise polynomials, and second, the surrogate model would need to be extended to incorporate additional inputs representing the approximation of the right-hand side. Future work also involves incorporating techniques from Principal Component Analysis (PCA) to reduce the complexity associated with the derivatives of the DtN operators. Additionally, the inclusion of higher-order derivatives could increase model accuracy. We are also planning on using higher-order optimizers like BFGS to accelerate convergence of the optimization process. Finally, we note that our methodology should, in principle, extend to three-dimensional problems. The dimensionality of the $\mathrm{DtN}_{H,i}$ operators can be kept small by using tetrahedral coarse meshes, which would, again, result in $\mathrm{DtN}_{H,i}$ maps acting from $\mathbb{R}^4$ to itself.

Acknowledgements

This work has been supported by ANR Project Top-up (ANR-20-CE46-0005).

References

[1] Babuska, I., Caloz, G., Osborn, J.E.: Special finite element methods for a class of second order elliptic problems with rough coefficients. SIAM Journal on Numerical Analysis **31**(4), 945–981 (1994)
[2] Babuska, I., Lipton, R.: Optimal local approximation spaces for generalized finite element methods with application to multiscale problems. Multiscale Modeling and Simulation **9**(1), 373–406 (2011)
[3] Badia, S., Li, W., Martín, A.F.: Finite element interpolated neural networks for solving forward and inverse problems. Computer Methods in Applied Mechanics and Engineering **418**, 116505 (2024)
[4] Chaturantabut, S., Sorensen, D.C.: Nonlinear model reduction via discrete empirical interpolation. SIAM Journal on Scientific Computing **32**(5), 2737–2764 (2010)
[5] E, W., Yu, B.: The deep ritz method: a deep learning-based numerical algorithm for solving variational problems. Communications in Mathematics and Statistics **6**(1), 1–12 (2018)
[6] Efendiev, Y., Galvis, J., Hou, T.Y.: Generalized multiscale finite element methods (gmsfem). Journal of computational physics **251**, 116–135 (2013)
[7] Hajibeygi, H., Bonfigli, G., Hesse, M.A., Jenny, P.: Iterative multiscale finite-volume method. Journal of Computational Physics **227**(19), 8604–8621 (2008)
[8] Hou, T.Y., Wu, X.H.: A multiscale finite element method for elliptic problems in composite materials and porous media. Journal of computational physics **134**(1), 169–189 (1997)
[9] Huge, B., Savine, A.: Differential machine learning. arXiv preprint arXiv:2005.02347 (2020)
[10] Jagtap, A.D., Karniadakis, G.E.: Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. In: AAAI Spring Symposium: MLPS (2021)
[11] Jagtap, A.D., Kharazmi, E., Karniadakis, G.E.: Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. Computer Methods in Applied Mechanics and Engineering **365**, 113028 (2020)
[12] Kharazmi, E., Zhang, Z., Karniadakis, G.E.: Variational physics-informed neural networks for solving partial differential equations. arXiv preprint arXiv:1912.00873 (2019)
[13] Leray, J., Lions, J.: Quelques résultats de visik sur les problèmes elliptiques non linéaires par les méthodes de minty-browder. Séminaire Jean Leray (1), 1–19 (1964)
[14] Lu, L., Jin, P., Pang, G., Zhang, Z., Karniadakis, G.E.: Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. Nature machine intelligence **3**(3), 218–229 (2021)
[15] Ma, C., Scheichl, R., Dodwell, T.: Novel design and analysis of generalized finite element methods based on locally optimal spectral approximations. SIAM Journal on Numerical Analysis **60**(1), 244–273 (2022)
[16] Målqvist, A., Peterseim, D.: Localization of elliptic multiscale problems. Mathematics of Computation **83**(290), 2583–2603 (2014)
[17] Moseley, B., Markham, A., Nissen-Meyer, T.: Finite basis physics-informed neural networks (fbpinns): a scalable domain decomposition approach for solving differential equations. Advances in Computational Mathematics **49**(4), 62 (2023)
[18] Vázquez, J.L.: The porous medium equation: mathematical theory. Oxford University Press (2007)
[19] Weinan, E., Engquist, B.: The heterogeneous multiscale methods. Communications in Mathematical Sciences **1**(1), 87–132 (2003)