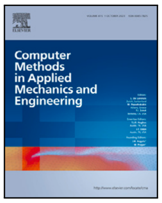
Computer Methods in Applied Mechanics and Engineering 419 (2024) 116595

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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



Residual-based error corrector operator to enhance accuracy and reliability of neural operator surrogates of nonlinear variational boundary-value problems

Prashant K. Jha

Oden Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX 78712, USA

ARTICLE INFO

Keywords:
Neural operators
Operator learning
Singular-value decomposition
Variational formulation
Surrogate modeling
Topology optimization

ABSTRACT

This work focuses on developing methods for approximating the solution operators of a class of parametric partial differential equations via neural operators. Neural operators have several challenges, including the issue of generating appropriate training data, cost-accuracy trade-offs, and nontrivial hyperparameter tuning. The unpredictability of the accuracy of neural operators impacts their applications in downstream problems of inference, optimization, and control. A framework based on the linear variational problem that gives the correction to the prediction furnished by neural operators is considered based on earlier work in JCP 486 (2023) 112104. The operator, called Residual-based Error Corrector Operator or simply Corrector Operator, associated with the corrector problem is analyzed further. Numerical results involving a nonlinear reaction–diffusion model in two dimensions with PCANet-type neural operators show almost two orders of increase in the accuracy of approximations when neural operators are corrected using the correction scheme. Further, topology optimization involving a nonlinear reaction–diffusion model is considered to highlight the limitations of neural operators and the efficacy of the correction scheme. Optimizers with neural operator surrogates are seen to make significant errors (as high as 80 percent). However, the errors are much lower (below 7 percent) when neural operators are corrected.

1. Introduction

This work focuses on neural operator-based surrogates constructed for a class of nonlinear parametric partial differential equations (PDEs). Specifically, neural operators that approximate the solution operator associated with the PDEs are considered. Working in a variational setting, consider a PDE (or system of PDEs) $\mathcal{R}(m,u)=0$, where $\mathcal{R}$ is a residual operator, m a parameter field, and u a solution of PDE. Assuming that for a given m, there is a unique solution $u=u(m)$, an operator $\mathcal{F}$ – referred to as solution operator – can be defined such that given m, $\mathcal{F}(m)$ satisfies $\mathcal{R}(m,\mathcal{F}(m))=0$. For nonlinear and computationally expensive PDEs, applications in which the solution $u=\mathcal{F}(m)$ is sought for large samples of parameter m becomes challenging, e.g., Bayesian inference, optimization, and control under uncertainty. A neural operator $\mathcal{F}_{NN}$ that maps a parameter m to an approximation $\tilde{u}=\mathcal{F}_{NN}(m)$ of a solution $u=u(m)$ is considered to cope with the computation cost of solving PDEs. To realize the full potential of neural operators in problems such as inference, optimization, and control possibly under uncertainties, it is essential to control the approximation errors within the required tolerance. Towards this, it is demonstrated in [1,2] that by utilizing the underlying structure of the solution operator, an operator $\mathcal{F}^C=\mathcal{F}^C(\cdot,\cdot)$ can be constructed – referred to as the Residual-based Error Corrector Operator or simply

E-mail address: prashant.jha@austin.utexas.edu.

https://doi.org/10.1016/j.cma.2023.116595
Received 21 June 2023; Received in revised form 3 November 2023; Accepted 4 November 2023
Available online 17 November 2023

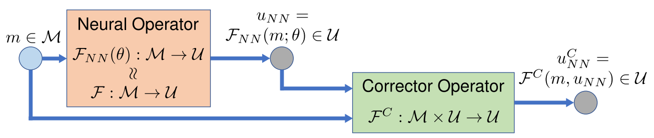

$m \in \mathcal{M}$
Neural Operator
$\mathcal{F}_{NN}(\theta): \mathcal{M} \to \mathcal{U}$
$\approx$
$\mathcal{F}: \mathcal{M} \to \mathcal{U}$
$u_{NN} = \mathcal{F}_{NN}(m;\theta) \in \mathcal{U}$
Corrector Operator
$\mathcal{F}^C: \mathcal{M} \times \mathcal{U} \to \mathcal{U}$
$u^C_{NN} = \mathcal{F}^C(m, u_{NN}) \in \mathcal{U}$

Fig. 1. Schematics of the framework where the neural operator is augmented with the corrector operator developed in earlier work [1]. Here, $\mathcal{M}$ and $\mathcal{U}$ are (Banach) function spaces of model parameters and solutions of the parameterized partial differential equation, $\mathcal{F}$ the solution operator of the PDE, and $\mathcal{F}_{NN}$ a neural operator with network parameters $\theta \in \mathbb{R}^{d_{NN}}$. $\mathcal{F}^C$ is a correction operator defined in Section 2.2.

Corrector Operator – that takes as input the parameter m of the model and the neural operator prediction $\mathcal{F}_{NN}(m)$ and provides a correction to the neural operator prediction; schematically shown in Fig. 1.

In the following, the key role of parametric PDEs in various sectors is highlighted, and the neural operators are surveyed briefly. Next, the limitations of neural operators are discussed, motivating this work. The section concludes with the details of the corrector approach and notations and the layout of the paper.

1.1. Surrogate techniques for PDEs

Partial Differential Equations are at the core of many engineering and scientific advancements and provide a robust and systematic means to represent physical systems or processes while encoding the fundamental laws of mechanics such as conservation of mass, momentum, energy, and thermodynamics principles. The models of physical reality based on PDEs may include parameters (model parameters) that could change depending on scenarios of interest or could be uncertain. As a result, one deals with a family of PDE-based models parameterized by model parameters – so-called parametric PDEs. Examples from engineering and biomedical sectors include multiphysics modeling of complex materials [3–17] and biophysical systems [18–24].

The parametric PDE-based model is a critical component in several computationally-intensive downstream problems such as parameter estimation under uncertainty [25–32], topology and design optimization and optimization under uncertainty [33–41], model selection [42–44], control under uncertainty [45,46], digital twins [47], and structural health monitoring [48]. One common aspect of these problems is that PDE solutions are required for large samples of model parameters. If the PDEs are coupled and nonlinear, computing solutions can take significant time and resources, restricting their application in downstream problems of optimization and control. For computationally expensive PDEs, several approximation techniques are available:

(i) low-fidelity approximations of the model [2,49];
(ii) reduced-order modeling [38,50–52];
(iii) regression techniques (e.g., polynomial chaos) to represent the quantities of interest as a function of the parameter [53,54]; and
(iv) neural operators – the main focus of this work – that approximate the solution operator [1,55–59,59–74,74–76].

Next, the neural operators are briefly surveyed, and their limitations are highlighted.

1.2. Neural operators as surrogates of solution operators of PDEs

There has been a remarkable growth in the development of neural operator-based approximations of the solution operator of parametric PDEs in recent years; for example, DeepONet [62,68,72], Derivative-informed Neural Operators (DINO) [69], Fourier Neural Operators (FNO) [65,66], Graph-based Neural Operators (GNO) [67,77], PCA/POD-based Neural Operators (PCANet/PODNet) [63,64], Physics-informed Neural Operator (PINO) [75], and Wavelet Neural Operator (WNO) [78]. Many of the neural operators can be characterized in a unified way; see [63,65,76]. Some applications of neural operators include Bayesian inference [1], design of materials and structures [58], digital twin [57], inverse problem [71,73], multiphase flow [61], multiscale modeling [60], optimal experimental design [56], PDE-constrained optimization [55], and phase-field modeling [59].

There are multiple choices of neural operators $\mathcal{F}_{NN}$ for approximating the solution operator $\mathcal{F}$. However, it is usually very difficult to predict and control the accuracy of the approximation, stemming from the fact that neural networks are trained to minimize an empirical error (the error is minimized in an average sense). Neural networks work well when the input parameter is in the subspace associated with the training data, and, therefore, the choice of distributions for sampling the training data becomes highly important. And, for downstream problems such as inference, optimization, and control, it is usually not possible to construct a training distribution *a priori* that is representative of the parameters that may be encountered during the solution of these problems. There are techniques to remedy this issue — the downstream problems can be solved with a crude approximation to extract crucial features of input parameters. Another option is actively updating the neural network parameters by generating new training samples. While these methods attempt to overcome the limitations of choosing appropriate training data, they may not be robust. Related to the issue of generating training data is also the trade-off between the cost and accuracy of neural operators [76].

While sufficient conditions for the existence of neural operators that are arbitrarily close to the target operator can be shown [79,80], in practice, constructing such neural operators is nontrivial. Increasing the number of training data or the complexity of neural networks may not necessarily increase the accuracy. Beyond a certain level of accuracy, it often becomes increasingly challenging to enhance accuracy, and trial-and-error approaches for tuning hyperparameters may only result in marginal gains. In this direction, adaptively increasing the complexity of neural networks and using the derivative information of the target map to create a reduced basis can help maximize the accuracy [70]. Lastly, consider a scenario in which the training data is limited and sparse, and the neural operators employed are purely data-driven and do not explicitly enforce solving the variational problem, albeit in an approximate sense. In such scenarios, not much can be done to improve the accuracy of neural operators.

1.3. Approach for enhancing accuracy of neural operators

Our goal towards improving the accuracy and reliability of neural operators is to utilize the underlying structure of the target map $\mathcal{F}$ and build a corrector framework externally from the neural operators. A computationally inexpensive framework is sought that can take neural operator predictions and provide a correction with increased accuracy and reliability at a low cost.

For the PDEs represented in variational form, $\mathcal{R}(m,u) = 0$, so-called goal-oriented error estimates [2,49,81] can provide a way forward. Following [2,49,81], given any approximation $\tilde{u}$ of the solution $u = u(m)$ of the variational problem, under certain usually reasonable assumptions, one can estimate the error $u - \tilde{u}$ by solving a linear variational problem $\delta_u \mathcal{R}(m,\tilde{u})(\tilde{e}) = -\mathcal{R}(m,\tilde{u})$ for approximate error $\tilde{e}$; $\delta_u \mathcal{R}(m,\tilde{u})(\tilde{e})$ being the variational derivative of $\mathcal{R}(m,\tilde{u})$ in the direction $\tilde{e}$. If $\mathcal{Q}(\cdot)$ is the Quantity of Interest (QoI) functional, then it is shown in [2] that goal-oriented error, $\mathcal{Q}(u) - \mathcal{Q}(\tilde{u})$, can be approximated by $\delta_u \mathcal{Q}(\tilde{u})(\tilde{e})$. There are different versions of estimates available for goal-oriented error, for example, $\mathcal{Q}(u) - \mathcal{Q}(\tilde{u}) \approx \langle \tilde{p}, \mathcal{R}(m,\tilde{u})\rangle$, $\tilde{p}$ being the approximation of the solution of the dual problem associated with the variational form $\mathcal{R}$ and the QoI functional $\mathcal{Q}$; see [2,49,81–86]

Once the error is estimated, a correction $u^C = \tilde{e} + \tilde{u}$ can be easily computed. Such an approach following [2] was developed in an earlier work [1] and it was shown in [1] that correcting predictions of trained neural operators leads to an increase in accuracy that simply cannot be achieved by hyperparameter tuning or training with larger samples of data. The steps of computing estimate $\tilde{e}$ of error and the correction u^C given m and $\tilde{u}$ can be combined to define an operator – referred to as the corrector operator – $\mathcal{F}^C : \mathcal{M} \times \mathcal{U} \to \mathcal{U}$ such that $u^C = \tilde{e} + \tilde{u} = \mathcal{F}^C(m,\tilde{u})$. If $\tilde{u}$ is already close to u, say $\tilde{u}$ is the prediction of the neural operator that has been trained to achieve a certain level of accuracy, the correction u^C is expected to have two orders of more accuracy as compared to $\tilde{u}$, owing to the Newton–Kantorovich theorem; see Section 2.2. Further, because one only solves the linear variational problem (linear in the error estimate), the added computation cost is smaller than solving the target forward problem.

1.4. Contributions of this work

Motivated by the above observations, this work analyzes the correction scheme developed in [1] and it is shown theoretically and numerically through different examples that the correcting neural operators can have significant effects on the performance of neural operator surrogates of PDEs. It is essential to highlight here that the correction framework is external to the neural operator and uses the neural operator as a black box. To demonstrate the utility of the corrector operator, a numerical example involving a nonlinear reaction–diffusion model is considered. The accuracy of neural operators and their corrections using the corrector operator for varying input and output reduced dimensions and training sample sizes is analyzed. Moreover, topology optimization of the diffusivity parameter field in a nonlinear reaction–diffusion model is taken up to highlight the limitations of neural operators and the efficacy of the correction scheme. Particularly, three different versions of optimization problems with (1) "true" (up to numerical discretization error) forward model, (2) neural operator surrogates of the forward model, and (3) neural operators with corrector operator are solved to compare the performance of neural operators and the improvements due to the corrector operator in the accuracy of optimizers. The results show a significant error reduction when the corrector operator is applied to neural operators. The error in the case of neural operator surrogates is as high as 80 percent while the error is seen to be below 7 percent when neural operators are corrected.

1.5. Notations

Let $\mathbb{N}, \mathbb{Z}, \mathbb{R}$ denote the space of natural numbers, integers, and real numbers, respectively, $\mathbb{R}^+$ the space of all nonnegative real numbers. $\mathbb{R}^n$ denotes the n-dimensional Euclidean space, $x, y \in \mathbb{R}^n$ generic points, and $\|x\|$ the Euclidean norm of $x \in \mathbb{R}^n$. Space of L^2-integrable functions $f : \Omega \subset \mathbb{R}^{d_i} \to \mathbb{R}^{d_o}$ is denoted by $L^2(\Omega;\mathbb{R}^{d_o})$; space $H^s(\Omega;\mathbb{R}^{d_o})$ for functions in $L^2(\Omega;\mathbb{R}^{d_o})$ with generalized derivatives up to order s in $L^2(\Omega;\mathbb{R}^{d_i \times_{i=1}^{s-1} d_i \times d_o})$. $\mathcal{L}(\mathcal{U};\mathcal{V})$ denotes the space of continuous linear maps from $\mathcal{U}$ to $\mathcal{V}$ and $C^1(U;\mathcal{V})$ space of continuous and differentiable maps from $U \subset \mathcal{U}$ to $\mathcal{V}$. $\mathcal{M}$ and $\mathcal{U}$ denote the generic Banach spaces of functions; for $u \in \mathcal{U}$, $\|u\|_{\mathcal{U}}$ denotes the norm of function $u \in \mathcal{U}$. The dual of $\mathcal{U}$ is the space of all linear continuous functionals on $\mathcal{U}$, $L : \mathcal{U} \to \mathbb{R}$, and is denoted by $\mathcal{U}^*$. $\langle a, b\rangle_{\mathcal{U}}$ denotes duality pairing $\mathcal{U}$ and $\mathcal{U}^*$, where $a \in \mathcal{U}$ and $b \in \mathcal{U}^*$. Given two Banach spaces $\mathcal{M}$ and $\mathcal{U}$, and a probability measure $\nu_{\mathcal{M}}$ on $\mathcal{M}$, the Bochner space of operators $\mathcal{F} : \mathcal{M} \to \mathcal{U}$ is denoted by $L^p(\mathcal{M}, \nu_{\mathcal{M}}; \mathcal{U})$, for $p \in [1, \infty]$ and the norm is given by, see [Section 1.2, [87]],

$$\|\mathcal{F}\|_{L^p(\mathcal{M},\nu_{\mathcal{M}};\mathcal{U})} = \begin{cases} \left(\mathbb{E}^{m\sim\nu_{\mathcal{M}}}\left[\|\mathcal{F}(m)\|_{\mathcal{U}}^p\right]\right)^{1/p}, & p \in [1,\infty), \\ \operatorname{esssup}_{m\sim\nu_{\mathcal{M}}} \|\mathcal{F}(m)\|_{\mathcal{U}}, & p = \infty\,. \end{cases} \tag{1}$$

Here, $\mathbb{E}^{m\sim\nu_{\mathcal{M}}}\left[\|\mathcal{F}(m)\|_{\mathcal{U}}\right]$ is the expectation with respect to the probability measure $\nu_{\mathcal{M}}$ and is defined as:

$$\mathbb{E}^{m\sim\nu_{\mathcal{M}}}\left[\|\mathcal{F}(m)\|_{\mathcal{U}}\right] = \int_{\mathcal{M}} \|\mathcal{F}(m)\|_{\mathcal{U}} \, \mathrm{d}\nu_{\mathcal{M}}(m)\,. \tag{2}$$

1.6. Layout of the paper

The paper is organized as follows: In Section 2, operators induced by PDEs in BVPs are discussed in an abstract setting, and the linear variational formulation is identified to compute the corrector in Section 2.2. Section 3 gives the overview of neural operators; in Section 3.1, corrector framework is described. For completeness, the issues of scalability and mesh dependence with vanilla neural operators are highlighted, and the scalable approach based on singular value decomposition is discussed in Section 3.2. In Section 4, numerical results are presented. The work is summarized and concluding remarks are provided in Section 5. Appendix A, Appendix B, and Appendix C include supplementary materials.

2. Variational boundary-value problems

Consider a class of parameterized variational boundary-value problems:

$$\text{Given } m \in \mathcal{M}, \text{ find } u \in \mathcal{U} \text{ such that} \qquad b(m,u;v) = l(v)\,, \quad \forall v \in \mathcal{U}\,, \tag{3}$$

where m denotes the parameter, u a solution of the problem given m, v a test function, and $\mathcal{M}$ and $\mathcal{U}$ are appropriate Banach function spaces associated with the parameter and solution of the problem, respectively. The semilinear form $b : \mathcal{M} \times \mathcal{U} \times \mathcal{U} \to \mathbb{R}$ could possibly be nonlinear in the first and second arguments and linear in the last argument, and $l \in \mathcal{U}^*$ is a continuous linear functional on $\mathcal{U}$; $\mathcal{U}^*$ being the topological dual of $\mathcal{U}$. The form $b(\cdot,\cdot;\cdot)$ is assumed to characterize weak forms or variational boundary-value problems corresponding to various PDE models of physical systems or processes, with boundary conditions embedded in $\mathcal{U}$ or the source term $l(\cdot)$.

It is convenient to represent (3) in terms of the residual operator $\mathcal{R} : \mathcal{M} \times \mathcal{U} \to \mathcal{U}^*$ defined via

$$\langle v, \mathcal{R}(m,\tilde{u})\rangle := b(m,\tilde{u};v) - l(v)\,, \qquad \forall v \in \mathcal{U}\,, \tag{4}$$

for any given $m \in \mathcal{M}$ and $\tilde{u} \in \mathcal{U}$. Problem (3) now reads:

$$\text{Given } m \in \mathcal{M}, \text{ find } u \in \mathcal{U} \text{ such that} \qquad \mathcal{R}(m,u) = 0\,. \tag{5}$$

Hereafter, the residual operator is assumed to be at least twice differentiable in the second argument in the variational, or, Gâteaux sense, with the first and second derivatives, $\delta_u \mathcal{R}(m,u) : \mathcal{U} \to \mathcal{U}^*$ and $\delta_u^2 \mathcal{R}(m,u) : \mathcal{U} \times \mathcal{U} \to \mathcal{U}^*$ given by the linear and quadratic forms,

$$\begin{aligned} \delta_u \mathcal{R}(m,u)(p) &:= \lim_{\epsilon \to 0} \frac{1}{\epsilon}\left[\mathcal{R}(m,u+\epsilon p) - \mathcal{R}(m,u)\right]\,, \\ \delta_u^2 \mathcal{R}(m,u)(p,q) &:= \lim_{\epsilon \to 0} \frac{1}{\epsilon}\left[\delta_u \mathcal{R}(m,u+\epsilon q)(p) - \delta_u \mathcal{R}(m,u)(p)\right]\,, \end{aligned} \tag{6}$$

for all $p, q \in \mathcal{U}$.

The corrector operator studied in this work is motivated by the recent work [2] on the application of so-called goal-oriented a-posteriori error estimates for the calibration of high-fidelity models using the lower fidelity approximate models. Therefore, some key aspects of goal-oriented estimates are reviewed following [2]. Following this discussion, the corrector approach is presented.

2.1. Goal-oriented A-posteriori error estimation

Consider a Quantity of Interest (QoI) $\mathcal{Q}(u) \in \mathbb{R}$ to be computed using the solution u of the variational problem (5). Suppose, there exists a lower fidelity model with the variational problem given by $\tilde{\mathcal{R}}(\tilde{u}) = 0 \in \mathcal{U}^*$, $\tilde{\mathcal{R}} : \mathcal{U} \to \mathcal{U}^*$ residual of lower fidelity model and $\tilde{u}$ a solution. If $\tilde{u} \approx u$, $\mathcal{Q}(u)$ can be approximated by $\mathcal{Q}(\tilde{u})$ with some error:

$$\mathcal{Q}(u) = \mathcal{Q}(\tilde{u}) + \underbrace{\mathcal{Q}(u) - \mathcal{Q}(\tilde{u})}_{\text{goal-oriented error}}\,.$$

The error $\mathcal{Q}(u) - \mathcal{Q}(\tilde{u})$ is referred to as the goal-oriented or modeling error as it pertains to the QoI, the goal of the analysis. Assuming that the high fidelity problem (5) is computationally expensive, the goal-oriented a-posteriori error estimation provides a means to estimate goal-oriented error using $\tilde{u}$ – a lower fidelity solution – and some version of formula can also involve $\tilde{p}$, the dual or adjoint solution of the dual problem associated with the lower fidelity model and QoI functional $\mathcal{Q}$; see [2,49,81–86].

Suppose $\tilde{e} = u - \tilde{u}$ is the error in the forward solution; then it is argued in [Section 2.1, [2]] that an estimate e^C of $\tilde{e}$ can be computed by solving the following linear variational problem:

$$\delta_u \mathcal{R}(m,\tilde{u})(e^C) = -\mathcal{R}(m,\tilde{u}) \tag{7}$$

and, using e^C, the following estimate of the goal-oriented error can be constructed,

$$\mathcal{Q}(u) - \mathcal{Q}(\tilde{u}) = \delta_u \mathcal{Q}(\tilde{u})(e^C) + r(u,\tilde{u},e^C),$$

where r collects the remainder terms [2]. The equation for e^C is based on the Taylor series expansion of the residual operator, as follows, see [Theorem 1.8, [88]]:

$$\mathcal{R}(m,u) = \mathcal{R}(m,\tilde{u}+\tilde{e}) = \mathcal{R}(m,\tilde{u}) + \delta_u \mathcal{R}(m,\tilde{u})(\tilde{e}) + \int_0^1 (1-s)\delta_u^2 \mathcal{R}(m,\tilde{u}+s\tilde{e})(\tilde{e},\tilde{e})\,\mathrm{d}s.$$

Noting that u solves $\mathcal{R}(m,u)=0$, it holds that, for any $\tilde{u}\in\mathcal{U}$,

$$\mathcal{R}(m,\tilde{u}+\tilde{e})=\mathcal{R}(m,\tilde{u})+\delta_u\mathcal{R}(m,\tilde{u})(\tilde{e})+\int_0^1(1-s)\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})(\tilde{e},\tilde{e})\,\mathrm{d}s=0.$$

If $\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})$ is bounded for $s\in[0,1]$, then the leading order term in the above is $O(\|\tilde{e}\|^2_{\mathcal{U}})$. It follows if $\|\tilde{e}\|_{\mathcal{U}}$ is small, then $\tilde{e}$ satisfies $\mathcal{R}(m,\tilde{u})+\delta_u\mathcal{R}(m,\tilde{e})(\tilde{e})\approx 0\in\mathcal{U}^*$. Thus, ignoring the leading order term gives (7) – an equation for the estimate e^C of the error $\tilde{e}$.

2.2. Corrector operator based on residuals

In this section, the residual-based error correction scheme developed for neural operators in an earlier work [1] based on [2] is reviewed. Suppose the variational problem (5) is expensive to solve (i.e., evaluation of the solution operator $\mathcal{F}(m)$), and, therefore, the operator $\mathcal{F}$ is approximated using an operator $\tilde{\mathcal{F}}:\mathcal{M}\to\mathcal{U}$ that is more easily evaluated. To ascertain how good or bad the approximation $\tilde{u}$ is, a straightforward way is to look at the exact error, $\tilde{e}=\mathcal{F}(m)-\tilde{\mathcal{F}}(m)=u-\tilde{u}$, and its norm, $\|\tilde{e}\|_{\mathcal{U}}$. However, from the previous subsection, a computationally inexpensive method is available to estimate the error $\tilde{e}$ by e^C, where e^C solves (7).

If e^C is the estimate of error $\tilde{e}=u-\tilde{u}$, then $u^C:=\tilde{u}+e^C\approx\tilde{u}+u-\tilde{u}=u$, i.e., u^C is another approximation of u. So given an approximation $\tilde{u}$ of u, a linear variational problem is solved to construct another approximation u^C as follows:

$$\text{Given } m\in\mathcal{M} \text{ and } \tilde{u}\in\mathcal{U}\text{, find } u^C \text{ such that}\quad u^C=\tilde{u}+e^C=\tilde{u}-\delta_u\mathcal{R}(m,\tilde{u})^{-1}\mathcal{R}(m,\tilde{u})\,, \tag{8}$$

assuming $\delta_u\mathcal{R}(m,\tilde{u})^{-1}$ exists. The equation above induces an operator $\mathcal{F}^C:\mathcal{M}\times\mathcal{U}\to\mathcal{U}$, referred to as the residual-based error corrector operator or simply corrector operator, defined as

$$\mathcal{F}^C(m,\tilde{u})=u^C=\tilde{u}-\delta_u\mathcal{R}(m,\tilde{u})^{-1}\mathcal{R}(m,\tilde{u})\,, \tag{9}$$

for any pair $(m,\tilde{u})\in\mathcal{M}\times\mathcal{U}$.

2.2.1. Corrector operator property

The reason $\mathcal{F}^C$ is referred to as the corrector operator is that under ideal conditions, given $m\in\mathcal{M}$ and an approximation $\tilde{u}$ of $u=\mathcal{F}(m)$, $\mathcal{F}^C$ produces another approximation $u^C=\mathcal{F}^C(m,\tilde{u})$ that has smaller error as compared to $\tilde{u}$, i.e.,

$$\|u-u^C\|_{\mathcal{U}}\le\|u-\tilde{u}\|_{\mathcal{U}}\,. \tag{10}$$

The following theorem provides a bound on correction error e^C in terms of prediction error $\tilde{e}$.

Theorem 1. *Let $\mathcal{M}$ and $\mathcal{U}$ be Banach spaces, and $\mathcal{R}:\mathcal{M}\times\mathcal{U}\to\mathcal{U}^*$ the residual functional. For a given fixed $m\in\mathcal{M}$, let $u=\mathcal{F}(m)$, i.e., u is such that $\mathcal{R}(m,u)=0$. For any arbitrary $\tilde{u}\in\mathcal{U}$, suppose $\mathcal{R}$ satisfies the following*

- *$\delta_u\mathcal{R}(m,\tilde{u}):\mathcal{U}\to\mathcal{U}^*$ is invertible, i.e., $\delta_u\mathcal{R}(m,\tilde{u})^{-1}$ exists; and*
- *$\delta_u^2\mathcal{R}(m,w):\mathcal{U}\times\mathcal{U}\to\mathcal{U}^*$ for all $w\in\{\tilde{u}+s(u-\tilde{u}):s\in[0,1]\}$ is bounded.*

If $u^C=\mathcal{F}^C(m,\tilde{u})$, $\mathcal{F}^C$ being the corrector operator defined in (9), and $e^C=u-u^C$ and $\tilde{e}=u-\tilde{u}$, then the following estimate holds:

$$\|e^C\|_{\mathcal{U}}\le\frac{1}{2}\left[\sup_{s\in[0,1]}\|\delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U},\mathcal{U})}\right]\|\tilde{e}\|^2_{\mathcal{U}},$$

where $\mathcal{L}(U,V)$ is the space of all continuous linear operators from U to V and $\|f\|_{\mathcal{L}(U,V)}=\sup_{\|v\|_U=1}\|f(v)\|_V$ is the operator norm. □

Theorem 1 is proved in Appendix A. From the above, two situations may arise:

- *Linear error reduction.* If $\tilde{u}$ is such that $\|\tilde{e}\|_{\mathcal{U}}$ is sufficiently small so that

$$\frac{1}{2}\left[\sup_{s\in[0,1]}\|\delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U},\mathcal{U})}\right]\|\tilde{e}\|_{\mathcal{U}}<1,$$

  then it holds $\|e^C\|_{\mathcal{U}}<\|\tilde{e}\|_{\mathcal{U}}$.
- *Quadratic error reduction.* More strictly, if $\mathcal{R}$ is such that, for given $(m,\tilde{u})$ and $u=\mathcal{F}(m)$,

$$\frac{1}{2}\left[\sup_{s\in[0,1]}\|\delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U},\mathcal{U})}\right]\le C<1,$$

  then $\|e^C\|_{\mathcal{U}}<\|\tilde{e}\|^2_{\mathcal{U}}$. Two orders of accuracy may thus be gained when $\|\tilde{e}\|_{\mathcal{U}}<1$.

Connection to the Newton's iteration. The corrector operator is directly related to Newton's iteration for solving $\mathcal{R}(m,u) = 0$. Let $u_0 \in \mathcal{U}$ be the initial guess, then the Newton step to solve $\mathcal{R}(m,u) = 0$, for a fixed $m \in \mathcal{M}$, is given by

$$u_k = u_{k-1} - \delta_u \mathcal{R}(m,u_{k-1})^{-1}\mathcal{R}(m,u_{k-1}) = \mathcal{F}^C(m,u_{k-1}), \qquad \forall k \geq 1\,. \tag{11}$$

Thus, the solution $u = \mathcal{F}(m)$ is a fixed-point of $\mathcal{F}^C(m,\cdot)$; i.e., if $u = \mathcal{F}^C(m,u)$ then $u = \mathcal{F}(m)$ or, equivalently, $\mathcal{R}(m,u) = 0$. Since $\mathcal{F}^C$ is the operator characterized in the Newton step, the convergence of iterations, $\{u_k\}_k$, to u and the reduction in error, $\|u - u_k\|_{\mathcal{U}}$, as k increases are ascertained by the Newton–Kantorovich theorem [89,90]. In particular, in ideal conditions when the initial guess is sufficiently close to u, Newton's iterations are expected to converge at a quadratic rate, and, thus, the error reduces by two orders in every iteration. One of the versions of the Newton–Kantorovich theorem from [90] is produced below.

Theorem 2. *For a fixed $m \in \mathcal{M}$, let $D(m)$ be an nonempty open set in $\mathcal{U}$, $\mathcal{U}$ being Banach space, and $u_0 \in D(m)$. Let $\mathcal{R}(m,\cdot) : \mathcal{U} \to \mathcal{U}^*$ be such that $\mathcal{R}(m,\cdot) \in C^1(D(m);\mathcal{U}^*)$ and $\delta_u\mathcal{R}(m,u_0) \in \mathcal{L}(\mathcal{U};\mathcal{U}^*)$ is bijective, i.e., $\delta_u\mathcal{R}(m,u_0)^{-1} \in \mathcal{L}(U^*;\mathcal{U})$. Further, suppose that there exists a constant $r > 0$ such that*

- *$\overline{B(u_0;r)} \subset D(m)$, $B(u_0;r)$ being an open ball of radius r centered at u_0;*
- *$\|\delta_u\mathcal{R}(m,u_0)^{-1}\mathcal{R}(m,u_0)\|_{\mathcal{U}} \leq \dfrac{r}{2}$;*
- *$\|\delta_u\mathcal{R}(m,u_0)^{-1}\left(\delta_u\mathcal{R}(m,\tilde{u}) - \delta_u\mathcal{R}(m,\hat{u})\right)\|_{\mathcal{L}(\mathcal{U};\mathcal{U})} \leq \dfrac{\|\tilde{u} - \hat{u}\|_{\mathcal{U}}}{r}$, for all $\tilde{u},\hat{u} \in B(u_0;r)$.*

Then, $\delta_u\mathcal{R}(m,\tilde{u}) \in \mathcal{L}(\mathcal{U};\mathcal{U}^*)$ is bijective and $\delta_u\mathcal{R}(m,\tilde{u})^{-1} \in \mathcal{L}(\mathcal{U}^*;\mathcal{U})$ at each $\tilde{u} \in B(u_0;r)$. The sequence $(u_k)_{k=0}^{\infty}$ defined by

$$u_k = u_{k-1} - \delta_u\mathcal{R}(m,u_{k-1})^{-1}\mathcal{R}(m,u_{k-1}) = \mathcal{F}^C(m,u_{k-1}), \qquad \forall k \geq 1,$$

is such that $u_k \in B(u_0;r)$ for all $k \geq 0$, and $u_k \to u$, where $u \in \overline{B(u_0;r)}$ is a zero of $\mathcal{R}(m,\cdot)$, i.e., $\mathcal{R}(m,u) = 0$. Further, for each $k \geq 0$,

$$\|u - u_k\|_{\mathcal{U}} \leq \frac{r}{2^k},$$

and the point $u \in \overline{B(u_0;r)}$ is the only zero of $\mathcal{R}(m,\cdot)$ in $\overline{B(u_0;r)}$.

For proof, see [Theorem 5, [90]].

2.3. Example of a nonlinear reaction–diffusion equation

To put the notations and ideas discussed so far into a context, a forward problem involving a nonlinear reaction–diffusion model with homogeneous Dirichlet boundary condition is considered. Suppose $\Omega \subset \mathbb{R}^d$, $d = 1,2,3$, denotes the open, bounded, and smooth domain, $u = u(\boldsymbol{x})$, $\boldsymbol{x} \in \Omega$, is the temperature field with $u \in \mathcal{U} := H^1_0(\Omega) = \{v \in H^1(\Omega) : v = 0 \text{ on } \partial\Omega\}$ governed by the reaction–diffusion model,

$$\begin{aligned} -\nabla \cdot (\kappa_0 m(\boldsymbol{x})\nabla u(\boldsymbol{x})) + \alpha u(\boldsymbol{x})^3 &= f(\boldsymbol{x}), \qquad && \boldsymbol{x} \in \Omega\,; \\ u(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \partial\Omega\,, \end{aligned}$$

where $\kappa_0, \alpha > 0$ are fixed constants, $m \in \mathcal{M} := \{v \in L^2(\Omega) \cap L^\infty(\Omega) : v \geq m_{\text{lw}}\}$ is the diffusivity field, and $f \in L^2(\Omega)$ external heat source that is fixed and given. The associated variational problem reads:

$$\text{Given } m \in \mathcal{M}, \text{ find } u \in \mathcal{U} \text{ such that } \quad \underbrace{\int_\Omega \left\{\kappa_0\, m\nabla u \cdot \nabla v + \alpha u^3 v\right\} \mathrm{d}x}_{=:\, b(m,u;v)} = \underbrace{\int_\Omega f\,v\,\mathrm{d}x}_{=:\, l(v)}, \qquad \forall v \in \mathcal{U}.$$

The corresponding residual functional $\mathcal{R} : \mathcal{M} \times \mathcal{U} \to \mathcal{U}^*$ is defined through action on $v \in \mathcal{U}$, for $(m,u) \in \mathcal{M} \times \mathcal{U}$, as follows

$$\langle v, \mathcal{R}(m,u)\rangle_{\mathcal{U}} := b(m,u;v) - l(v) = \int_\Omega \left\{\kappa_0\, m\nabla u \cdot \nabla v + \alpha u^3 v\right\} \mathrm{d}x - \int_\Omega f\,v\,\mathrm{d}x.$$

The first and second derivatives, in this case, have the form:

$$\begin{aligned} \langle v, \delta_u\mathcal{R}(m,u)(p)\rangle_{\mathcal{U}} &= \int_\Omega \left\{\kappa_0\, m\nabla p \cdot \nabla v + 3\alpha u^2 p v\right\} \mathrm{d}x\,, \\ \langle v, \delta_u^2\mathcal{R}(m,u)(p,q)\rangle_{\mathcal{U}} &= \int_\Omega \{6\alpha u p q v\}\, \mathrm{d}x\,. \end{aligned} \tag{12}$$

The existence of solutions of the above variational problem for $d \geq 3$ can be established by following the Theorem 1.6.6 in [91] and the underlying arguments.

2.3.1. Constants in the corrector operator bound for the nonlinear reaction–diffusion equation

The theorem below provides the bound on the first two derivatives of $\mathcal{R}$ and shows that the inverse operator $\delta_u\mathcal{R}(m,\tilde{u})^{-1} : \mathcal{U}^* \to \mathcal{U}$ can also be bounded.

Theorem 3. *For any $m \in \mathcal{M}$ and $\tilde{u} \in \mathcal{U}$, the following holds*

(i) *Upper and lower bound on the norm of $\delta_u \mathcal{R}(m,\tilde{u})$:*

$$\hat{C}_{\delta R}\, \|v\|_{\mathcal{U}} \le \|\delta_u \mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{U}^*} \le \bar{C}_{\delta R}(\tilde{u})\, \|v\|_{\mathcal{U}},$$

where $\hat{C}_{\delta R}$ and $\bar{C}_{\delta R} = \bar{C}_{\delta R}(\tilde{u})$ are constants given by

$$\hat{C}_{\delta R} := \frac{\kappa_0 m_{lw}}{2}\min\{1, C_P^{-2}\}, \qquad \bar{C}_{\delta R}(\tilde{u}) := \kappa_0 \|m\|_{L^\infty(\Omega)} + 3\alpha C_S^4 \|\tilde{u}\|^2_{\mathcal{U}},$$

and C_P, C_S are constants from the Poincaré inequality and the Sobolev embedding. Taking the operator norm of $\delta_u \mathcal{R}(m,\tilde{u})(v)$, it holds

$$\hat{C}_{\delta R} \le \|\delta_u \mathcal{R}(m,\tilde{u})\|_{\mathcal{L}(\mathcal{U};\mathcal{U}^*)} \le \bar{C}_{\delta R}(\tilde{u}).$$

(ii) *Upper bound on the norm of the inverse of $\delta_u \mathcal{R}(m,\tilde{u})$:*

$$\|\delta_u \mathcal{R}(m,\tilde{u})^{-1}\|_{\mathcal{L}(\mathcal{U}^*;\mathcal{U})} \le \frac{1}{\hat{C}_{\delta R}}.$$

(iii) *Upper bound on the norm of $\delta_u^2 \mathcal{R}(m,\tilde{u})$:*

$$\|\delta_u^2 \mathcal{R}(m,\tilde{u})(p,q)\|_{\mathcal{U}^*} \le \bar{C}_{\delta^2 R}(\tilde{u})\, \|p\|_{\mathcal{U}}\, \|q\|_{\mathcal{U}}, \qquad \bar{C}_{\delta^2 R}(\tilde{u}) := 6\alpha C_S^4 \|\tilde{u}\|_{\mathcal{U}},$$

and, therefore,

$$\|\delta_u^2 \mathcal{R}(m,\tilde{u})\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U};\mathcal{U}^*)} \le \bar{C}_{\delta^2 R}(\tilde{u}).$$

The theorem above is proved in Appendix B.

Using Theorem 3, the norm of the correction error, $\|e^C\|_{\mathcal{U}}$, in terms of the norm of the prediction error $\|\tilde{e}\|_{\mathcal{U}}$ can be estimated for the example of the nonlinear reaction–diffusion model. Let $\tilde{u} \in \mathcal{U}$ be an approximation (prediction) of the solution u of $\mathcal{R}(m,u) = 0$ for a given $m \in \mathcal{M}$. Let $u^C = \mathcal{F}^C(m,\tilde{u})$ be the correction, where $\mathcal{F}^C$ is the corrector operator defined in (9). Then, combining Theorems 1 and 3, recalling that $e^C = u - u^C$ and $\tilde{e} = u - \tilde{u}$, gives

$$\begin{aligned}
\|e^C\|_{\mathcal{U}} &\le \frac{1}{2}\left[\sup_{s\in[0,1]} \|\delta_u \mathcal{R}(m,\tilde{u})^{-1}\delta_u^2 \mathcal{R}(m,\tilde{u}+s\tilde{e})\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U},\mathcal{U})}\right] \|\tilde{e}\|^2_{\mathcal{U}} \\
&\le \frac{1}{2}\left[\|\delta_u \mathcal{R}(m,\tilde{u})^{-1}\|_{\mathcal{L}(\mathcal{U};\mathcal{U}^*)}\right]\left[\sup_{s\in[0,1]} \|\delta_u^2 \mathcal{R}(m,\tilde{u}+s\tilde{e})\|_{\mathcal{U}^*}\right] \|\tilde{e}\|^2_{\mathcal{U}} \\
&\le \frac{1}{2\hat{C}_{\delta R}}\left[\sup_{s\in[0,1]} \bar{C}_{\delta^2 R}(\tilde{u}+s\tilde{e})\right] \|\tilde{e}\|^2_{\mathcal{U}} \\
&\le \underbrace{\frac{3C_S^4}{\hat{C}_{\delta R}}}_{=:C} \left[\|\tilde{u}\|_{\mathcal{U}} + \|\tilde{e}\|_{\mathcal{U}}\right] \|\tilde{e}\|^2_{\mathcal{U}},
\end{aligned}$$

i.e., under conditions in Theorem 3, a constant $C \in \mathbb{R}^+$ exists such that

$$\|e^C\|_{\mathcal{U}} \le C\|\tilde{u}\|_{\mathcal{U}}\, \|\tilde{e}\|^2_{\mathcal{U}} + C\, \|\tilde{e}\|^3_{\mathcal{U}}.$$

3. Neural operators and corrector scheme

Consider a case when a map $\mathcal{F} : \mathcal{M} \to \mathcal{U}$ is induced by a variational problem $\mathcal{R}(m,\mathcal{F}(m)) = 0$ in $\mathcal{U}^*$. In practice, the variational problem is solved numerically in finite-dimensional subspaces of $\mathcal{M}$ and $\mathcal{U}$. In an abstract setting, the discrete variational problem can be written as:

$$\text{Given } m \in \mathcal{M}_h, \text{ find } u \in \mathcal{U}_h \text{ such that} \qquad \langle v, \mathcal{R}(m,u)\rangle = 0, \quad \forall v \in \mathcal{U}_h, \tag{13}$$

where $\mathcal{M}_h \subset \mathcal{M}$ and $\mathcal{U}_h \subset \mathcal{U}$ are finite dimensional subspaces of $\mathcal{M}$ and $\mathcal{U}$, respectively. For example, in a finite element approximation, if $\{\phi_i\}_{i=1}^{q_m}$ are the basis functions, where $q_m = \dim(\mathcal{M}_h)$, then $\mathcal{M}_h = \mathrm{span}\left(\{\phi_i\}\right) \subset \mathcal{M}$. An element $m \in \mathcal{M}_h$ is expressed as the linear combinations of interpolation functions, i.e., $m = \sum_i m_i\phi_i$, where $(m_1, m_2, \dots, m_{q_m}) \in \mathbb{R}^{q_m}$ and $\mathbb{R}^{q_m}$ being the coefficient space of $\mathcal{M}_h$. Similarly, if $\{\psi_i\}_{i=1}^{q_u}$ are the basis functions such that $\mathcal{U}_h = \mathrm{span}\left(\{\psi_i\}\right)$ and $q_u = \dim(\mathcal{U}_h)$, then $u \in \mathcal{U}_h$ has a representation $u = \sum_{i=1}^{q_u} u_i\psi_i$ with $(u_1, u_2, \dots, u_{q_u}) \in \mathbb{R}^{q_u}$. In what follows, $m \in \mathcal{M}_h$ and $u \in \mathcal{U}_h$ will be interchanged with $m = (m_1, m_2, \dots, m_{q_m}) \in \mathbb{R}^{q_m}$ and $u = (u_1, u_2, \dots, u_{q_u}) \in \mathbb{R}^{q_u}$ when convenient while keeping in mind that given coefficient vectors (m_i) and (u_i), the functions m and u are given by $\sum m_i\phi_i$ and $\sum u_i\psi_i$, respectively.

The numerical discretization furnishes an approximation $\mathcal{F}_h : \mathbb{R}^{q_m} \to \mathbb{R}^{q_u}$ (technically, $\mathcal{F}_h : \mathcal{M}_h \to \mathcal{U}_h$) of $\mathcal{F} : \mathcal{M} \to \mathcal{U}$. Next, consider a family of neural operators $\mathcal{F}_{h,NN}(\cdot;\theta) : \mathbb{R}^{q_m} \to \mathbb{R}^{q_u}$ parameterized by $\theta \in \Theta_{NN} \subset \mathbb{R}^{d_{NN}}$, d_{NN} being the number of trainable parameters in the neural network architecture. Here, $m = (m_1, \dots, m_{q_m}) \in \mathbb{R}^{q_m}$ is the input and $u = (u_1, \dots, u_{q_u}) \in \mathbb{R}^{q_u}$ is the output to the neural operator. Broadly, θ is chosen such that the error $\mathcal{F}_{h,NN}(\cdot;\theta) - \mathcal{F}_h(\cdot)$ is minimized in some sense.

While $\mathcal{F}_h$ can be applied to any $m \in \mathbb{R}^{q_m}$, for practical purposes, some probability distribution measure ν on $\mathcal{M}$ (ν_h after finite dimensional approximation of $\mathcal{M}$) is assumed to sample m and compute $\mathcal{F}_h(m)$. Given a sampling probability distribution ν_h, the optimization problem to train neural operator $\mathcal{F}_{h,NN}$ can be written as:

$$\theta_{h,NN} = \operatorname*{arg\,min}_{\theta\in\Theta_{NN}} J_h(\theta) := \mathbb{E}^{m\sim\nu_h}\left[\|\mathcal{F}_h(m) - \mathcal{F}_{h,NN}(m;\theta)\|_{\mathcal{U}_h}\right]. \tag{14}$$

In the above, the computation of $J_h(\theta)$ is intractable due to the nature of integration. Moreover, because $\mathcal{F}_h(m)$ is expensive to compute, requiring $\mathcal{F}_h(m)$ for large samples should be avoided. Thus, in practice, a finite number of samples $m^i \sim \nu_h$, $i = 1, \dots, N$ (assumed independent and identically distributed) is considered to approximate the cost function as follows:

$$\tilde{\theta}_{h,NN} = \underset{\theta \in \Theta_{NN}}{\arg\min}\ \tilde{J}_h(\theta) := \frac{1}{N} \sum_{i=1}^{N} \|\mathcal{F}_h(m^i) - \mathcal{F}_{h,NN}(m^i;\theta)\|_{\mathcal{U}_h} . \tag{15}$$

Here, $\{(m^i, u^i) = \mathcal{F}_h(m^i)\}_{i=1}^N$ are the training data, and, for each i, $(m^i, u^i) \in \mathbb{R}^{q_m} \times \mathbb{R}^{q_u}$. Assuming the above optimization problem can be solved for $\tilde{\theta}_{h,NN}$, an "optimal" neural operator, $\tilde{\mathcal{F}}_{h,NN}(\cdot)$, can be defined according to:

$$\tilde{\mathcal{F}}_{h,NN}(\cdot) := \mathcal{F}_{h,NN}(\cdot; \tilde{\theta}_{h,NN}) . \tag{16}$$

3.1. Correcting neural operators using corrector operator

Suppose the optimization problem (15) is solved to obtain the neural operator $\tilde{\mathcal{F}}_{h,NN}$. Since the neural operator is trained to minimize the average error, see (15), the error $\|\mathcal{F}_h(m) - \tilde{\mathcal{F}}_{h,NN}(m)\|_{\mathcal{U}_h}$ for any arbitrary sample $m \sim \nu_h$ (or m can be any element of $\mathcal{M}_h$) can be significantly large. Depending on the application of the neural operator-based surrogate, this unpredictability of accuracy of $\tilde{\mathcal{F}}_{h,NN}(m)$ can pose a serious challenge. To improve the accuracy of $\tilde{\mathcal{F}}_{h,NN}(m)$ further, one direction is to fine-tune the network architecture and hyperparameters through trial and error. However, it is seen in practice that the accuracy of a fixed neural network cannot be enhanced beyond a certain limit, and often fine-tuning hyperparameters is not straightforward and may give only marginal gains [1,76].

To enhance the accuracy and reliability beyond what can be achieved by hyperparameter tuning or increasing data samples, in an earlier work [1], neural operator predictions are corrected using the residual-based error correction; see Section 2.2. This approach does not seek to modify the existing neural operator architecture and has the potential to be used with neural operators with limited accuracy and trained with sparse data.

Given $m \in \mathcal{M}_h$ and corresponding neural operator prediction $u_{NN} = \tilde{\mathcal{F}}_{h,NN}(m) \in \mathcal{U}_h$, the correction u^C_{NN} is computed as follows:

$$u^C_{NN} = \mathcal{F}^C(m, u_{NN}) = u_{NN} - (\delta_u \mathcal{R}(m, u_{NN}))^{-1} \mathcal{R}(m, u_{NN}).$$

The above entails solving the following linear variational problem:

$$\begin{aligned} &\text{Given } m \in \mathcal{M}_h \text{ and } u_{NN} = \tilde{\mathcal{F}}_{h,NN}(m) \in \mathcal{U}_h, \text{ find } u^C_{NN} \in \mathcal{U}_h \text{ such that} \\ &\quad \langle v, \delta_u \mathcal{R}(m, u_{NN})(u^C_{NN} - u_{NN}) \rangle = -\langle v, \mathcal{R}(m, u_{NN}) \rangle, \qquad \forall v \in \mathcal{U}_h . \end{aligned} \tag{17}$$

If u_{NN} is sufficiently close to the true solution $u = \mathcal{F}_h(m)$, as will be the case for a trained neural operator, the error $\|u - u^C_{NN}\|_{\mathcal{U}_h}$ is expected to be at least two orders smaller than the neural operator error $\|u - u_{NN}\|_{\mathcal{U}_h}$ when conditions of the Newton Kantorovich theorem hold; see Theorems 1 and 2. In a scenario when $\mathcal{F}_h(m)$ is sought for an input m far from the subspace generated by the training input data $\{m^i\}_{i=1}^N$, the neural operator prediction is expected to have a large error, as this corresponds to extrapolation. In this case, the correction u^C_{NN} is hoped to keep the error small. This is demonstrated numerically for the topological optimization of input parameter m in a nonlinear reaction–diffusion equation. For the example of topological optimization, since it is difficult to construct *a priori* a probability distribution ν_h that includes samples representative of m_{optim}, m_{optim} being the solution of the topological optimization problem, the neural operator is expected to make large errors during the optimization iterations. This is the case for the topological optimization example in Section 4.3. However, when a corrector operator is used together with a neural operator, the accuracy of optimization solutions is seen to increase significantly.

3.2. Scalable and mesh-independent neural operators

The neural operator $\tilde{\mathcal{F}}_{h,NN}(m) : \mathbb{R}^{q_m} \to \mathbb{R}^{q_u}$ described so far, see (16), has two key limitations. First, it is not scalable; for fine discretizations of variational problems, q_m and q_u could be large which makes training neural networks difficult due to the fact that the map is now between two very high-dimensional spaces. The second problem is mesh dependence, as $\mathcal{F}_{h,NN}$ is coupled to the underlying mesh used in $\mathcal{M}_h$ and $\mathcal{U}_h$.

An approach based on dimensional reduction techniques to make the neural operators scalable and mesh-independent is discussed following [63,65,69,70]. The key idea is to construct a neural network-based mapping between the low-dimensional subspaces $\mathbb{R}^{r_m}$ and $\mathbb{R}^{r_u}$ of $\mathbb{R}^{q_m}$ and $\mathbb{R}^{q_u}$, respectively; see Fig. 2. To make this more precise, suppose $\Pi^{\mathcal{M}}_{r_m} : \mathbb{R}^{q_m} \to \mathbb{R}^{r_m}$, $q_m = \dim(\mathcal{M}_h)$ and $r_m \ll q_m$ is the dimension of the reduced subspace, and similarly, $\Pi^{\mathcal{U}}_{r_u} : \mathbb{R}^{q_u} \to \mathbb{R}^{r_u}$, where $q_u = \dim(\mathcal{U}_h)$ and $r_u \ll q_u$. Next, a parameterized neural operator $\mathcal{F}_{r,h,NN} : \mathbb{R}^{r_m} \times \Theta_{NN} \to \mathbb{R}^{r_u}$ is considered with the corresponding optimization problem defined as:

$$\tilde{\theta}_{r,h,NN} = \underset{\theta \in \Theta_{NN}}{\arg\min}\ \tilde{J}_{r,h}(\theta) := \frac{1}{N} \sum_{i=1}^{N} \|\mathcal{F}_h(m_i) - (\Pi^{\mathcal{U}}_{r_u})^T \left(\mathcal{F}_{r,h,NN}(\Pi^{\mathcal{M}}_{r_m}(m^i); \theta) \right) \|_{\mathcal{U}_h} . \tag{18}$$

The trained neural operator is then defined as

$$\tilde{\mathcal{F}}_{r,h,NN}(\cdot) := \mathcal{F}_{r,h,NN}(\cdot; \tilde{\theta}_{r,h,NN}) . \tag{19}$$

In the term $(\Pi^{\mathcal{U}}_{r_u})^T \left(\mathcal{F}_{r,h,NN}(\Pi^{\mathcal{M}}_{r_m}(m^i); \theta) \right)$, firstly, the input parameter $m^i \in \mathbb{R}^{q_m}$ is projected into the reduced subspace $\mathbb{R}^{r_m}$; secondly, the reduced input vector is fed to the neural operator which returns $u_r = \mathcal{F}_{r,NN}(\Pi^{\mathcal{M}}_{r_m}(m^i); \theta) \in \mathbb{R}^{r_u}$; and, thirdly, u_r is projected into the full space $\mathbb{R}^{q_u}$ using $(\Pi^{\mathcal{U}}_{r_u})^T$.

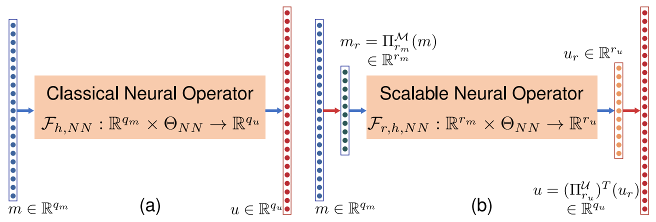

Classical Neural Operator
$\mathcal{F}_{h,NN} : \mathbb{R}^{q_m} \times \Theta_{NN} \to \mathbb{R}^{q_u}$
$m \in \mathbb{R}^{q_m}$
(a)
$u \in \mathbb{R}^{q_u}$
$m_r = \Pi^{\mathcal{M}}_{r_m}(m) \in \mathbb{R}^{r_m}$
Scalable Neural Operator
$\mathcal{F}_{r,h,NN} : \mathbb{R}^{r_m} \times \Theta_{NN} \to \mathbb{R}^{r_u}$
$u_r \in \mathbb{R}^{r_u}$
$m \in \mathbb{R}^{q_m}$
(b)
$u = (\Pi^{\mathcal{U}}_{r_u})^T(u_r) \in \mathbb{R}^{q_u}$

Fig. 2. Schematics of the two neural operators. In (a), a classical approach is shown which maps the coefficient space of $\mathcal{M}_h$ into the coefficient space of $\mathcal{U}_h$. In (b), a neural operator is taken as a map between the low-dimensional subspaces of input and output spaces, and, as a result, is relatively easier to train and is independent of the mesh [65,92]. There are two additional steps in this approach: first, the given input is compressed using the projector $\Pi^{\mathcal{M}}_{r_m}$, and, second, the output of the neural operator is decompressed using the transpose of the projector $\Pi^{\mathcal{U}}_{r_u}$.

3.2.1. *Singular-value decomposition (SVD) for projectors*

In this work, SVD is used to construct the projectors $\Pi^{\mathcal{M}}_{r_m}$ and $\Pi^{\mathcal{U}}_{r_u}$ for dimensional reduction. For completeness, key aspects of SVD are reviewed in this subsection. In several works, for example, [63], Principal Component Analysis (PCA) provides a natural way to reduce the dimensions of discretized input and output spaces. PCA begins with the covariance or correlation matrix associated with the data matrix (see matrix A below associated with the input samples for which the covariance matrix will be $\frac{1}{N-1}AA^T$) and step-by-step constructs a set of orthonormal bases that are called principal components. More precisely, the ith principal component w^i, such that $\|w^i\| = 1$, maximizes the variance of dataset projected on w^i and is orthogonal to previous principal components $w^1, w^2, \dots, w^{i-1}$. In SVD, instead, one works with the data matrix directly, and the singular values and right and left singular vectors of the data matrix are sought. The singular vectors are then used as the principal bases. Following [Section 3.5, [93]], the orthonormal bases computed in SVD can be shown to be related to the principal components. Therefore, the SVD can be seen as a method for performing PCA.

Suppose $\{(m^i, u^i)\}_{i=1}^N$ are the training data for the neural operator, where $m^i \in \mathbb{R}^{q_m}$ and $u^i = \mathcal{F}_h(m^i) \in \mathbb{R}^{q_u}$. Further, suppose that $\{m^i\}$ and $\{u^i\}$ are centered so that mean of $\{m^i\}$ and $\{u^i\}$, $\frac{1}{N}\sum_{i=1}^N m^i$ and $\frac{1}{N}\sum_{i=1}^N u^i$, respectively, are zero. Focusing on the input space $\mathbb{R}^{q_m}$, let A denote an $q_m \times N$ matrix such that:

$$A = \begin{bmatrix} | & | & & | \\ m^1 & m^2 & \cdots & m^N \\ | & | & & | \end{bmatrix}. \tag{20}$$

Next, consider a singular value decomposition of A, $A = UDV^T$, where U and V are column-orthonormal matrices of sizes $q_m \times q_m$ and $N \times N$, respectively, and D is a $q_m \times N$ diagonal matrix. The columns of U and V are referred to as left and right singular vectors, respectively, while the diagonal elements of D, $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r \geq 0$, are the singular values. Here, $r = \min\{q_m, N\}$, and some λ_i can be zero. There exists an integer $r_A \leq \min\{q_m, N\}$ such that $\lambda_j = 0$ for all $j > r_A$, and $r_A = \mathrm{rank}(A)$. Focusing on the matrix U, it has the following structure

$$U = \begin{bmatrix} | & | & & | \\ w^1 & w^2 & \cdots & w^{q_m} \\ | & | & & | \end{bmatrix}, \tag{21}$$

where $w^i \in \mathbb{R}^{q_m}$ are orthonormal vectors, i.e., $w^i \cdot w^j = \delta_{ij}$, δ_{ij} being the Kronecker delta function. The columns of U, i.e., $\{w^i\}$, form a bases for $\mathbb{R}^{q_m}$.

Let $r_m > 0$ such that $r_m \leq \mathrm{rank}(A)$ is the integer of the reduced dimension $\mathbb{R}^{r_m}$ for which a projector $\Pi^{\mathcal{M}}_{r_m} : \mathbb{R}^{q_m} \to \mathbb{R}^{r_m}$ is sought. Given r_m, a matrix U_{r_m} is constructed as follows by removing the last $q_m - r_m$ columns of U:

$$U_{r_m} = \begin{bmatrix} | & | & & | \\ w^1 & w^2 & \cdots & w^{r_m} \\ | & | & & | \end{bmatrix}. \tag{22}$$

The matrix U_{r_m} has the following notable properties:

- *Projection into the reduced space.* $U^T_{r_m}(m)$ projects an element $m \in \mathbb{R}^{q_m}$ into a lower dimensional subspace of $\mathbb{R}^{q_m}$, i.e., $U^T_{r_m} : \mathbb{R}^{q_m} \to \mathbb{R}^{r_m}$. To see this, consider

$$U^T_{r_m}(m) = \begin{bmatrix} — & (w^1)^T & — \\ — & (w^2)^T & — \\ & \vdots & \\ — & (w^{r_m})^T & — \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_{q_m} \end{bmatrix} = \begin{bmatrix} m \cdot w^1 \\ m \cdot w^2 \\ \vdots \\ m \cdot w^{r_m} \end{bmatrix} \in \mathbb{R}^{r_m}. \tag{23}$$

- *Projection into the full space.* $U_{r_m} : \mathbb{R}^{r_m} \to \mathbb{R}^{q_m}$ and this is confirmed as follows: take a vector $\tilde{m} = (\tilde{m}_1, \tilde{m}_2, \dots, \tilde{m}_{r_m}) \in \mathbb{R}^{r_m}$ and note

$$U_{r_m}(\tilde{m}) = \begin{bmatrix} | & | & & | \\ w^1 & w^2 & \cdots & w^{r_m} \\ | & | & & | \end{bmatrix} \begin{bmatrix} \tilde{m}_1 \\ \tilde{m}_2 \\ \vdots \\ \tilde{m}_{r_m} \end{bmatrix} = \sum_{i=1}^{r_m} \tilde{m}_i w^i \qquad \in \mathbb{R}^{q_m}, \tag{24}$$

as $w^i \in \mathbb{R}^{q_m}$ for each i.
- *Approximation of an identity matrix.* $U_{r_m} U_{r_m}^T \approx I_{q_m}$ and $U_{r_m}^T U_{r_m} = I_{r_m}$, I_n being the identity matrix in $\mathbb{R}^n$.
- *Optimal reconstruction.* The matrix $U_{r_m}^T$ minimizes the reconstruction error over all possible projection operators of rank r_m. To define a reconstruction error, first note that if $m \in \mathbb{R}^{q_m}$ then $U_{r_m}^T(m) \in \mathbb{R}^{r_m}$ and $U_{r_m}(U_{r_m}^T(m)) \in \mathbb{R}^{q_m}$, i.e., $U_{r_m}(U_{r_m}^T(m))$ projects m back into the same space. Thus, $U_{r_m}(U_{r_m}^T(m))$ is the reconstruction of m. The error, $\|m - U_{r_m} U_{r_m}^T m\|$, in general, is not zero. The reconstruction error for a given data matrix A with columns $\{m^i\}_{i=1}^N$ is defined as the sum of the square of individual reconstruction errors:

$$e_{r_m} := \frac{1}{N} \sum_{i=1}^{N} \|m^i - U_{r_m} U_{r_m}^T m^i\|^2 = \frac{1}{N} \|A - U_{r_m} U_{r_m}^T A\|_F^2, \tag{25}$$

where $\|A\|_F = \sqrt{\sum_i \sum_j A_{ij}^2}$ is the Frobenius norm of the matrix A. It can be shown that U_{r_m} solves the following optimization problem (Eckart–Young theorem)

$$U_{r_m} = \operatorname*{arg\,min}_{V \in \mathbb{R}^{q_m \times r_m}} \|A - V V^T A\|_F. \tag{26}$$

For proof, see [Theorem 2, [94]].

Due to the properties listed above, it makes sense to take $U_{r_m}^T$ as the projector, i.e., $\Pi_{r_m}^{\mathcal{M}} := U_{r_m}^T$.

In a similar fashion, let A is now written in terms of the output data $\{u^i\}_{i=1}^N$, i.e.,

$$A = \begin{bmatrix} | & | & & | \\ u^1 & u^2 & \cdots & u^N \\ | & | & & | \end{bmatrix}, \tag{27}$$

and $A = U D V^T$, where U and V are $q_u \times q_u$ and $N \times N$ orthonormal matrices, respectively, and D an $q_u \times N$ diagonal matrix of singular values of A. Further, let r_u, such that $r_u \le \mathrm{rank}(A) \le \min\{q_u, N\}$, be the given dimension of the desired reduced space. The projector $\Pi_{r_u}^{\mathcal{U}} : \mathbb{R}^{q_u} \to \mathbb{R}^{r_u}$ is defined using the matrix $U_{r_u}^T$, where

$$U_{r_u} = \begin{bmatrix} | & | & & | \\ w^1 & w^2 & \cdots & w^{r_u} \\ | & | & & | \end{bmatrix} \tag{28}$$

is the truncation of U.

4. Numerical examples

The example of a nonlinear reaction–diffusion equation presented in Section 2.2 with slight modifications is taken up for the demonstration of the efficacy of the proposed corrector approach. The first example concerns the temperature field in a square domain with a prescribed heat source and the Dirichlet boundary condition on the bottom edge of the domain. Neural operators with varying input and output reduced dimensions and sizes of training samples are tested for accuracy. In the same tests, the corrector operator that takes neural operator prediction and model parameter as input and computes new prediction is analyzed and it is shown that the corrector operator consistently produces a new approximation of a solution of the problem with increased accuracy. The second example considers a slightly more complex geometry of a square domain with two circular voids. The forward problem now has heat flux prescribed on the outer boundary, and the temperature is fixed to zero in the inner boundaries. In this setup, the topology optimization problem on the diffusivity parameter field is posed. For this optimization problem, the accuracy of neural operators as surrogates of the forward model is examined, and it is shown that neural operators lead to high errors in optimizers. However, when the corrector operator is used in conjunction with neural operators, the accuracy increases significantly. In what follows, first, the neural network architecture and some details about the libraries used in this work are discussed. Following that, the two subsections present the key results.

4.1. Neural network architecture and software details

Neural operators in this work are based on the projectors from SVD and consist of ResNet (residual network [95]) layers following [1,70]. Particularly, the number of residual network blocks is fixed to five with rank 20; see Fig. 3. Let $\{(m^i, u^i)\}_{i=1}^N$ be training data with q_u and q_m such that $m^i \in \mathbb{R}^{q_m}$ and $u^i \in \mathbb{R}^{q_u}$, respectively, and $\bar{m} := \frac{1}{N}\sum_i m^i$ and $\bar{u} = \frac{1}{N}\sum_i u^i$. Also, let $r_m \le q_m$ and $r_u \le q_u$ be the dimension of reduced input and output spaces, respectively. In addition to the five residual network blocks, the neural network has the following four affine (an identity activation function) layers:

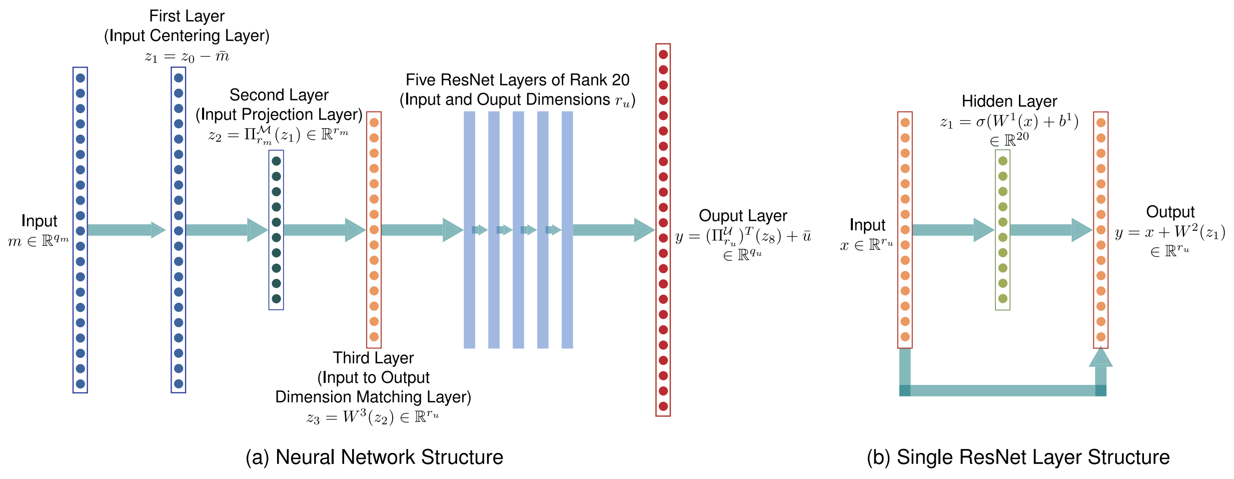

First Layer
(Input Centering Layer)
$z_1 = z_0 - \bar{m}$
Second Layer
(Input Projection Layer)
$z_2 = \Pi^{\mathcal{M}}_{r_m}(z_1) \in \mathbb{R}^{r_m}$
Five ResNet Layers of Rank 20
(Input and Ouput Dimensions r_u)
Input
$m \in \mathbb{R}^{q_m}$
Ouput Layer
$y = (\Pi^{\mathcal{U}}_{r_u})^T(z_8) + \bar{u}$
$\in \mathbb{R}^{q_u}$
Third Layer
(Input to Output
Dimension Matching Layer)
$z_3 = W^3(z_2) \in \mathbb{R}^{r_u}$
(a) Neural Network Structure
Hidden Layer
$z_1 = \sigma(W^1(x) + b^1)$
$\in \mathbb{R}^{20}$
Input
$x \in \mathbb{R}^{r_u}$
Output
$y = x + W^2(z_1)$
$\in \mathbb{R}^{r_u}$
(b) Single ResNet Layer Structure

Fig. 3. Neural network structure (a) and the structure of the residual network (ResNet) layer (b). In (a), data centering and projection steps are depicted. In (b), $\sigma = \sigma(z)$ is an `Softplus` ($\sigma(z) = \log(\exp(z) + 1)$) activation function.

- *Input centering layer.* The first hidden layer has identity matrix as weights and $-\bar{m}$ (negative of $\bar{m}$) as bias. Given an input $m \in \mathbb{R}^{q_m}$ to this layer, the output is $m - \bar{m} \in \mathbb{R}^{q_m}$.
- *Input projector layer.* The second hidden layer has input projector matrix $\Pi^{\mathcal{M}}_{r_m} \in \mathbb{R}^{r_m \times q_m}$ as weight, and the bias is fixed to zero. This layer projects the centered input data onto a reduced dimensional space.
- *Input-to-output dimension matching layer.* The third layer is a `Dense` layer with r_u neurons which takes $\mathbb{R}^{r_m}$ element and outputs $\mathbb{R}^{r_u}$ element. The bias is fixed to zero.
- *Output centering and projector layer.* The last (output) layer consists of $(\Pi^{\mathcal{U}}_{r_u})^T \in \mathbb{R}^{q_u \times r_u}$ as weight and $\bar{u}$ as bias. This layer takes the output $u \in \mathbb{R}^{r_u}$ of the second last layer and projects it onto the output space $\mathbb{R}^{q_u}$ and translates by $\bar{u}$.

The weights and biases of the above layers except for the third layer for handling dimension mismatch are frozen, however, it is possible to learn the projectors (weights) by making these layers part of training [70]. The resulting neural network with the above three layers and the five hidden layers based on ResNet is depicted in Fig. 3. The parameters that will be varied in the numerical examples are dimensions of reduced spaces (r_m and r_u) and the number of data (N). The N samples of data are divided into $\lfloor 0.1N \rfloor$ number of validation data $N - \lfloor 0.1N \rfloor$ number of training data, and the testing data sample in addition to N training samples is fixed to $\lfloor 0.25N \rfloor$. The implementation of neural networks is based on hIPPYflow[1] [1,69,70] and TensorFlow.[2] [96] To solve the variational problems and sample from a prior ν_h, FEniCS[3] [97,98] and hiPPYlib[4] [99] are used.

4.2. Accuracy comparison for a nonlinear reaction–diffusion equation

Consider a square domain $\Omega = (0,1)^2$ with the bottom edge denoted by $\Gamma_b = [0,1] \times 0$. The equation for the temperature field $u = u(\boldsymbol{x})$ over the domain Ω is taken as

$$\begin{aligned} -\nabla \cdot (e^{m(\boldsymbol{x})} \nabla u(\boldsymbol{x})) + u(\boldsymbol{x})^3 &= f(\boldsymbol{x}), && \boldsymbol{x} \in \Omega\,; \\ u(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \Gamma_b\,; \\ e^{m(\boldsymbol{x})} \nabla u(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \partial\Omega - \Gamma_b\,; \end{aligned} \tag{29}$$

where

$$f(\boldsymbol{x}) = e^{-4(1-x_1)^2} \sin(4\pi x_2)^2, \qquad \boldsymbol{x} = (x_1, x_2) \in \Omega,$$

is an external heat source. Let the parameter and solution function spaces be given by:

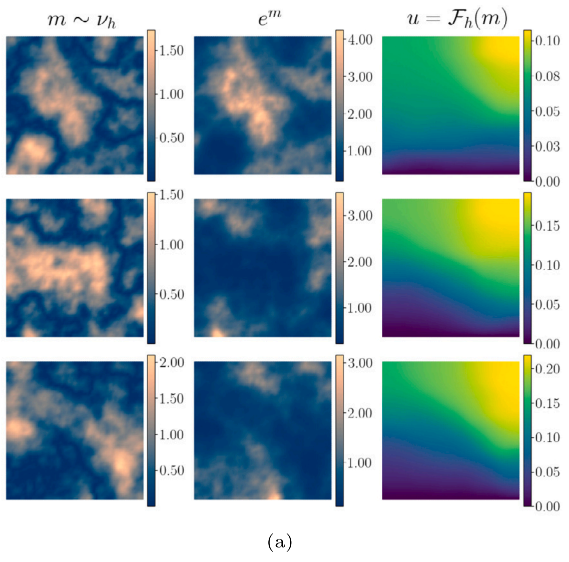
$$m \in \mathcal{M} := L^2(\Omega) \cap L^\infty(\Omega), \qquad u \in \mathcal{U} := \{v \in H^1(\Omega) : v(\boldsymbol{x}) = 0, \, \boldsymbol{x} \in \Gamma_b\}.$$

The variational problem associated with (29) reads:

Given $m \in \mathcal{M}$, find $u \in \mathcal{U}$ such that

$$\langle v, \mathcal{R}(m,u) \rangle := \int_\Omega e^{m(\boldsymbol{x})} \nabla u(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} + \int_\Omega u(\boldsymbol{x})^3 v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} - \int_\Omega f(\boldsymbol{x}) v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = 0\,, \quad \forall v \in \mathcal{U}\,. \tag{30}$$

1 https://github.com/hippylib/hippyflow
2 https://www.tensorflow.org/
3 https://fenicsproject.org/
4 https://github.com/hippylib/hippylib



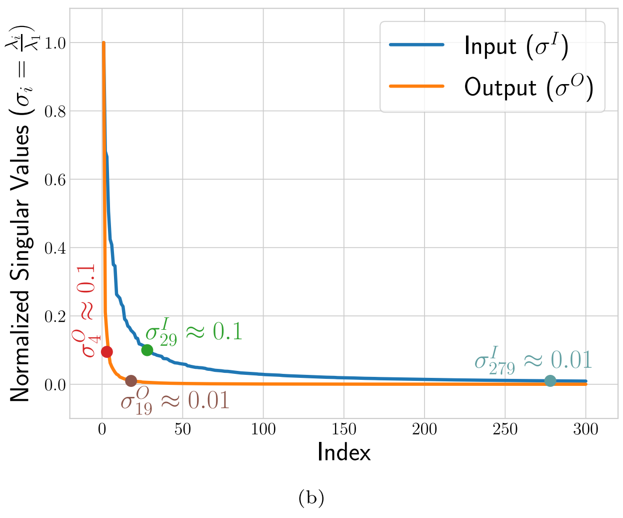


Fig. 4. (a) Visualization of three representative samples. (b) Normalized singular values for the input and output data of sample size $N = 4096$. The indices associated with the singular values near 0.1 and 0.01 are also shown.

Expressions for $\delta_u \mathcal{R}(m,u)(p)$ and $\delta^2_u \mathcal{R}(m,u)(p,q)$ can be derived following Section 2.3.

4.2.1. Data generation and neural operators

A bilinear finite element space $\mathcal{U}_h$ on a quadrilateral mesh of Ω consisting of 64 × 64 elements is considered. $\mathcal{M}_h$ is identical to $\mathcal{U}_h$. By replacing $\mathcal{M}$ and $\mathcal{U}$ with $\mathcal{M}_h$ and $\mathcal{U}_h$, respectively, the discrete version of the problem is obtained.

Data for neural operator training. To generate training data, probability distribution ν is assumed to be $\nu = \mathcal{N}(0, \mathcal{C})$, where $\mathcal{C} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is a covariance operator taking the form

$$\mathcal{C} = \begin{cases} (-\gamma \nabla \cdot \nabla + \delta)^{-d} & \text{in } \Omega\,, \\ \gamma \boldsymbol{n} \cdot \nabla + \eta & \text{on } \partial\Omega\,, \end{cases}$$

where γ, δ, η, d are hyperparameters of a covariance operator and $\boldsymbol{n}$ unit outward normal. Covariance parameters are fixed as follows: $\gamma = 0.08, \delta = 2, \eta = 1/1.42, d = 2$. Let $\mathcal{C}_h : \mathcal{M}_h \times \mathcal{M}_h \to \mathbb{R}$ is the covariance operator in the finite dimensional setting and $\nu_h = \mathcal{N}(0, \mathcal{C}_h)$. The set $\{(m^i, u^i)\}_{i=1}^N$, where $m^i \sim \nu_h \in \mathcal{M}_h$ and $u^i = \mathcal{F}_h(m^i)$ is the solution of the discretized variational problem, is the data for neural operator learning. In Fig. 4(a), three representative data samples along with the singular values of input and output data from $N = 4096$ samples are depicted.

Neural operators. To test the effect of sample size and approximations due to dimension reductions on the accuracy of neural operators, neural operators with

$$(r_m, r_u) \in \{(50, 25), (50, 50), (100, 25), (100, 50)\} \text{ and } N \in \{256, 512, 1024, 2048, 4096\}$$

are trained. Here, r_u is kept small compared to r_m based on the relatively faster decay of output singular values; see Fig. 4(b). Thus, a total of 20 neural operators are trained and tested.

4.2.2. Comparing neural operator and corrector operator accuracy

Given a sample of input parameter $m \in \mathbb{R}^{q_m}$, suppose $u = u(m) \in \mathbb{R}^{q_u}$ is the finite element solution, $u_{NN} = u_{NN}(m)$ is the approximation furnished by neural operator, and $u^C_{NN} = u^C_{NN}(m)$ is the correction of u_{NN} obtained through corrector operator. The normalized percentage error can be defined as

$$e_{NN}(m) := \frac{\|u(m) - u_{NN}(m)\|_{l^2}}{\|u(m)\|_{l^2}} \times 100\,, \qquad e^C_{NN}(m) := \frac{\|u(m) - u^C_{NN}(m)\|_{l^2}}{\|u(m)\|_{l^2}} \times 100\,, \tag{31}$$

where $\|a\|_{l^2} = \sqrt{\sum_i a_i^2}$. In Table 1, the statistics of errors due to neural operators and corrector is shown. Analyzing the highlighted columns in Table 1 corresponding to the mean of e_{NN} and e^C_{NN}, the corrector approach is seen to consistently decrease the errors. In fact, for neural operators trained on small data (see rows with Numbers 1, 6, 11, and 16), the corrector does a great job of keeping the average error below 0.1 percentage. For the neural operators trained with the smallest and largest datasets, samples of $m \sim \nu_h$ are drawn randomly, and the solutions from the true model, neural operator, and correction of the neural operator are visualized in Figs. 5 and 6.

Table 1
Comparing errors due to the neural operator approximations and corrections of neural operators for the first example (see Section 4.2). For each neural operator and the corrector of the neural operator, the errors are computed for a total of twenty samples, and the minimum, maximum, and mean were computed from the resulting twenty e^C_{NN} and e^C_{NN} values. The errors are defined as in (31). Particularly, columns corresponding to the mean of e_{NN} and e^C_{NN} errors are highlighted; from the results, the corrector is seen to consistently enhance the accuracy of neural operators by almost two orders. The neural operators trained with the smallest and largest data samples for different (r_m, r_u) pairs are highlighted. Generally, increasing the sample size increases the accuracy of neural operators, as seen from the results.

				e_{NN}			e^C_{NN}		
Number	r_m	r_u	N	min	max	mean	min	max	mean
1	50	25	256	2.45728	16.84489	6.06698	0.00024	0.31488	0.04303
2	50	25	512	1.40506	7.82594	3.53658	0.00016	0.16437	0.03716
3	50	25	1024	1.65162	6.06503	3.18435	0.00005	0.09237	0.01783
4	50	25	2048	1.40497	4.91186	2.56645	0.00141	0.20477	0.07224
5	50	25	4096	1.16622	9.80585	2.43871	0.00001	0.19827	0.03807
6	50	50	256	3.86182	29.05746	8.91953	0.00008	0.24863	0.07290
7	50	50	512	2.89625	10.76244	5.54185	0.00014	0.19987	0.05248
8	50	50	1024	1.92660	8.37186	3.77333	0.00036	0.28288	0.06235
9	50	50	2048	1.88078	4.93965	3.07723	0.00011	0.18041	0.03552
10	50	50	4096	1.74788	4.84569	3.19339	0.00012	0.20722	0.06781
11	100	25	256	2.76545	11.96616	5.23490	0.00937	0.20000	0.05499
12	100	25	512	1.74937	5.60931	3.70606	0.00072	0.26213	0.07850
13	100	25	1024	1.65959	6.97599	3.37611	0.00019	0.23674	0.05536
14	100	25	2048	1.01940	5.27190	2.60825	0.00012	0.20808	0.06776
15	100	25	4096	1.22295	4.00625	2.00097	0.00005	0.27715	0.05748
16	100	50	256	2.30358	23.87869	5.58995	0.00033	0.23157	0.05828
17	100	50	512	2.26997	12.24735	5.29943	0.00077	0.27386	0.05864
18	100	50	1024	1.39031	7.22452	3.02298	0.00008	0.25664	0.06455
19	100	50	2048	1.08747	6.17697	3.15330	0.00012	0.24090	0.06358
20	100	50	4096	1.00965	4.63677	2.27597	0.00005	0.20564	0.07560

4.3. Topology optimization involving a nonlinear reaction–diffusion equation

To further test the utility of the predictor–corrector approach and highlight the limitation of neural operators in optimization problems, topological optimization of the diffusivity field in a nonlinear reaction–diffusion model is considered in this subsection. The domain Ω is a square domain with two circular voids: $\Omega = (0,1)^2 - \overline{B(\boldsymbol{x}_{c_1}, R_1)} - \overline{B(\boldsymbol{x}_{c_2}, R_2)}$, where $B(\boldsymbol{x}, R) = \{\boldsymbol{y} \in \mathbb{R}^2 : |\boldsymbol{y} - \boldsymbol{x}| < R\}$ denotes the ball of radius R centered at $\boldsymbol{x}$. Here, $\boldsymbol{x}_{c_1} = (0.2, 0.8)$, $\boldsymbol{x}_{c_2} = (0.7, 0.3)$, $R_1 = 0.1$, and $R_2 = 0.2$; see Fig. 7. Let $\partial\Omega = \Gamma_{in} \cup \Gamma_{out}$, Γ_{in} and Γ_{out} being the inner and outer boundaries, respectively. In the inner boundary, Γ_{in}, temperature is fixed to zero, while, in the outer boundary, Γ_{out}, the heat flux $g(\boldsymbol{x}) := 0.1$ is prescribed. Keeping the model same as in (29), but now with heat source zero, $f = 0$, the strong form of the forward model reads as:

Given a diffusivity field $m = m(\boldsymbol{x})$, find temperature u such that

$$\begin{aligned} -\nabla \cdot (m(\boldsymbol{x})\nabla u(\boldsymbol{x})) + u(\boldsymbol{x})^3 &= 0, && \boldsymbol{x} \in \Omega\,; \\ u(\boldsymbol{x}) &= 0, && \boldsymbol{x} \in \Gamma_{in}\,; \\ m(\boldsymbol{x})\nabla u(\boldsymbol{x}) \cdot \boldsymbol{n}(\boldsymbol{x}) &= 0.1 =: g(\boldsymbol{x}), && \boldsymbol{x} \in \Gamma_{out}\,. \end{aligned} \tag{32}$$

As before, function spaces associated with the parameter and solution are taken as:

$$m \in \mathcal{M} := L^2(\Omega) \cap L^\infty(\Omega), \qquad u \in \mathcal{U} := \{v \in H^1(\Omega) : v(\boldsymbol{x}) = 0, \, \boldsymbol{x} \in \Gamma_{in}\}\,. \tag{33}$$

The variational problem corresponding to the forward problem reads:

Given $m \in \mathcal{M}$, find $u \in \mathcal{U}$ such that

$$\langle v, \mathcal{R}(m,u)\rangle := \int_\Omega m(\boldsymbol{x})\nabla u(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} + \int_\Omega u(\boldsymbol{x})^3 v(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} - \int_{\Gamma_{out}} g\, v(\boldsymbol{x})\, \mathrm{d}S(\boldsymbol{x}) = 0\,, \quad \forall v \in \mathcal{U}\,. \tag{34}$$

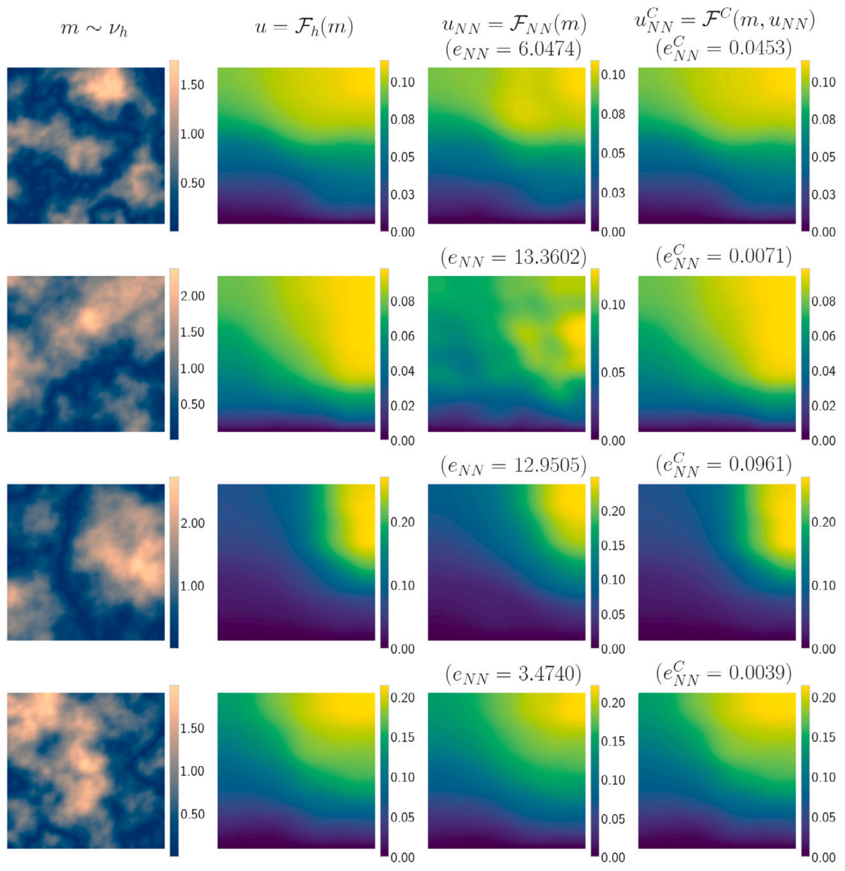


Fig. 5. Comparing true solution, neural operator prediction, and the correction of neural operator prediction for networks $1, 6, 11, 16$ (see Table 1) trained with smaller samples of data.

Topological optimization problem. Given a temperature u satisfying the above equation, the compliance – external working – is defined as

$$J(m) = \int_{\Gamma_{out}} g\, u(\boldsymbol{x})\, \mathrm{d}S(\boldsymbol{x})\,. \tag{35}$$

In this example, diffusivity m is optimized to minimize the compliance J. Let $\mathcal{M}_{ad} = \{m \in \mathcal{M} : 0 < m_{\mathrm{lw}} \le m \le 1\} \subset \mathcal{M}$ be the admissible space, $m_{\mathrm{lw}} > 0$ being a small number suitably chosen to ensure wellposedness of the problem (34). Further, let $\eta \in (0, 1]$ is the target average diffusivity, $g = 0.1$ an external heat flux on Γ_{out}, and $\mathcal{F}$ a forward solution operator. The topology optimization problem reads

$$\tilde{m} = \underset{m \in \mathcal{M}_{ad}}{\arg\min}\, J(m) := \int_{\Gamma_{out}} g\, \mathcal{F}(m)\, \mathrm{d}S(\boldsymbol{x}) \quad \text{such that} \quad \frac{1}{|\Omega|}\int_{\Omega} m(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} = \eta\,, \tag{36}$$

where the optimization problem is assumed to be wellposed and there exists a minimizer $\tilde{m}$.

Next, let $\mathcal{F}_{NN}$ is a neural operator approximation of $\mathcal{F}$, and $J_{NN}(m)$ and $J^C_{NN}(m)$ are two approximations of $J(m)$ given by

$$J_{NN}(m) := \int_{\Gamma_{out}} g\, \mathcal{F}_{NN}(m)\, \mathrm{d}S(\boldsymbol{x}), \qquad J^C_{NN}(m) := \int_{\Gamma_{out}} g\, \mathcal{F}^C(m, \mathcal{F}_{NN}(m))\, \mathrm{d}S(\boldsymbol{x}).$$

Let the minimizers of (36) with the above two cost functions are denoted by $\tilde{m}_{NN}$ and $\tilde{m}^C_{NN}$, respectively. The main objective of this example is to compare the accuracy of $\tilde{m}_{NN}$ and $\tilde{m}^C_{NN}$ with $\tilde{m}$.

4.3.1. Data generation, neural operators, and numerical method for the optimization problem

The domain Ω is triangulated using Gmsh[5] [100] with 20,614 triangular elements and 10,301 vertices. The mesh is converted into a Fenics-friendly format using Meshio.[6] [101] The finite element spaces for the parameter and solution fields are based on the

[5] https://gmsh.info/
[6] https://github.com/nschloe/meshio

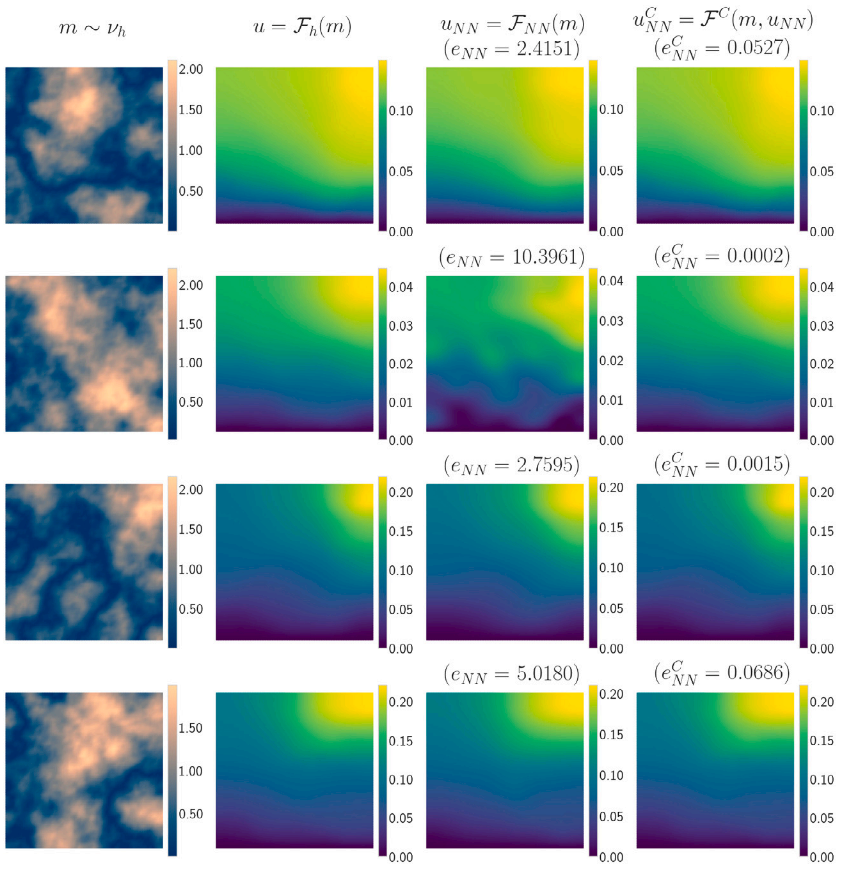


Fig. 6. Comparing true solution, neural operator prediction, and the correction of neural operator prediction for networks 5, 10, 15, 20 (see Table 1) trained with larger samples of data.

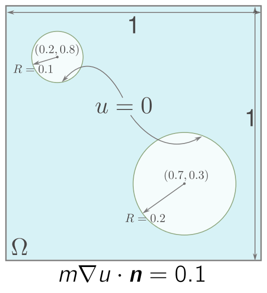


Fig. 7. Setup for the second example of topological optimization of diffusivity in nonlinear reaction–diffusion equation.

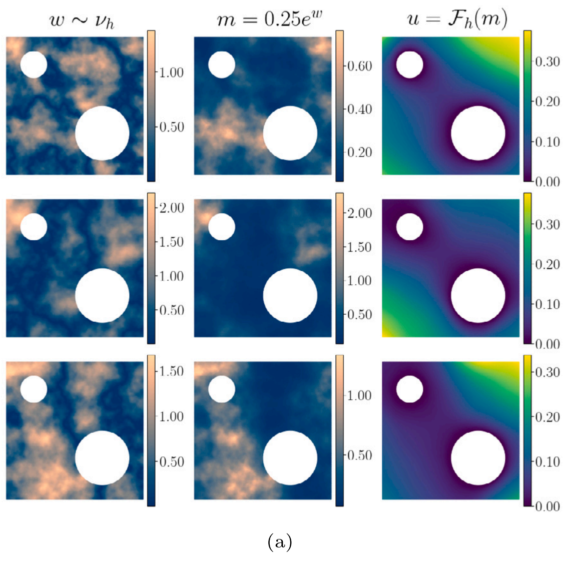
first-order Lagrange basis functions over the mesh of domain Ω. If $\mathcal{M}_h \subset \mathcal{M}$ and $\mathcal{U}_h \subset \mathcal{U}$ are the finite element function spaces, then the variational problem in discrete setting reads:

$$\text{Given } m \in \mathcal{M}_h, \text{ find } u \in \mathcal{U}_h \text{ such that} \qquad \langle v, \mathcal{R}(m,u) \rangle = 0\,, \quad \forall v \in \mathcal{U}_h\,. \tag{37}$$

Let, as before $\mathcal{F}_h$, denote the finite-dimensional approximation of $\mathcal{F}$.

Data for neural operator training. Let the probability distribution ν and its finite element approximation ν_h be the same as in Section 4.2.1. The training samples $\{(m^i, u^i)\}_{i=1}^N$ are generated as follows:

$$\text{Draw } w^i(\boldsymbol{x}) \sim \nu_h \qquad \text{and} \qquad m^i(\boldsymbol{x}) = 0.25 e^{w^i(\boldsymbol{x})}\,, \quad u^i = \mathcal{F}_h(m^i)\,. \tag{38}$$



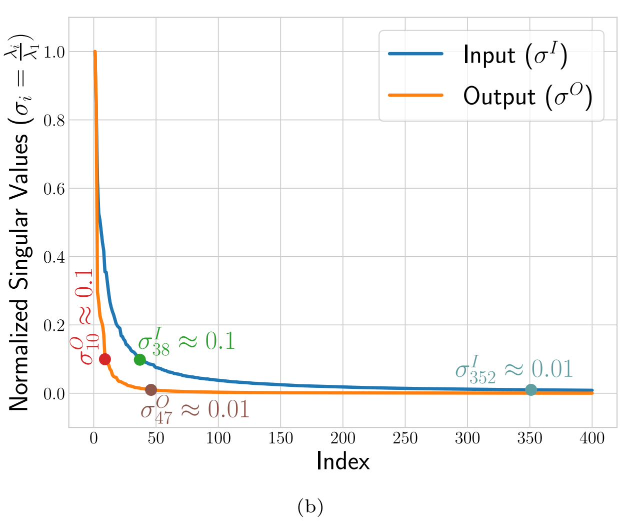


Fig. 8. (a) Visualization of three representative samples, where w and m are from (38) and u is the solution of (37) given m. (b) Normalized singular values for input and output data of sample size $N = 4096$ for the second example. The indices associated with the singular values near 0.1 and 0.01 are also displayed. In (a), the first column displays samples w generated from prior ν_h, the second column shows the diffusivity $m = 0.25e^{w^i(\boldsymbol{x})}$, and the last column is the finite element solution of the nonlinear reaction–diffusion equation with m as diffusivity. We note that the pair $(m, u = \mathcal{F}_h(m))$ constitutes a data point.

In contrast to the example in Section 4.2, here, diffusivity $m = 0.25e^w$ is taken as the model parameter, where w is some function. To generate training data, w is sampled using the probability distribution ν_h. In Fig. 8(a), three representative data samples along with the singular values of input and output data from $N = 4096$ samples are depicted.

Neural operators. Same as in the first example, a total of twenty neural operators with

$$(r_m, r_u) \in \{(50, 25), (50, 50), (100, 25), (100, 50)\} \text{ and } N \in \{256, 512, 1024, 2048, 4096\}$$

are trained. The plot of singular values in Fig. 8(b) is similar to the first example as expected and shows that the singular values of output data decay faster relative to the input data. Table 2 compares the accuracy of all twenty neural operators and corrections of these neural operators. Increasing data samples have a more dominant effect on the accuracy of neural operators in this example. It is however noted that increasing the reduced space dimension does not necessarily increase the accuracy. For the neural operators trained on the smallest and largest data samples, forward solutions and their approximations for a randomly drawn sample of w with $m = 0.25e^w$ are compared in Figs. 9 and 10.

Numerical solution of the optimization problem. The numerical method is based on the relaxation of the optimization problem (36) in a finite-dimensional setting:

$$\begin{aligned} &\min_{m \in \mathcal{M}_h, \lambda \in \mathbb{R}} && \hat{J}(m, \lambda, u) := \int_{\Gamma_{out}} g\, u \, \mathrm{d}\boldsymbol{x} + \lambda \left(\int_{\Omega} m \, \mathrm{d}\boldsymbol{x} - \eta |\Omega| \right), \\ &\text{where } u = u(m) \in \mathcal{U}_h \text{ satisfies} && \langle v, \mathcal{R}(m, u) \rangle = 0, \qquad \forall v \in \mathcal{U}_h, \\ &\text{and} && 0 < m_{\mathrm{lw}} \le m \le 1. \end{aligned} \tag{39}$$

By replacing $J_h(m)$ with

$$J_{h,NN}(m) = \int_{\Gamma_{out}} g \; \mathcal{F}_{h,NN}(m) \, \mathrm{d}S(\boldsymbol{x}) \qquad \text{and} \qquad J^C_{h,NN}(m) = \int_{\Gamma_{out}} g \; \mathcal{F}^C(m, \mathcal{F}_{h,NN}(m)) \, \mathrm{d}S(\boldsymbol{x}),$$

respectively, optimization problems with surrogates of the forward model are obtained. For the results in the next section, the numerical minimizers with cost functions J_h, $J_{h,NN}$, $J^C_{h,NN}$ are denoted by $\tilde{m}$, $\tilde{m}_{NN}$, and $\tilde{m}^C_{NN}$, respectively.

Optimization problem (39) is solved using a bi-level iterative scheme, wherein the outer iteration, u and pair (m, λ) are solved sequentially. For a given outer iteration step k and variables $m_k, \lambda_k, u_k = u(m_k)$, first, the new updated values (m_{k+1}, λ_{k+1}) are computed using the inner iteration, and then using m_{k+1}, $u_{k+1} = u(m_{k+1})$ is computed. The outer iteration stops when $\|m_k - m_{k+1}\|_{L^2(\Omega)} \le \gamma_{tol}$ or when $k = n_{max}$. The numerical method is detailed in Appendix C.

In the numerical experiments, the target average diffusivity is taken to be $\eta = 0.4$, and the initial guess for the parameter m is a constant function $m(\boldsymbol{x}) = 0.1$. Lagrange multiplier λ is initialized as $\lambda = 1$. The tolerance in Algorithm 2 is set to $m_{\mathrm{tol}} = 0.005$. Finally, $m_{\mathrm{lw}} = 0.001$.

Table 2
Comparing errors due to neural operator approximations of the forward model and corrections of neural operators for the second example. For more details on the table and colors in columns and rows, see Table 1. Compared to the neural operators in the first example, the effect of increasing data samples on the average neural operator and corrector operator errors is more evident. However, increasing the input and output reduced dimensions have very little positive effect on the accuracy of neural operators.

				e_{NN}			e^C_{NN}		
Number	r_m	r_u	N	min	max	mean	min	max	mean
1	50	25	256	7.15830	21.25784	12.47695	0.00153	0.84201	0.13296
2	50	25	512	4.59103	15.99608	9.09779	0.00907	0.38084	0.06826
3	50	25	1024	4.56197	10.36562	6.26587	0.00459	0.16858	0.03795
4	50	25	2048	3.75125	7.91939	5.54681	0.00139	0.23467	0.04583
5	50	25	4096	2.91057	8.09030	4.94291	0.00178	0.12398	0.02647
6	50	50	256	9.15900	19.87726	13.10120	0.02228	0.48271	0.14268
7	50	50	512	5.60731	24.31812	11.78976	0.00593	0.77939	0.15389
8	50	50	1024	4.50428	16.09015	7.89234	0.00262	1.34911	0.10651
9	50	50	2048	3.58687	11.70707	6.41672	0.00263	0.29747	0.03527
10	50	50	4096	3.11247	6.86457	4.22486	0.00139	0.17169	0.02497
11	100	25	256	6.66220	22.61565	11.11228	0.00673	1.41224	0.20579
12	100	25	512	5.56698	21.51665	10.51670	0.00973	0.68513	0.09869
13	100	25	1024	5.25971	12.44923	7.97554	0.00135	0.59833	0.07555
14	100	25	2048	4.53138	11.74423	7.21405	0.00342	0.96087	0.11523
15	100	25	4096	3.24184	6.81036	4.38627	0.00031	0.18123	0.02263
16	100	50	256	7.72910	18.16715	11.80013	0.00926	0.33257	0.09902
17	100	50	512	6.11802	12.84410	8.83076	0.00718	0.28607	0.05695
18	100	50	1024	4.99326	10.96458	7.80839	0.00267	0.25835	0.04400
19	100	50	2048	3.81907	15.93567	6.45772	0.00198	0.91764	0.06216
20	100	50	4096	3.41474	8.98157	5.17495	0.00113	0.12705	0.02403

4.3.2. *Optimization results*

The optimization problem was solved numerically for the following three cases:

- when $u = u(m)$ is obtained by solving the forward problem in which case the numerical minimizer is denoted by $\tilde{m}$;
- when neural operator prediction was used to approximate u by $u_{NN} = \mathcal{F}_{NN}(m)$ in which case the minimizers are identified by $\tilde{m}_{NN}$; and
- when u was approximated by $u^C_{NN} = \mathcal{F}^C(m, \mathcal{F}_{NN}(m))$ using the corrector operator in which case the minimizer is denoted by $\tilde{m}^C_{NN}$.

The percentage errors between numerical minimizers can be defined as:

$$\tilde{\varepsilon}_{NN} := \frac{\|\tilde{m} - \tilde{m}_{NN}\|_{l^2}}{\|\tilde{m}\|_{l^2}} \times 100\,, \qquad \tilde{\varepsilon}^C_{NN} := \frac{\|\tilde{m} - \tilde{m}^C_{NN}\|_{l^2}}{\|\tilde{m}\|_{l^2}} \times 100\,. \tag{40}$$

Similarly, the errors in forward solutions when using the minimizers as the model parameter are defined as

$$\tilde{e}_{NN} := \frac{\|u(\tilde{m}) - u_{NN}(\tilde{m}_{NN})\|_{l^2}}{\|u(\tilde{m})\|_{l^2}} \times 100\,, \qquad \tilde{e}^C_{NN} := \frac{\|u(\tilde{m}) - u^C_{NN}(\tilde{m}^C_{NN})\|_{l^2}}{\|u(\tilde{m})\|_{l^2}} \times 100\,. \tag{41}$$

The numerical minimizer $\tilde{m}$ and the forward solution at $\tilde{m}$ are shown in Fig. 11. The history of the cost function, the volumetric average of m, and the Lagrange multiplier are plotted in Fig. 12. In Fig. 13, the minimizers for the neural operators trained with smaller and larger datasets are compared. The figure also shows the minimizers when neural operators are corrected using the corrector operator $\mathcal{F}^C$. Finally, for all the twenty neural operators and the corrector of those neural operators, the percentage errors $\tilde{\varepsilon}_{NN}$, $\tilde{\varepsilon}^C_{NN}$, $\tilde{e}_{NN}$, and $\tilde{e}^C_{NN}$ are plotted in Fig. 14. The errors are tabulated in Table 3 to allow easier comparison of the accuracy of neural operators with and without corrections. From the error results in Table 3, it is clear that the neural operators consistently lead to minimizers with high error (as high as 80 percent). The corrector on the other hand provides an approximation of minimizers with errors below seven percent.

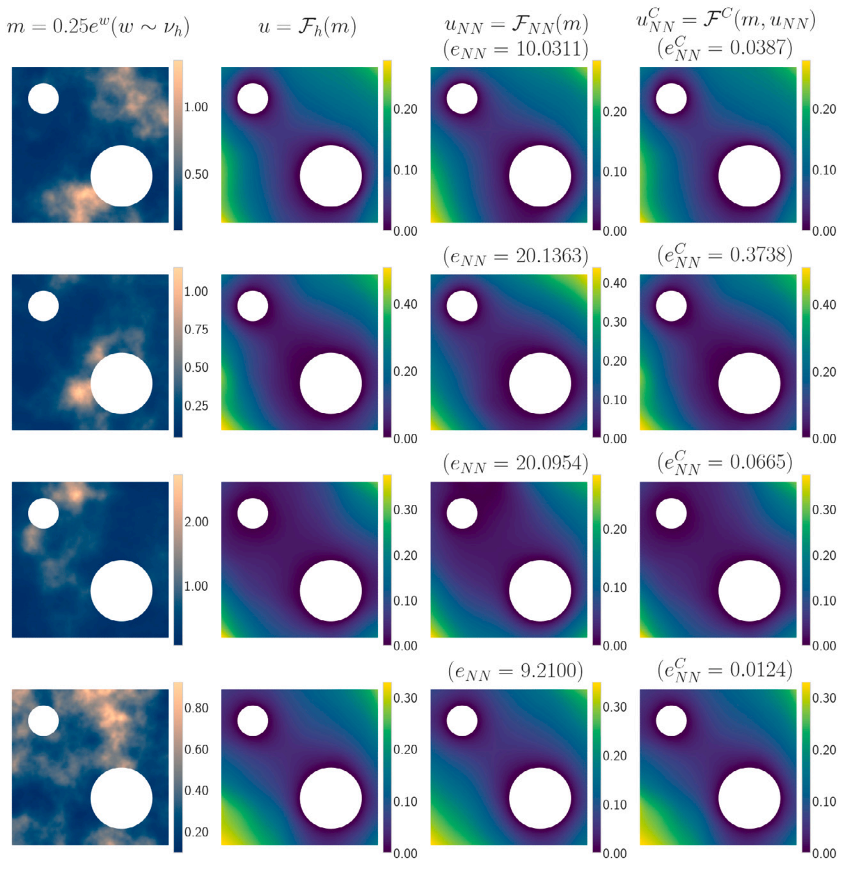


Fig. 9. Comparing true solution, neural operator prediction, and the correction of neural operator prediction for networks $1, 6, 11, 16$ (see Table 2) trained with smaller samples of data.

Table 3
Comparison of the percentage errors in minimizers and forward solutions at the minimizers for neural operators and corrected neural operators. Networks trained with small datasets are highlighted in blue, while the ones trained with relatively large datasets are highlighted in green.

Number	$\tilde{\varepsilon}_{NN}$	$\tilde{\varepsilon}^C_{NN}$	$\tilde{e}_{NN}$	$\tilde{e}^C_{NN}$	Number	$\tilde{\varepsilon}_{NN}$	$\tilde{\varepsilon}^C_{NN}$	$\tilde{e}_{NN}$	$\tilde{e}^C_{NN}$
1	72.14426	5.46892	22.43047	0.56382	11	75.05329	5.16826	14.79259	0.45834
2	68.70190	3.41497	10.39295	0.08882	12	71.45512	2.99042	11.51458	0.28541
3	71.45517	2.53612	10.25896	0.10772	13	74.81585	3.15803	11.99392	0.14022
4	63.65149	3.36944	9.54038	0.19412	14	65.21906	3.63778	12.64035	0.10310
5	65.68216	3.62392	9.73351	0.14817	15	66.73859	3.70730	13.78046	0.14303
6	80.76804	6.22848	30.17278	0.63044	16	77.65764	6.99276	35.45096	1.16036
7	75.65322	3.72890	16.44920	0.07026	17	75.78578	3.19518	14.10578	0.29605
8	79.60037	3.17328	14.71594	0.18724	18	70.69823	4.06247	13.14865	0.12392
9	75.74455	2.96135	13.32289	0.09598	19	68.44559	3.34363	10.35292	0.11209
10	74.11825	3.67819	11.89410	0.18198	20	66.36491	5.09965	8.65368	0.56261

5. Conclusion

The work considers a powerful approach for enhancing the accuracy and reliability of neural operators, especially when the neural operator accuracy is impacted by the unavailability of appropriate training distributions and sparse data. The approach is based on the corrector operator, which requires solving the linear variational problem given the input parameter m and the prediction

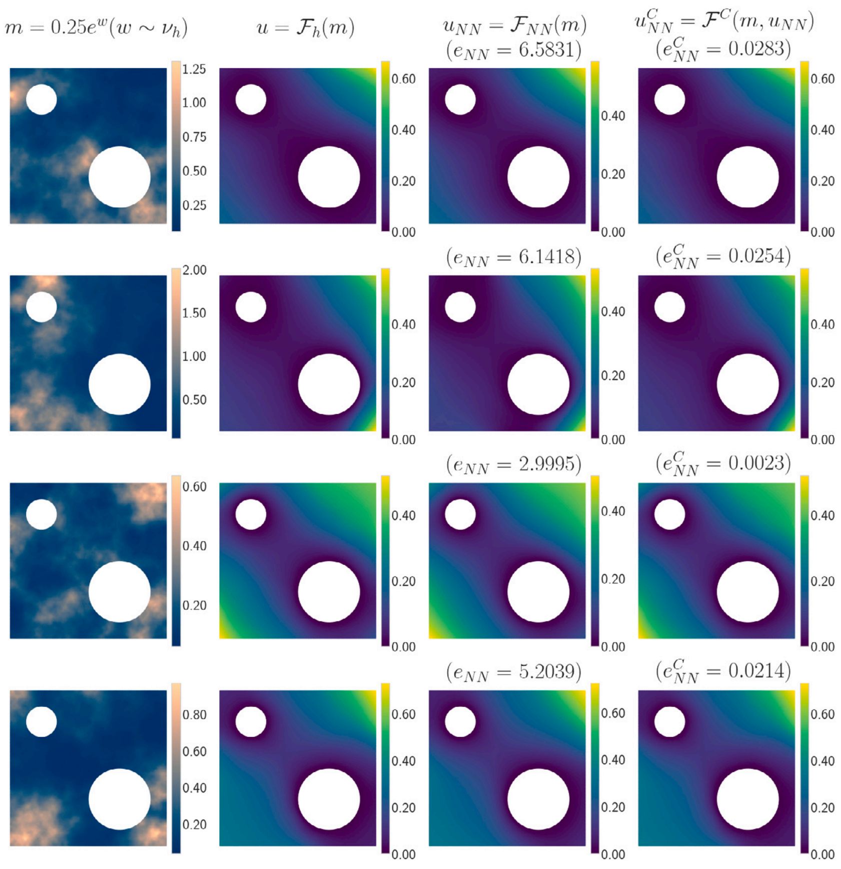


Fig. 10. Comparing true solution, neural operator prediction, and the correction of neural operator prediction for networks 5, 10, 15, 20 (see Table 2) trained with larger samples of data.

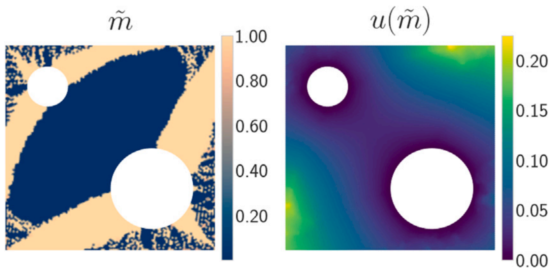


Fig. 11. Plot of "true" minimizer (up to numerical discretization error in forward and optimization problems) $\tilde{m}$ and corresponding forward solution $u(\tilde{m})$.

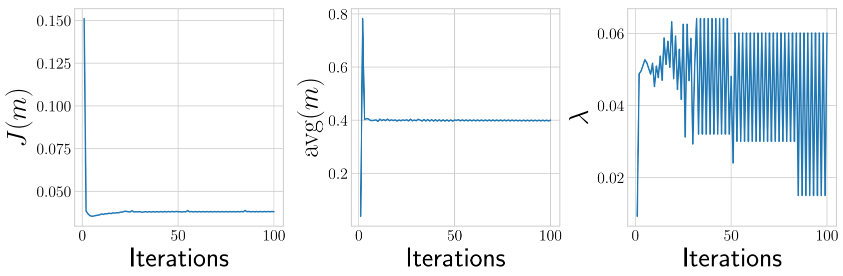


Fig. 12. History of compliance function $J(m)$, the volume average of m (note that the target volume average is $\eta = 0.4$), and the Lagrange multiplier during iterations.

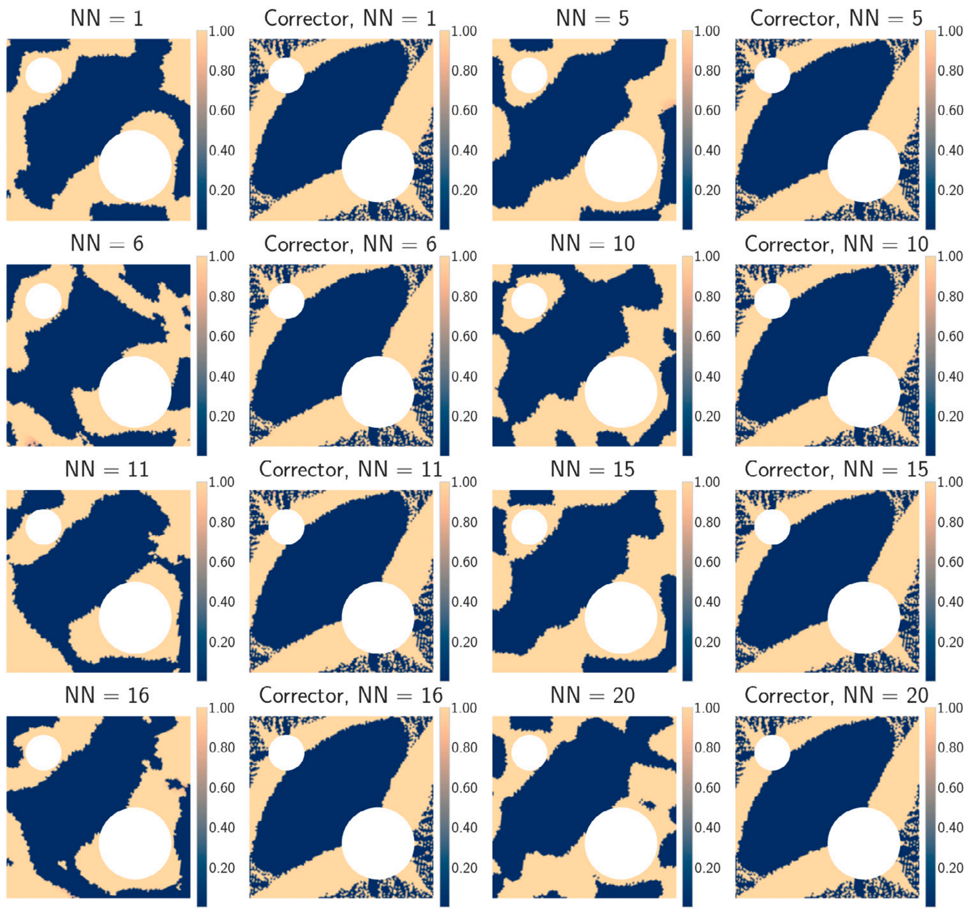


Fig. 13. Comparing the minimizers for neural networks trained with a smaller dataset (left two columns) and the larger dataset (right two columns). The odd columns (1 and 3) correspond to the minimizers obtained by employing neural operator surrogates in the cost function. In contrast, the even columns (2 and 4) are those where neural operator predictions are corrected using the corrector operator $\mathcal{F}^C$. For the properties of neural operators, see to Table 2.

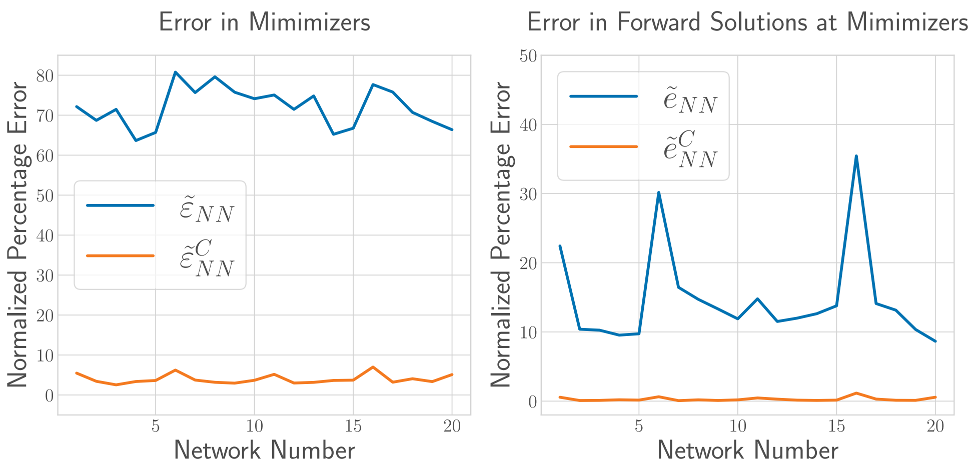


Fig. 14. Normalized minimizer errors $\tilde{\varepsilon}_{NN}$ and $\tilde{\varepsilon}^C_{NN}$ due to surrogate approximations of the forward problem (see (40)) and the error in forward solutions $\tilde{e}_{NN}$ and $\tilde{e}^C_{NN}$ defined in (41).

furnished by the neural operator. For the two examples considered in this work, the increase in accuracy obtained via the corrector operator is not possible to attain by simply tuning the hyperparameters of the neural operators and increasing the training data samples. For the topology optimization problem, the work highlights the limitations of neural operators in choosing the appropriate training samples and neural operators leading to results with large errors. The corrector operator for the topology optimization problem increased the accuracy of optimizers significantly. In summary, the approach seems to do a great job of increasing the accuracy and reliability of neural operators.

In the present work, the correction is computed external to neural operators. In contrast, future work will explore the possibility of integrating the correction step into the neural operators. Further, goal-oriented error estimates can be used to enhance the accuracy of neural operators with respect to specific quantities of interest. Finally, downstream applications of parameter estimation and design optimization of complex materials for mechanical loading and actuation are of interest where the neural operators will be used as surrogates of highly nonlinear parametric multiphysics models of mechanical deformation.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Prashant K Jha. reports financial support was provided by US Department of Energy.

Data availability

Data will be made available on request.

Acknowledgments

The author is thankful to the Late Professor John Tinsley Oden for editing and making corrections in several versions of the draft before it could be ready for publication. Professor Oden also helped write part of the second section of the article for which the author is grateful. This work was supported by the U. S. Department of Energy, Office of Science, USA, Office of Advanced Scientific Computing Research, Mathematical Multifaceted Integrated Capability Centers (MMICCS), under Award Number DE-SC0019303.

Appendix A. Corrector operator error analysis

In this section, Theorem 1 is proved. Let $m \in \mathcal{M}$, and $\tilde{u} \in \mathcal{U}$ be arbitrary approximation of $u = \mathcal{F}(m)$, where $\mathcal{F}$ is the solution operator. From the definition of corrector operator $\mathcal{F}^C$ (see (9)), it can be shown that

$$\begin{aligned}
& \mathcal{F}^C(m) = u^C = \tilde{u} - \delta_u\mathcal{R}(m,\tilde{u})^{-1}\mathcal{R}(m,\tilde{u}) \\
\Rightarrow \quad & \underbrace{u - u^C}_{=e^C} = \underbrace{u - \tilde{u}}_{=\tilde{e}} + \delta_u\mathcal{R}(m,\tilde{u})^{-1}\left(\mathcal{R}(m,\tilde{u}) - \underbrace{\mathcal{R}(m,u)}_{=0}\right) \\
\Rightarrow \quad & e^C = \tilde{e} - \delta_u\mathcal{R}(m,\tilde{u})^{-1}\left(\mathcal{R}(m,u) - \mathcal{R}(m,\tilde{u})\right) .
\end{aligned} \tag{A.1}$$

Using the identity $\tilde{e} = \delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u\mathcal{R}(m,\tilde{u})(\tilde{e})$ and the Taylor series expansion

$$\mathcal{R}(m,u) - \mathcal{R}(m,\tilde{u}) = \delta_u\mathcal{R}(m,\tilde{u})(\tilde{e}) + \int_0^1 (1-s)\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})(\tilde{e},\tilde{e})\,\mathrm{d}s$$

in (A.1), it holds

$$\begin{aligned}
e^C &= \delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u\mathcal{R}(m,\tilde{u})(\tilde{e}) - \delta_u\mathcal{R}(m,\tilde{u})^{-1}\left(\mathcal{R}(m,u) - \mathcal{R}(m,\tilde{u})\right) \\
&= \delta_u\mathcal{R}(m,\tilde{u})^{-1}\left[\delta_u\mathcal{R}(m,\tilde{u})(\tilde{e}) - \left(\mathcal{R}(m,u) - \mathcal{R}(m,\tilde{u})\right)\right] \\
&= \delta_u\mathcal{R}(m,\tilde{u})^{-1}\left[-\int_0^1 (1-s)\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})(\tilde{e},\tilde{e})\,\mathrm{d}s\right] \\
&= \int_0^1 (s-1)\delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})(\tilde{e},\tilde{e})\,\mathrm{d}s .
\end{aligned} \tag{A.2}$$

Next, define a bilinear operator $A(s) : \mathcal{U}\times\mathcal{U} \to \mathcal{U}$ for $s \in [0,1]$ as follows

$$A(s)(p,q) := \delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})(p,q)$$

and note that

$$\|A(s)(p,q)\|_{\mathcal{U}} \le \|A(s)\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U};\mathcal{U})}\,\|p\|_{\mathcal{U}}\,\|q\|_{\mathcal{U}} \le \left[\sup_{s\in[0,1]} \|A(s)\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U};\mathcal{U})}\right]\|p\|_{\mathcal{U}}\,\|q\|_{\mathcal{U}}.$$

Taking the norm of both sides in (A.2) and using the above estimate, the following can be shown

$$\|e^C\|_{\mathcal{U}} \le \int_0^1 (1-s)\|A(s)(\tilde{e},\tilde{e})\|_{\mathcal{U}} \le \frac{1}{2}\left[\sup_{s\in[0,1]} \|\delta_u\mathcal{R}(m,\tilde{u})^{-1}\delta_u^2\mathcal{R}(m,\tilde{u}+s\tilde{e})\|_{\mathcal{L}(\mathcal{U}\times\mathcal{U};\mathcal{U})}\right]\|\tilde{e}\|_{\mathcal{U}}^2 . \quad \square \tag{A.3}$$

Appendix B. Bounds on derivatives of a residual for nonlinear diffusion example

In this section, Theorem 3 is established. In what follows, the preliminary results needed in the proof are first collected, and then Theorem 3 is proved.

B.1. Preliminary results

Let X and Y be two Banach spaces. X is said to be embedded continuously in Y, written as $X \hookrightarrow Y$, if

- $X \subseteq Y$;
- the canonical injection $i : X \to Y$ is a continuous (linear) operator, i.e., there exists a constant $C > 0$ such that

$$\|i(u)\|_Y \le C\|u\|_X, \qquad \forall u \in X.$$

  Here, C is independent of u.

Embedding of X into Y is compact if, in addition to the above conditions, the canonical injection operator i is a compact operator. Relevant Sobolev embedding results from Theorems 2.6.1 and 2.6.2 in [91] are collected in the following theorem:

Theorem 4. *Let $\Omega \subset \mathbb{R}^d$ be the open, bounded, and smooth domain where $d = 2, 3$. It holds that*

$$H^1(\Omega) \hookrightarrow L^q(\Omega),$$

for every $q \in [1,\infty)$ when $d = 2$ and for every $q \in [1,6]$ when $d = 3$. Further, the embedding is compact for every $q \in [1,\infty)$ when $d = 2$ and for every $q \in [1,6)$ when $d = 3$.

For $d = 3$, if Ω is open and bounded subset of $\mathbb{R}^d$ or $\Omega = \mathbb{R}^d$, then

$$H^1_0(\Omega) \hookrightarrow L^q(\Omega)$$

for every $d \in [1,6]$. Further, the embedding is compact for every $q \in [1,6)$.

Consider the case when Ω is open, bounded, and smooth with $d = 2$ or $d = 3$. Let $\mathcal{U} := H^1_0(\Omega)$. Then for every $u \in \mathcal{U}$, using the above Sobolev embedding theorem, it holds

$$\|u\|_{L^4(\Omega)} \le C_S\|u\|_{\mathcal{U}}\,, \tag{B.1}$$

where C_S independent of u is the Sobolev embedding constant.

Next, the Poincaré inequality theorem following [Theorem 2.6.3, [91]] is stated:

Theorem 5. *Let $\Omega \subset \mathbb{R}^d$ be open and bounded. Then there exists a constant $C_P > 0$, depending only on Ω, such that*

$$\|u\|_{L^2(\Omega)} \le C_P\|\nabla u\|_{L^2(\Omega)}, \qquad \forall u \in H^1_0(\Omega)\,. \tag{B.2}$$

B.2. Proof of Theorem 3

The proof is divided into three steps as follows.

Step 1. First, the upper bound on $\|\delta_u\mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{U}^*}$ is established. Using the definition of operator norm gives

$$\|\delta_u\mathcal{R}(m,\tilde{u})\|_{\mathcal{L}(\mathcal{U};\mathcal{U}^*)} = \sup_{\|v\|_{\mathcal{U}}=1} \|\delta_u\mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{U}^*}\,. \tag{B.3}$$

Further,

$$\|\delta_u\mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{U}^*} = \sup_{\|w\|_{\mathcal{U}}=1} |\langle w, \delta_u\mathcal{R}(m,\tilde{u})(v)\rangle_{\mathcal{U}}|\,. \tag{B.4}$$

Noting the expression of $\delta_u\mathcal{R}(m,u)$ in (12) and setup of the problem described in Section 2.3, it holds

$$\begin{aligned}
|\langle w, \delta_u\mathcal{R}(m,\tilde{u})(v)\rangle_{\mathcal{U}}| &\le \int_\Omega \left\{ \left(\kappa_0 \sup_{x\in\Omega}|m(x)|\right)|\nabla v|\,|\nabla w| + 3\alpha\tilde{u}^2 v w \right\} \mathrm{d}x \\
&\le \left[\kappa_0\|m\|_{L^\infty(\Omega)}\|\nabla v\|_{L^2(\Omega)}\,\|\nabla w\|_{L^2(\Omega)} + 3\alpha\left(\int_\Omega \tilde{u}^4\,\mathrm{d}x\right)^{1/2}\left(\int_\Omega v^2w^2\,\mathrm{d}x\right)^{1/2}\right] \\
&\le \kappa_0\|m\|_{L^\infty(\Omega)}\|v\|_{\mathcal{U}}\,\|w\|_{\mathcal{U}} + 3\alpha\|\tilde{u}\|^2_{L^4(\Omega)}\,\|v\|_{L^4(\Omega)}\,\|w\|_{L^4(\Omega)} \\
&\le \kappa_0\|m\|_{L^\infty(\Omega)}\|v\|_{\mathcal{U}}\,\|w\|_{\mathcal{U}} + 3\alpha C_S^4\|\tilde{u}\|^2_{\mathcal{U}}\,\|v\|_{\mathcal{U}}\,\|w\|_{\mathcal{U}}\,,
\end{aligned}$$

where Hölder inequality is used in the second and third equations, and the Sobolev embedding property (see (B.1)) in the last equation. Using the above estimate in (B.4) and combining the result with (B.3), it can be shown that

$$\|\delta_u\mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{U}^*} \le \underbrace{\left(\kappa_0\|m\|_{L^\infty(\Omega)} + 3\alpha C_S^4\|\tilde{u}\|^2_{\mathcal{U}}\right)}_{\bar{C}_{\delta R}}\,\|v\|_{\mathcal{U}}\,. \tag{B.5}$$

For the lower bound on $\|\delta_u \mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{V}^*}$, proceeding as follows

$$\begin{aligned}
\|\delta_u \mathcal{R}(m,\tilde{u})(v)\|_{\mathcal{V}^*} &= \sup_{\|w\|_{\mathcal{V}}=1} |\langle w, \delta_u \mathcal{R}(m,\tilde{u})(v)\rangle_{\mathcal{V}}| \\
&\geq \frac{|\langle v, \delta_u \mathcal{R}(m,\tilde{u})(v)\rangle_{\mathcal{V}}|}{\|v\|_{\mathcal{V}}} \\
&= \frac{1}{\|v\|_{\mathcal{V}}} \left| \int_\Omega \left\{ \kappa_0 m(x) |\nabla v|^2 + 3\alpha \tilde{u}^2 v^2 \right\} \mathrm{d}x \right| \\
&\geq \frac{\kappa_0 m_{\mathrm{lw}}}{\|v\|_{\mathcal{V}}} \|\nabla v\|^2_{L^2(\Omega)} = \frac{\kappa_0 m_{\mathrm{lw}}}{2\|v\|_{\mathcal{V}}} \left[\|\nabla v\|^2_{L^2(\Omega)} + \|\nabla v\|^2_{L^2(\Omega)} \right] \\
&\geq \frac{\kappa_0 m_{\mathrm{lw}}}{2\|v\|_{\mathcal{V}}} \left[\|\nabla v\|^2_{L^2(\Omega)} + C_P^{-2} \|v\|^2_{L^2(\Omega)} \right] \\
&\geq \underbrace{\frac{\kappa_0 m_{\mathrm{lw}}}{2} \min\{1, C_P^{-2}\}}_{=:\hat{C}_{\delta R}} \frac{1}{\|v\|_{\mathcal{V}}} \|v\|^2_{\mathcal{V}} = \hat{C}_{\delta R} \|v\|_{\mathcal{V}},
\end{aligned}$$

where $\sup_{\|w\|_{\mathcal{V}}=1} |\langle w, \delta_u \mathcal{R}(m,\tilde{u})(v)\rangle_{\mathcal{V}}| \geq \frac{|\langle v, \delta_u \mathcal{R}(m,\tilde{u})(v)\rangle_{\mathcal{V}}|}{\|v\|_{\mathcal{V}}}$ is used in the second equation. In the fourth equation, the positive term is dropped, and the property that $m \geq m_{\mathrm{lw}}$ is used, since $m \in \mathcal{M} = \{g \in L^2(\Omega) \cap L^\infty(\Omega) : g \geq m_{\mathrm{lw}}\}$. In the fifth step, the Poincaré inequality from (B.2) is applied. Combining the above with (B.5), the proof of (i) of Theorem 3 is complete.

Step 2. In this step, the upper bound on the norm of $\delta_u \mathcal{R}(m,\tilde{u})^{-1}$ is established. Let $A := \delta_u \mathcal{R}(m,\tilde{u})$ and $r = A(v) \in \mathcal{V}^*$, then

$$\|A^{-1}(r)\|_{\mathcal{V}} = \|A^{-1}(A(v))\|_{\mathcal{V}} = \|v\|_{\mathcal{V}}.$$

However, since $r = A(v)$, from the previous calculations, it holds that $\hat{C}_{\delta R}\|v\|_{\mathcal{V}} \leq \|r\|_{\mathcal{V}^*}$, i.e., $\|v\|_{\mathcal{V}} \leq \frac{1}{\hat{C}_{\delta R}} \|r\|_{\mathcal{V}^*}$. Combining this with the equation above, the following can be shown

$$\|A^{-1}(r)\|_{\mathcal{V}} = \|v\|_{\mathcal{V}} \leq \hat{C}_{\delta R} \|r\|_{\mathcal{V}^*}.$$

The above establishes (ii) of Theorem 3.

Step 3. To prove the last result, i.e., (iii) of Theorem 3, the definition of an operator norm is expanded as follows

$$\|\delta_u^2 \mathcal{R}(m,\tilde{u})\|_{\mathcal{L}(\mathcal{V}\times\mathcal{V};\mathcal{V}^*)} = \sup_{\|v\|_{\mathcal{V}}=1, \|w\|_{\mathcal{V}}=1} \|\delta_u^2 \mathcal{R}(m,\tilde{u})(v,w)\|_{\mathcal{V}^*}.$$

Focusing on the $\|\delta_u^2 \mathcal{R}(m,\tilde{u})(v,w)\|_{\mathcal{V}^*}$, it holds that

$$\begin{aligned}
\|\delta_u^2 \mathcal{R}(m,\tilde{u})(v,w)\|_{\mathcal{V}^*} &= \sup_{\|q\|_{\mathcal{V}}=1} |\langle q, \delta_u^2 \mathcal{R}(m,\tilde{u})(v,w)\rangle_{\mathcal{V}}| \\
&= \sup_{\|q\|_{\mathcal{V}}=1} \left| \int_\Omega 6\alpha \tilde{u} v w q \, \mathrm{d}x \right| \leq 6\alpha \sup_{\|q\|_{\mathcal{V}}=1} \|\tilde{u}\|_{L^4(\Omega)} \|v\|_{L^4(\Omega)} \|w\|_{L^4(\Omega)} \|q\|_{L^4(\Omega)} \\
&\leq 6\alpha C_S^4 \sup_{\|q\|_{\mathcal{V}}=1} \|\tilde{u}\|_{\mathcal{V}} \|v\|_{\mathcal{V}} \|w\|_{\mathcal{V}} \|q\|_{\mathcal{V}} \\
&= \underbrace{\left(6\alpha C_S^4 \|\tilde{u}\|_{\mathcal{V}}\right)}_{=:\tilde{C}_{\delta^2 R}} \|v\|_{\mathcal{V}} \|w\|_{\mathcal{V}},
\end{aligned} \tag{B.6}$$

where Hölder inequality is applied twice, and the Sobolev embedding property is used. This completes the proof of Theorem 3.

Appendix C. Numerical method for the topology optimization problem

A bi-level iteration scheme is developed for solving the optimization problem (39). The method employed here is based on the so-called SIMP (Solid Isotropic Material with Penalization) method and closely follows the topology optimization example[7] in [102].

Since $u = u(m) \in \mathcal{V}$ solves $\langle v, \mathcal{R}(m,u)\rangle = 0$ for all $v \in \mathcal{V}$, taking $v = u$, and using the definition of $\mathcal{R}$ from (34), it can be easily shown that

$$\int_{\Gamma_{out}} g\, u(m) \,\mathrm{d}S(\boldsymbol{x}) = \int_\Omega m \nabla u \cdot \nabla u \,\mathrm{d}\boldsymbol{x} + \int_\Omega u^4 \,\mathrm{d}\boldsymbol{x}.$$

Let $\boldsymbol{q} = \boldsymbol{q}(m) = m\nabla u(m)$ denote the flux and $e = e(m) = \boldsymbol{q}(m) \cdot \boldsymbol{q}(m)$, then from the above, the compliance can be expressed as

$$\int_{\Gamma_{out}} g\, u(m) \,\mathrm{d}S(\boldsymbol{x}) = \int_\Omega \left[\frac{e(m)}{m} + u(m)^4 \right] \mathrm{d}\boldsymbol{x}.$$

[7] https://comet-fenics.readthedocs.io/en/latest/demo/topology_optimization/simp_topology_optimization.html

Using the relation above, the optimization problem (39) can be re-written as follows

$$\begin{aligned}
&\min_{m\in\mathcal{M}_h,\lambda\in\mathbb{R}} && \hat{J}(m,\lambda,u) = \int_\Omega \left[\frac{e(m)}{m} + u(m)^4\right] \mathrm{d}\boldsymbol{x} + \lambda\left(\int_\Omega m\,\mathrm{d}\boldsymbol{x} - \eta|\Omega|\right), \\
&\text{where } u = u(m) \in \mathcal{U}_h \text{ satisfies} && \langle v, \mathcal{R}(m,u)\rangle = 0, \qquad \forall v \in \mathcal{U}_h, \\
&\text{and} && 0 < m_{\text{lw}} \le m \le 1.
\end{aligned} \tag{C.1}$$

The problem above is coupled in variables m, λ, and u. To simplify the computation, an iterative scheme is sought in which the variables can be uncoupled and sequentially computed. One possible approach is to consider an iteration (referring to this as outer iteration) where given solutions from the previous iteration, i.e., $m_k, \lambda_k, u_k = u(m_k)$, first, the pair (m_{k+1}, λ_{k+1}) is updated by solving the approximate minimization problem with fixed $u = u_k$, i.e.,

$$\begin{aligned}
&(m_{k+1},\lambda_{k+1}) = \operatorname*{arg\,min}_{m\in\mathcal{M}_h,\lambda\in\mathbb{R}} && \hat{J}(m,\lambda,u_k) = \int_\Omega \left[\frac{e(m_k)}{m} + u(m_k)^4\right] \mathrm{d}\boldsymbol{x} + \lambda\left(\int_\Omega m\,\mathrm{d}\boldsymbol{x} - \eta|\Omega|\right), \\
&\text{and} && 0 < m_{\text{lw}} \le m \le 1.
\end{aligned} \tag{C.2}$$

Next, given m_{k+1}, $\mathcal{R}(m_{k+1}, u_{k+1}) = 0$ is solved for u_{k+1}. The iteration over k is continued until $\|m_{k+1} - m_k\|_{L^2(\Omega)} \le \gamma_{tol}$ or until k reaches the maximum number of iterations. The algorithm for this outer iteration is presented in Algorithm 1.

Algorithm 1: Outer iteration for solving (39) or equivalently (C.1).

```
Input:
(a) set up mesh, variational problem, and optimization parameters ;
(b) take initial guess $m_0$, $\lambda_0 = 1$;
Outer iteration:
$k = 0$;
while $k < n^{out}_{max\ iter}$ do
    (1) given $m_k$, $\lambda_k$, and $u_k$, solve for $(m_{k+1}, \lambda_{k+1}) \in \mathcal{M}_h \times \mathbb{R}$ following Algorithm 2 ;
    (2) solve $\langle v, \mathcal{R}(m_{k+1}, u_{k+1})\rangle = 0$, $\forall v \in \mathcal{U}_h$, for $u_{k+1} \in \mathcal{U}_h$ ;
    (3) update flux $\boldsymbol{q}_{k+1} = m_{k+1}\nabla u_{k+1}$ and $e_{k+1} = \boldsymbol{q}_{k+1} \cdot \boldsymbol{q}_{k+1}$ ;
    (4) $m_k = m_{k+1}$, $\lambda_k = \lambda_{k+1}$, $u_k = u_{k+1}$, and $k \leftarrow k+1$ ;
    if $||m_{k+1} - m_k||_{L^2(\Omega)} < \gamma_{tol}$ then break ;
end
Return: $\tilde{m} = m_k$.
```

Focusing now on (C.2), the problem is solved iteratively (inner iteration) where in each iteration i, first $m^{(i)}$ is updated to compute $m^{(i+1)}$ while keeping $\lambda^{(i)}$ fixed and then $\lambda^{(i)}$ is updated into $\lambda^{(i+1)}$. During the inner iteration inside the outer iteration step k, u is fixed to $u_k = u(m_k)$ throughout; Algorithm 1. The key question here is how to compute the updated value $m^{(i+1)}$ given $\lambda = \lambda^{(i)}$ and $u = u_k$. Towards this, from Eq. (C.2), when the variables λ and u are fixed, the optimization problem on m becomes

$$\begin{aligned}
&m^{(i+1)} = \operatorname*{arg\,min}_{m\in\mathcal{M}} && \hat{J}(m,\lambda^{(i)},u_k) = \int_\Omega \left[\frac{e_k}{m} + u_k^4\right] \mathrm{d}\boldsymbol{x} + \lambda^{(i)}\left(\int_\Omega m\,\mathrm{d}\boldsymbol{x} - \eta|\Omega|\right), \\
&\text{and} && 0 < m_{\text{lw}} \le m \le 1,
\end{aligned} \tag{C.3}$$

where $e_k = \boldsymbol{q}(m_k) \cdot \boldsymbol{q}(m_k) = (m_k\nabla u_k) \cdot (m_k\nabla u_k)$. Taking the variation of $\hat{J}(\cdot, \lambda^{(i)}, u_k)$ in the direction of arbitrary $w \in \mathcal{M}$, and setting it to zero, gives

$$\int_\Omega \left[-\frac{e_k}{2} + \lambda^{(i)}\right] w\,\mathrm{d}\boldsymbol{x} = 0, \qquad \forall w \in \mathcal{M} \qquad \Rightarrow \qquad m = \sqrt{\frac{e_k}{\lambda^{(i)}}}.$$

Thus, the formula for updating m given u_k and $\lambda^{(i)}$ is given by

$$m^{(i+1)} = \min\left\{1, \max\left\{m_{\text{lw}}, \sqrt{\frac{e_k}{\lambda^{(i)}}}\right\}\right\}, \tag{C.4}$$

where the upper and lower bound constraints on m are enforced strongly. The algorithm for the inner iteration where λ and m are successively updated for a given outer iteration step k is presented in Algorithm 2. This algorithm is based on the bisection method; see [103].

Algorithm 2: Inner iteration for pair (m_{k+1}, λ_{k+1}) given m_k, λ_k, u_k.

```
Input:
(a) $\eta$, lower bound $m_{\mathrm{lw}}$, and tolerance $m_{\mathrm{tol}}$ to check volumetric constraint ;
(b) outer iteration step $k$ and solutions $m_k, \lambda_k, u_k$ ;
Setup:
(a) let $m^{(0)} = m_k$, $\lambda^{(0)} = \lambda_k$, and let $\bar{m}^{(i)}$ denote the volume average of $m^{(i)}$ ;
(b) let $m$ and $\lambda$ are current values of variables, and let $\lambda_{min} = 0$ and $\lambda_{max} = 0$ ;
Inner iteration (bracketing):
if $\bar{m}^{(0)} < \eta$ then
    $\lambda_{min} = \lambda^{(0)}$, $i = 0$ ;
    while $\bar{m}^{(i)} < \eta$ do
        (1) update $\lambda$: $\lambda^{(i+1)} = \frac{\lambda^{(i)}}{2}$ ;
        (2) update $m$: given $m_k$, $u_k$, and $\lambda^{(i+1)}$, compute $m^{(i+1)}$ using (C.4) ;
        (3) $m^{(i)} = m^{(i+1)}$, $\lambda^{(i)} = \lambda^{(i+1)}$, and $i \leftarrow i + 1$ ;
    end
    $\lambda = \lambda^{(i)}$, $m = m^{(i)}$, and $\lambda_{max} = \lambda$ ;
end
else
    $\lambda_{max} = \lambda^{(0)}$, $i = 0$ ;
    while $\bar{m}^{(i)} > \eta$ do
        (1) update $\lambda$: $\lambda^{(i+1)} = 2\lambda^{(i)}$ ;
        (2) update $m$: given $m_k$, $u_k$, and $\lambda^{(i+1)}$, compute $m^{(i+1)}$ using (C.4) ;
        (3) $m^{(i)} = m^{(i+1)}$, $\lambda^{(i)} = \lambda^{(i+1)}$, $i \leftarrow i + 1$ ;
    end
    $\lambda = \lambda^{(i)}$, $m = m^{(i)}$, and $\lambda_{min} = \lambda$ ;
end
Inner iteration (bisection):
$i = 0$, $\lambda^{(0)} = \lambda$, and $m^{(0)} = m$ ;
while $|\bar{m}^{(0)} - \eta| > \eta\, m_{tol}$ do
    (1) update $\lambda$: $\lambda^{(i+1)} = \frac{\lambda_{min}+\lambda_{max}}{2}$ ;
    (2) update $m$: given $m_k$, $u_k$, and $\lambda^{(i+1)}$, compute $m^{(i+1)}$ using (C.4) ;
    (3) update $\lambda_{min}$ and $\lambda_{max}$ :
    if $\bar{m}^{(i+1)} < \eta$ then $\lambda_{min} = \lambda^{(i+1)}$ ;
    else $\lambda_{max} = \lambda^{(i+1)}$ ;
    (4) $m^{(i)} = m^{(i+1)}$, $\lambda^{(i)} = \lambda^{(i+1)}$, and $i \leftarrow i + 1$ ;
end
$m_{k+1} = m^{(i)}$, $\lambda_{k+1} = \lambda^{(i)}$ ;
Return: $(m_{k+1}, \lambda_{k+1})$.
```

References

[1] L. Cao, T. O'Leary-Roseberry, P.K. Jha, J.T. Oden, O. Ghattas, Residual-based error correction for neural operator accelerated infinite-dimensional Bayesian inverse problems, J. Comput. Phys. 486 (2023) 112104.

[2] P.K. Jha, J.T. Oden, Goal-oriented a-posteriori estimation of model error as an aid to parameter estimation, J. Comput. Phys. 470 (2022) 111575.

[3] R. Zhao, Y. Kim, S.A. Chester, P. Sharma, X. Zhao, Mechanics of hard-magnetic soft materials, J. Mech. Phys. Solids 124 (2019) 244–263.

[4] A.H. Rahmati, R. Jia, K. Tan, X. Zhao, Q. Deng, L. Liu, P. Sharma, Theory of hard magnetic soft materials to create magnetoelectricity, J. Mech. Phys. Solids 171 (2023) 105136.

[5] F. Darbaniyan, K. Dayal, L. Liu, P. Sharma, Designing soft pyroelectric and electrocaloric materials using electrets, Soft Matter 15 (2019) 262–277.

[6] A. Nandy, C. Jog, A monolithic finite-element formulation for magnetohydrodynamics, Sādhanā 43 (2018) 151.

[7] L. Cao, O. Ghattas, J.T. Oden, A globally convergent modified Newton method for the direct minimization of the Ohta–Kawasaki energy with application to the directed self-assembly of diblock copolymers, SIAM J. Sci. Comput. 44 (2022) B51–B79.

[8] R.P. Lipton, R.B. Lehoucq, P.K. Jha, Complex fracture nucleation and evolution with nonlocal elastodynamics, J. Peridyn. Nonlocal Model. 1 (2019) 122–130.

[9] P.K. Jha, R.P. Lipton, Kinetic relations and local energy balance for lefm from a nonlocal peridynamic model, Int. J. Fract. 226 (2020) 81–95.

[10] K. Dayal, K. Bhattacharya, Kinetics of phase transformations in the peridynamic formulation of continuum mechanics, J. Mech. Phys. Solids 54 (2006) 1811–1842.

[11] T. Breitzman, K. Dayal, Bond-level deformation gradients and energy averaging in peridynamics, J. Mech. Phys. Solids 110 (2018) 192–204.

[12] R. Lipton, E. Said, P. Jha, Free damage propagation with memory, J. Elasticity 133 (2018) 129–153.

[13] P.K. Jha, R. Lipton, Numerical analysis of nonlocal fracture models in holder space, SIAM J. Numer. Anal. 56 (2018) 906–941.

[14] P. Jha, R. Lipton, Finite element approximation of nonlocal dynamic fracture models, Discrete Contin. Dyn. Syst. Ser. B 26 (2021) 1675.

[15] B.E. Abali, F. Aldakheel, T.I. Zohdi, Multiphysics computation of thermomechanical fatigue in electronics under electrical loading, in: Current Trends and Open Problems in Computational Mechanics, Springer, 2022, pp. 1–14.

[16] S. Dutta, C. Jog, A monolithic arbitrary Lagrangian–Eulerian-based finite element strategy for fluid–structure interaction problems involving a compressible fluid, Internat. J. Numer. Methods Engrg. 122 (2021) 6037–6102.
[17] P.K. Jha, P.S. Desai, D. Bhattacharya, R. Lipton, Peridynamics-based discrete element method (peridem) model of granular systems involving breakage of arbitrarily shaped particles, J. Mech. Phys. Solids 151 (2021) 104376.
[18] M. Torbati, K. Mozaffari, L. Liu, P. Sharma, Coupling of mechanical deformation and electromagnetic fields in biological cells, Rev. Modern Phys. 94 (2022) 025003.
[19] D.A. Hormuth, A.M. Jarrett, E.A. Lima, M.T. McKenna, D.T. Fuentes, T.E. Yankeelov, Mechanism-based modeling of tumor growth and treatment response constrained by multiparametric imaging data, JCO Clin. Cancer Inform. 3 (2019) 1–10.
[20] M. Fritz, P.K. Jha, T. Köppl, J.T. Oden, B. Wohlmuth, Analysis of a new multispecies tumor growth model coupling 3d phase-fields with a 1d vascular network, Nonlinear Anal. RWA 61 (2021) 103331.
[21] M. Fritz, P.K. Jha, T. Köppl, J.T. Oden, A. Wagner, B. Wohlmuth, Modeling and simulation of vascular tumors embedded in evolving capillary networks, Comput. Methods Appl. Mech. Engrg. 384 (2021) 113975.
[22] J.T. Oden, E.A. Lima, R.C. Almeida, Y. Feng, M.N. Rylander, D. Fuentes, D. Faghihi, M.M. Rahman, M. DeWitt, M. Gadde, et al., Toward predictive multiscale modeling of vascular tumor growth: Computational and experimental oncology for tumor prediction, Arch. Comput. Methods Eng. 23 (2016) 735–779.
[23] J.T. Oden, A. Hawkins, S. Prudhomme, General diffuse-interface theories and an approach to predictive tumor growth modeling, Math. Models Methods Appl. Sci. 20 (2010) 477–517.
[24] P.K. Jha, L. Cao, J.T. Oden, Bayesian-based predictions of Covid-19 evolution in texas using multispecies mixture-theoretic continuum models, Comput. Mech. 66 (2020) 1055–1068.
[25] N. Petra, J. Martin, G. Stadler, O. Ghattas, A computational framework for infinite-dimensional bayesian inverse problems, part ii: Stochastic newton mcmc with application to ice sheet flow inverse problems, SIAM J. Sci. Comput. 36 (2014) A1525–A1555.
[26] A. Hawkins-Daarud, S. Prudhomme, K.G. van der Zee, J.T. Oden, Bayesian calibration, validation, and uncertainty quantification of diffuse interface models of tumor growth, J. Math. Biol. 67 (2013) 1457–1485.
[27] L. Cao, K. Wu, J.T. Oden, P. Chen, O. Ghattas, Bayesian model calibration for diblock copolymer thin film self-assembly using power spectrum of microscopy data, 2023, arXiv preprint arXiv:2306.05398.
[28] M. Karimi, M. Massoudi, K. Dayal, M. Pozzi, High-dimensional nonlinear Bayesian inference of poroelastic fields from pressure data, Math. Mech. Solids (2023) 10812865221140840.
[29] M. Karimi, K. Dayal, M. Pozzi, Hessian-informed hamiltonian Monte Carlo for high-dimensional problems, 2023, arXiv preprint arXiv:2305.01576.
[30] L. Bi, J. Sovizi, K. Mathieu, W. Stefan, S. Thrower, J. Hazle, D. Fuentes, Bayesian inference and model selection for physiologically-based pharmacokinetic modeling of superparamagnetic iron oxide nanoparticles, in: Medical Imaging 2018: Biomedical Applications in Molecular, Structural, and Functional Imaging. Vol. 10578, SPIE, 2018, pp. 584–589.
[31] B. Liang, J. Tan, L. Lozenski, D.A. Hormuth II, T.E. Yankeelov, U. Villa, D. Faghihi, Bayesian inference of tissue heterogeneity for individualized prediction of glioma growth, IEEE Trans. Med. Imaging (2023).
[32] J.T. Oden, Adaptive multiscale predictive modelling, Acta Numer. 27 (2018) 353–450.
[33] A.K. Nandy, C. Jog, Optimization of vibrating structures to reduce radiated noise, Struct. Multidiscip. Optim. 45 (2012) 717–728.
[34] P. Chen, M.R. Haberman, O. Ghattas, Optimal design of acoustic metamaterial cloaks under uncertainty, J. Comput. Phys. 431 (2021) 110114.
[35] B.S. Cohen, A.I. March, K.E. Willcox, D.W. Miller, A level set-based topology optimization approach for thermally radiating structures, Struct. Multidiscip. Optim. 65 (2022) 167.
[36] R.B. Haber, C.S. Jog, M.P. Bendsøe, A new approach to variable-topology shape design using a constraint on perimeter, Struct. Optim. 11 (1996) 1–12.
[37] C.S. Jog, Topology design of structures subjected to periodic loading, J. Sound Vib. 253 (2002) 687–709.
[38] O. Ghattas, K. Willcox, Learning physics-based models from data: Perspectives from inverse problems and model reduction, Acta Numer. 30 (2021) 445–554.
[39] R. Lipton, A.P. Velo, Optimal design of gradient fields with applications to electrostatics, Stud. Math. Appl. 31 (2002) 509.
[40] R. Lipton, Design of functionally graded composite structures in the presence of stress constraints, Int. J. Solids Struct. 39 (2002) 2575–2586.
[41] M.P. Bendsøe, A.R. Díaz, R. Lipton, J.E. Taylor, Optimal design of material properties and material distribution for multiple loading conditions, Internat. J. Numer. Methods Engrg. 38 (1995) 1149–1170.
[42] E. Lima, J. Oden, D. Hormuth, T. Yankeelov, R. Almeida, Selection, calibration, and validation of models of tumor growth, Math. Models Methods Appl. Sci. 26 (2016) 2341–2368.
[43] E. Lima, J. Oden, B. Wohlmuth, A. Shahmoradi, D. Hormuth II, T. Yankeelov, L. Scarabosio, T. Horger, Selection and validation of predictive models of radiation effects on tumor growth based on noninvasive imaging data, Comput. Methods Appl. Mech. Engrg. 327 (2017) 277–305.
[44] D. Luo, L. Cao, P. Chen, O. Ghattas, J.T. Oden, Optimal design of chemoepitaxial guideposts for the directed self-assembly of block copolymer systems using an inexact newton algorithm, J. Comput. Phys. 485 (2023) 112101.
[45] P. Chen, U. Villa, O. Ghattas, Taylor approximation and variance reduction for pde-constrained optimal control under uncertainty, J. Comput. Phys. 385 (2019) 163–186.
[46] A. Alexanderian, N. Petra, G. Stadler, O. Ghattas, Mean–variance risk-averse optimal control of systems governed by pdes with random parameter fields using quadratic approximations, SIAM/ASA J. Uncertain. Quantif. 5 (2017) 1166–1192.
[47] M.G. Kapteyn, D.J. Knezevic, D. Huynh, M. Tran, K.E. Willcox, Data-driven physics-based digital twins via a library of component-based reduced-order models, Internat. J. Numer. Methods Engrg. 123 (2022) 2986–3003.
[48] C.R. Farrar, K. Worden, An introduction to structural health monitoring, Phil. Trans. R. Soc. A 365 (2007) 303–315.
[49] J.T. Oden, S. Prudhomme, Estimation of modeling error in computational mechanics, J. Comput. Phys. 182 (2002) 496–515.
[50] N.C. Nguyen, J. Peraire, An efficient reduced-order modeling approach for non-linear parametrized partial differential equations, Internat. J. Numer. Methods Engrg. 76 (2008) 27–55.
[51] E. Qian, I.-G. Farcas, K. Willcox, Reduced operator inference for nonlinear partial differential equations, SIAM J. Sci. Comput. 44 (2022) A1934–A1959.
[52] R. Geelen, S. Wright, K. Willcox, Operator inference for non-intrusive model reduction with quadratic manifolds, Comput. Methods Appl. Mech. Engrg. 403 (2023) 115717.
[53] Y.M. Marzouk, H.N. Najm, Dimensionality reduction and polynomial chaos acceleration of Bayesian inference in inverse problems, J. Comput. Phys. 228 (2009) 1862–1902.
[54] X. Huan, Y.M. Marzouk, Simulation-based optimal bayesian experimental design for nonlinear systems, J. Comput. Phys. 232 (2013) 288–317.
[55] D. Luo, T. O'Leary-Roseberry, P. Chen, O. Ghattas, Efficient pde-constrained optimization under high-dimensional uncertainty using derivative-informed neural operators, 2023, arXiv preprint arXiv:2305.20053.
[56] K. Wu, T. O'Leary-Roseberry, P. Chen, O. Ghattas, Large-scale bayesian optimal experimental design with derivative-informed projected neural network, J. Sci. Comput. 95 (2023) 30.
[57] T. Zohdi, A digital-twin and machine-learning framework for precise heat and energy management of data-centers, Comput. Mech. 69 (2022) 1501–1516.
[58] X. Du, J.R. Martins, T. O'Leary-Roseberry, A. Chaudhuri, O. Ghattas, K.E. Willcox, Learning optimal aerodynamic designs through multi-fidelity reduced-dimensional neural networks, in: AIAA SCITECH 2023 Forum, p. 0334.

[59] S. Goswami, C. Anitescu, S. Chakraborty, T. Rabczuk, Transfer learning enhanced physics informed neural network for phase-field modeling of fracture, Theor. Appl. Fract. Mech. 106 (2020) 102447.
[60] F. Aldakheel, E.S. Elsayed, T.I. Zohdi, P. Wriggers, Efficient multiscale modeling of heterogeneous materials using deep neural networks, Comput. Mech. (2023) 1–17.
[61] G. Wen, Z. Li, K. Azizzadenesheli, A. Anandkumar, S.M. Benson, U-fno—an enhanced fourier neural operator-based deep-learning model for multiphase flow, Adv. Water Resour. 163 (2022) 104180.
[62] L. Lu, P. Jin, G. Pang, Z. Zhang, G.E. Karniadakis, Learning nonlinear operators via deeponet based on the universal approximation theorem of operators, Nat. Mach. Intell. 3 (2021) 218–229.
[63] K. Bhattacharya, B. Hosseini, N.B. Kovachki, A.M. Stuart, Model reduction and neural networks for parametric PDE, SMAI J. Comput. Math. 7 (2021).
[64] S. Fresca, A. Manzoni, POD-DL-ROM: Enhancing deep learning–based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, Comput. Methods Appl. Mech. Engrg. 388 (2022) 114181.
[65] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: L earning maps between function spaces, 2021, arXiv preprint arXiv:2108.08481.
[66] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, in: International Conference on Learning Representations, 2021.
[67] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Multipole graph neural operator for parametric partial differential equations, in: Neural Information Processing Systems, 2020.
[68] L. Lu, P. Jin, G. Pang, G.E. Karniadakis, DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, Nat. Mach. Intell. (2021).
[69] T. O'Leary-Roseberry, U. Villa, P. Chen, O. Ghattas, Derivative-informed projected neural networks for high-dimensional parametric maps governed by PDEs, Comput. Methods Appl. Mech. Engrg. 388 (2022) 114199.
[70] T. O'Leary-Roseberry, X. Du, A. Chaudhuri, J.R. Martins, K. Willcox, O. Ghattas, Learning high-dimensional parametric maps via reduced basis adaptive residual networks, Comput. Methods Appl. Mech. Engrg. 402 (2022) 115730.
[71] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[72] S. Wang, H. Wang, P. Perdikaris, Learning the solution operator of parametric partial differential equations with physics-informed DeepONets, Sci. Adv. 7 (2021) eabi8605.
[73] J. Yu, L. Lu, X. Meng, G.E. Karniadakis, Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems, Comput. Methods Appl. Mech. Engrg. 393 (2022) 114823.
[74] J. Hesthaven, S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, J. Comput. Phys. 363 (2018) 55–78.
[75] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, A. Anandkumar, Physics-informed neural operator for learning partial differential equations, 2021, arXiv preprint arXiv:2111.03794.
[76] M. De Hoop, D.Z. Huang, E. Qian, A.M. Stuart, The cost-accuracy trade-off in operator learning with neural networks, 2022, arXiv preprint arXiv:2203.13181.
[77] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Graph kernel network for partial differential equations, 2020, arXiv preprint arXiv:2003.03485.
[78] T. Tripura, S. Chakraborty, Wavelet neural operator: A neural operator for parametric partial differential equations, 2022, arXiv preprint arXiv:2205.02191.
[79] T. Chen, H. Chen, Approximations of continuous functionals by neural networks with application to dynamic systems, IEEE Trans. Neural Netw. 4 (1993) 910–918.
[80] T. Chen, H. Chen, Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems, IEEE Trans. Neural Netw. 6 (1995) 911–917.
[81] J.T. Oden, S. Prudhomme, Goal-oriented error estimation and adaptivity for the finite element method, Comput. Math. Appl. 41 (2001) 735–756.
[82] K.G. van der Zee, J.T. Oden, S. Prudhomme, A. Hawkins-Daarud, Goal-oriented error estimation for Cahn–Hilliard models of binary phase transition, Numer. Methods Partial Differential Equations 27 (2011) 160–196.
[83] S. Prudhomme, J.T. Oden, On goal-oriented error estimation for elliptic problems: Application to the control of pointwise errors, Comput. Methods Appl. Mech. Engrg. 176 (1999) 313–331.
[84] S. Prudhomme, J.T. Oden, Computable error estimators and adaptive techniques for fluid flow problems, in: Error estimation and adaptive discretization methods in computational fluid dynamics, Springer, 2003, pp. 207–268.
[85] R. Rannacher, F.-T. Suttmeier, A feed-back approach to error control in finite element methods: Application to linear elasticity, Comput. Mech. 19 (1997) 434–446.
[86] M.B. Giles, E. Süli, Adjoint methods for pdes: A posteriori error analysis and postprocessing by duality, Acta Numer. 11 (2002) 145–236.
[87] T. Hytönen, J. Van Neerven, M. Veraar, L. Weis, Analysis in Banach spaces. Vol. 12, Springer, 2016.
[88] S.-N. Chow, J.K. Hale, Methods of bifurcation theory. Vol. 251, Springer Science & Business Media, 2012.
[89] J.M. Ortega, The Newton-Kantorovich theorem, Amer. Math. Monthly 75 (1968) 658–660.
[90] P.G. Ciarlet, C. Mardare, On the Newton–Kantorovich theorem, Anal. Appl. 10 (2012) 249–269.
[91] M. Badiale, E. Serra, Semilinear Elliptic Equations for Beginners: Existence Results via the Variational Approach, Springer Science & Business Media, 2010.
[92] D. Bhattacharya, R.P. Lipton, Simulating grain shape effects and damage in granular media using peridem, SIAM J. Sci. Comput. 45 (2023) B1–B26.
[93] I. Jolliffe, Principal component analysis, in: Springer Series in Statistics, Springer, 2002.
[94] J.S. Chipman, Proofs and proofs of the Eckart–Young theorem, in: Stochastic processes and functional analysis, CRC Press, 2020, pp. 71–83.
[95] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.
[96] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Vié, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015, Software available from tensorflow.org.
[97] M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M.E. Rognes, G.N. Wells, The fenics project version 1.5, Arch. Num. Softw. 3 (2015).
[98] M.S. Alnæs, A. Logg, K.B. Ølgaard, M.E. Rognes, G.N. Wells, Unified form language: A domain-specific language for weak formulations of partial differential equations, ACM Trans. Math. Softw. 40 (2014) 1–37.
[99] U. Villa, N. Petra, O. Ghattas, Hippylib: An extensible software framework for large-scale inverse problems, J. Open Source Softw. 3 (2018).
[100] C. Geuzaine, J.-F. Remacle, Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities, Internat. J. Numer. Methods Engrg. 79 (2009) 1309–1331.
[101] N. Schlömer, Meshio: Tools for mesh files, 2022, If you use this software, please cite it as below.
[102] J. Bleyer, Numerical tours of computational mechanics with FEniCS, 2018.
[103] T. Kumar, K. Suresh, Direct lagrange multiplier updates in topology optimization revisited, Struct. Multidiscip. Optim. 63 (2021) 1563–1578.