Corrector Operator to Enhance Accuracy and Reliability of Neural Operator Surrogates of Nonlinear Variational Boundary-Value Problems

Prashant K. Jha¹*, J. Tinsley Oden¹²

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 is proposed based on the linear variational problem that gives the correction to the prediction furnished by neural operators. The operator associated with the corrector problem is referred to as the corrector operator. Numerical results involving a nonlinear 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 proposed scheme. Further, topology optimization involving a nonlinear 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 following the proposed method.

Keywords: neural operators, operator learning, singular-value decomposition, variational formulation, surrogate modeling, topology optimization, finite element method

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) \( R(m,u) = 0 \), where \( m \) is a parameter field, \( u \) a solution of PDE, and \( R \) a residual operator. Assuming that for a given \( m \), there is a unique solution \( u = u(m) \), an operator \( F \) – referred to as solution operator – can be defined such that given \( m \), \( F(m) \) satisfies \( R(m,F(m)) = 0 \). For nonlinear and computationally expensive PDEs, applications in which the solution \( u = F(m) \) is sought for large samples of parameter \( m \) becomes challenging, e.g., Bayesian inference, optimization, and control under uncertainty. A neural operator \( F_{NN} \) that maps a parameter \( m \) to an approximation \( \tilde{u} = F_{NN}(m) \) of a solution \( u = u(m) \) is increasingly 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 that by utilizing the underlying structure of the solution operator, an operator \( F^C = F^C(\cdot,\cdot) \) can be constructed – referred to as the 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 Figure 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. Crucial Role of Parametric 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 [1–15] and biophysical systems [16–22].

The parametric PDE-based model is a critical component in several computationally-intensive downstream problems such as parameter estimation under uncertainty [23–30], topology and design optimization and optimization under uncertainty [31–39], model selection [40–42], control under uncertainty [43, 44], digital twins [45], and structural health monitoring [46]. 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. For PDEs with high computation costs, their application in problems such as real-time control of industrial processes and robotic locomotion, calibration of climate models in a time-bound manner, and real-time control of adaptive materials is hindered. For computationally expensive PDEs, several approximation techniques are available:

(i) low-fidelity approximations of the model [47, 48];

(ii) reduced-order modeling [36, 49–51];

(iii) regression techniques (e.g., polynomial chaos) to represent the quantities of interest as a function of the parameter [52, 53]; and

(iv) neural operators – the main focus of this work – that approximate the solution operator [54–58, 58–74, 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 [61], design of materials and structures [57], digital twin [56], inverse problem [71, 73], multiphase flow [60], multiscale modeling [59], optimal experimental design [55], PDE-constrained optimization [54], and phase-field modeling [58].

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 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. Proposed 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 [47, 48, 81] can provide a way forward. Following [47, 48, 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 $\delta_\tilde{u}\mathcal{R}(m, \tilde{u})(\tilde{e}) = -\mathcal{R}(m, \tilde{u})$ for approximate error $\tilde{e}$; $\delta_\tilde{u}\mathcal{R}(m, \tilde{u})(\tilde{e})$ being the variational derivative of $\mathcal{R}(m, \tilde{u})$ in the direction $\tilde{e}$. If $\mathcal{Q}(\cdot)$ is the Quantity of Interests (QoIs) functional, then it is shown in [47] that goal-oriented error, $\mathcal{Q}(u) - \mathcal{Q}(\tilde{u})$, can be approximated by $\delta_\tilde{u}\mathcal{Q}(\tilde{u})(\tilde{e})$. There are different versions of estimates available for goal-oriented error, for example, $\mathcal{Q}(u) - \mathcal{Q}(\tilde{u}) \approx \tilde{g}, \mathcal{R}(m, \tilde{u})$, $\tilde{p}$ being the approximation of the solution of the dual problem associated with the variational form $\mathcal{R}$ and the QoI functional $\mathcal{Q}$; see [47, 48, 81–86]

Once the error is estimated, a correction $u^{C} = \tilde{e} + \tilde{u}$ can be easily computed; 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 $-\mathcal{F}^{C} : \mathcal{M} \times \mathcal{U} \rightarrow \mathcal{U}$ 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.

Motivated by the above observations, this work proposes a corrector framework based on the corrector operator. It is essential to highlight here that the 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 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 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” (upto 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.

1.4. 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 \( f : \Omega \subset \mathbb{R}^d \rightarrow \mathbb{R}^{d_u} \) is denoted by \( L^2(\Omega; \mathbb{R}^{d_u}) \); space \( H^s(\Omega; \mathbb{R}^{d_u}) \) for functions in \( L^2(\Omega; \mathbb{R}^{d_u}) \) with generalized derivatives up to order \( s \) in \( L^2(\Omega; \mathbb{R}^{d_u}) \) is denoted by \( L^2(\Omega; \mathbb{R}^{d_u}) \). \( \mathcal{L}(\mathcal{U}; \mathcal{V}) \) denotes the space of continuous linear maps from \( \mathcal{U} \) to \( \mathcal{V} \) and \( \mathcal{C}^1(\mathcal{U}; \mathcal{V}) \) space of continuous and differentiable maps from \( \mathcal{U} \subset \mathcal{U} \) to \( \mathcal{V} \). \( \mathcal{M} \) and \( \mathcal{U} \) denote the generic Banach spaces of functions; for \( u \in \mathcal{U} \), \( ||u|| \) denotes the norm of function \( u \in \mathcal{U} \). The dual of \( \mathcal{U} \) is the space of all linear continuous functionals on \( \mathcal{U} \), \( L : \mathcal{U} \rightarrow \mathbb{R} \), and is denoted by \( \mathcal{U}^* \) and duality pairing on \( \mathcal{U} \times \mathcal{U}^* \) is denoted \( \langle a, b \rangle_\mathcal{U} \) where \( a \in \mathcal{U} \) and \( b \in \mathcal{U}^* \). Given two Banach spaces \( \mathcal{M} \) and \( \mathcal{U} \), and a probability measure \( \nu_M \) on \( \mathcal{M} \), the Bochner space of operators \( \mathcal{F} : \mathcal{M} \rightarrow \mathcal{U} \) is denoted by \( L^p(\mathcal{M}, \nu_M; \mathcal{U}) \), for \( p \in [1, \infty] \) and the norm is given by, see [Section 1.2, [87]],

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

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

\[
\mathbb{E}^{m\sim \nu_M} [||\mathcal{F}(m)||_{\mathcal{U}}] = \int_{\mathcal{M}} ||\mathcal{F}(m)||_{\mathcal{U}} d\nu_M(m).
\]

1.5. 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 proposed. 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, and Appendix B, Appendix C include supplementary materials.

2. Variational Boundary-Value Problems

Consider a class of parameterized variational boundary-value problems:

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

\[
b(m, u; v) = l(v), \quad \forall v \in \mathcal{U},
\]

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 \( b : \mathcal{M} \times \mathcal{U} \times \mathcal{U} \rightarrow \mathbb{R} \) 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

$$\langle v, \mathcal{R}(m, \bar{u}) \rangle := b(m, \bar{u}; v) - l(v), \quad \forall v \in \mathcal{U},$$

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

Given $m \in \mathcal{M}$, find $u \in \mathcal{U}$ such that $\mathcal{R}(m, u) = 0$ $\in \mathcal{U}^*$.

(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,

$$\delta_u \mathcal{R}(m, u)(p) := \lim_{\epsilon \to 0} \frac{1}{\epsilon} [\mathcal{R}(m, u + \epsilon p) - \mathcal{R}(m, u)],$$

$$\delta_u^2 \mathcal{R}(m, u)(p, q) := \lim_{\epsilon \to 0} \frac{1}{\epsilon} [\delta_u \mathcal{R}(m, u + \epsilon q)(p) - \delta_u \mathcal{R}(m, u)(p)],$$

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

The corrector operator studied in this work is motivated by the recent work [47] 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 [47]. Following this discussion, the corrector approach is presented.

2.1. Goal-Oriented A-Posteriori Error Estimation

Consider a Quantity of Interest (QoI) $Q(u) \in \mathbb{R}$ 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 $\tilde{u} \approx u$, $Q(u)$ can be approximated by $Q(\tilde{u})$ with some error:

$$Q(u) = Q(\tilde{u}) + Q(u) - Q(\tilde{u}).$$

The error $Q(u) - Q(\tilde{u})$ 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 $Q$; see [47, 48, 81–86].

Suppose $\tilde{e} = u - \tilde{u}$ is the error in the forward solution; then it is argued in [Section 2.1, 47] that an estimate $\tilde{e}^C$ of $\tilde{e}$ can be computed by solving the following linear variational problem:

$$\delta_u \mathcal{R}(m, \tilde{u})(\tilde{e}^C) = -\mathcal{R}(m, \tilde{u}) \quad \in \mathcal{U}^*$$

(7)

and, using $\tilde{e}^C$, the following estimate of the goal-oriented error can be constructed,

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

where $r$ collects the remainder terms [47]. The equation for $\tilde{e}^C$ is based on the Taylor series expansion of the residual operator, as follows:

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

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

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

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 $O(||\tilde{e}||_{\mathcal{U}}^2)$. 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) as an equation for the estimate $\tilde{e}^C$ of the error $\tilde{e}$. 

5
2.2. Corrector Operator Based on Residuals

Suppose the variational problem (5) is expensive to solve (i.e., evaluation of the solution operator \( F(m) \)), and, therefore, the operator \( F \) is approximated using an operator \( \tilde{F} : \mathcal{M} \to \mathcal{U} \) 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, \( \tilde{e} = F(m) - \tilde{F}(m) = u - \tilde{u} \), and its norm, \( ||\tilde{e}||_U \). 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:

Given \( m \in \mathcal{M} \) and \( u \in \mathcal{U} \), find \( u^C \) such that

\[
u^C = \tilde{u} + e^C = \tilde{u} - \delta_u \mathcal{R}(m, \tilde{u})^{-1} \mathcal{R}(m, \tilde{u}), \tag{8}\]

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 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 \( u = F(m) \), \( \mathcal{F}^C \) produces another approximate solution \( u^C = \mathcal{F}^C(m, \tilde{u}) \) that has smaller error as compared to \( \tilde{u} \), i.e.,

\[
||u - u^C||_U \leq ||u - \tilde{u}||_U . \tag{10}\]

The following theorem provides a bound on correction error 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 \( u = F(m) \) be the solution of \( \mathcal{R}(m, u) = 0 \). 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:

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

where \( \mathcal{L}(U, V) \) is the space of all continuous linear operators from \( U \) to \( V \) and \( ||f||_{\mathcal{L}(U, V)} = \sup_{||v||_V = 1} ||f(v)||_V \) 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 \( ||\tilde{e}||_U \) is sufficiently small so that

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

then it holds \( ||e^C||_U < ||\tilde{e}||_U \).

- **Quadratic error reduction.** More strictly, if \( \mathcal{R} \) is such that, for given \((m, \tilde{u})\) and \( u = F(m) \),

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

then \( ||e^C||_U < ||\tilde{e}||_U^2 \). Two orders of accuracy may thus be gained when \( ||\tilde{e}||_U < 1 \).
Connection to the Newton Iteration. The corrector operator is directly related to Newton iteration for solving \( R(m, u) = 0 \). Let \( u_0 \in \mathcal{U} \) be the initial guess, then the Newton step to solve \( R(m, u) = 0 \), for a fixed \( m \in \mathcal{M} \), is given by
\[
    u_k = u_{k-1} - \delta_u R(m, u_{k-1})^{-1} R(m, u_{k-1}) = F^C(m, u_{k-1}), \quad \forall k \geq 1.
\]
Thus, the solution \( u = F(m) \) is a fixed-point of \( F^C(m, \cdot) \); i.e., if \( u = F^C(m, u) \) then \( u = F(m) \) or, equivalently, \( R(m, u) = 0 \). Since \( F^C \) 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. In particular, in ideal conditions, Newton 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 \([88, 89]\) is produced below.

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

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

Then, \( \delta_u R(m, \hat{u}) \in \mathcal{L}(\mathcal{U}; \mathcal{U}^*) \) is bijective and \( \delta_u R(m, \hat{u})^{-1} \in \mathcal{L}(\mathcal{U}^*; \mathcal{U}) \) at each \( \hat{u} \in \mathcal{B}(u_0; r) \). The sequence \( (u_k)_{k=0}^\infty \) defined by
\[
    u_k = u_{k-1} - \delta_u R(m, u_{k-1})^{-1} R(m, u_{k-1}) = F^C(m, u_{k-1}), \quad \forall k \geq 1,
\]
is such that \( u_k \in \mathcal{B}(u_0; r) \) for all \( k \geq 0 \), and \( u_k \to u \), where \( u = \overline{\mathcal{B}(u_0; r)} \) is a zero of \( R(m, \cdot) \), i.e., \( R(m, u) = 0 \). Further, for each \( k \geq 0 \),
\[
    \|u - u_k\|_\mathcal{U} \leq \frac{r}{2^k},
\]
and the point \( u \in \overline{\mathcal{B}(u_0; r)} \) is the only zero of \( R(m, \cdot) \) in \( \overline{\mathcal{B}(u_0; r)} \).

For proof, see [Theorem 5, \([89]\)].

2.3. Example of a Nonlinear Diffusion Equation

To put the notations and ideas discussed so far into a context, a forward problem involving a nonlinear 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, \( u = u(x) \), \( x \in \Omega \), is the temperature field with \( u \in \mathcal{U} := H_0^1(\Omega) = \{v \in H^1(\Omega) : v = 0 \text{ on } \partial \Omega\} \) governed by the diffusion-reaction model,
\[
    - \nabla \cdot (\kappa_0 m(x) \nabla u(x)) + \alpha u(x)^3 = f(x), \quad x \in \Omega; \\
    u(x) = 0, \quad x \in \partial \Omega,
\]
where \( \kappa_0, \alpha > 0 \) are fixed constants, \( m \in \mathcal{M} := \{v \in L^2(\Omega) \cap L^\infty(\Omega) : v \geq m_0 \} \) is the diffusivity field, and \( f \in L^2(\Omega) \) external heat source that is fixed and given. The associated variational problem reads:
\[
    \text{Given } m \in \mathcal{M}, \text{ find } u \in \mathcal{U} \text{ such that } \int_{\Omega} \left\{ \kappa_0 m \nabla u \cdot \nabla v + \alpha u^3 v \right\} \, dx = \int_{\Omega} f v \, dx, \quad \forall v \in \mathcal{U}.
\]

The corresponding residual functional \( R : \mathcal{M} \times \mathcal{U} \to \mathcal{U}^* \) is defined through action on \( v \in \mathcal{U} \), for \( (m, u) \in \mathcal{M} \times \mathcal{U} \), as follows
\[
    \langle v, R(m, u) \rangle := b(m, u; v) - l(v) = \int_{\Omega} \left\{ \kappa_0 m \nabla u \cdot \nabla v + \alpha u^3 v \right\} \, dx - \int_{\Omega} f v \, dx.
\]
The first and second derivatives, in this case, have the form:

\[
\langle v, \delta_1 \mathcal{R}(m, u)(p) \rangle_{\mathcal{U}} = \int_{\Omega} \{ \kappa_0 m \nabla p \cdot \nabla v + 3\alpha u^2 pv \} \, dx ,
\]

\[
\langle v, \delta_2 \mathcal{R}(m, u)(p, q) \rangle_{\mathcal{U}} = \int_{\Omega} \{ 6\alpha upqv \} \, dx .
\] (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 [90] and the underlying arguments.

2.3.1. Constants in the Corrector Operator Bound for the Nonlinear Diffusion Equation

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

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

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

\[
\hat{C}_{\delta R} := \frac{\kappa_0 M_{Rm}}{2} \min \{ 1, C_P^2 \} , \quad \bar{C}_{\delta R}(\tilde{u}) := \kappa_0 ||m||_{L^\infty(\Omega)} + 3\alpha C_S^2 ||\tilde{u}||_{\mathcal{U}}^2 ,
\]

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

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

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

\[
||\delta_u \mathcal{R}(m, \tilde{u})^{-1}||_{\mathcal{L}(\mathcal{U}; \mathcal{U}^*)} \leq \frac{1}{\bar{C}_{\delta R}} .
\]

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

\[
||\delta_u^2 \mathcal{R}(m, \tilde{u})(p, q)||_{\mathcal{U}^*} \leq \tilde{C}_{\delta^2 R}(\tilde{u}) ||p||_{\mathcal{U}} ||q||_{\mathcal{U}} , \quad \tilde{C}_{\delta^2 R}(\tilde{u}) := 6\alpha C_S^2 ||\tilde{u}||_{\mathcal{U}} ,
\]

and, therefore,

\[
||\delta_u^2 \mathcal{R}(m, \tilde{u})||_{\mathcal{L}(\mathcal{U}^*; \mathcal{U}^*; \mathcal{U})} \leq \tilde{C}_{\delta^2 R}(\tilde{u}) .
\]

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 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 Theorem 1 and Theorem 3, recalling that \(e^C = u - u^C\) and \(\tilde{e} = u - \tilde{u}\), gives

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

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

\[
\leq \frac{1}{2\bar{C}_{\delta R}} \left[ \sup_{s \in [0,1]} \tilde{C}_{\delta^2 R}(\tilde{u} + s\tilde{e}) \right] ||\tilde{e}||_{\mathcal{U}}^2
\]

\[
\leq \frac{3C_S^4}{\bar{C}_{\delta R}} ||\tilde{u}||_{\mathcal{U}} + ||\tilde{e}||_{\mathcal{U}} ||\tilde{e}||_{\mathcal{U}}^2 .
\]
i.e., under conditions in Theorem 3, a constant $C \in \mathbb{R}^+$ exists such that 
\[
||e^C||_U \leq C||\tilde{u}||_U ||e||_U^2 + C ||e||_U^3.
\]

3. Neural Operators and Corrector Scheme

Consider a case when a map $F : \mathcal{M} \rightarrow \mathcal{U}$ is expensive to compute, requiring 
\[
\mathcal{M}_h = \text{span}\{\phi_i\} \subset \mathcal{M}.
\] 
An element $m \in \mathcal{M}_h$ is expressed as the linear combinations of interpolation 
functions, i.e., $m = \sum m_i \phi_i$, where $(m_1, m_2, ..., m_q) \in \mathbb{R}^{q_m}$ and $\mathbb{R}^{q_m}$ being the coefficient space of $\mathcal{M}_h$. 
Similarly, if $\{\psi_i\}_{i=1}^{q_u}$ are the interpolation functions such that $\mathcal{U}_h = \text{span}\{\psi_i\}$ and $q_u = \dim(\mathcal{U}_h)$, then $u \in \mathcal{U}_h$ is has a representation $u = \sum_{i=1}^{q_u} u_i \psi_i$ with $(u_1, u_2, ..., u_{q_u}) \in \mathbb{R}^{q_u}$. In what follows, $m \in \mathcal{M}_h$ and $u \in \mathcal{U}_h$ will be interchanged with $m = (m_1, m_2, ..., m_{q_m}) \in \mathbb{R}^{q_m}$ and $u = (u_1, u_2, ..., u_{q_u}) \in \mathbb{R}^{q_u}$ 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 $F_h : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{n}$ (technically, $F_h : \mathcal{M}_h \rightarrow \mathcal{U}_h$) of $F : \mathcal{M} \rightarrow \mathcal{U}$. Next, consider a family of neural operators $F_{h,NN}(\cdot; \theta) : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{n}$ parameterized by 
\[
\theta \in \Theta_{NN} \subset \mathbb{R}^{d_{NN}},
\]
where $m = (m_1, ..., m_{q_m}) \in \mathbb{R}^{q_m}$ is the input to the neural operator and $u = (u_1, u_2, ..., u_{q_u}) \in \mathbb{R}^{q_u}$ is the output. Broadly, $\theta$ is chosen such that the error $F_{h,NN}(\cdot; \theta) - F_h(\cdot)$ is minimized in some sense. 

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

\[
\theta_{h,NN} = \arg \min_{\theta \in \Theta_{NN}} J_h(\theta) := \mathbb{E}^{m \sim \nu_h} [||F_h(m) - F_{h,NN}(m; \theta)||_{\mathcal{U}_h}] . \tag{14}
\]

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

\[
\tilde{\theta}_{h,NN} = \arg \min_{\theta \in \Theta_{NN}} \tilde{J}_h(\theta) := \frac{1}{N} \sum_{i=1}^{N} ||F_h(m^i) - F_{h,NN}(m^i; \theta)||_{\mathcal{U}_h} . \tag{15}
\]

Here, $\{(m^i, u^i = F_h(m^i))\}_{i=1}^{N}$ 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, $\mathcal{F}_{h,NN}(\cdot)$, can be defined according to:

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

3.1. Correcting Neural Operators using Corrector Operator

Suppose the optimization problem (15) is solved to obtain the neural operator $\mathcal{F}_{h,NN} : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{q_u}$. Since the neural operator is trained to minimize the average error, see (15), the error $||F_h(m) - \mathcal{F}_{h,NN}(m)||_{\mathcal{U}_h}$ 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 $\mathcal{F}_{h,NN}(m)$ can pose a serious challenge. To improve the accuracy of $\mathcal{F}_{h,NN}(m)$ 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 can not be enhanced beyond a certain limit, and often fine-tuning hyperparameters is not straightforward and may give only marginal gains [61, 76].

To enhance the accuracy and reliability, the use of the corrector operator developed in Section 2.2 on the neural operator prediction is proposed. The proposed approach does not seek to modify the existing neural operator architecture and has the potential to be used with neural operators with limited accuracy trained with sparse data.

Given \( m \in \mathcal{M}_b \) and corresponding neural operator prediction \( u_{NN} = \hat{F}_{NN}(m) \in \mathcal{U}_b \), the correction \( \hat{u}_{NN} \) is computed:

\[
\hat{u}_{NN} = \mathcal{F}^C(m, u_{NN}) = u_{NN} - (\delta_u \mathcal{R}(m, u_{NN}))^{-1} \mathcal{R}(m, u_{NN}).
\]

The above entails solving the following linear variational problem:

Given \( m \in \mathcal{M}_b, u_{NN} = \hat{F}_{h,NN}(m) \), find \( \hat{u}_{NN} \in \mathcal{U}_h \) such that

\[
\langle v, \delta_u \mathcal{R}(m, u_{NN})(\hat{u}_{NN} - u_{NN}) \rangle = -\langle v, \mathcal{R}(m, u_{NN}) \rangle, \quad \forall v \in \mathcal{U}_h.
\] (17)

If \( u_{NN} \) is sufficiently close to the true solution \( u = \mathcal{F}_b(m) \), as will be the case for a trained neural operator, the error \( \|u - u_{NN}\|_{\mathcal{U}_h} \) 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 Theorem 1 and Theorem 2. In a scenario when \( \mathcal{F}(m) \) 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 \( \hat{u}_{NN} \) is hoped to keep the error small. This is demonstrated numerically for the topological optimization of input parameter \( m \) in a nonlinear diffusion equation. For the example of topological optimization, since it is difficult to construct a priori a probability distribution \( \nu_b \) that includes samples representative of \( m_{\text{optimal}} \), \( m_{\text{optimal}} \) 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 operators \( \mathcal{F}_{h,NN} : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{d_u} \) described so far as has two key limitations. First, they are not scalable; for fine discretizations of variational problems, \( q_m \) and \( q_u \) could be large which results in a neural network with a large computational complexity (complexity is defined as the number of training parameters, i.e., dim(\( \Theta_{NN} \))). The second problem is mesh dependence, as \( \mathcal{F}_{h,NN} \) is coupled to the underlying mesh used in \( \mathcal{M}_b \) and \( \mathcal{U}_b \).

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 consider low-dimensional subspaces \( \mathbb{R}^{q_m} \) and \( \mathbb{R}^{r_u} \) of \( \mathbb{R}^{q_m} \) and \( \mathbb{R}^{d_u} \), respectively, and construct a neural operator between the low dimensional subspaces; see Figure 2. To make this more precise, suppose \( \Pi_{r_m}^m : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{r_m} \), \( q_m = \text{dim}(\mathcal{M}_b) \) and \( r_m << q_m \) is the dimension of the reduced subspace, and similarly, \( \Pi_{r_u}^u : \mathbb{R}^{d_u} \rightarrow \mathbb{R}^{r_u} \), where \( q_u = \text{dim}(\mathcal{U}_b) \) and \( r_u << q_u \). Next, a parameterized neural operator \( \mathcal{F}_{r,h,NN} : \mathbb{R}^{r_m} \times \Theta_{NN} \rightarrow \mathbb{R}^{r_u} \) is considered with the corresponding optimization problem defined as:

\[
\hat{\theta}_{r,h,NN} = \arg \min_{\theta \in \Theta_{NN}} \hat{J}_{r,h}(\theta) := \frac{1}{N} \sum_{i=1}^N \| \mathcal{F}_h(m_i) - (\Pi_{r_u}^u)^T (\mathcal{F}_{r,h,NN}(\Pi_{r_m}^m(m_i); \theta)) \|_{\mathcal{U}_h}. \tag{18}
\]

The trained neural operator is then defined as

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

In the term, \( (\Pi_{r_u}^u)^T (\mathcal{F}_{r,h,NN}(\Pi_{r_m}^m(m); \theta)) \), 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 \( u_r = \mathcal{F}_{r,NN}(\Pi_{r_m}^m(m); \theta) \in \mathbb{R}^{r_u} \); and, thirdly, \( u_r \) is projected into the full space \( \mathbb{R}^{d_u} \) using \( (\Pi_{r_u}^u)^T \).
Figure 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, has a lower computational complexity and is independent of the mesh \cite{65, 91}. There are two additional steps in this approach: first, the given input is compressed using the projector $\Pi_{r_m}^{\mathcal{M}}$, and, second, the output of the neural operator is decompressed using the transpose of the projector $\Pi_{r_m}^{\mathcal{U}}$.

3.2.1. Singular-Value Decomposition (SVD) for Projectors

In this work, SVD is used to construct the projectors $\Pi_{r_m}^{\mathcal{M}}$ and $\Pi_{r_m}^{\mathcal{U}}$ for dimensional reduction. For completeness, key aspects of SVD are reviewed in this subsection. 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 $u_i = F_h(m_i) \in \mathbb{R}^{q_u}$. Further, suppose that $\{ m_i \}$ and $\{ u_i \}$ are centered so that mean of $\{ m_i \}$ and $\{ u_i \}$, respectively, are zero. Focusing on the input space $\mathbb{R}^{q_m}$, let $A$ denote an $q_m \times N$ matrix such that:

$$ A = \begin{bmatrix} m_1 & m_2 & \cdots & m_N \end{bmatrix}. $$

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 called the singular values. Here, $r = \min\{q_m, N\}$, and some $\lambda_i$ can be zero. There exist an integer $r_A \leq \min\{q_m, N\}$ such that $\lambda_j = 0$ for all $j > r_A$, and $r_A = \text{rank}(A)$. Focusing on the matrix $U$, it has the following structure:

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

where $w^i \in \mathbb{R}^{q_m}$ are orthonormal vectors, i.e., $w^i \cdot w^j = \delta_{ij}$, $\delta_{ij}$ 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 $r_m \leq \text{rank}(A)$ is the integer of the reduced dimension $\mathbb{R}^{r_m}$ for which a projector $\Pi_{r_m}^{\mathcal{M}} : \mathbb{R}^{q_m} \to \mathbb{R}^{r_m}$ is sought. Given $r_m$, a matrix $U_{r_m}$ is constructed as follows by removing the last $q_m - r_m$ columns of $U$:

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

The matrix $U_{r_m}$ has the following notable properties:
Due to the properties listed above, it makes sense to take $U_{r_m}^T (m)$ projects an element $m \in \mathbb{R}^{q_m}$ into a lower dimensional subspace of $\mathbb{R}^{q_m}$, i.e., $U_{r_m}^T : \mathbb{R}^{q_m} \rightarrow \mathbb{R}^{r_m}$. To see this, consider

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

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

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

$$U (\tilde{m}) = \begin{bmatrix} w^1 \\ w^2 \\ \vdots \\ w^{r_m} \end{bmatrix} \begin{bmatrix} \tilde{m}_1 \\ \tilde{m}_2 \\ \vdots \\ \tilde{m}_{r_m} \end{bmatrix} = \sum_{i=1}^{r_m} \tilde{m}_i w^i \in \mathbb{R}^{q_m}, \quad (24)$$

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 referred to as 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 $m^i$ is defined as the sum of the square of individual reconstruction errors:

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

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

$$U_{r_m} = \arg \min_{V \in \mathbb{R}^{q_m \times r_m}} ||A - V V^T A||_F. \quad (26)$$

For proof, see [Theorem 2, [92]].

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

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

$$A = \begin{bmatrix} u^1 \\ u^2 \\ \vdots \\ u^N \end{bmatrix}. \quad (27)$$
and $A = UDV^T$, 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 $r_u \leq \text{rank}(A) \leq \min\{q_u, N\}$ be the given dimension of the desired reduced space. Then the projector $\Pi_{r_u} : \mathbb{R}^{q_u} \rightarrow \mathbb{R}^{r_u}$ is defined using the matrix $U_{r_u}^T$, where

$$U_{r_u} = \begin{bmatrix} w^1 & w^2 & \ldots & w^{r_u} \\
\end{bmatrix}$$

is the truncation of $U$.

4. Numerical Examples

The example of a nonlinear 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 taking neural operator prediction and model parameter as input is analyzed and it is shown that the corrector operator consistently leads to increased accuracy. The second example considers a slightly 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 are 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 is seen to significantly increased. 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 [93]) layers following [61, 70]. Particularly, the number of residual network blocks is fixed to five.
with rank 20; see Figure 3. Let \( \{(m^i, u^i)\}_{i=1}^{N} \) are training data, \( q_u \) and \( q_m \) are such that \( m^i \subset \mathbb{R}^{m} \) and \( u^i \subset \mathbb{R}^{n} \), respectively, and \( \bar{m} := \frac{1}{N} \sum m^i \) and \( \bar{u} := \frac{1}{N} \sum u^i \). Also, let \( r_m \leq q_m \) and \( r_u \leq q_u \) 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:

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

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 Figure 3. The parameters that will be varied in the numerical examples are dimensions of reduced spaces \((r_m \text{ and } r_u)\) and the number of data \((N)\). The \( N \) samples of data are divided into \([0.1N]\) number of validation data \( N - [0.1N]\) number of training data, and the testing data sample in addition to \( N \) training samples is fixed to \([0.25N]\). The implementation of neural networks is based on hIPPYflow\(^3\) [61, 69, 70] and TensorFlow\(^4\) [94]. To solve the variational problems and sample from a prior \( \nu_b \), FEniCS\(^5\) [95, 96] and hiPPYlib\(^6\) [97] are used.

### 4.2. Accuracy Comparison for a Nonlinear Diffusion Equation

Consider a square domain \( \Omega = (0, 1)^2 \) and denote the bottom edge by \( \Gamma_b = [0, 1] \times 0 \). The equation for the temperature field \( u = u(\mathbf{x}) \) is taken as

\[
\begin{align*}
-\nabla \cdot (e^{m(\mathbf{x})} \nabla u(\mathbf{x})) + u(\mathbf{x})^3 &= f(\mathbf{x}), & \mathbf{x} \in \Omega; \\
u(\mathbf{x}) &= 0, & \mathbf{x} \in \Gamma_b; \\
e^{m(\mathbf{x})} \nabla u(\mathbf{x}) \cdot \mathbf{n}(\mathbf{x}) &= 0, & \mathbf{x} \in \partial\Omega - \Gamma_b;
\end{align*}
\]

(29)

where

\[
f(\mathbf{x}) = e^{-4(1-x_1)^2 \sin(4\pi x_2)^2}, \quad \mathbf{x} = (x_1, x_2) \subset \Omega,
\]

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

\[
m \subset \mathcal{M} := L^2(\Omega) \cap L^\infty(\Omega), \quad u \subset \mathcal{U} := \{ v \in H^1(\Omega) : v(\mathbf{x}) = 0, \mathbf{x} \in \Gamma_b \}.
\]

The variational problem associated to (29) is written as:

Given \( m \subset \mathcal{M} \), find \( u \subset \mathcal{U} \) such that

\[
\langle v, \mathcal{R}(m, u) \rangle := \int_\Omega e^{m(\mathbf{x})} \nabla u(\mathbf{x}) \cdot \nabla v(\mathbf{x}) \, d\mathbf{x} + \int_\Omega u(\mathbf{x})^3 v(\mathbf{x}) \, d\mathbf{x} - \int_\Omega f(\mathbf{x}) v(\mathbf{x}) \, d\mathbf{x} = 0, \quad \forall v \subset \mathcal{U}.
\]

(30)

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

---

\(^3\) [https://github.com/hippylib/hippyflow](https://github.com/hippylib/hippyflow)

\(^4\) [https://www.tensorflow.org/](https://www.tensorflow.org/)

\(^5\) [https://fenicsproject.org/](https://fenicsproject.org/)

\(^6\) [https://github.com/hippylib/hippylib](https://github.com/hippylib)
4.2.1. Data Generation and Neural Operators

A 64 × 64 quadrilateral mesh of Ω is considered, and \( U_h \) and \( M_h \) are finite element function spaces based on the first-order Lagrange interpolation. By replacing \( M \) and \( U \) with \( M_h \) and \( U_h \), 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 \( ν = \mathcal{N}(0,C) \), where \( C : M \times M \rightarrow \mathbb{R} \) is a covariance operator taking the form

\[
C = \begin{cases} \frac{(-\gamma \nabla \cdot \nabla + \delta)^{-d}}{} & \text{in } \Omega, \\ \gamma \mathbf{n} \cdot \nabla + \eta & \text{on } \partial \Omega, \end{cases}
\]

where \( \gamma, \delta, \eta, d \) are hyperparameters and \( \mathbf{n} \) unit outward normal. Covariance parameters are fixed as follows:

\( \gamma = 0.08, \delta = 2, \eta = 1/1.42, d = 2 \). Let \( C_h : M_h \rightarrow M_h \rightarrow \mathbb{R} \) is the covariance operator in the finite dimensional setting and \( \nu_h = \mathcal{N}(0, C_h) \). The set \( \{(m^i, u^i)\}_{i=1}^N \), where \( m^i \sim \nu_h \in M_h \) and \( u^i = F_h(m^i) \) is the solution of the discretized variational problem, is the data for neural operator learning. In Figure 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

\[
(m_u, r_u) \in \{(50, 25), (50, 50), (100, 25), (100, 50)\} \text{ and } N \in \{256, 512, 1024, 2048, 4096\}
\]

are trained. Here, \( r_u \) is kept small compared to \( r_m \) based on the relatively faster decay of output singular values; see Figure 4(b).

4.2.2. Comparing Neural Operator and Corrector Operator Accuracy

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

\[
e_{NN}(m) := \frac{||u(m) - u_{NN}(m)||_2}{||u(m)||_2} \times 100, \quad e_{NN}^C(m) := \frac{||u(m) - u_{NN}^C(m)||_2}{||u(m)||_2} \times 100, \quad (31)
\]
where \(\|a\|_2 = \sqrt{\sum a_i^2}\). 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 \(m \sim \nu_b\) are drawn randomly, and the solutions from the true model, neural operator, and correction of the neural operator are visualized in Figure 5 and Figure 6.

![Figure 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.](image)

4.3. Topology Optimization Involving a Nonlinear 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 diffusion model is considered in this subsection. The domain \(\Omega\) is a square domain with two circular voids: \(\Omega = (0,1)^2 - B(x_{c1}, R_1) - B(x_{c2}, R_2)\), where \(B(x, R) = \{y \in \mathbb{R}^2 : |y - x| < R\}\) denotes the ball of radius \(R\) centered at \(x\). Here, \(x_{c1} = (0.2, 0.8), x_{c2} = (0.7, 0.3)\), \(R_1 = 0.1\), and \(R_2 = 0.2\); see Figure 7. Let \(\partial\Omega = \Gamma_{in} \cup \Gamma_{out}\), \(\Gamma_{in}\) and \(\Gamma_{out}\) being the inner and outer boundaries, respectively. In the inner boundary, \(\Gamma_{in}\), temperature is fixed to zero, while, in the outer boundary, \(\Gamma_{out}\), the heat flux \(g(x) := 0.1\) 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 \(m = m(x)\), find temperature \(u\) such that

\[
-\nabla \cdot (m(x)\nabla u(x)) + u(x)^3 = 0, \\
u(x) = 0, \\
m(x)\nabla u(x) \cdot n(x) = 0.1 =: g(x),
\]

\(x \in \Omega;\) 
\(x \in \Gamma_{in};\) 
\(x \in \Gamma_{out}.
\]
### 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 was computed from the resulting twenty $e_{NN}$ 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.

<table>
<thead>
<tr>
<th>Number</th>
<th>$r_m$</th>
<th>$r_u$</th>
<th>$N$</th>
<th>$e_{NN}$ min</th>
<th>$e_{NN}$ max</th>
<th>$e_{NN}$ mean</th>
<th>$e_{C_{NN}}$ min</th>
<th>$e_{C_{NN}}$ max</th>
<th>$e_{C_{NN}}$ mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50</td>
<td>25</td>
<td>256</td>
<td>2.45728</td>
<td>16.84489</td>
<td>6.06698</td>
<td>0.00024</td>
<td>0.31488</td>
<td>0.04303</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>25</td>
<td>512</td>
<td>1.40506</td>
<td>7.82594</td>
<td>3.53658</td>
<td>0.00016</td>
<td>0.16437</td>
<td>0.03716</td>
</tr>
<tr>
<td>3</td>
<td>50</td>
<td>25</td>
<td>1024</td>
<td>1.65162</td>
<td>6.06503</td>
<td>3.18435</td>
<td>0.00005</td>
<td>0.09237</td>
<td>0.01783</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>25</td>
<td>2048</td>
<td>1.40497</td>
<td>4.91186</td>
<td>2.56645</td>
<td>0.00141</td>
<td>0.20477</td>
<td>0.07224</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>25</td>
<td>4096</td>
<td>1.16622</td>
<td>9.80585</td>
<td>2.43871</td>
<td>0.00001</td>
<td>0.19827</td>
<td>0.03807</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>50</td>
<td>256</td>
<td>3.86182</td>
<td>29.05746</td>
<td>8.91953</td>
<td>0.00008</td>
<td>0.24863</td>
<td>0.07290</td>
</tr>
<tr>
<td>7</td>
<td>50</td>
<td>50</td>
<td>512</td>
<td>2.89625</td>
<td>10.76244</td>
<td>5.54185</td>
<td>0.00014</td>
<td>0.19987</td>
<td>0.05248</td>
</tr>
<tr>
<td>8</td>
<td>50</td>
<td>50</td>
<td>1024</td>
<td>1.92660</td>
<td>8.37186</td>
<td>3.77333</td>
<td>0.00036</td>
<td>0.28288</td>
<td>0.06235</td>
</tr>
<tr>
<td>9</td>
<td>50</td>
<td>50</td>
<td>2048</td>
<td>1.88078</td>
<td>4.93965</td>
<td>3.07723</td>
<td>0.00011</td>
<td>0.18041</td>
<td>0.03552</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>50</td>
<td>4096</td>
<td>1.74788</td>
<td>4.84569</td>
<td>3.19339</td>
<td>0.00012</td>
<td>0.20722</td>
<td>0.06781</td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>25</td>
<td>256</td>
<td>2.76545</td>
<td>11.96616</td>
<td>5.23490</td>
<td>0.00937</td>
<td>0.20000</td>
<td>0.05499</td>
</tr>
<tr>
<td>12</td>
<td>100</td>
<td>25</td>
<td>512</td>
<td>1.74937</td>
<td>5.69031</td>
<td>3.70606</td>
<td>0.00072</td>
<td>0.26213</td>
<td>0.07850</td>
</tr>
<tr>
<td>13</td>
<td>100</td>
<td>25</td>
<td>1024</td>
<td>1.65059</td>
<td>6.97599</td>
<td>3.37611</td>
<td>0.00019</td>
<td>0.23674</td>
<td>0.05536</td>
</tr>
<tr>
<td>14</td>
<td>100</td>
<td>25</td>
<td>2048</td>
<td>1.01940</td>
<td>5.27190</td>
<td>2.60825</td>
<td>0.00012</td>
<td>0.20808</td>
<td>0.06776</td>
</tr>
<tr>
<td>15</td>
<td>100</td>
<td>25</td>
<td>4096</td>
<td>1.22295</td>
<td>4.06625</td>
<td>2.00097</td>
<td>0.00005</td>
<td>0.27715</td>
<td>0.05748</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
<td>50</td>
<td>256</td>
<td>2.30358</td>
<td>23.87869</td>
<td>5.38995</td>
<td>0.00033</td>
<td>0.23157</td>
<td>0.05828</td>
</tr>
<tr>
<td>17</td>
<td>100</td>
<td>50</td>
<td>512</td>
<td>2.26997</td>
<td>12.24735</td>
<td>5.29943</td>
<td>0.00077</td>
<td>0.27386</td>
<td>0.05864</td>
</tr>
<tr>
<td>18</td>
<td>100</td>
<td>50</td>
<td>1024</td>
<td>1.39031</td>
<td>7.22452</td>
<td>3.02298</td>
<td>0.00008</td>
<td>0.25664</td>
<td>0.06455</td>
</tr>
<tr>
<td>19</td>
<td>100</td>
<td>50</td>
<td>2048</td>
<td>1.08747</td>
<td>6.17697</td>
<td>3.15330</td>
<td>0.00012</td>
<td>0.24090</td>
<td>0.06358</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>50</td>
<td>4096</td>
<td>1.00065</td>
<td>4.63677</td>
<td>2.27597</td>
<td>0.00005</td>
<td>0.20564</td>
<td>0.07560</td>
</tr>
</tbody>
</table>
As before, function spaces associated with the parameter and solution are taken as:

\[ m \in M := \mathcal{L}^2(\Omega) \cap \mathcal{L}^\infty(\Omega), \quad u \in U := \{ v \in H^1(\Omega) : v(x) = 0, x \in \Gamma_i \} \]

(33)

The variational problem corresponding to the forward problem reads:

\[
\langle v, R(m, u) \rangle := \int_{\Omega} m(x) \nabla u(x) \cdot \nabla v(x) \, dx + \int_{\Omega} u(x)^3 v(x) \, dx - \int_{\Gamma_{out}} g v(x) \, dS(x) = 0, \quad \forall v \in U.
\]

(34)

**Topological Optimization Problem.** Given a temperature \( u \) satisfying the above equation, the compliance – external working – is defined as

\[
J(m) = \int_{\Gamma_{out}} g u(x) \, dS(x).
\]

(35)

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

\[
\tilde{m} = \arg \min_{m \in \mathcal{M}_{ad}} J(m) := \int_{\Gamma_{out}} g F(m) \, dS(x) \quad \text{such that} \quad \frac{1}{|\Omega|} \int_{\Omega} m(x) \, dx = \eta,
\]

(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

\[
J_{NN}(m) := \int_{\Gamma_{out}} g \mathcal{F}_{NN}(m) dS(x), \quad J^C_{NN}(m) := \int_{\Gamma_{out}} g \mathcal{F}^C(m, \mathcal{F}_{NN}(m)) dS(x).
\]

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 $\Omega$ is triangulated using Gmsh\(^7\) [98] with 20614 triangular elements and 10301 vertices. The mesh is converted into a Fenics-friendly format using Meshio\(^8\) [99]. The parameter and solution spaces are approximated using the first-order Lagrange interpolation on the mesh. 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:

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

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

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

\[
\text{Draw } w^i(x) \sim \nu_h \quad \text{and} \quad m^i(x) = 0, 25e^{w^i(x)}, \quad u^i = \mathcal{F}_h(m^i). \tag{38}
\]

In Figure 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 Figure 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 are compared in Figure 9 and Figure 10.

\(^7\)https://gmsh.info/
\(^8\)https://github.com/nschloe/meshio
Figure 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.

Figure 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>
<thead>
<tr>
<th>Number</th>
<th>( r_m )</th>
<th>( r_u )</th>
<th>( N )</th>
<th>( e_{NN} ) min</th>
<th>( e_{NN} ) max</th>
<th>( e_{NN} ) mean</th>
<th>( e_{NN}^C ) min</th>
<th>( e_{NN}^C ) max</th>
<th>( e_{NN}^C ) mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>50</td>
<td>25</td>
<td>256</td>
<td>7.15830</td>
<td>21.25784</td>
<td>12.47695</td>
<td>0.00153</td>
<td>0.84201</td>
<td>0.13296</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
<td>25</td>
<td>512</td>
<td>4.59103</td>
<td>15.99608</td>
<td>9.09779</td>
<td>0.00907</td>
<td>0.38084</td>
<td>0.06826</td>
</tr>
<tr>
<td>3</td>
<td>50</td>
<td>25</td>
<td>1024</td>
<td>4.56197</td>
<td>10.36562</td>
<td>6.26587</td>
<td>0.00459</td>
<td>0.16858</td>
<td>0.03795</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>25</td>
<td>2048</td>
<td>3.75125</td>
<td>7.91939</td>
<td>5.54681</td>
<td>0.00139</td>
<td>0.23467</td>
<td>0.04583</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>25</td>
<td>4096</td>
<td>2.91057</td>
<td>8.09030</td>
<td>4.94291</td>
<td>0.00178</td>
<td>0.12398</td>
<td>0.02647</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>50</td>
<td>256</td>
<td>9.15900</td>
<td>19.87726</td>
<td>13.10120</td>
<td>0.02228</td>
<td>0.48271</td>
<td>0.14268</td>
</tr>
<tr>
<td>7</td>
<td>50</td>
<td>50</td>
<td>512</td>
<td>5.60731</td>
<td>24.31812</td>
<td>11.78976</td>
<td>0.00593</td>
<td>0.77939</td>
<td>0.15389</td>
</tr>
<tr>
<td>8</td>
<td>50</td>
<td>50</td>
<td>1024</td>
<td>4.50428</td>
<td>10.0015</td>
<td>7.89234</td>
<td>0.00262</td>
<td>1.34911</td>
<td>0.16561</td>
</tr>
<tr>
<td>9</td>
<td>50</td>
<td>50</td>
<td>2048</td>
<td>3.58687</td>
<td>11.7070</td>
<td>6.41672</td>
<td>0.00263</td>
<td>0.29747</td>
<td>0.03527</td>
</tr>
<tr>
<td>10</td>
<td>50</td>
<td>50</td>
<td>4096</td>
<td>3.11247</td>
<td>6.86458</td>
<td>4.22486</td>
<td>0.00139</td>
<td>0.17169</td>
<td>0.02497</td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>25</td>
<td>256</td>
<td>6.66220</td>
<td>22.61565</td>
<td>11.11228</td>
<td>0.00673</td>
<td>1.41224</td>
<td>0.20579</td>
</tr>
<tr>
<td>12</td>
<td>100</td>
<td>25</td>
<td>512</td>
<td>5.56698</td>
<td>21.51665</td>
<td>10.51670</td>
<td>0.00973</td>
<td>0.6513</td>
<td>0.09869</td>
</tr>
<tr>
<td>13</td>
<td>100</td>
<td>25</td>
<td>1024</td>
<td>5.25971</td>
<td>12.44923</td>
<td>7.97554</td>
<td>0.00135</td>
<td>0.59833</td>
<td>0.07555</td>
</tr>
<tr>
<td>14</td>
<td>100</td>
<td>25</td>
<td>2048</td>
<td>4.53138</td>
<td>11.74423</td>
<td>7.21405</td>
<td>0.00342</td>
<td>0.60607</td>
<td>0.11523</td>
</tr>
<tr>
<td>15</td>
<td>100</td>
<td>25</td>
<td>4096</td>
<td>3.24184</td>
<td>6.81036</td>
<td>4.38627</td>
<td>0.00331</td>
<td>0.18123</td>
<td>0.02263</td>
</tr>
<tr>
<td>16</td>
<td>100</td>
<td>50</td>
<td>256</td>
<td>7.72910</td>
<td>18.16715</td>
<td>11.80013</td>
<td>0.00926</td>
<td>0.32357</td>
<td>0.09902</td>
</tr>
<tr>
<td>17</td>
<td>100</td>
<td>50</td>
<td>512</td>
<td>6.11802</td>
<td>12.84410</td>
<td>8.83076</td>
<td>0.00718</td>
<td>0.28067</td>
<td>0.05695</td>
</tr>
<tr>
<td>18</td>
<td>100</td>
<td>50</td>
<td>1024</td>
<td>4.99526</td>
<td>10.96458</td>
<td>7.80839</td>
<td>0.00267</td>
<td>0.25835</td>
<td>0.04400</td>
</tr>
<tr>
<td>19</td>
<td>100</td>
<td>50</td>
<td>2048</td>
<td>3.81907</td>
<td>15.93567</td>
<td>6.45772</td>
<td>0.00198</td>
<td>0.91764</td>
<td>0.06216</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>50</td>
<td>4096</td>
<td>3.41474</td>
<td>8.98157</td>
<td>5.17495</td>
<td>0.00113</td>
<td>0.12705</td>
<td>0.02403</td>
</tr>
</tbody>
</table>

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, 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.
Figure 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.

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

\[
\min_{m \in \mathcal{M}_h, \lambda \in \mathbb{R}} J(m, \lambda, u) := \int_{\Gamma_{\text{out}}} g \, v \, d\mathbf{x} + \lambda \left( \int_\Omega m \, d\mathbf{x} - \eta |\Omega| \right),
\]

where \( u = u(m) \in \mathcal{U}_h \) satisfies

\[
\langle v, \mathcal{R}(m, u) \rangle = 0, \quad \forall v \in \mathcal{U}_h,
\]

and

\[
0 < m_{lw} \leq m \leq 1.
\]

By replacing \( J_h(m) \) with

\[
J_{h,NN}(m) = \int_{\Gamma_{\text{out}}} g \mathcal{F}_{h,NN}(m) \, dS(\mathbf{x}) \quad \text{and} \quad J_{h,NN}^C(m) = \int_{\Gamma_{\text{out}}} g \mathcal{F}^C(m, \mathcal{F}_{h,NN}(m)) \, dS(\mathbf{x}),
\]

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

Optimization problem (39) is solved using a bi-level iterative scheme, wherein the outer iteration, \( u \) and pair \((m, \lambda)\) 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 \((m_{k+1}, \lambda_{k+1})\) 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 \( \|m_k - m_{k+1}\|_{L^2(\Omega)} \leq \gamma_{\text{tol}} \) or when \( k = n_{\text{max}} \).

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 \( m(\mathbf{x}) = 0.1 \). Lagrange multiplier \( \lambda \) is initialized as \( \lambda = 1 \). The tolerance in Algorithm 2 is set to \( n_{\text{tol}} = 0.005 \). Finally, \( m_{lw} = 0.001 \).
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 $u_{NN}^C = \mathcal{F}^C(m, \mathcal{F}_{NN}(m))$ using the corrector operator in which case the minimizer is denoted by $\tilde{m}_{NN}^C$.

The percentage errors between numerical minimizers can be defined as:

$$\tilde{e}_{NN} := \frac{||\tilde{m} - \tilde{m}_{NN}||_2}{||\tilde{m}||_2} \times 100, \quad \tilde{e}_{NN}^C := \frac{||\tilde{m} - \tilde{m}_{NN}^C||_2}{||\tilde{m}||_2} \times 100.$$  \hspace{1cm} (40)

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

$$\tilde{e}_{NN} := \frac{||u(\tilde{m}) - u_{NN}(\tilde{m}_{NN})||_2}{||u(\tilde{m})||_2} \times 100, \quad \tilde{e}_{NN}^C := \frac{||u(\tilde{m}) - u_{NN}^C(\tilde{m}_{NN}^C)||_2}{||u(\tilde{m})||_2} \times 100.$$  \hspace{1cm} (41)

The numerical minimizer $\tilde{m}$ and the forward solution at $\tilde{m}$ are shown in Figure 11. The history of the cost function, the volumetric average of $m$, and the Lagrange multiplier are plotted in Figure 12. In Figure 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 $\tilde{e}_{NN}$, $\tilde{e}_{NN}^C$, $\tilde{e}_{NN}$, and $\tilde{e}_{NN}^C$ are plotted in Figure 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.

5. Conclusion

The work provides 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 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
Figure 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.

Figure 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 $F^C$. For the properties of neural operators, see Table 2.
Figure 14: Normalized minimizer errors $\tilde{e}_{NN}$ and $\tilde{e}_{C\,NN}$ 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).

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.
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.

Acknowledgement

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.

References


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 \( u = \mathcal{F}(m) \), i.e., \( \mathcal{R}(m, u) = 0 \). From the definition of corrector operator \( \mathcal{F}^C \) (see (9)), it can be shown that

\[
\mathcal{F}^C(m) = u^C = \tilde{u} - \delta_u \mathcal{R}(m, \tilde{u})^{-1} \mathcal{R}(m, \tilde{u})
\]

\[
\Rightarrow \quad u - u^C = u - \tilde{u} + \delta_u \mathcal{R}(m, \tilde{u})^{-1} \left( \mathcal{R}(m, \tilde{u}) - \mathcal{R}(m, u) \right) \geq 0
\]

\[
\Rightarrow \quad e^C = \tilde{e} - \delta_u \mathcal{R}(m, \tilde{u})^{-1} (\mathcal{R}(m, u) - \mathcal{R}(m, \tilde{u})). \tag{A.1}
\]

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

\[
\mathcal{R}(m, u) - \mathcal{R}(m, \tilde{u}) = \delta_u \mathcal{R}(m, \tilde{u})(\tilde{e}) + \int_0^1 (1 - s) \delta_u^2 \mathcal{R}(m, \tilde{u} + s\tilde{e})(\tilde{e}, \tilde{e}) \, ds
\]
in (A.1), it holds

\[ e^C = \delta_u \mathcal{R}(m, \tilde{u})^{-1} \mathcal{R}(m, \tilde{u})(\tilde{e}) - \delta_u \mathcal{R}(m, \tilde{u})^{-1} (\mathcal{R}(m, u) - \mathcal{R}(m, \tilde{u})) \]

\[ = \delta_u \mathcal{R}(m, \tilde{u})^{-1} [\mathcal{R}(m, \tilde{u})(\tilde{e}) - (\mathcal{R}(m, u) - \mathcal{R}(m, \tilde{u}))] \]

\[ = \delta_u \mathcal{R}(m, \tilde{u})^{-1} \left[ - \int_0^1 (1 - s) \delta_u^2 \mathcal{R}(m, \tilde{u} + s \tilde{e})(\tilde{e}, \tilde{e}) \, ds \right] \]

(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

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

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

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

\[ \square \]

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.

Appendix 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

\[ \| i(u) \|_Y \leq C \| u \|_X, \quad \forall u \in X . \]

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 [90] 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 \( \Omega \) is open and bounded subset of \( \mathbb{R}^d \) or \( \Omega \equiv \mathbb{R}^d \), 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 $\Omega$ 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
\[
||u||_{L^2(\Omega)} \leq C_S ||u||_{\mathcal{U}},
\]  
(\text{B.1})
where $C_S$ independent of $u$ is the Sobolev embedding constant.

Next, the Poincaré inequality theorem following [Theorem 2.6.3, [90]] 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 $\Omega$, such that
\[
||u||_{L^2(\Omega)} \leq C_P ||\nabla u||_{L^2(\Omega)}, \quad \forall u \in H^1_0(\Omega).
\]  
(\text{B.2})

**Appendix B.2. Proof of Theorem 3**

The proof is divided into three steps as follows.

**Step 1.** First, the upper bound on $||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'}$ is established. Using the definition of operator norm gives
\[
||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'} = \sup_{||v||_{\mathcal{U}} = 1} ||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'}.
\]  
(\text{B.3})
Further,
\[
||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'} = \sup_{||w||_{\mathcal{U}} = 1} ||(w, \delta_u \mathcal{R}(m, \hat{u})(v))||_{\mathcal{U}}.
\]  
(\text{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
\[
||\langle w, \delta_u \mathcal{R}(m, \hat{u})(v) ||_{\mathcal{U}} \leq \int_\Omega \left\{ \left( \kappa_0 \sup_{x \in \Omega} |m(x)| \right) |\nabla v|^2 + 3 \alpha \hat{u}^2 |w|^2 \right\} dx
\leq \left[ \kappa_0 |m|_{L^\infty(\Omega)} ||\nabla v||_{L^2(\Omega)} ||\nabla w||_{L^2(\Omega)} + 3 \alpha \left( \int_{\Omega} \hat{u}^4 dx \right)^{1/2} \left( \int_{\Omega} v^2 w^2 dx \right)^{1/2} \right]
\leq \kappa_0 |m|_{L^\infty(\Omega)} ||v||_{\mathcal{U}} ||w||_{\mathcal{U}} + 3 \alpha \hat{u}_m^2 ||v||_{L^4(\Omega)} ||w||_{L^4(\Omega)}
\leq \kappa_0 |m|_{L^\infty(\Omega)} ||v||_{\mathcal{U}} ||w||_{\mathcal{U}} + 3 \alpha C_\delta^2 ||\hat{u}||^2 ||v||_{\mathcal{U}} ||w||_{\mathcal{U}},
\]  
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 is shown that
\[
||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'} \leq \left( \kappa_0 |m|_{L^\infty(\Omega)} + 3 \alpha C_\delta^4 ||\hat{u}||^2 ||v||_{\mathcal{U}} \right) ||v||_{\mathcal{U}}.
\]  
(\text{B.5})

For the lower bound on $||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'}$, proceeding as follows
\[
||\delta_u \mathcal{R}(m, \hat{u})(v)||_{\mathcal{U}'} = \sup_{||v||_{\mathcal{U}} = 1} ||\langle v, \delta_u \mathcal{R}(m, \hat{u})(v) ||_{\mathcal{U}}
\geq \frac{1}{||v||_{\mathcal{U}}} \left| \int_\Omega \left\{ \kappa_0 m(x) |\nabla v|^2 + 3 \alpha \hat{u}^2 |v|^2 \right\} dx \right|
\geq \frac{\kappa_0 m_{\text{lw}}}{||v||_{\mathcal{U}}} ||\nabla v||_{L^2(\Omega)}^2 + \left( \frac{\kappa_0 m_{\text{lw}}}{2 ||v||_{\mathcal{U}}} \right) \left[ ||\nabla v||_{L^2(\Omega)}^2 + ||\nabla v||_{L^2(\Omega)}^2 \right]
\geq \frac{\kappa_0 m_{\text{lw}}}{2 ||v||_{\mathcal{U}}} \left[ ||\nabla v||_{L^2(\Omega)}^2 + C_P^{-2} ||v||_{L^2(\Omega)}^2 \right]
\geq \frac{\kappa_0 m_{\text{lw}}}{2} \min\{1, C_P^{-2}\} \frac{1}{||v||_{\mathcal{U}}} ||v||_{\mathcal{U}}^2 = \hat{C}_t \hat{C}_R ||v||_{\mathcal{U}},
\]  
for $\hat{C}_t, \hat{C}_R > 0$.
Let \( \frac{1}{\epsilon} \left( w, \delta_\epsilon \mathcal{R}(m, \tilde{u})(v) \right)_{\mathcal{U}^*} \geq \frac{|\langle v, \delta_\epsilon \mathcal{R}(m, \tilde{u})(v) \rangle_{\mathcal{U}^*}|}{|v|_{\mathcal{U}^*}} \) is used in the second equation. In the fourth equation, the positive term is dropped, and the property that \( m \geq m_{tw} \) is used, since \( m \in \mathcal{M} = \{ g \in L^2(\Omega) : \lambda g \leq m_{tw} \} \). In the fifth step, the Poincare 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 \( \delta_\epsilon \mathcal{R}(m, \tilde{u})^{-1} \) is established. Let \( A := \delta_\epsilon \mathcal{R}(m, \tilde{u}) \) and \( r = A(v) \in \mathcal{U}^* \), then

\[
\| A^{-1}(r) \|_{\mathcal{U}} = \| A^{-1}(A(v)) \|_{\mathcal{U}} = \| v \|_{\mathcal{U}}.
\]

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

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

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

\[
\| \delta_\epsilon^2 \mathcal{R}(m, \tilde{u}) \|_{\mathcal{L}(\mathcal{U}^*)} = \sup_{\| v \|_{\mathcal{U}} = 1} \| \delta_\epsilon^2 \mathcal{R}(m, \tilde{u})(v) \|_{\mathcal{U}^*}. 
\]

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

\[
\| \delta_\epsilon^2 \mathcal{R}(m, \tilde{u})(v, w) \|_{\mathcal{U}^*} = \sup_{\| q \|_{\mathcal{U}} = 1} | \langle q, \delta_\epsilon^2 \mathcal{R}(m, \tilde{u})(v, w) \rangle_{\mathcal{U}} | \\
= \sup_{\| q \|_{\mathcal{U}} = 1} \left| \int_\Omega 6\alpha \nabla \tilde{u} \cdot \nabla w \, dx \right| \leq 6\alpha \sup_{\| q \|_{\mathcal{U}} = 1} \| \tilde{u} \|_{L^4(\Omega)} \| v \|_{L^4(\Omega)} \| w \|_{L^4(\Omega)} \| q \|_{L^4(\Omega)} \\
\leq 6\alpha C_S^4 \sup_{\| q \|_{\mathcal{U}} = 1} \| \tilde{u} \|_{\mathcal{U}} \| v \|_{\mathcal{U}} \| w \|_{\mathcal{U}} \| q \|_{\mathcal{U}} \\
= \left( 6\alpha C_S^4 \| \tilde{u} \|_{\mathcal{U}} \right) \| v \|_{\mathcal{U}} \| w \|_{\mathcal{U}},
\]

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**

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 example9 in [100].

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

\[
\int_{\Gamma_{out}} g(u(m)) \, dS(x) = \int_\Omega m \nabla u \cdot \nabla u \, dx + \int_\Omega u^4 \, dx.
\]

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

\[
\int_{\Gamma_{out}} g(u(m)) \, dS(x) = \int_\Omega \left[ \frac{e(m)}{m} + u(m)^4 \right] \, dx.
\]

---

Thus, the formula for updating arbitrary inner iteration inside the outer iteration step is presented in

\[ m_{\text{next}} = m \]

Next, given problem with fixed, i.e., \( m_k, \lambda_k, u_k = u(m_k) \), first, the pair \((m_k+1, \lambda_{k+1})\) is updated by solving the approximate minimization problem with fixed \( u = u_k \), i.e.,

\[
(m_{k+1}, \lambda_{k+1}) = \arg \min_{m \in \mathcal{M}, \lambda \in \mathbb{R}} \tilde{J}(m, \lambda, u_k) = \int_{\Omega} \left[ \frac{e(m)}{m} + u(m)^4 \right] \, dx + \lambda \left( \int_{\Omega} \, dx - \eta |\Omega| \right)
\]

and \( 0 < m_{\text{lw}} \leq m \leq 1 \).

The problem above is coupled in variables \( m, \lambda, \) 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 \((m_{k+1}, \lambda_{k+1})\) is updated by solving the approximate minimization problem with fixed \( u = u_k \), i.e.,

\[
(m_{k+1}, \lambda_{k+1}) = \arg \min_{m \in \mathcal{M}, \lambda \in \mathbb{R}} J(m, \lambda, u_k) = \int_{\Omega} \left[ \frac{e(m)}{m} + u(m)^4 \right] \, dx + \lambda \left( \int_{\Omega} \, dx - \eta |\Omega| \right)
\]

and \( 0 < m_{\text{lw}} \leq m \leq 1 \).

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 \( \|m_{k+1} - m_k\|_{L^2(\Omega)} \leq \gamma_{\text{tol}} \) 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 \( k < n_{\text{outer iter}} \) do
  (1) given \( m_k, \lambda_k, \) and \( u_k \), solve for \((m_{k+1}, \lambda_{k+1}) \in \mathcal{M} \times \mathbb{R} \) following Algorithm 2;
  (2) solve \((v, \mathcal{R}(m_{k+1}, u_{k+1})) = 0\), \( \forall v \in \mathcal{U}_h \), for \( u_{k+1} \in \mathcal{U}_h \);
  (3) update flux \( q_{k+1} = m_{k+1} \nabla u_{k+1} \) and \( e_{k+1} = q_{k+1} \cdot q_{k+1} \);
  (4) \( m_k = m_{k+1}, \lambda_k = \lambda_{k+1} \), \( u_k = u_{k+1} \), and \( k \leftarrow k + 1 \);
  if \( \|m_{k+1} - m_k\|_{L^2(\Omega)} \leq \gamma_{\text{tol}} \) 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 Equation (C.2), when the variables \( \lambda \) and \( u \) are fixed, the optimization problem on \( m \) becomes

\[
m^{(i+1)} = \min_{m \in \mathcal{M}} \tilde{J}(m, \lambda^{(i)}, u_k) = \int_{\Omega} \left[ \frac{e_k}{m} + u_k^4 \right] \, dx + \lambda^{(i)} \left( \int_{\Omega} \, dx - \eta |\Omega| \right)
\]

and \( 0 < m_{\text{lw}} \leq m \leq 1 \),

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

\[
\int_{\Omega} \left[ - \frac{e_k}{2} + \lambda^{(i)} \right] w \, dx = 0, \quad \forall w \in \mathcal{M} \quad \Rightarrow \quad m = \sqrt{\frac{e_k}{\lambda^{(i)}}}
\]

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

\[
m^{(i+1)} = \min \left\{ 1, \max \left\{ m_{\text{lw}}, \sqrt{\frac{e_k}{\lambda^{(i)}}} \right\} \right\}
\]

33
where the upper and lower bound constraints on $m$ are enforced strongly. The algorithm for the inner iteration where $\lambda$ 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 [101].

**Algorithm 2**: Inner iteration for pair $(m_{k+1}, \lambda_{k+1})$ given $m_k, \lambda_k, u_k$.

**Input:**
(a) $\eta$, lower bound $m_{lw}$, and tolerance $m_{tol}$ 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 $\lambda_{max} = \lambda$ ;
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:** $(m_{k+1}, \lambda_{k+1})$. 