Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 404 (2023) 115783
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Wavelet Neural Operator for solving parametric partial differential equations in computational mechanics problems

Tapas Tripura[a,*], Souvik Chakraborty[a,b,*]

[a] Department of Applied Mechanics, Indian Institute of Technology Delhi, Hauz Khas, Delhi, 110016, India
[b] Yardi School of Artificial Intelligence (ScAI), Indian Institute of Technology Delhi, Hauz Khas, Delhi, 110016, India

Received 5 August 2022; received in revised form 15 November 2022; accepted 16 November 2022
Available online 6 December 2022

Dataset link: https://github.com/csccm-iitd/WNO

Abstract

With massive advancements in sensor technologies and Internet-of-things (IoT), we now have access to terabytes of historical data; however, there is a lack of clarity on how to best exploit the data to predict future events. One possible alternative in this context is to utilize an operator learning algorithm that directly learns the nonlinear mapping between two functional spaces; this facilitates real-time prediction of naturally arising complex evolutionary dynamics. In this work, we introduce a novel operator learning algorithm referred to as the Wavelet Neural Operator (WNO) that blends integral kernel with wavelet transformation. WNO harnesses the superiority of the wavelets in time–frequency localization of the functions and enables accurate tracking of patterns in the spatial domain and effective learning of the functional mappings. Since the wavelets are localized in both time/space and frequency, WNO can provide high spatial and frequency resolution. This offers learning of the finer details of the parametric dependencies in the solution for complex problems. The efficacy and robustness of the proposed WNO are illustrated on a wide array of problems involving Burger's equation, Darcy flow, Navier–Stokes equation, Allen–Cahn equation, and Wave advection equation. A comparative study with respect to existing operator learning frameworks is presented. Finally, the proposed approach is used to build a digital twin capable of predicting Earth's air temperature based on available historical data.

Keywords: Nonlinear mappings; Operator learning; Wavelet; Wavelet neural operator; Scientific machine learning

1. Introduction

The application of partial differential equations (PDEs) for modeling various physical and complex phenomena in science and engineering problems (e.g., fluid flow, traffic flow, chemical reactions, biological growth, etc. [1–3]) is well acknowledged. Analytical solutions for PDEs are rarely available, and hence, one often has to rely on numerical techniques such as finite element [4], finite difference [5], and finite volume [6] based approaches. Among recent developments, nonlocal operator [7,8] and isogeometric analysis [9,10] have also gained popularity. However, almost

* Corresponding authors.
E-mail addresses: tapas.t@am.iitd.ac.in (T. Tripura), souvik@am.iitd.ac.in (S. Chakraborty).
URL: https://www.csccm.in/ (S. Chakraborty).

https://doi.org/10.1016/j.cma.2022.115783
0045-7825/

all the classical techniques are discretization dependent and computationally expensive. Additionally, with changes in parametric values, the solvers need to be run from scratch; this severely limits the applicability of the previously mentioned approaches. In recent years, neural networks (NN) have emerged as a possible alternative to popular classical solvers [11,12]. NNs are universal approximators [13–16] and approximate highly non-linear functions by passing the weighted sum of inputs through a non-linear activation function. The learning of NNs can either be data-driven [17–20] or physics-informed [21–24] in nature. Physics-informed NNs are very accurate and exploit governing law of the physical problem during training. Therefore, if the physics of the underlying processes are known in advance, physics-informed NN can be used. Unfortunately, for many natural processes, the governing physics is not known a priori, and data-driven algorithms are the only feasible option. The data-driven algorithms, however, do not ensure the governing physics of the underlying problem and hence, fail to generalize beyond the training data distribution. To address these challenges, the neural operator was proposed in [25].

Neural operators learn the mapping between two infinite-dimensional function spaces. Once trained, the neural operators can predict the solution for any given input function. Neural operators also share the same network parameters between different discretizations, and therefore neural operators are discretization invariant. Overall, the neural operators are computationally advantageous compared to the classical PDE solvers. Similar to universal function approximation theory, the neural operators work on the theorem of universal operator approximation proposed in [12].

DeepONet [25,26] was the first operator learning framework proposed by the researchers. It consists of two neural networks, which are referred to as branch and trunk NN. The branch NN is responsible for the input function, and the trunk NN corresponds to the output function at some sensor point. The output in the DeepONet is then obtained from the inner product of these NNs. Its application can be found in various domains including reliability analysis [27], bubble dynamics [28], and fracture mechanics [29]. In a different approach, the graph neural operator (GNO) was proposed in [30]. The GNO focuses on learning infinite-dimensional mapping by the composition of nonlinear activation functions and a certain class of integral operators. The kernel integration in the GNO is learned through message passing between the graph networks. However, GNO has the tendency to become unstable with the increase in the number of hidden layers. As an improvement over the GNO, Fourier neural operator (FNO) was proposed to learn the parameters of the network in Fourier space [31]. The heart of the FNO is the spectral decomposition of the input using a fast Fourier transform (FFT) and computation of the convolution integral kernel in the Fourier space. The efficacy of the FNO was demonstrated on various state-of-the-art problems, including Burgers, Darcy, and Navier–Stokes equations.

One major shortcoming of the FNO is that the basis functions in FFTs are generally frequency localized with no spatial resolution [32]. Consequently, the performance of the FNO gets hindered in complex boundary conditions. In this context, one can think of wavelets that are both space and frequency-localized [32,33]. Due to the availability of spatial information, the wavelets can better handle signals with discontinuity and/or spikes and, therefore, can learn the patterns in images better than FFTs. To explain further, the wavelet provides us with characteristic frequency and characteristic space information, with the help of which we can not only tell the frequencies present in the signal as a function of characteristic frequency but also the location of the frequency as a function of characteristic space information. The application of wavelets can be found in various aspects of data compression, such as fingerprint compression [34], compressed sensing [35], iris recognition [36], signal denoising [37], motion analysis [38], and fault detection [39] to name few. The use of wavelets in neural networks can also be found in the literature [40–42]. In Ref. [43], a multiwavelet decomposition (MWT) based operator is proposed where the wavelet kernels are computed using four independent neural networks. This network combines a fully connected neural network (FNN), a Fourier integral layer similar to FNO, and a convolution neural network (CNN). While the FNN represents the coarse scale, the latter three are used in combinations to compute the detailed wavelet coefficients. Therefore, the MWT can be regarded as an improvement over FNO. However, MWT works only with inputs having resolutions in powers of two. In this context, we hypothesize that integrating wavelet decomposition with kernel integral operator can result in a robust neural operator and will alleviate the limitation associated with MWT.

In this work, we propose a new spectral decomposition-based neural operator, referred to as the Wavelet Neural Operator (WNO), for learning mappings between infinite-dimensional spaces of functions. In the proposed WNO, the snapshots are first transformed to their high and low-frequency components using discrete wavelet transform (DWT) [44,45]. In the DWT, the signal is decomposed at different levels using wavelets of different scales. In general, the decomposed parts of a signal at smaller levels predominantly consist of noise. Therefore, the key

features of a signal at these levels are masked. On the other hand, the wavelet coefficients at higher levels contain the most useful information. Since our primary aim is to learn the most relevant information from a signal, we create an information subset using wavelet coefficients at higher levels of discrete wavelet decomposition only. This subset is then utilized to learn the solution operators. Since the proposed WNO harnesses the benefits of the wavelets, it can learn solution operators of highly nonlinear parametric PDEs having an irregular domain and boundary conditions. It can further be shown that if the mother wavelet basis in WNO is replaced by trigonometric functions, then FNO becomes the special case of WNO. Overall, the salient points of the proposed WNO architecture are as follows:

- The network parameters in the proposed WNO are learned in the wavelet space. Since the wavelet space is both frequency and spatial localized, WNO can learn the patterns in the images and/or signals more effectively.
- The proposed WNO can handle highly nonlinear family of PDEs with discontinuities and abrupt changes in the solution domain and the boundary.
- The accuracy of the proposed WNO is consistent for both smooth and complex geometry.
- The proposed WNO can learn the frequency of changes in the patterns over the solution domain; this makes it amenable for online implementations without requiring the entire dataset. Thus, the proposed WNO can be generalized to images and videos and can also be implemented for short-to-medium-range climate modeling and weather forecast purposes.

Due to these reasons, WNO can be straightforwardly applied to a wide array of problems, including fluid mechanics, traffic flow modeling, and climate modeling.

The remainder of the paper is arranged as follows: in Section 2, the general idea of a neural operator is presented. In Section 3, the proposed wavelet neural operator is briefly discussed. In Section 4, a variety of numerical examples are considered to demonstrate the efficacy and robustness of the proposed neural operator. These examples include Burger's equation, Darcy flow, Navier–Stokes equation, and Wave advection equation. A comparative study with respect to other existing neural operators has also been presented in this section. In Section 5, we utilize the proposed WNO to develop a digital twin capable of predicting Earth's air temperature at 2 m above the ground at a resolution of $2^\circ \times 2^\circ$ based on historical data. Finally, in Section 6, we review the salient features of the proposed neural operator and conclude the paper.

2. Problem formulation and neural operator

An operator is responsible for mapping between infinite-dimensional input and output function spaces. To understand it better, let $D \in \mathbb{R}^d$ be the n-dimensional domain with boundary ∂D. For a fixed domain $D = (a, b)$ and $x \in D$, consider a PDE which maps the input function spaces, i.e., the space containing the source term $f(x,t) : D \mapsto \mathbb{R}$, the initial condition $u(x,0) : D \mapsto \mathbb{R}$, and the boundary conditions $u(\partial D, t) : D \mapsto \mathbb{R}$ to the solution space $u(x,t) : D \mapsto \mathbb{R}$, with t being the time coordinate. In the present work, we focus on learning the operator that maps the input functions to the solution space $u(x,t)$. Now, let us define two complete normed vector spaces $(\mathcal{A}, \|.\|_{\mathcal{A}})$ and $(\mathcal{U}, \|.\|_{\mathcal{U}})$ of functions taking values in $\mathbb{R}^{d_a}$ and $\mathbb{R}^{d_u}$ with given number of partial derivatives. These function spaces are often called Banach spaces and denoted as $\mathcal{A} := \mathcal{C}(D; \mathbb{R}^{d_w})$ and $\mathcal{U} := \mathcal{C}(D; \mathcal{R}^{d_u})$. For the given domain $D \subset \mathbb{R}^d$ and normed spaces $(\mathcal{A}, \|.\|_{\mathcal{A}})$ and $(\mathcal{U}, \|.\|_{\mathcal{U}})$, we aim at learning the nonlinear operator $\mathcal{D} : \mathcal{A} \mapsto \mathcal{U}$. One elegant approach is to learn the nonlinear operator $\mathcal{D}$ so that the solutions to a family of PDEs can be obtained for different sets of input parameters $\boldsymbol{a} \in \mathcal{A}$.

Let the input and output functions in the function spaces $\mathcal{A}$ and $\mathcal{U}$ be denoted by $\boldsymbol{a} : D \mapsto a(x \in D) \in \mathbb{R}^{d_a}$ and $\boldsymbol{u} : D \mapsto u(x \in D) \in \mathbb{R}^{d_u}$. Further, consider that $u_j = \mathcal{D}(a_j)$ and we have access to N number of pairs $\{(a_j, u_j)\}_{j=1}^N$. Then, we aim to approximate $\mathcal{D}$ by neural networks as,

$$\mathcal{D} : \mathcal{A} \times \boldsymbol{\theta}_{NN} \mapsto \mathcal{U}; \quad \boldsymbol{\theta}_{NN} = \{\mathcal{W}_{NN}, \boldsymbol{b}_{NN}\}, \tag{1}$$

where $\boldsymbol{\theta}_{NN}$ denotes the finite-dimensional parameter space of the neural networks. Although the input and output functions a_j and u_j are continuous, for numerical implementation we assume the n_D point discretization $\{x_p\}_{p=1}^{n_D}$ of the domain D. Therefore, for the N collection of the input–output pair, we assume that we have access to the datasets $\{a_j \in \mathbb{R}^{n_D \times d_a}, u_j \in \mathbb{R}^{n_D \times d_u}\}_{j=1}^N$. With this setup, the objective here is to develop a network that can exploit the data to learn the operator $\mathcal{D}$. To that end, we first lift the inputs $a(x,t) \in \mathcal{A}$ to some high dimensional space through a local transformation $P(a(x)) : \mathbb{R}^{d_a} \mapsto \mathbb{R}^{d_v}$ and denote the high dimensional space as $v_0(x) = P(a(x))$, where $v_0(x)$ takes values in $\mathbb{R}^{d_v}$. In a neural network framework, the local transformation $P(a(x))$ can be achieved by

constructing a shallow fully connected neural network (FNN). Further, let us denote that a total number of l steps are required to achieve satisfactory convergence. Therefore, we apply l numbers of updates $v_{j+1} = G(v_j) \forall j = 1, \ldots, l$ on $v_0(x)$, where the function $G : \mathbb{R}^{d_v} \mapsto \mathcal{R}^{d_v}$ takes value in $\mathbb{R}^{d_v}$. Once all the updates are performed, we define another local transformation $Q(v_l(x)) : \mathbb{R}^{d_v} \mapsto \mathbb{R}^{d_u}$ and transform back the output $v_l(x)$ to the solution space $u(x) = Q\left(v_l(x)\right)$. The step-wise updates $v_{j+1} = G(v_j)$ are defined as,

$$v_{j+1}(x) := g\left(\left(K(a;\phi) * v_j\right)(x) + W v_j(x)\right); \quad x \in D, \quad j \in [1, l], \tag{2}$$

where $g(\cdot) : \mathbb{R} \to \mathbb{R}$ is a non-linear activation function, $W : \mathbb{R}^{d_v} \to \mathbb{R}^{d_v}$ is a linear transformation, $K : \mathcal{A} \times \boldsymbol{\theta} \to \mathcal{L}\left(\mathcal{U}, \mathcal{U}\right)$ is the integral operator on $\mathcal{C}\left(D; \mathbb{R}^{d_v}\right)$. In the neural network setup, we represent $K(a;\phi)$ to be a kernel integral operator parameterized by $\phi \in \boldsymbol{\theta}$. The kernel integral operator $K(a;\phi)$ is defined as,

$$\left(K(a;\phi) * v_j\right)(x) := \int_{D \in \mathbb{R}^d} k\left(a(x), x, y; \phi\right) v_j(y) \mathrm{d}y; \quad x \in D, \quad j \in [1, l], \tag{3}$$

where $k_\phi : \mathbb{R}^{2d+d_a} \mapsto \mathbb{R}^{d_v \times d_v}$ is a neural network parameterized by $\phi \in \boldsymbol{\theta}$. Suppose there is some kernel $k \in \mathcal{C}\left(D; \mathbb{R}^{d_v}\right)$, then we want to employ the concept of degenerate kernels to learn the κ_ϕ from the data such that,

$$\| \left(k_\phi(a) - k(a)\right) v \| \leq \| \left(k_\phi(a) - k(a)\right) \| \| v \|, \tag{4}$$

and $k_\phi \to k$. Since the kernel, $k \in \mathcal{C}(D)$ and $K \in \mathcal{L}(\mathcal{C}(D; \mathbb{R}^{d_v}))$ together defines the infinite-dimensional spaces, Eq. (2) can learn the mapping of any infinite-dimensional function space. As far as the approximation of nonlinear operators is concerned, the structure of Eq. (2) follows the conventional neural networks. We propose a new neural operator referred to as the WNO that learns the mapping with a higher order of accuracy. One key component of the proposed WNO is the wavelet kernel integral layer. The overarching framework of the proposed WNO is presented in Fig. 1. While Fig. 1(a) provides the schematic description of the proposed WNO, in Fig. 1(b), a simplistic WNO with wavelet kernel integral layer is shown for easy understanding.

For training the network, we define an appropriate loss function $\mathcal{L} = \mathcal{U} \times \mathcal{U}$ as follows:

$$\boldsymbol{\theta}_{NN} = \underset{\boldsymbol{\theta}_{NN}}{\arg\min} \mathcal{L}\left(\mathcal{D}(\boldsymbol{a}), \mathcal{D}(\boldsymbol{a}, \boldsymbol{\theta})\right). \tag{5}$$

We leverage the universal approximation theorems of the neural networks [11,12] and try to learn the parameter space (weights-$\mathcal{W}_{NN}$ and biases-$\boldsymbol{b}_{NN}$ of neural networks) by minimizing the loss function from a data-driven perspective.

3. Wavelet neural operator for parametric partial differential equations

We aim to perform the kernel integration in Eq. (3) by parameterizing the kernel $k(a(x, y), x, y)$ with $\phi \in \boldsymbol{\theta}$. Before constructing the framework for parameterization in wavelet space, let us revisit the convolution integral. Let us drop the term $a(x, y)$ in Eq. (3) and modify the kernel as $k_\phi(x - y)$. By doing this we rewrite the Eq. (3) as,

$$\left(K(\phi) * v_j\right)(x) := \int_{D \in \mathbb{R}^d} k\left(x - y; \phi\right) v_j(y) \mathrm{d}y; \quad x \in D, \quad j \in [1, l], \tag{6}$$

which is consistent with the convolution operator. In this work, we propose to learn the kernel k_ϕ by parameterizing it in the wavelet domain. By learning kernel in the wavelet domain, we intend to perform the above convolution on the coefficients of wavelet decomposition rather than direct convolution in physical space. Let $\psi(x) \in \mathcal{L}^2(\mathbb{R})$ be an orthonormal mother wavelet that is localized both in the time and frequency domain. Also, assume $\mathcal{W}(\Gamma)$ and $\mathcal{W}^{-1}(\Gamma_W)$ be the forward and inverse wavelet transforms of an arbitrary function $\Gamma : D \mapsto \mathbb{R}^{d_v}$. Then the forward and inverse wavelet transforms of the function Γ with the scaling and translational parameters $\alpha \in \mathbb{R}^{+*}$ and $\beta \in \mathbb{R}$ are given by the following integral pairs,

$$\begin{aligned} \left(\mathcal{W}\Gamma\right)_j(\alpha, \beta) &= \int_D \Gamma(x) \frac{1}{|\alpha|^{1/2}} \psi\left(\frac{x - \beta}{\alpha}\right) dx, \\ \left(\mathcal{W}^{-1}\Gamma_w\right)_j(x) &= \frac{1}{C_\psi} \int_0^\infty \int_D (\Gamma_w)_j(\alpha, \beta) \frac{1}{|\alpha|^{1/2}} \tilde{\psi}\left(\frac{x - \beta}{\alpha}\right) d\beta \frac{d\alpha}{\alpha^2}, \end{aligned} \tag{7}$$

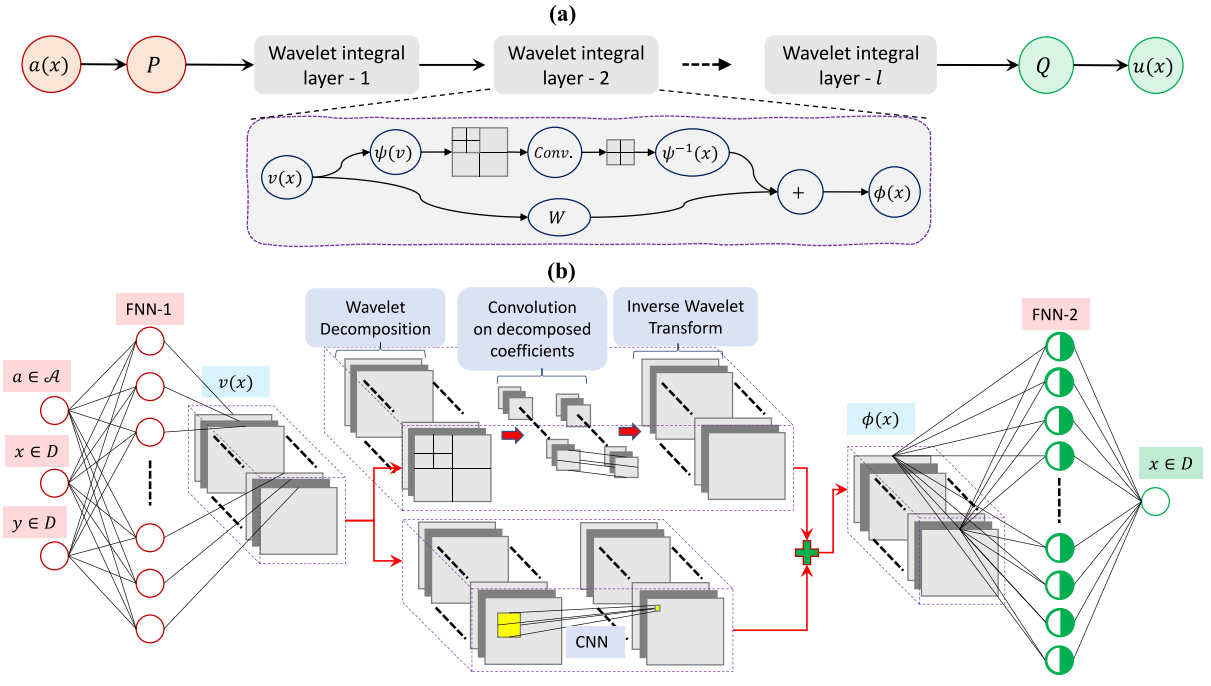

Fig. 1. Illustration of the architecture of the wavelet neural operator (WNO). (a) **Schematic of the proposed neural operator**. First, lift the inputs to a higher dimension by a local transformation $P(\cdot)$. Then pass the lifted inputs to a series of wavelet kernel integral layers. Transform back the final output of the wavelet integral layer using a local transformation $Q(\cdot)$. Finally, activate the output of $Q(\cdot)$, which will provide the solution $u(x)$. In the wavelet kernel integral layer, the multilevel wavelet decomposition of the inputs is performed first, resulting in the horizontal, vertical, and diagonal coefficients at different levels. The convolution between neural network weights and the subband coefficients of the last level is then performed. The inverse wavelet transform is performed on the convoluted coefficients to transform the reduced inputs to the original dimension. Suitable activation is done on the output of the wavelet kernel integration layer. (b) **A simple WNO with one wavelet kernel integral layer**. The inputs contain the initial parameters and space information. The local transformations $P(\cdot)$ and $Q(\cdot)$ are modeled as shallow fully connected neural networks. The output of $P(\cdot)$ is fed into the wavelet integral layer. The integral layer consists of two separate branches. In the first branch, the wavelet decomposition of inputs, followed by parameterization of the integral kernel is done. In the second branch, a convolution neural network (CNN) with kernel size 1 is constructed. The outputs of the two branches are then summed, and activations are performed. Then the outputs are passed through the transformation $Q(\cdot)$, which provides the target solution $u(x)$. Similarly, a WNO with an arbitrary number of wavelet integral layers can be constructed.

where $(\Gamma_w)_j(\alpha,\beta) = (\mathcal{W}\Gamma)_j(\alpha,\beta)\ \psi((x-\beta)\alpha) \in \mathcal{L}^2(\mathbb{R})$ is a scaled and translated mother wavelet, often called as the daughter wavelet. The desired wavelets can be obtained from the mother wavelet by scaling and shifting. The term C_ψ is called the admissible constant whose range is given as $0 < C_\psi < \infty$. The expression for C_ψ is given as,

$$C_\psi = 2\pi \int_D \frac{|\psi(\omega)|^2}{|\omega|} d\omega, \tag{8}$$

where $\psi(\omega)$ denotes the Fourier transform of $\psi(x)$. Applying the convolution theorem, we, therefore, obtain that,

$$\left(K(a;\phi) * v_j\right)(x) = \mathcal{W}^{-1}\left(\mathcal{W}(k_\phi)\cdot\mathcal{W}(v_j)\right)(x); \quad x \in D. \tag{9}$$

Since we propose to parameterize the kernel k_ϕ directly in wavelet space instead of performing wavelet transform of k_ϕ, we represent $R_\phi = \mathcal{W}(k_\phi)$, where R_ϕ represents the wavelet transform of the kernel function $k : D \mapsto \mathbb{R}^{d_v \times d_v}$. With the above information, we define the wavelet neural operator as,

$$\left(\mathcal{K}(\phi) * v_j\right)(x) = \mathcal{W}^{-1}\left(R_\phi \cdot \mathcal{W}(v_j)\right)(x); \quad x \in D. \tag{10}$$

One issue with continuous wavelet transform (CWT) is that it estimates the wavelet coefficients at all possible scales, i.e., the cwt contains information along an infinite number of wavelets, which may be redundant and computationally expensive. However, the same equivalent accuracy with computational speedup can be achieved

by performing the discrete wavelet transform (DWT). In DWT, we modify the mother wavelet to calculate the wavelet coefficients at scales with powers of two. In this case, the wavelet $\phi(x)$ is defined as,

$$\psi_{m,\tau}(x) = \frac{1}{\sqrt{2^m}}\psi\left(\frac{x-\tau 2^m}{2^m}\right), \quad (m,\tau) \in \mathbb{Z}^2, \tag{11}$$

where the parameter $m \in \mathbb{Z}^+$ and $\tau \in \mathbb{Z}^{+*}$ are the scaling and translational parameters. The mother wavelet $\psi(\cdot)$ is translated and scaled by the desired factor to form wavelet basis functions of different families. The mother wavelet $\psi(\cdot)$ is required to satisfy the following properties- (i) $\psi \in L^2(\mathbb{R})$, $(ii)\int_{-\infty}^{\infty}\psi(x)dx = 0$, and (iii) $\|\psi\|_2 = 1$. During scaling, the positive or negative sign on m represents the expansion and contraction, respectively. The admissible constant in DWT is given by,

$$C_\psi = \int_0^\infty \frac{|\psi(\omega)|^2}{|\omega|}d\omega, \tag{12}$$

The forward DWT is given as,

$$(\mathcal{W}\Gamma)_j(m,\tau) = \frac{1}{\sqrt{2^m}}\int_D \Gamma(x)\psi\left(\frac{x-\tau 2^m}{2^m}\right)dx. \tag{13}$$

By fixing the scale parameter m at a particular integer and shifting the τ, only the DWT coefficients at level m can be obtained. Though there are many different flavors of DWT, we particularly chose the filterbank implantations of DWT. The filterbank implementation decomposes the signal into detail and approximation coefficients by passing the signal through a low-pass and a high-pass filter. If $r(n_D)$ and $s(n_D)$ denote the low-pass and high-pass filter, respectively, then two convolutions of the form $z_{low}(n_D) = (x * r)(n_D) \downarrow 2$ and $z_{high}(n_D) = (z * s)(n_D) \downarrow 2$ are executed, where n_D is the number of discretization points. While the detail coefficients $z_{high}(n_D)$ are retained, the approximation coefficients are recursively filtered by passing them through the low-pass and high-pass filters until the total number of decomposition levels is exhausted. At each level, the length of the signal is halved by a factor of 2 due to the conjugate symmetry. At the end of the entire operation, the length of the support width of wavelet coefficients is given by the formula: $\zeta = n_s/2^m + (2n_w - 2)$, where, n_s is the length of the signal, m is the decomposition level in DWT and n_w is the length of the vanishing moment of the undertaken wavelet.

In the DWT, the coefficients in smaller scales, such as 2^0 and 2^1 are usually associated with high frequencies; thus, the smaller scales are not the canonical choice for kernel parameterization. On the other hand, the high-scale wavelet coefficients generally correspond to low-frequency information, thus containing the most important features of the input signal. Therefore we choose to parameterize R_ϕ on higher scales of the discrete wavelet transform, possibly the last level. For this purpose, a finite-dimensional parameterization space is obtained by keeping the information from only the highest scale wavelet coefficients. For $n_D \in D$, we have $v_t \in \mathbb{R}^{n_D \times d_v}$, $\mathcal{W}(v_t) \in \mathbb{R}^{n_D \times d_v}$ and $R_\phi(m) \in \mathbb{R}^{d_v \times d_v}$. Since we want to parameterize R_ϕ only on the highest level of decomposition, we have $\mathcal{W}(v_t) \in \mathbb{R}^{n/2^m \times d_v}$ in an m-level of DWT. In general, the length of wavelet coefficients is also influenced by the number of vanishing moments of the orthogonal mother wavelet. Thus, it is more appropriate to write that $\mathcal{W}(v_t, m) \in \mathbb{R}^{\zeta \times d_v}$. The direct parameterization of R_ϕ in the wavelet space is therefore carried out by integrating out the parameter ϕ as the convolution of $(\zeta \times d_v \times d_v)$-valued tensor. Rephrasing Eq. (10), we can write the convolution of weight tensor $R \in \mathbb{R}^{\zeta \times d_v \times d_v}$ and $\mathcal{W}(v_t, m) \in \mathbb{R}^{\zeta \times d_v}$ as,

$$\left(R \cdot \mathcal{W}(v_j, m)\right)_{I_1,I_2}(x) = \sum_{I_3=1}^{d_v} R_{I_1,I_2,I_3}\mathcal{W}(v_j,m)_{I_1,I_3}; \quad I_1 \in [1,\zeta], \quad I_2, I_3 \in d_v. \tag{14}$$

An algorithm for implementation of the proposed wavelet neural operator is provided in Algorithm 1.

3.1. Multiwavelet decomposition and parameterization of the spatial field

The spatial field after the discretization of the domain D into $n_D \in \mathbb{N}$ points is generally high-dimensional. To facilitate computational speedup and learning of the key features in data, the parameterization of neural networks is performed on a subset of the spatial domain. In the FNO, the parameterization space is created by truncating the higher-order Fourier modes. Say, $\mathcal{F}$ denotes the Fourier transform and $k \in D$ are the frequency modes, then we have $v_t \in \mathbb{R}^{n_D \times d_v}$ and $\mathcal{F}(v_t) \in \mathbb{C}^{n_D \times d_v}$. In the FNO setup, a finite-dimensional parameterization space is constructed by

Algorithm 1 Algorithm of the WNO

Input: N-samples of the pair $\{a(x) \in \mathbb{R}^{n_D \times d_a}, u(x) \in \mathbb{R}^{n_D \times d_u}\}$, coordinates $x \in D$, and network hyperparameters.

Stack the inputs: $\{a(x), x\} \in \mathbb{R}^{n_D \times 2d_a}$.
for epoch $= 1, \ldots,$ epochs **do**
  Uplift the input using transformation $P(\cdot)$: $v_0(x) \in \mathbb{R}^{n_D \times d_v} = P\left(\{a(x), x\} \in \mathbb{R}^{n_D \times 2d_a}\right)$.
  for $j = 1, \ldots, l$ perform the iterations: v_{j+1}=$G(v_j)$ **do**
    Decompose the input using wavelet decomposition: $\mathcal{W}(v_j(x)) \in \mathbb{R}^{n_D/2^m \times d_v}$.
    Parameterize the NN kernel k_ϕ in the wavelet space: $R_\phi * \mathcal{W}(v_j(x))$. ▷ Eq. (14)
    Reconstruct the convoluted input: $v^1_{j+1}(x) = \mathcal{W}^{-1}\left(R_\phi * \mathcal{W}(v_j(x))\right)$. ▷ Eq. (10)
    Perform the linear transform: $v^2_{j+1}(x) = W * v_j(x)$ using CNN.
    Add the outputs of step 7 and 8: $\tilde{v}_{j+1}(x) \in \mathbb{R}^{n_D \times d_v} = \left(v^1_{j+1} + v^2_{j+1}\right)(x)$.
    if $j \neq l$ **then**
      Apply the activation to complete the iteration: $v_{j+1} \in \mathbb{R}^{n_D \times d_v} = g\left(\tilde{v}_{j+1}(x)\right)$. ▷ Eq. (2)
    end if
  end for
  Apply an activated lifting transformation: $r(x) \in \mathbb{R}^{n_D \times d_r}$=$g\left(Q_1(v_l(x))\right)$ ($Q_1(\cdot) : \mathbb{R}^{d_v} \mapsto \mathbb{R}^{d_r}$ is a FNN).
  Compute the final output: $\hat{u}(x) \in \mathbb{R}^{n_D \times d_u}$=$Q_2\left(r(x)\right)$ ($Q_2(\cdot) : \mathbb{R}^{d_r} \mapsto \mathbb{R}^{d_u}$ is a FNN).
  Compute the loss: $\mathcal{L}\left(u, \hat{u}\right)$.
  Compute the gradient of the loss: $\frac{\partial \mathcal{L}\left(u,\hat{u}\right)}{\partial \theta_{NN}}$.
  Update the parameters of the network using the gradient.
end for

Output: Solution space $u \in \mathcal{U}$, parameters of NN θ_{NN}.

truncating the Fourier series at a maximal number of modes $k_{\max} = |\left\{k \in \mathbb{Z}^d : \left|k_j\right| \leq k_{\max,j}, \text{ for } j = 1, \ldots, d\right\}|$. Afterward, the operator parameters are learned by performing convolution of v_t with $\mathcal{F}\left(v_t\right) \in \mathbb{C}^{k_{\max} \times d_v}$ which has $k_{\max}$ Fourier modes. When working with inputs containing higher frequency ranges, this setup results in a loss of information on higher frequency bands.

In this work, we aim to create a multi-resolution representation of the parameterization space by using wavelet basis functions. As stated earlier, the spatial domain is decomposed into a projected domain using multilevel wavelet decomposition. Once the decomposition is performed up to a level-m, the convolution is performed between $\mathcal{W}\left(v_t\right) \in \mathbb{R}^{n_D \times d_v}$ and $R_\phi \in \mathbb{R}^{\zeta \times d_v \times d_v}$. The wavelets have many advantages over the more traditional Fourier methods [46], e.g., wavelets are (i) localized in both spatial and frequency domains, (ii) provide sparse representations, and
(iii) structured hierarchically. Due to the hierarchical structure, while reconstructing using the inverse wavelet transform, the learned parameters will get convoluted automatically to the lower decomposition levels. This ensures no loss of information and compact learning. There exists a variety of families of wavelets, each with differing properties. Although the proposed framework is equally applicable to any of the families of wavelets, we restrict ourselves to the Daubechies wavelets. The Daubechies wavelets are simple yet compact, orthogonal, robust in texture retrieval, and can be chosen according to the level of smoothness required [47].

Let $L^2\left(\mathbb{R}^n\right)$ be the space of square-integrable functions, which is complete with respect to the inner product- $\langle \cdot, \cdot \rangle$ and L^2 norm- $\| \cdot \|_2$. In this space, the wavelet basis functions are defined by Eq. (11). In conjunction with Eq. (11), we also introduce the scaling (for a chosen father wavelet $\varphi(\cdot)$),

$$\varphi_{m,\tau}(x) = \frac{1}{\sqrt{2^m}} \varphi\left(\frac{x - \tau 2^m}{2^m}\right), \quad (m, \tau) \in \mathbb{Z}^2. \tag{15}$$

Together the mother $\psi(\cdot)$ and father $\varphi(\cdot)$ wavelets provide- (i) approximation spaces $V_m \subset L^2(\mathbb{R})$ and (ii) detail spaces $W_m \subset L^2(\mathbb{R})$ at each scale of the m-level multiwavelet decomposition. The subspaces at each level-m are constructed as,

$$V_m = \text{span}\left\{\phi_{m,\tau}\right\}_{\tau \in \mathbb{Z}}, \quad W_m = \text{span}\left\{\psi_{m,\tau}\right\}_{\tau \in \mathbb{Z}}, \quad V_m \perp W_m. \tag{16}$$

Therefore if the wavelet function $\psi(\cdot)$ and the scaling function $\varphi(\cdot)$ are chosen appropriately, we have a multi-resolution decomposition of the spatial field, for which the following properties hold,

$$\{0\} \subset \dots V_1 \subset V_0 \subset L^2(\mathbb{R}), \quad V_{j-1} = V_j \oplus W_j, \tag{17}$$

where $\oplus$ denotes the usual orthogonal sum. If $\mathcal{W}(\cdot) \in L^2(\mathbb{R})$ denotes the wavelet decomposition then at the coarsest scale $m = m_0$, $\mathcal{W}(\cdot)$ admits the following expansion,

$$\mathcal{W}(x) = \sum_{\tau=-\infty}^{\infty} \left\langle \mathcal{W}(\cdot), \phi_{m_0,\tau} \right\rangle \phi_{m_0,\tau}(x) + \sum_{\tau=-\infty}^{\infty} \sum_{j=-\infty}^{j_0} \left\langle \mathcal{W}(\cdot), \psi_{\tau,m} \right\rangle \psi_{\tau,m}(x), \tag{18}$$

The above expansion can similarly be extended to any ℓ-dimensional wavelet basis functions V_m^ℓ and W_m^ℓ. However, we restrict ourselves to the two-dimensional case only. Towards this, at each level of decomposition, we define the two-dimensional approximation and detail coefficients V_m^2 and W_m^2, as,

$$V_m^2 = V_m \otimes V_m, \quad \text{and,} \quad V_{m-1}^2 = V_m^2 \oplus W_m^2, \tag{19}$$

where $\otimes$ denotes the usual tensor product. Using the above results, the equation for the detail coefficients can be derived as,

$$W_j^2 = \left(V_j \otimes W_j\right) \oplus \left(W_j \otimes V_j\right) \oplus \left(W_j \otimes W_j\right). \tag{20}$$

In the above expression, $\left(V_j \otimes W_j\right)$, $\left(W_j \otimes V_j\right)$, and $\left(W_j \otimes W_j\right)$ represents the two-dimensional horizontal, vertical and diagonal wavelet coefficients. Thereafter, the detailed wavelet coefficients are obtained from the mother and father wavelets as,

$$\psi_{horizontal}(x, y) = \varphi(x)\psi(y) \tag{21a}$$

$$\psi_{vertical}(x, y) = \psi(x)\varphi(y) \tag{21b}$$

$$\psi_{diagonal}(x, y) = \psi(x)\psi(y), \tag{21c}$$

and the two-dimensional approximate basis functions are obtained as,

$$\varphi_{approximate}(x, y) = \varphi(x)\varphi(y). \tag{22}$$

To find an expansion for $\mathcal{W}(\cdot, \cdot)$, we derive the two-dimensional wavelet basis functions and the two-dimensional scaling basis functions as,

$$\psi_{m,\ell,\tau,\varrho}(x, y) = \frac{1}{\sqrt{2^m}} \psi_{\ell=\{0,1,2\}} \left(\frac{x - \tau 2^m}{2^m}, \frac{y - \varrho 2^m}{2^m} \right), \tag{23a}$$

$$\varphi_{m,\tau,\varrho}(x, y) = \frac{1}{\sqrt{2^m}} \varphi \left(\frac{x - \tau 2^m}{2^m}, \frac{y - \varrho 2^m}{2^m} \right), \tag{23b}$$

where $\tau \in \mathbb{Z}$ and $\varrho \in \mathbb{Z}$ are the shifting parameters. In Fig. 2, the two-dimensional multiwavelet decomposition for a given function $v(x)$ is illustrated. Using these basis we write the expansion of $\mathcal{W}(\cdot, \cdot) \in L^2\left(\mathbb{R}^2\right)$, in similar manner to Eq. (18) as follows,

$$\mathcal{W}(x, y) = \sum_{\tau=-\infty}^{\infty} \sum_{\varrho=-\infty}^{\infty} \left\langle \mathcal{W}(\cdot), \phi_{m_0,\tau,\varrho} \right\rangle \phi_{m_0,\tau}(x, y) + \sum_{m=-\infty}^{m_0} \sum_{\ell=0}^{2} \sum_{\tau=-\infty}^{\infty} \sum_{\varrho=-\infty}^{\infty} \left\langle \mathcal{W}(\cdot), \psi_{m,\ell,\tau,\varrho} \right\rangle \psi_{m,\ell,\tau,\varrho}(x, y). \tag{24}$$

In this setup we define four $(\zeta \times d_v \times d_v)$-valued tensor to learn the approximate function $\varphi(x, y)_{approximate}$, the horizontal function $\psi_{horizontal}$, the vertical function $\psi_{vertical}$ and the diagonal wavelet function $\psi_{diagonal}$. Using these weight tensors, we aim to learn the multi-resolution representation of the parameterization space of the spatial field. The convolution of these weight tensors with wavelet functions is already defined in Eq. (10).

3.2. Relation with fourier neural operator (FNO)

The proposed WNO is closely related to FNO; in fact, we will show that the FNO is a special case of the proposed WNO. To understand the relation, let us consider the generalized wavelet transform that is given as,

$$\mathcal{W}(\alpha, \beta) = \int_D \Gamma(x)\psi_{\alpha,\beta}(x)dx, \tag{25}$$

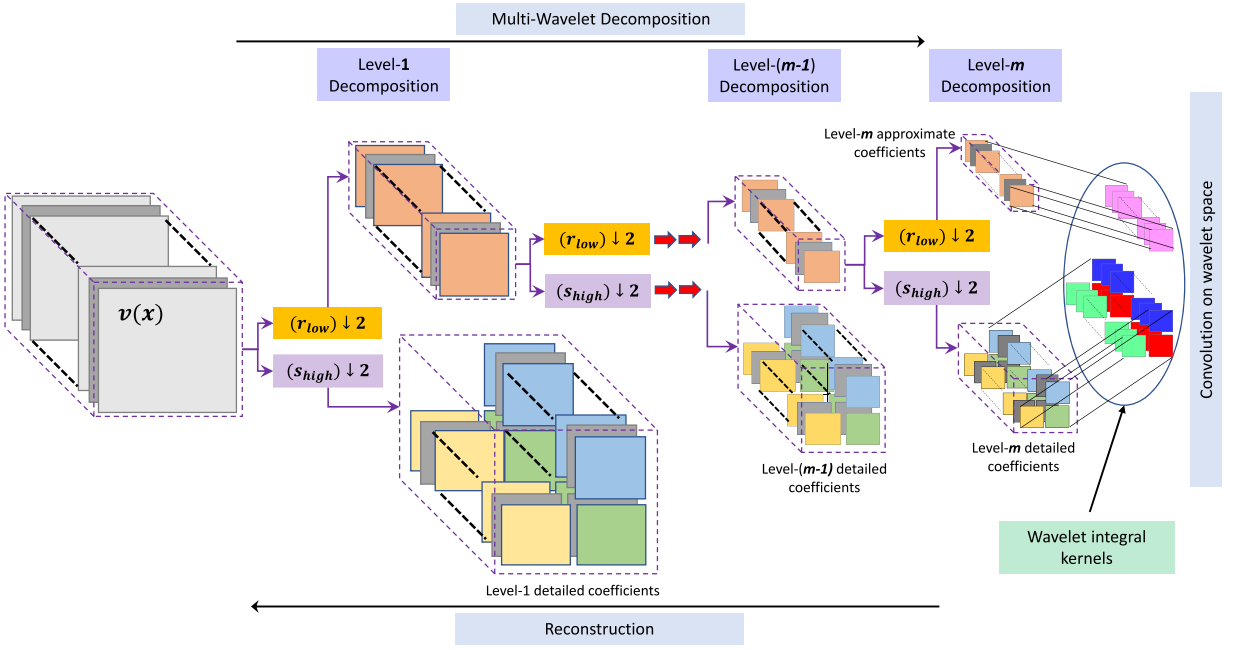

Fig. 2. Construction of the parametric space using multiwavelet decomposition. When working with CNN, the hierarchical wavelet decomposition in two dimensions can be represented by using approximate, horizontal, vertical, and diagonal wavelet coefficients in a binary tree configuration. In the binary tree at each level of decomposition, the approximate and detailed (horizontal, vertical, and diagonal) coefficients are obtained by passing the spatial coordinates through a quadrature mirror filter. At each level, the time resolution is halved, and the frequency resolution is doubled, i.e., at each decomposition, the output has half the frequency band of the input. At each resolution, the red block represents the approximate coefficient, the top right blue block corresponds to horizontal wavelets, the yellow bottom left block corresponds to vertical wavelets, and the green bottom right block corresponds to the diagonal wavelets. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

where $\psi_{\alpha,\beta}(x) \in \mathcal{L}^2(\mathbb{R})$ is the orthonormal wavelet and α, β, $\Gamma(x)$ denote the same as above. In wavelet transform, the aim is to transform the continuous function $\Gamma(x)$ into a continuous function of scale and translational variables α and β. The transformation is generally done by projecting $\Gamma(x)$ into different wavelet coefficients using the following form of $\psi_{\alpha,\beta}(x)$,

$$\psi_{\alpha,\beta}(x) = \frac{1}{\sqrt{\alpha}}\psi\left(\frac{x-\beta}{\alpha}\right) \tag{26}$$

where $\psi_{\alpha,\beta}(x)$ will decide the nature of waveforms. Based on the requirements, if one chooses to project the function $\Gamma(x)$ in terms of its frequency components, the translation parameter can be dropped, and the orthonormal basis wavelet can be replaced using the triangular/harmonic basis functions. If one chose to consider only the sine and cosine waves as the orthonormal wavelets, i.e., to replace the orthonormal wavelet $\psi_{\alpha,\beta}(x)$ by $e^{-2i\pi\langle x,\omega\rangle}$, Eq. (25) will result in a standard Fourier transform. If we denote $\mathcal{F}(\omega)$ as the Fourier transform and $\mathcal{F}^{-1}(x)$ to be its inverse operation, then for the function $\Gamma(x)$, we have,

$$(\mathcal{F}\Gamma)_j(\omega) = \int_D f_j(x)e^{-2i\pi\langle x,\omega\rangle}dx, \quad \& \qquad \left(\mathcal{F}^{-1}\Gamma\right)_j(x) = \int_D f_j(\omega)e^{2i\pi\langle x,\omega\rangle}\mathrm{d}\omega; \quad j = 1,\ldots,d_v \tag{27}$$

Since the Fourier transform can translate between the convolution of multiple functions, Eq. (9) can be written in Fourier space as,

$$\left(\omega(a;\phi)v_j\right)(x) = \mathcal{F}^{-1}\left(\mathcal{F}(k_\phi)\cdot\mathcal{F}(v_j)\right)(x); \quad x \in D. \tag{28}$$

Therefore, the kernel k_ϕ can be directly parameterized in Fourier space by choosing a sufficient number of Fourier modes $\omega_{\max} \in D$.

Table 1
Dataset size for each problem unless otherwise stated. WNO architecture for each problem, unless otherwise stated.

Examples	Number of data		Wavelet	m	Network dimensions				$g(.)$
	Training	Testing			FNN1	FNN2	CNN	WNO	
Burgers[4.1]	1000	100	db6	8	64	128	4	4	GeLU
Darcy (rectangular)[4.2]	1000	100	db4	4	64	128	4	4	GeLU
Darcy (triangular)[4.3]	1900	100	db6	3	64	128	4	4	GeLU
Navier–Stokes[4.4]	1000	100	db4	3	26	128	4	4	GeLU
Allen–Cahn[4.5]	1400	100	db4	1	64	128	4	4	GeLU
Advection[4.6]	900	100	db6	3	96	128	4	4	GeLU

3.3. Note on the complexity of the wavelet integral layer

Without loss of generality, we restrict the complexity analysis to a 1D problem. Recall that n_D is the number of discretization points in the domain D. In the discrete wavelet decomposing, the input signals are decomposed simultaneously using a low-pass filter $r(n_D)$ and a high-pass filter $s(n_D)$. In the case of Daubechies wavelets, the filters $r(n_D)$ and $s(n_D)$ have a constant length, each of which performs the convolutions $(x * r)(n_D)$ and $(x * s)(n_D)$, thereby has the complexity $\mathcal{O}(n_D)$. In the single tree format, the discrete wavelet decomposition uses these two filters for recursive multi-level decomposition of the output of the branch convolved with $r(n_D)$. This recursive filterbank implementation has the overall complexity of $\mathcal{O}(n_D)$. Decomposition of the input using wavelet with a M level results in a signal of length $n_D/2^M$. Convolution of decomposed coefficients and weight tensor inside the wavelet integral layer takes $\mathcal{O}(n_D/2^M)$ time. The convolution in the CNN with kernel size takes $\mathcal{O}(n_D)$ time. Therefore, the majority of the computational demand arises from the convolution in the wavelet integral. In general, there is a trade-off between the number of decompositions and the integral kernel size, and their relation can be represented as a reciprocal relation, i.e., the more the decomposition level less the integral kernel size and vice-versa. However, we observed that the level of decomposition plays a significant role in the speed-up of the proposed neural operator compared to the integral kernel's size. For discrete wavelet decomposition, we make use of the PyTorch-wavelet and pywavelet libraries, more details on which can be found in Ref. [48,49].

4. Operator learning of example problems through WNO

Numerical case studies on a wide variety of systems representing various classes of problems in fluid dynamics, gas dynamics, traffic flow, atmospheric science, and phase-field modeling are considered in this section. In all the cases, the performance of the network architectures is compared using the L^2 relative error of the predictions. We compare the results obtained with four popular neural operators and their variants: (a) DeepONet [25,26], (b) Graph Neural Operator (GNO), (c) Fourier Neural Operator (FNO) [31], and (d) Multiwavelet Transformation operator (MWT) [43]. The overall WNO architecture consists of four layers of WNO, each of which is activated by the GeLU activation function [50]. The mother wavelet is chosen from the Daubechies family [47,51]. For all the cases, the sizes of the individual datasets, the wavelet, the level of the wavelet tree, and the number of layers in the network against each problem are listed in the Table 1. For optimization of the parameters of the WNO, the ADAM optimizer with an initial learning rate of 0.001 and a weight decay of 10^{-6} is utilized. During the optimization, decay in the learning rate of each parameter group is introduced at every 50 epochs at a rate of 0.75. The total number of epochs used for training the WNO architecture against the undertaken examples are given in Table 2. The batch size in the data loader varies according to the underlying system between 10–25. The numerical experiments are performed on a single RTX A2000 6 GB GPU. A summary of results for different example problems solved is shown in Table 3. The summary of the learning process of the proposed wavelet neural operator is illustrated in Fig. 3. It is observed that the error for the proposed WNO varies in the range of 0.21%–1.75%, with WNO yielding the best results (lowest error) in four out of the six cases. In the remaining two problems also, the results obtained using the proposed WNO are highly accurate (and close to the best results).

Table 2
Number of epochs required to obtain the best prediction results. Note that the number of epochs here for DeepONet and FNO, except the Allen–Cahn example, are reported from their respective original papers.

Architectures	Number of epochs					
	Burgers'	Darcy flow	Navier–Stokes	AllenCahn	Darcy (Notch)	Advection
DeepONet	500 000	100 000	100 000	100 000	20 000	250 000
POD-DeepONet	500 000	100 000	100 000	100 000	20 000	250 000
FNO	500	500	500	500	500	500
MWT	500	500	500	500	500	500
WNO	500	500	500	500	500	500

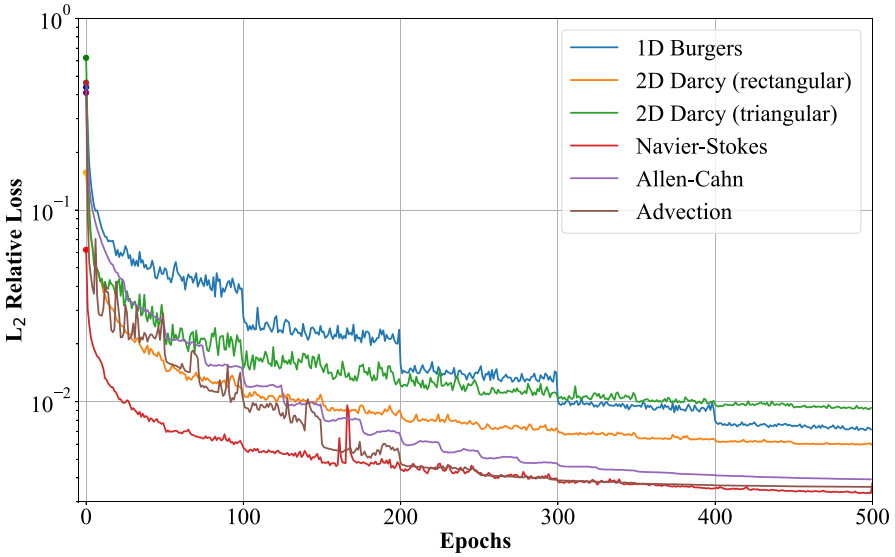

Fig. 3. Learning process of the wavelet neural operator.

4.1. 1D burgers equation

In the first example, we consider the 1D Burgers equation. The 1D Burgers equation is popularly used to model 1D flows in fluid mechanics, gas dynamics, and traffic flow. With periodic boundary conditions, the 1D Burgers equation is given by the form,

$$\begin{aligned} \partial_t u(x,t) + \frac{1}{2}\partial_x u^2(x,t) &= \nu \partial_{xx} u(x,t), \quad && x \in (0,1), t \in (0,1] \\ u(x=0,t) &= u(x=1,t), && x \in (0,1), t \in (0,1] \\ u(x,0) &= u_0(x), && x \in (0,1). \end{aligned} \tag{29}$$

where, $\nu \in \mathbb{R}^{+*}$ is the viscosity of the flow and $u_0(x) \in L^2_{per}((0,1);\mathbb{R})$ is the initial condition. The initial condition $u_0(x)$ is generated using a Gaussian random field as $u_0(x) \sim \mathcal{N}\left(0, 625(-\Delta + 25I)^{-2}\right)$. A periodic boundary condition of the form, $u(x-\pi,t) = u(x+\pi,t); x \in (0,1), t \in (0,1]$ is considered. The aim here is to learn the operator, $\mathcal{D} : u_0(x) \mapsto u(x,1)$. The datasets are taken from the Ref. [31], where $\nu = 0.1$ and a spatial resolution of 1024 is considered. **Results** : the prediction results, along with their accuracy are illustrated in Fig. 4 and Table 3. From the error values in Table 3, it is evident that the MWT obtains the lowest error followed by FNO. The proposed WNO has a slightly higher error than FNO but is lesser than DeepONet and POD-DeepONet. Although from the quantitative standpoint, the differences in the error in the predictions using WNO are slightly higher than MWT and

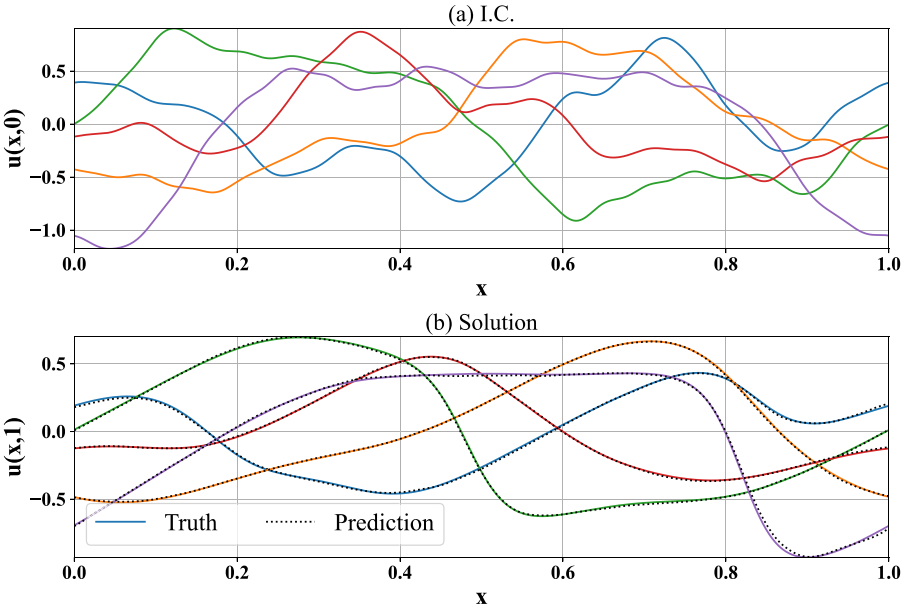

Fig. 4. 1D Burgers' equation with periodic boundary condition on a 1024 spatial resolution. (a) The initial boundary conditions at $t = 0$. (b) The solution to the 1D Burgers flow equation at $t = 1$. The aim was to learn the Burgers' operator that maps the input function, i.e., the initial condition to the final solution at some time T. For given input functions, the predictions are shown above, and it can be seen that the predictions of the proposed WNO architecture match the true solution almost exactly. This means the proposed WNO can solve 1D flow problems.

FNO, from Fig. 4, the differences in the predictions and true results are not discerned. This indicates that, although not the best, the proposed WNO performs reasonably well in this problem.

Case study: Burgers equation with discontinuity in the solution field: For small values of the viscosity parameter ν, the Burgers equation can lead to the formation of sharp jumps in the velocity field. Modeling such sharp jumps in the solution field is very difficult, primarily when the governing physics is not known a priori [21]. In this example, we aim to learn the operator which can model such jumps for a given initial condition. The sharp jump in the velocity field can be modeled using the Burgers equation for the following conditions,

$$\begin{aligned} \partial_t u(x,t) + u(x,t)\partial_x u(x,t) &= \frac{0.01}{\pi}\partial_{xx} u(x,t), && x \in [-1,1], t \in [0,1] \\ u(x=-1,t) = u(x=1,t) &= 0, && x \in [-1,1], t \in [0,1] \\ u(x,0) &= -sin(\pi x) + \zeta sin(\pi x), && x \in [-1,1]. \end{aligned} \tag{30}$$

where $\zeta \in \mathbb{R}$ is random chosen from [0,0.5]. Previously, the WNO was trained to map from the initial condition to a given time-step; however, in this case, the objective is to learn the operator that maps the velocity fields at the time steps $t \in [0, 10]$ to the velocity fields up to time steps $t \in (10, 50]$, i.e., $u|_{(-1,1)\times[0,10]} \mapsto u|_{(-1,1)\times[10,50]}$. This necessitates retraining the model. We have used a time step of $\Delta t = 0.02$ and a spatial resolution of 512 for generating the training data. The pde solver toolbox in Matlab is used to generate the training data.

Results : the prediction result for one instant of the initial condition is illustrated in Fig. 5. The true and predicted Spatio-temporal solutions are presented in the top two subplots. Visually, the results obtained using the proposed WNO and numerical solver are indistinguishable. For a more detailed assessment, four temporal snapshots at t = 0.2 s, 0.4 s, 0.6 s, and 0.8 s, represented by the black solid line in the top subplots, are also depicted at the bottom of the figure. These detailed assessments reveal that the proposed WNO could accurately model the time evolution of the notorious behavior of the Burgers equation at all four temporal snapshots. This illustrates the accuracy of the proposed WNO and its capability to capture sharp jumps in the solution field.

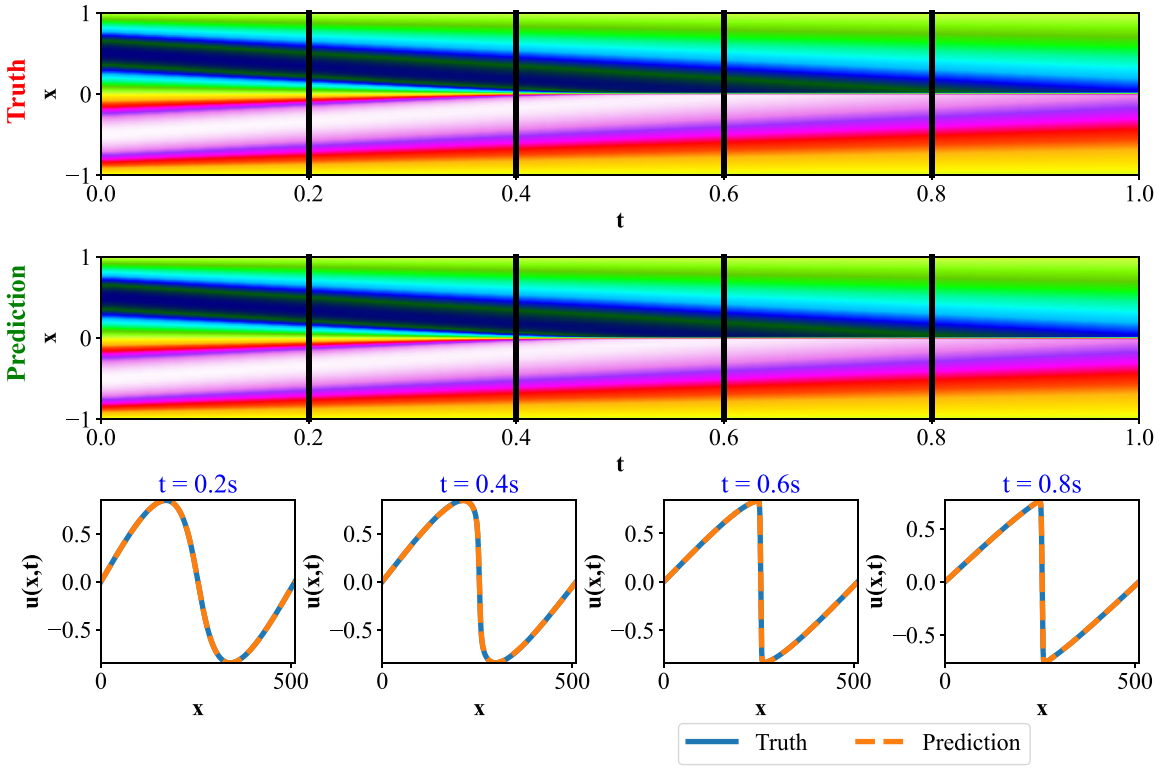

Fig. 5. 1D Burgers' equation with discontinuity in the velocity field. The aim in this example is to learn the operator which predicts the time-marching solutions to Burgers' equation for the given initial condition. The truth and prediction results are shown above. Four temporal snapshots are also portrayed. The mean L^2 relative testing loss is 0.23%. The predictions show that the proposed WNO can model sharp jumps in the solution fields with very high accuracy.

Table 3
Mean L^2 relative error between the true and predicted results.

PDE	Network architectures						
	GNO	DeepONet	FNO	MWT	POD-DeepONet[a]	dgFNO+[b]	WNO
Burgers' equation	≈6.15%	≈2.15%	≈1.60%	≈0.19%	≈1.94%	–	≈1.75%
Darcy (rectangular)	≈3.46%	≈2.98%	≈1.08%	≈0.89%	≈2.38%	–	≈0.84%
Darcy (triangular)	–	≈2.64%	–	≈0.87%	≈1.00%	≈7.82%	≈0.77%
Navier–Stokes	–	≈1.78%	≈1.28%	≈0.63%	≈1.36%	–	≈0.31%
Allen-Cahn	–	≈17.7%	≈0.93%	≈4.84%	–	–	≈0.21%
Wave-advection	–	≈0.32%	≈47.7%	≈10.22%	≈0.40%	≈0.60%	≈0.62%

[a]POD-DeepONet is the modified version of DeepONet, proposed in Ref. [26].
[b]Note that in complex geometric conditions, FNO does not work. Thus we use dgFNO+, which is a modified version of FNO (Ref. [26]).

4.2. 2D darcy flow equation in a rectangular domain

The 2D Darcy flow equation is widely used in modeling flow, whether liquid or gas, through a porous medium. In the absence of the time component, it represents a second-order nonlinear elliptic PDE, and the form in a 2D domain is given as,

$$\begin{aligned} -\nabla \cdot (a(x,y)\nabla u(x,y)) &= f(x,y), \quad x,y \in (0,\mathbb{R}) \\ u(x,y) &= u_0(x,y), \quad x,y \in \partial(0,\mathbb{R}) \end{aligned} \tag{31}$$

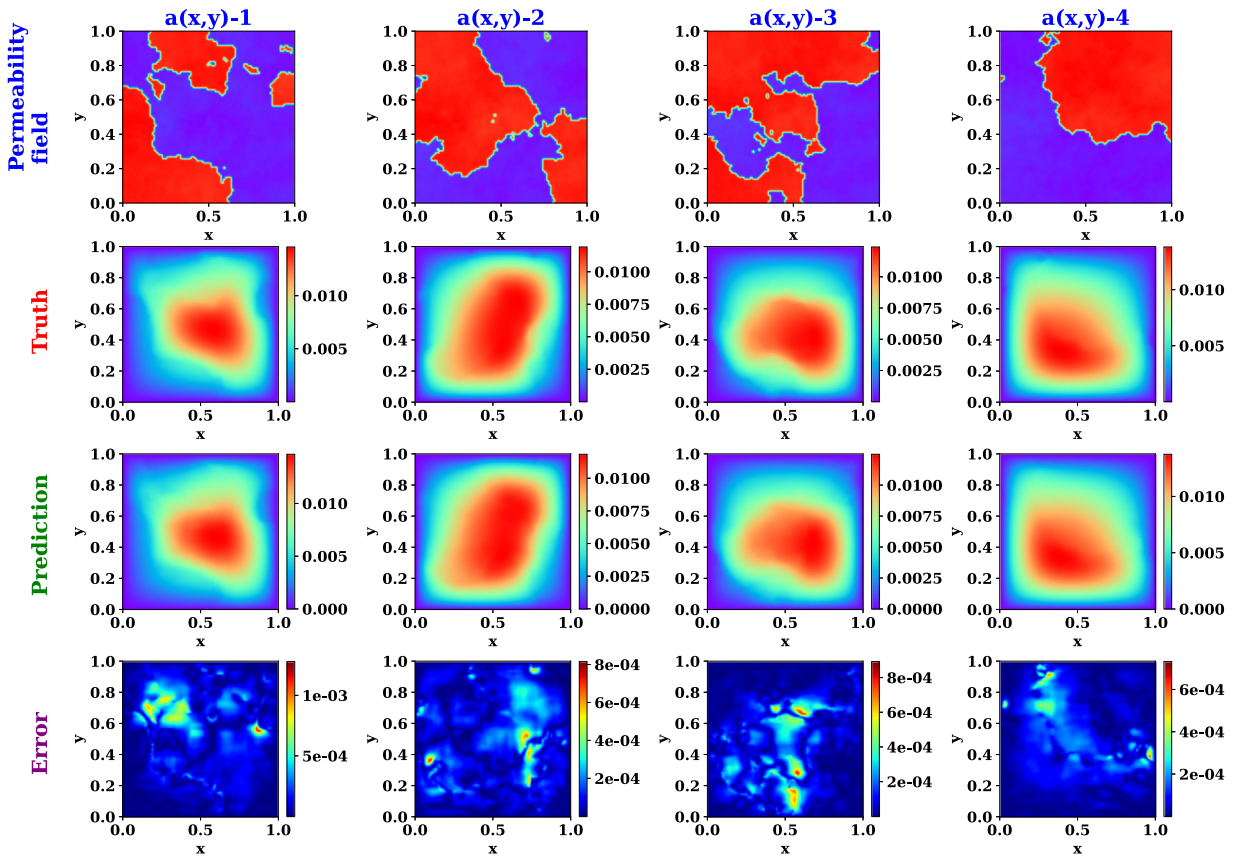

Fig. 6. Darcy flow in the rectangular domain with spatial resolution of 85×85. The aim here is to learn the operator of the Darcy equation that maps the given initial condition to the corresponding solution. The output pressure fields are obtained using the proposed WNO for different input permeability conditions. The prediction results show nearly an exact match with the true solution. This indicates that the proposed WNO can learn operators of 2-dimensional PDEs.

where $u(x, y) = u_0(x, y)$ is the initial condition, $a(x, y)$ is the permeability field, $u(x, y)$ is the pressure, $\nabla u(x, y)$ is the pressure gradient, and $f(x, y)$ is a source function. In this problem definition, the Darcy flow is defined on a unit box domain with $x \times y \in (0, 1)^2$ with $u_0(x, y) = 0$ (zero Dirichlet boundary condition). The source $f(x, y) = 1$ is considered. The aim is to learn the operator, $\mathcal{D} : a(x, y) \mapsto u(x, y)$. The dataset for training the network architectures is taken from Ref. [31]. During training, a spatial resolution of 85×85 is used. **Results** : the predictions for the 2D Darcy flow in the rectangular domain are plotted in Fig. 6, while the associated errors are given in Table 3. In this case, it is observed that the mean prediction error for WNO is the lowest among all the considered methods. It is further evident in Fig. 6, where the differences between true and predicted solutions cannot be discerned by the naked eye.

4.3. 2D darcy flow equation with a notch in triangular domain

This example emulates the previous 2D Darcy problem but with a more complex boundary condition. The setup and datasets for this case are taken from the Ref. [26], where the boundary conditions for the triangular domain are generated using the following Gaussian process (GP),

$$\begin{aligned} u(x) &\sim \mathrm{GP}(0, \mathcal{K}(x, x')), \\ \mathcal{K}(x, x') &= \exp\left(-\frac{(x - x')^2}{2l^2}\right), \quad l = 0.2, \quad x, x' \in [0, 1] \end{aligned} \tag{32}$$

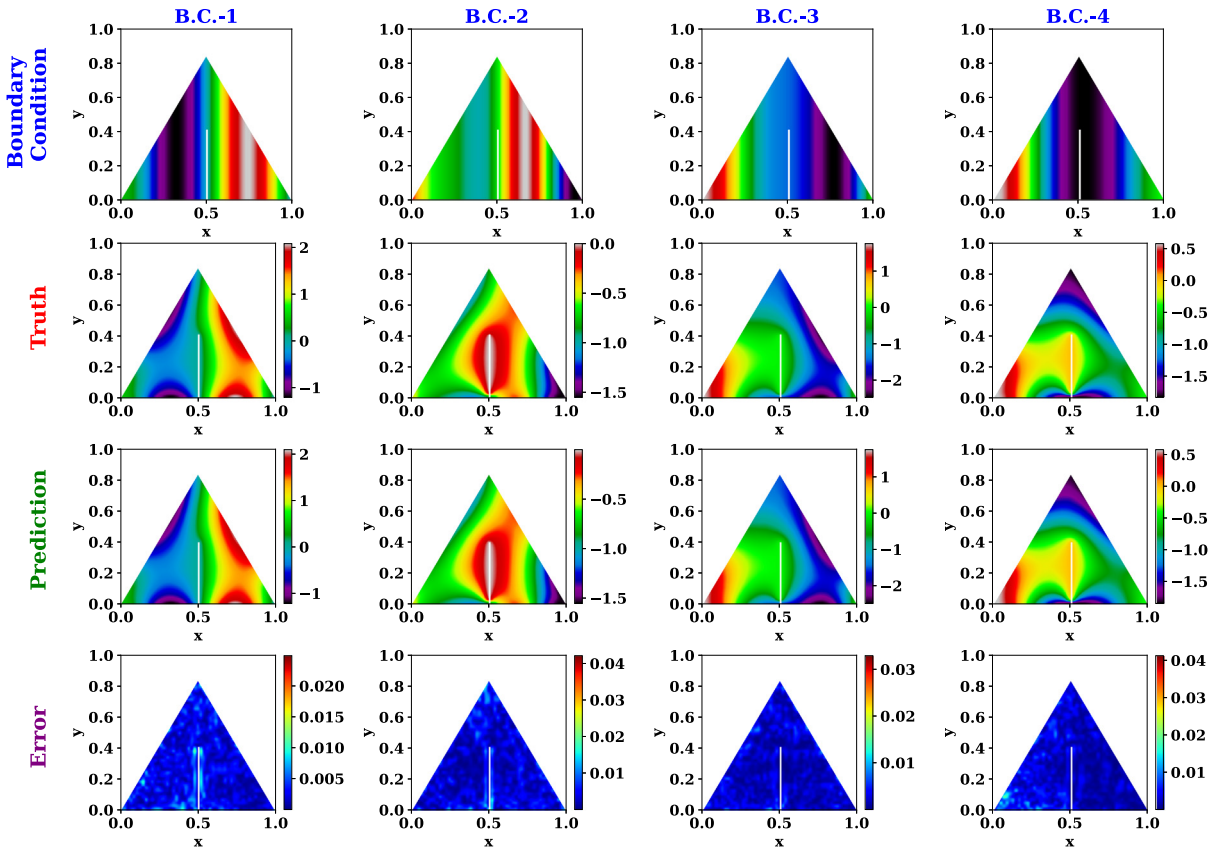

Fig. 7. Darcy flow in a triangular domain with a notch on a spatial resolution of 51×51. The aim of this problem is to learn the Darcy operator in complex geometry and map the arbitrary input permeability conditions to pressure fields. The pressure fields obtained using the proposed WNO for different boundary conditions are shown in the figure. The L^2 relative errors of the predictions in the pressure field for all the cases are also shown against the respective plots. The colorbars represent the amount of errors present in the predictions. While the color distribution varies differently in different predictions, it can be observed from the colorbars that the errors corresponding to each of the predictions are negligible. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In addition to the triangular geometry, a notch is introduced in the flow medium. The permeability field $a(x, y)$ and the forcing function $f(x, y)$ are set as 0.1 and -1, respectively. The aim here is to learn the operator responsible for the mapping of the boundary conditions to the pressure field in the entire domain, represented as,

$$\mathcal{D} : u(x, y)|_{\partial\omega} \mapsto u(x, y). \tag{33}$$

Results : the testing results with corresponding testing errors are given in Fig. 7 and Table 3, respectively. From the error values in the table, it can be observed that our proposed WNO has the lowest prediction error, which is more than threefold lower than DeepONet and tenfold less than FNO. The errors between the truth and prediction can also be observed in Fig. 7, which shows that the WNO can map the given boundary conditions to the permeability filed correctly. Upon careful observation, it can further be noticed that the predictions are not only correct in the smooth regions but also provides near-perfect match near the slit. This indicates that the proposed WNO can be applied to effectively and accurately learn nonlinear operators in complex geometric conditions.

4.4. 2D time-dependent Navier–Stokes equation

The Navier–Stokes equation is a second-order nonlinear parabolic PDE, and its application can be found in various facets of fluid flow problems such as airflow around the airplane wing, ocean currents, thermodynamics,

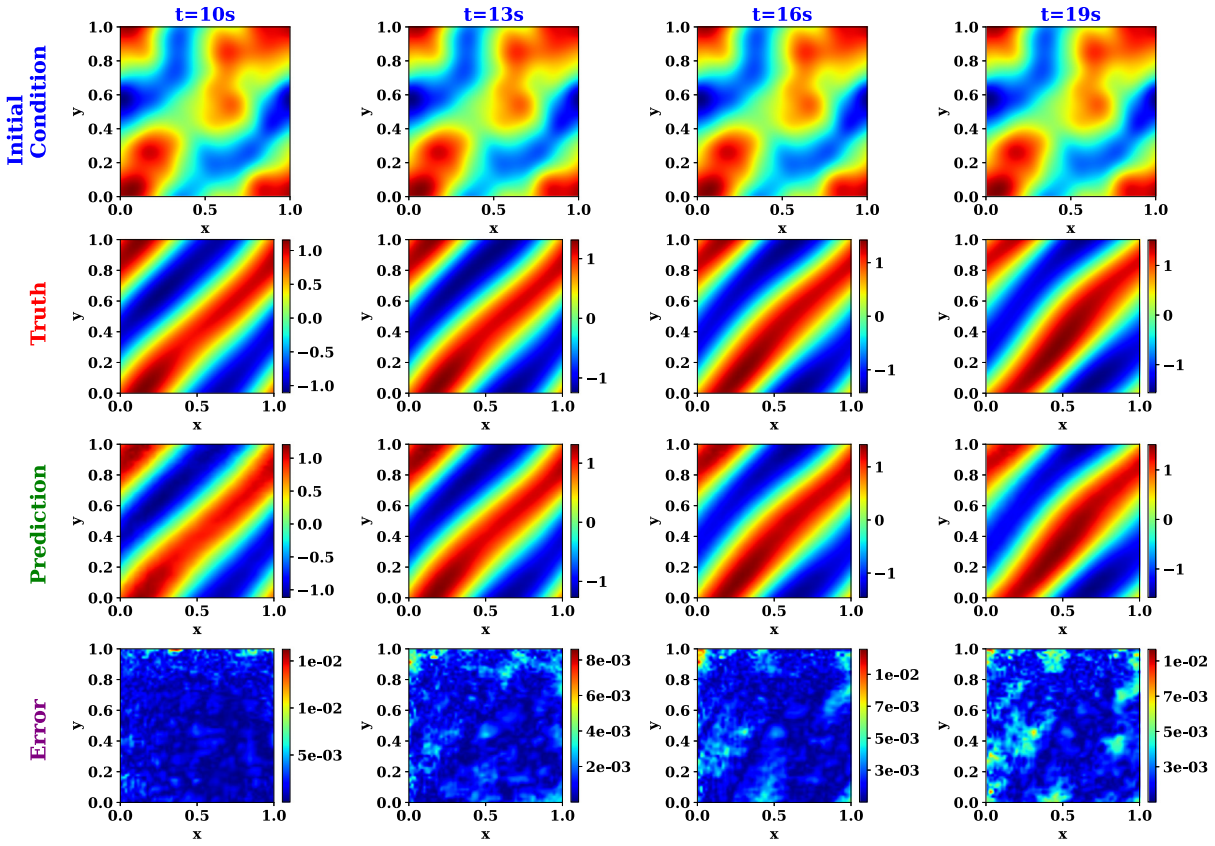

Fig. 8. Navier–Stokes equation with spatial resolution of 64×64. The aim here is to learn the Navier–Stokes operator from a few initial time steps and predict solutions for some later time steps. The training is done with vorticity data for $t \in [0, 10]$, and the predictions for vorticity are made for $t \in [11, 20]$. The predictions for the true results in the above figure cannot be discerned, which means the proposed WNO can learn highly nonlinear operators. Following the results, it can be conjectured that the proposed WNO can learn operators of 2D time-dependent PDEs very effectively.

etc. The 2D incompressible Navier–Stokes equation in the vorticity–velocity form is given by the equations,

$$\begin{aligned} \partial_t \omega(x, y, t) + u(x, y, t) \cdot \nabla \omega(x, y, t) &= \nu \Delta \omega(x, y, t) + f(x, y), \quad && x, y \in (0, 1), t \in (0, T] \\ \nabla \cdot u(x, y, t) &= 0, && x, y \in (0, 1), t \in [0, T] \\ \omega(x, y, 0) &= \omega_0(x, y), && x, y \in (0, 1) \end{aligned} \tag{34}$$

where $\nu \in \mathbb{R}$ and $f(x, y)$ are the viscosity of the fluid and source function. The terms $u(x, y, t)$ and $\omega(x, y, t)$ are the velocity and vorticity field of the fluid, respectively. The initial vorticity field is taken as $\omega(x, y, 0) = \omega_0(x, y)$; $x, y \in (0, 1)$. The viscosity is taken as $\nu = 10^{-3}$. The aim here is to learn the operator which maps the vorticity fields at the time steps $t \in [0, 10]$ to the vorticity fields up to time steps $t \in (10, 20]$, i.e., $\omega|_{(0,1)^2 \times [0,10]} \mapsto \omega|_{(0,1)^2 \times [10,20]}$. The initial vorticity $\omega_0(x, y)$ is simulated from a Gaussian random field as $\omega_0(x, y) = \mathcal{N}\left(0, 7^{3/2}(-\Delta + 49I)^{-2.5}\right)$. The source function is taken fixed as $f(x, y) = 0.1\,(\sin(2\pi\,(x + y)) + \cos(2\pi\,(x + y)))$. The Crank–Nicolson scheme with a time-step of 10^{-4} is used for computing the solution. For training the network, the data are generated by stacking the solution at every $t = 1$ s. For this example, the datasets for initial vorticity $\omega_0(x, y)$ and the solutions $\omega_{1:20}(x, y)$ at $t \in [0, 20]$ are taken from Ref. [31]. For both training and testing, the spatial resolution of the vorticity fields is fixed at resolutions 64×64. **Results** : the prediction results for vorticity field at $t \in \{10, 20\}$ are given in Fig. 8, along with the mean prediction errors in Table 3. The results show that the prediction results of the proposed WNO have the lowest error and emulate the true solutions almost accurately.

4.5. 2D Allen-Cahn equation

The Allen–Cahn equation is a reaction–diffusion equation and is often used in the context of chemical reactions and modeling of phase separation in the multi-component alloy. The Allen–Cahn equation in 2-dimensional space is given as,

$$\begin{aligned} \partial_t u(x,y,t) &= \epsilon \Delta u(x,y,t) + u(x,y,t) - u(x,y,t)^3, && x,y \in (0,3), t \in [0,20] \\ u(x=0,y,t) &= u(x=3,y,t), && y \in (0,3), t \in [0,20] \\ u(x,y=0,t) &= u(x,y=3,t), && x \in (0,3), t \in [0,20] \\ u(x,y,0) &= u_0(x,y) && x,y \in (0,3) \end{aligned} \tag{35}$$

where $\epsilon \in \mathbb{R}^{+*}$ is a real positive constant and responsible for the amount of diffusion. The problem is defined on a periodic boundary with $\epsilon = 1 \times 10^{-3}$, and the initial condition is simulated from the Gaussian Random Field using the following kernel,

$$\mathcal{K}(x,y) = \tau^{(\alpha-1)} \left(\pi^2(x^2+y^2) + \tau^2\right)^{\frac{\alpha}{2}}, \tag{36}$$

where the parameters for the kernel are taken as $\tau = 15$ and $\alpha = 1$. The aim here is to learn the operator $\mathcal{D} : u_0(x,y) \mapsto u(x,y,t)$. In this case, $t = 20$ s is taken and the solution is obtained on a grid of 43×43. The spectral Galerkin method is used to compute the training and testing data [52]. **Results** : The prediction results with mean prediction error of the proposed WNO for different input functions are given in Fig. 9 and Table 3. The mean errors in Table 3 indicate that the DeepONet and MWT have significant errors. The FNO has shown good performance with ≈1% error; however, our proposed WNO obtains the predictions with the lowest error, which is close to 0.2%. In Fig. 9, it is straightforward to see that the true and prediction results are non-differentiable for all the given input functions. This further illustrates that the proposed WNO can be successfully implemented for any 2D time-dependent highly nonlinear problems.

4.6. 1D time-dependent wave advection equation

The wave advection equation is a hyperbolic PDE and primarily describes the solution of a scalar under some known velocity field. The advection equation with periodic boundary condition is given by the expression,

$$\begin{aligned} \partial_t u(x,t) + v \partial_x u(x,t) &= 0, && x \in (0,1), t \in (0,1) \\ u(x-\pi,t) &= u(x+\pi,t), && x \in (0,1), t \in (0,1). \end{aligned} \tag{37}$$

Here $v \in \mathbb{R}^{+*}$ represents the speed of the flow. The initial condition is chosen as,

$$u(x,0) = h 1_{\{c-\frac{\omega}{2}, c+\frac{\omega}{2}\}} + \sqrt{\max(h^2 - (a(x-c))^2, 0)}. \tag{38}$$

where the variables ω and h represent the width and height of the square wave, respectively, and the wave is centered at $x = c$. The values of $\{c, \omega, h\}$ are randomly chosen from $[0.3, 0.7] \times [0.3, 0.6] \times [1, 2]$. For $v = 1$, the solution to the advection equation is given as $u(x,t) = u_0(x-t)$. A time step of 0.025 and 40 spatial discretization points are used to obtain the solution on a Spatio-temporal resolution of 40×40. Similar to example 4.4, the aim here is to learn the operator mapping, which maps the solution at first 10 time steps to final time $T = 1$. The operator is denoted as $\mathcal{D} : u|_{(0,1)\times[0,0.25]} \mapsto u|_{(0,1)\times(0.25,1]}$. The mapping is achieved by devising an RNN structure using the proposed WNO, where a moving window of 10-time steps is used to obtain the future predictions. The datasets for this example are taken from the Ref. [26].

Results : The prediction results along with their L^2 relative error are given in Fig. 10 and Table 3. Similar to the previous examples, it can be seen in Fig. 10 that the WNO can predict the scalar field for arbitrary input functions. The corresponding errors show that the FNO and MWT suffer the most with ≈47% and ≈10% errors, respectively. The DeepONet yields the lowest error. However, the POD-DeepONet, the modified FNO, and the proposed WNO also obtain relatively small errors (very close to the DeepONet results).

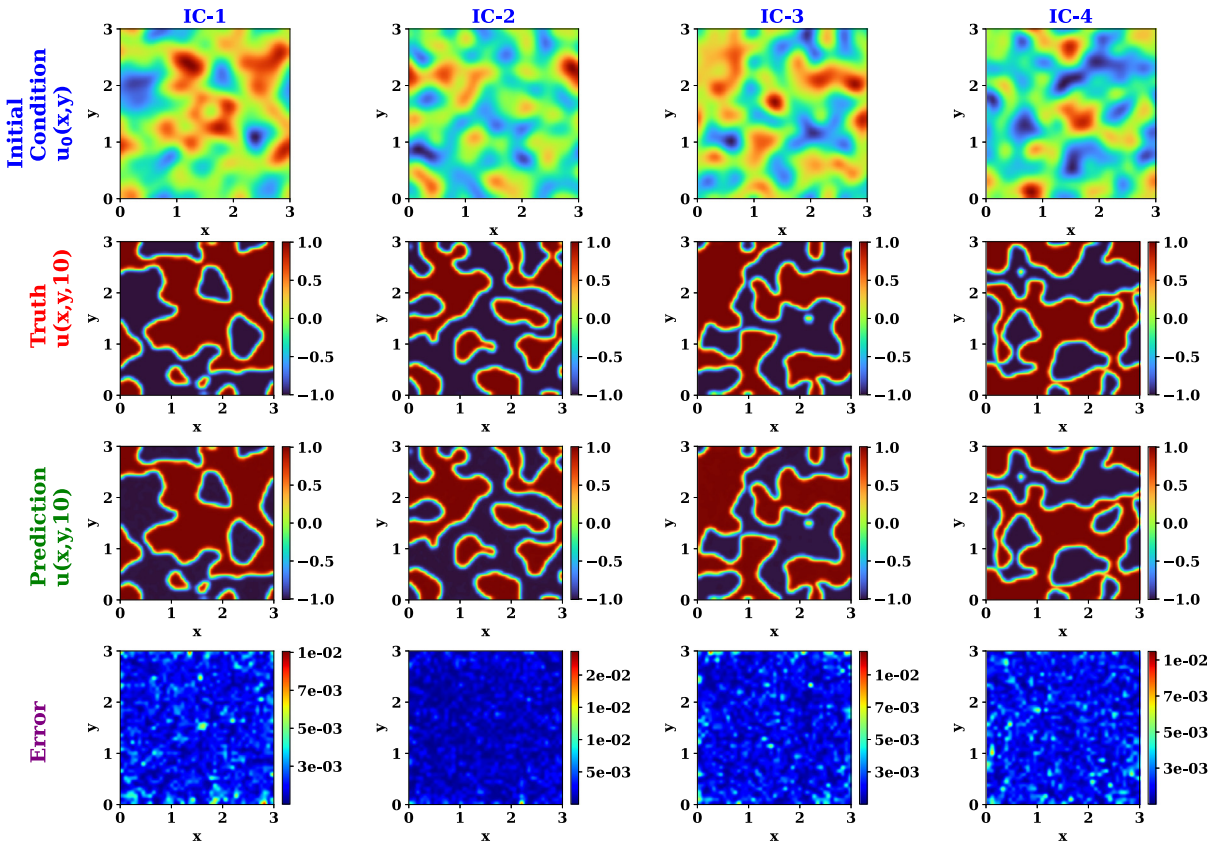

Fig. 9. Allen–Cahn reactor-diffusion equation under periodic boundary condition with a spatial resolution of 43×43. The aim here is to learn the operator of the Allen–Cahn equation and map the given unseen input functions to output solutions. The solutions obtained using the proposed WNO are shown in the figure for the given boundary conditions. The prediction results match almost exactly with the true solution. The error in the prediction results is presented in Table 3.

5. Weather forecast: prediction of the monthly mean 2 m air-temperature

The previous examples have showcased the effectiveness of the proposed WNO framework in accurately solving parametric PDEs. In a general sense, however, the solutions of the PDEs can be imagined as videos of snapshots of the time-evolving process. Since the images and videos can be used to represent the state and motion of an object, the images and videos can be idealized as 2D and 2D time-dependent problems. Thus, the applications of WNO can also be extended for learning dynamic systems from images and videos recorded using a phone, high-speed camera,s or satellites. Further, once the WNO is trained, the prediction stage is extremely efficient compared to traditional methods; this makes WNO an ideal candidate for real-time prediction. In this section, we implement the proposed WNO to develop a digital twin to predict Earth's temperature. In particular, based on historical data, we use WNO for accurate short to medium-range prediction of the hourly and mean monthly 2 m temperature of the air. The datasets are taken from European Centre for Medium-Range Weather Forecasts (ECMWF), which provides publicly available hourly and monthly averaged datasets (so-called ERA5) of various climate parameters. The ERA5 datasets are the fifth generation ECMWF reanalysis of various parameters of the atmosphere that uses data assimilation techniques to combine the numerical weather prediction data with freshly obtained observations to create new and improved predictions of the various atmosphere states. The ERA5 database is a huge collection of hourly and monthly measurements for a large number of atmospheric, ocean-wave, and land-surface parameters. The hourly and monthly datasets are stored both on single levels (atmospheric, ocean-wave, and land surface quantities) and at different pressure levels (upper air fields). The resolution of the datasets are maintained as $0.25° \times 0.25°$ for atmosphere, $0.5° \times 0.5°$ for ocean waves and $0.1° \times 0.1°$ for land. While many variables are present and many

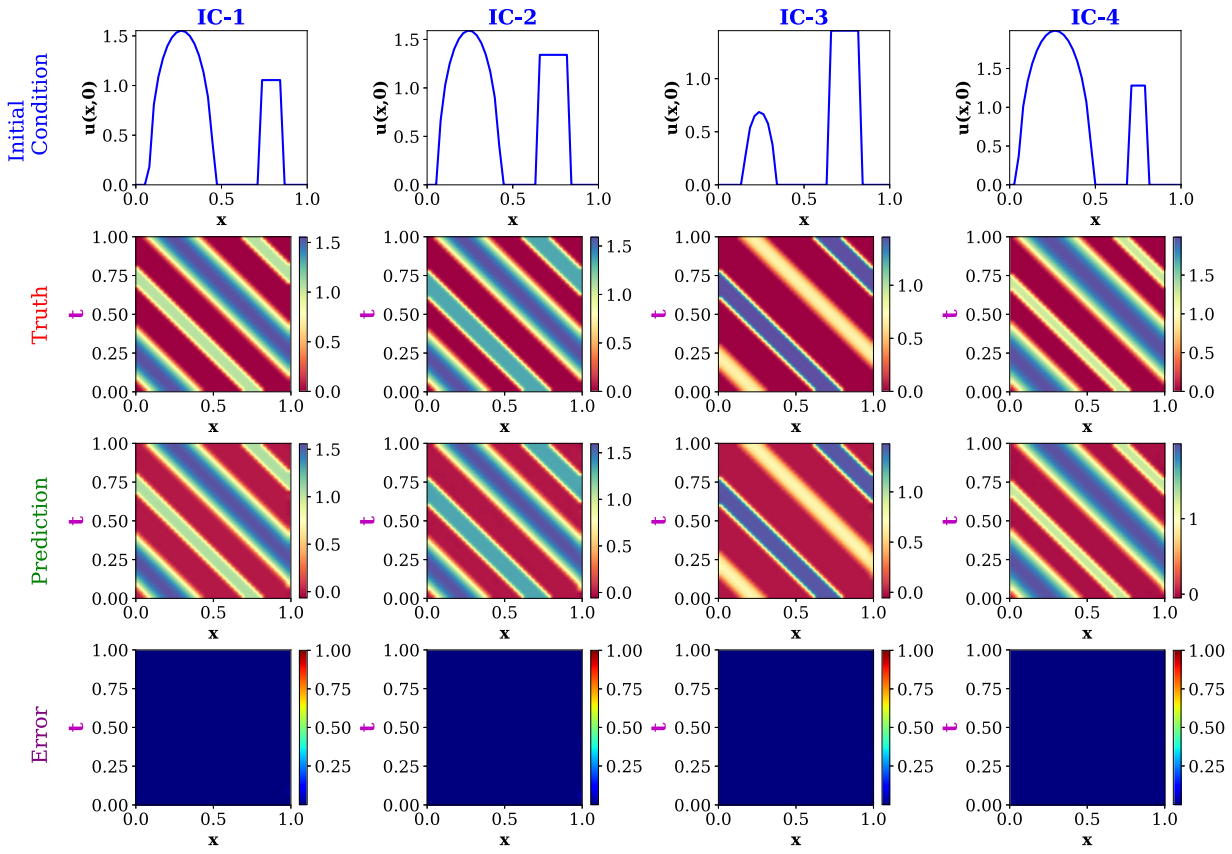

Fig. 10. Wave advection equation with periodic boundary condition on Spatio-temporal resolution of 40×40. The aim of this example is to learn the operator for wave propagation through a continuous medium. The initial conditions in Eq. (38) and the corresponding predictions $u(x,t)$ at $t = 1$ using the proposed WNO, along with their corresponding true solutions, are shown.

combinations are possible for various pressure levels, we only focus on one parameter with a spatial resolution of $2^\circ \times 2^\circ$. Combining all the parameters to develop a complete digital twin for Earth's climate is left as a future research direction.

The framework for short-term prediction of the 2 m temperature is provided in Fig. 11. The predictions are made at 12.00 Universal Time Coordinated (UTC) standard. For this purpose, the daily measurement of 2 m temperature at 12.00 UTC standards from Jan 1st, 2017, to Dec 31st, 2021, are taken from the ERA5 database. We first scaled down the data from $0.25^\circ \times 0.25^\circ$ to $2^\circ \times 2^\circ$ and arranged it into batches of 7, where the 7 denotes the number of days in a week. We then feed the dataset to the proposed WNO. Given the observations of the previous week, the aim here is to predict the 2 m temperatures at 12.00 UTC standards for the next 7 days. For this problem, a recurrent neural network (RNN) structure is formulated within the WNO. The WNO architecture consists of 4 WNO layers where db4 Daubechies wavelet with 2 levels of decomposition is performed. A total number of 276 training samples, 6 testing samples, and 500 epochs with a batch size of 3 are used. The initial learning rate is kept at 0.001 with a step decay constant of 0.75 with a step size of 50. From the previous 7 days database, the WNO performs the prediction for Day 8 and discards the Day 1 information from the previous record by appending newly obtained Day 8 information to the previous database. This process is continued until the complete prediction for the next 7 days is obtained. The prediction results are compared with the resolution-reduced results of the Numerical Weather Prediction (NWP) system from ECMWF. The prediction results with their corresponding error are presented in Fig. 12. At the accuracy of $2^\circ \times 2^\circ$, the short-term weekly prediction results obtained using the proposed WNO is highly accurate. Close observations of the results reveal that the WNO has almost accurately captured the finest details present in the true results at $2^\circ \times 2^\circ$ resolution. Additionally, once trained, the predictions for the entire week took only 3 s; this indicates the computational efficiency of the proposed WNO.

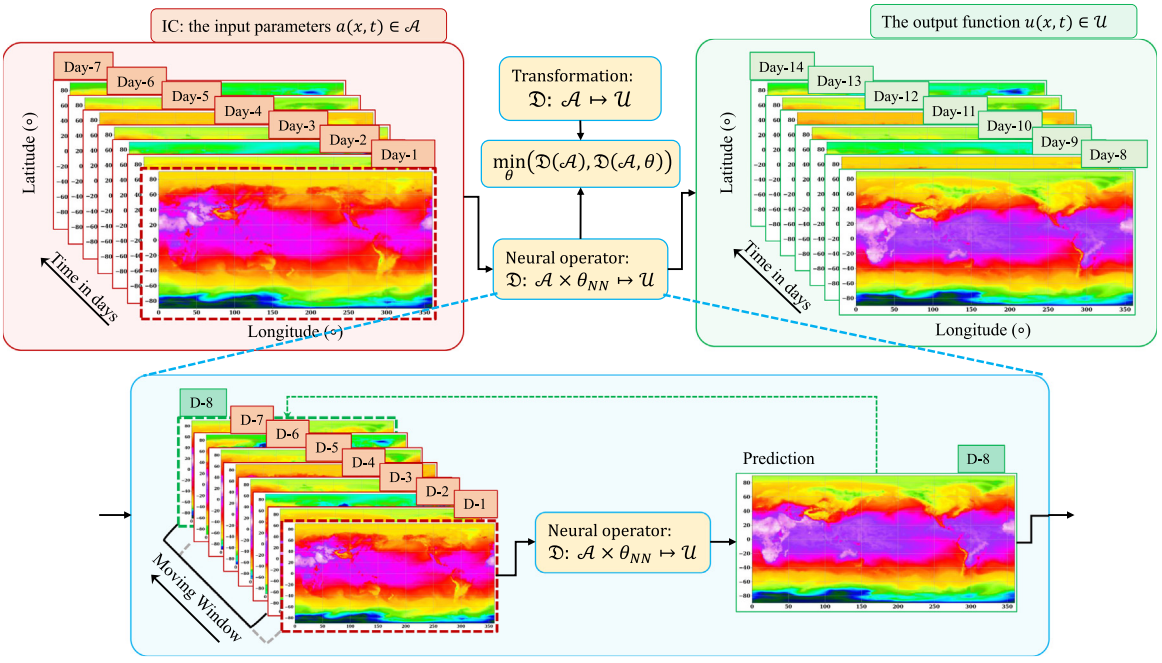

Fig. 11. Methodology for the short-term weekly forecast of 2 m air temperature on a resolution of 2° × 2°. The aim of the short-term forecast is to train the WNO using the previous 7 days' data and then predict the 2 m temperature for the next 7 days. To do this, a recurrent neural network (RNN) structure is employed within the WNO. The RNN framework takes the previous 7 days' data as input and then performs prediction for Day 8. For prediction, the parameters are learned using the proposed WNO architecture. The Day 8 prediction data is then appended to the previous 7-day database. To maintain the recurrent structure, i.e., to transfer the information from Day 8 to future predictions, the information of Day 1 from previous data is discarded, and the process is continued until the next 7-day predictions are made. This results in a highly efficient and accurate.

Next, we focus on the medium-range prediction of monthly mean 2 m temperature at 12.00 UTC standard. For this purpose, the monthly averaged datasets from 1979 to 2022 at 12.00 UTC standard are taken from the ECMWF-ERA5 database. The problem is formulated as a simple 2D problem, and the proposed WNO is trained from the previous datasets to perform the predictions for the next monthly mean 2 m temperature. For training and testing, the total dataset is sliced into 480 and 40 samples, and a total of 500 epochs with batch size 5 is used. The db4 Daubechies wavelet with 5 levels of decomposition is utilized. The initial learning rate is set as 0.001, which is later decayed with a decay constant of 0.75 at a step size of 50. The prediction results with corresponding errors are given in Fig. 13.

From the presented results, it can be conjectured that the proposed WNO can successfully learn the operators of the various parameters of climate and atmosphere, thereby can be used for computationally faster and more accurate predictions of arbitrary high-time scale parameters. We agree that the low-resolution-based prediction is not enough to capture the finest details of climate dynamics accurately. However, in this example, we only take a small subset of all the possible applications and showcase the possibility of the proposed WNO for weather prediction. Given computational facilities, the WNO can also learn the climate dynamics at desired resolutions that will facilitate the predictions of small-scale features in the climate variables.

6. Discussions

This article proposed a new neural operator for learning highly nonlinear operators that arise naturally in day-to-day scientific and engineering problems. Theoretically, the proposed WNO performs a local transformation on the input and then performs a series of updates on the locally transformed inputs. The series of updates is formulated using a combination of nonlinear activation functions and kernel integral operators. The integral kernels are parameterized by the neural network, where the parameterization of the integral kernel is done using a convolution-type architecture; however, instead of performing the convolution directly in the physical space, we propose a

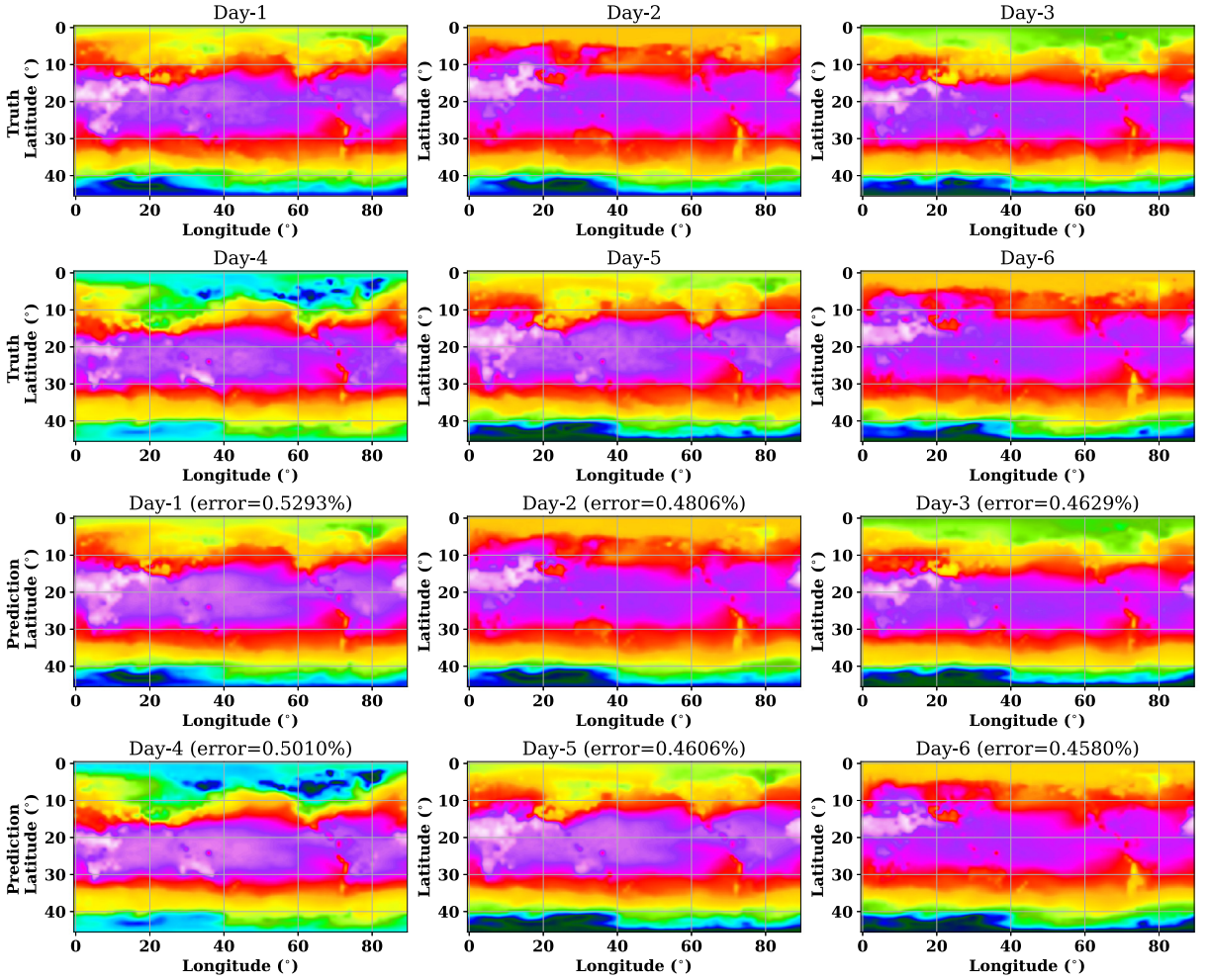

Fig. 12. WNO for the forecast of the weekly 2 m air temperature on a resolution of 2° × 2°. The predictions, along with their corresponding errors for the weekly forecast of 2 m temperature at 12.00 UTC between 10th to 15th April, are shown in the above figure. The errors are obtained with respect to the forecast made by the Integrated Forecast System (IFS) of ECMWF. The predictions using the proposed WNO are obtained with an error of the magnitude of <1%, proving the nearly accurate match with the actual forecast data. Once the training is done, the predictions for the entire week are made within 3 s on the given GPU, which is evidence of the computational advantage of the proposed WNO framework.

wavelet kernel integral operator. Within the wavelet kernel integral operator, the inputs are first transformed into the spectral domain using the wavelet, and after that, the convolutions are performed. Together with nonlinear activation functions and wavelet kernel integral updates, the proposed neural operator can represent any infinite-dimensional function space. While the parameterization of the integral kernel, the proposed WNO exploits the wavelets' spatial and frequency localization property to learn the frequency of change in pixels over the spatial domain. The use of wavelet space for learning of integral kernel thus allows tracking the finest patterns in the inputs; this allows the proposed WNO to learn the highly nonlinear operators even in complex geometric and boundary conditions. In terms of applications, we applied the proposed WNO to various time-independent and time-dependent 1D and 2D PDE examples, exhibiting nonlinear spatial and Spatio-temporal behaviors in flow, wave, and phase-field problems. The results obtained using the proposed WNO are compared with recently developed state-of-the-art neural operators like DeepONet, FNO, and MWT. The results suggest that the proposed WNO can learn the highly nonlinear operators very effectively even when complex geometry like triangular domain and notch is involved. Quantitatively, for the

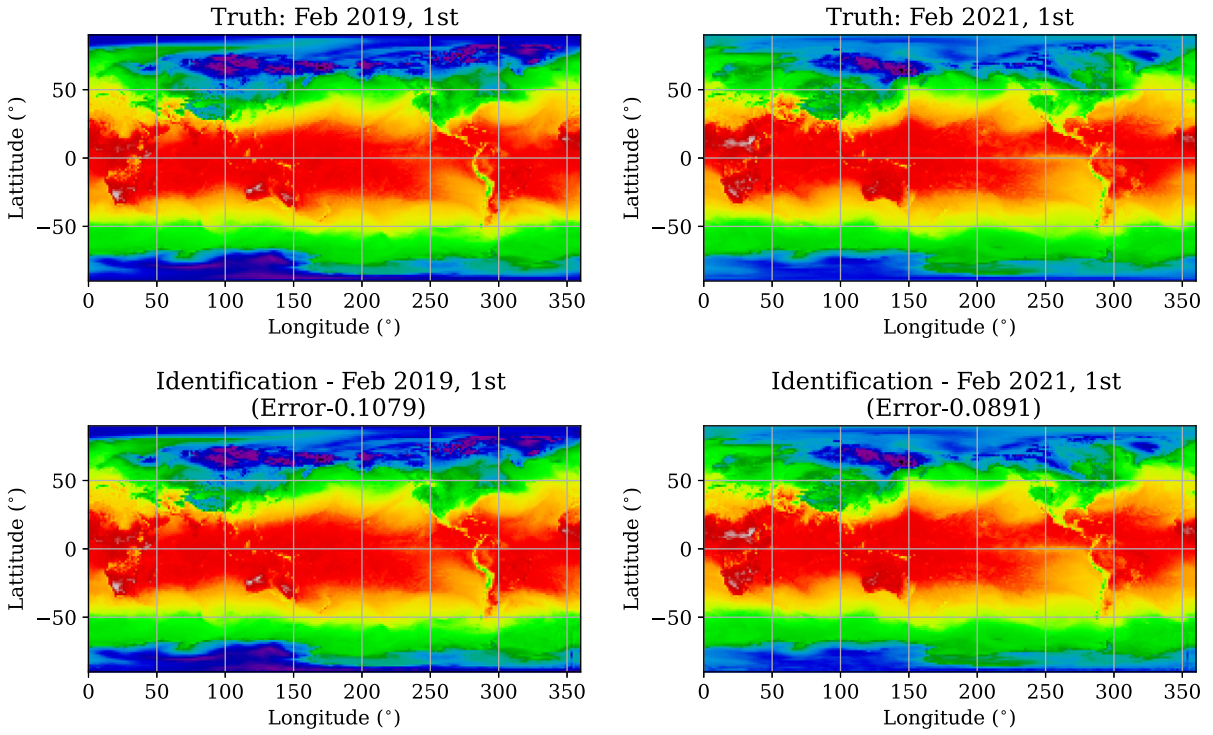

Fig. 13. WNO for the forecast of the monthly averaged 2 m air temperature on a resolution of $2^\circ \times 2^\circ$. The predictions for monthly averaged 2 m temperature for 1st February 2019 are performed. The datasets from Numerical Weather Prediction and the proposed WNO are shown in the above figure. The predictions accurately match the actual data with an error of the magnitude of $\approx$0.1%. On close observation, it can be observed that in addition to the actual temperature gradients, the proposed WNO has also captured the finer details like white spots. Even in small resolutions, the WNO has learned the small-scale features.

six examples presented, the error in results obtained using WNO varies in the range of 0.21%–1.75%, with WNO outperforming the other operator learning approaches in four problems.

The proposed neural operator can learn the mapping between infinite-dimensional function spaces by combining the nonlinear activation functions with wavelet integral layers. We mainly exploit this fact and extend the application of the proposed WNO beyond PDEs to satellite images. In a broader sense, the images and videos can also be contemplated as 2D and time-dependent 2D problems describing the motion or dynamics of an object/process. Thus, the proposed WNO can also be implemented on images and videos captured from cameras and satellites. As a possible application, we demonstrated the proposed WNO for predicting weekly and monthly mean 2 m air temperature at a resolution of $2^\circ \times 2^\circ$. The results are highly encouraging in the given resolution, and the proposed framework produces highly accurate predictions for the weekly and monthly forecasts.

It is worth noting that the proposed WNO is highly efficient and is ideally suited for scenarios where real-time predictions are necessary. In this context, two interesting directions of future work include the application of WNO for the development of digital twins and WNO-based active vibration control. The weather prediction used-case presented in this paper can be regarded as a weak digital twin for Earth's climate. However, the framework in its current form can only predict the Earth's temperature (weekly and mean monthly). An extremely useful extension will be to develop a fully functional digital twin for Earth's climate that can also predict other climate-related parameters such as rainfall precipitation, storms, etc. Such digital twins can assist humankind in its fight against climate change and global warming.

Having postulated above the possible sophisticated real-time applications of the proposed WNO, it will be interesting to perform a case study on the choice of different wavelet families. The proposed WNO constructs the parameterization space by performing a multi-resolution wavelet transform on the spatial coordinates. While in this work, we introduced the idea of learning PDEs using Daubechies wavelets, there are other wavelet families,

too, e.g., Haar, Symlets, Coiflets, Biorthogonal, etc. Therefore, in the future, it will be interesting to note the effect of different wavelets on the performance of the proposed WNO.

Code availability

The source codes for reproducing the results presented in this paper can be accessed at https://github.com/csccm-iitd/WNO.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Souvik Chakraborty reports financial support was provided by Science and Engineering Research Board.

Data availability

The data associated with this paper can be downloaded from https://github.com/csccm-iitd/WNO.

Acknowledgments

T. Tripura acknowledges the financial support from the Ministry of Education (MoE), India, in the form of the Prime Minister's Research Fellowship (PMRF). S. Chakraborty acknowledges the financial support received from Science and Engineering Research Board (SERB) via grant no. SRG/2021/000467 and seed grant received from IIT Delhi.

References

[1] M. Renardy, R.C. Rogers, An Introduction to Partial Differential Equations, Vol. 13, Springer Science & Business Media, 2006.
[2] A. Sommerfeld, Partial Differential Equations in Physics, Academic Press, 1949.
[3] D.S. Jones, M. Plank, B.D. Sleeman, Differential Equations and Mathematical Biology, Chapman and Hall/CRC, 2009.
[4] T.J. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Courier Corporation, 2012.
[5] J.C. Strikwerda, Finite Difference Schemes and Partial Differential Equations, SIAM, 2004.
[6] R. Eymard, T. Gallouët, R. Herbin, Finite Volume Methods, in: Handbook of Numerical Analysis, Vol. 7, Elsevier, 2000, pp. 713–1018.
[7] H. Ren, X. Zhuang, T. Rabczuk, A nonlocal operator method for solving partial differential equations, Comput. Methods Appl. Mech. Engrg. 358 (2020) 112621.
[8] H. Ren, X. Zhuang, T. Rabczuk, A higher order nonlocal operator method for solving partial differential equations, Comput. Methods Appl. Mech. Engrg. 367 (2020) 113132.
[9] T.J. Hughes, J.A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (39–41) (2005) 4135–4195.
[10] J.A. Cottrell, T.J. Hughes, Y. Bazilevs, Isogeometric Analysis: Toward Integration of CAD and FEA, John Wiley & Sons, 2009.
[11] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Netw. 2 (5) (1989) 359–366.
[12] T. Chen, H. Chen, Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems, IEEE Trans. Neural Netw. 6 (4) (1995) 911–917.
[13] J. Zhang, G.G. Walter, Y. Miao, W.N.W. Lee, Wavelet neural networks for function learning, IEEE Trans. Signal Process. 43 (6) (1995) 1485–1497.
[14] S. Karumuri, R. Tripathy, I. Bilionis, J. Panchal, Simulator-free solution of high-dimensional stochastic elliptic partial differential equations using deep neural networks, J. Comput. Phys. 404 (2020) 109120.
[15] K. Wu, D. Xiu, Data-driven deep learning of partial differential equations in modal space, J. Comput. Phys. 408 (2020) 109307.
[16] V.M. Nguyen-Thanh, X. Zhuang, T. Rabczuk, A deep energy method for finite deformation hyperelasticity, Eur. J. Mech. A Solids 80 (2020) 103874.
[17] M. Raissi, Deep hidden physics models: Deep learning of nonlinear partial differential equations, J. Mach. Learn. Res. 19 (1) (2018) 932–955.
[18] S.H. Rudy, S.L. Brunton, J.L. Proctor, J.N. Kutz, Data-driven discovery of partial differential equations, Sci. Adv. 3 (4) (2017) e1602614.
[19] N. Parashar, B. Srinivasan, S.S. Sinha, Modeling the pressure-Hessian tensor using deep neural networks, Phys. Rev. Fluids 5 (11) (2020) 114604.
[20] N. Navaneeth, S. Chakraborty, Koopman operator for time-dependent reliability analysis, Probabilistic Engineering Mechanics 70 (2022) 103372.

[21] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[22] S. Goswami, C. Anitescu, S. Chakraborty, T. Rabczuk, Transfer learning enhanced physics informed neural network for phase-field modeling of fracture, Theor. Appl. Fract. Mech. 106 (2020) 102447.
[23] E. Samaniego, C. Anitescu, S. Goswami, V.M. Nguyen-Thanh, H. Guo, K. Hamdia, X. Zhuang, T. Rabczuk, An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications, Comput. Methods Appl. Mech. Engrg. 362 (2020) 112790.
[24] S. Chakraborty, Transfer learning based multi-fidelity physics informed deep neural network, J. Comput. Phys. 426 (2021) 109942.
[25] L. Lu, P. Jin, G.E. Karniadakis, Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators, 2019, arXiv preprint arXiv:1910.03193.
[26] L. Lu, X. Meng, S. Cai, Z. Mao, S. Goswami, Z. Zhang, G.E. Karniadakis, A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data, Comput. Methods Appl. Mech. Engrg. 393 (2022) 114778.
[27] S. Garg, H. Gupta, S. Chakraborty, Assessment of DeepONet for reliability analysis of stochastic nonlinear dynamical systems, 2022, arXiv preprint arXiv:2201.13145.
[28] C. Lin, M. Maxey, Z. Li, G.E. Karniadakis, A seamless multiscale operator neural network for inferring bubble dynamics, J. Fluid Mech. 929 (2021).
[29] S. Goswami, M. Yin, Y. Yu, G.E. Karniadakis, A physics-informed variational DeepONet for predicting crack path in quasi-brittle materials, Comput. Methods Appl. Mech. Engrg. 391 (2022) 114587.
[30] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Neural operator: Graph kernel network for partial differential equations, 2020, arXiv preprint arXiv:2003.03485.
[31] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkumar, Fourier neural operator for parametric partial differential equations, 2020, arXiv preprint arXiv:2010.08895.
[32] G. Bachman, L. Narici, E. Beckenstein, Fourier and Wavelet Analysis, Vol. 586, Springer, 2000.
[33] A. Boggess, F.J. Narcowich, A First Course in Wavelets with Fourier Analysis, John Wiley & Sons, 2015.
[34] K. Wirsing, Time frequency analysis of wavelet and Fourier transform, in: Wavelet Theory, IntechOpen London, UK, 2020.
[35] S. Mallat, W.L. Hwang, Singularity detection and processing with wavelets, IEEE Trans. Inform. Theory 38 (2) (1992) 617–643.
[36] S.-W. Cho, Method of recognizing human iris using daubechies wavelet transform, 2002, uS Patent App. 09/946, 714.
[37] E. Sheybani, Dimensionality reduction and noise removal in wireless sensor networks, in: 2011 4th IFIP International Conference on New Technologies, Mobility and Security, IEEE, 2011, pp. 1–5.
[38] E. Martin, Novel method for stride length estimation with body area network accelerometers, in: 2011 IEEE Topical Conference on Biomedical Wireless Technologies, Networks, and Sensing Systems, IEEE, 2011, pp. 79–82.
[39] J. Liu, Shannon wavelet spectrum analysis on truncated vibration signals for machine incipient fault detection, Meas. Sci. Technol. 23 (5) (2012) 055604.
[40] A.K. Alexandridis, A.D. Zapranis, Wavelet neural networks: A practical guide, Neural Netw. 42 (2013) 1–27.
[41] B. Xu, H. Shen, Q. Cao, Y. Qiu, X. Cheng, Graph wavelet neural network, 2019, arXiv preprint arXiv:1904.07785.
[42] N. Shervani-Tabar, N. Zabaras, Physics-constrained predictive molecular latent space discovery with graph scattering variational autoencoder, 2020, arXiv preprint arXiv:2009.13878.
[43] G. Gupta, X. Xiao, P. Bogdan, Multiwavelet-based operator learning for differential equations, Adv. Neural Inf. Process. Syst. 34 (2021).
[44] I.W. Selesnick, R.G. Baraniuk, N.C. Kingsbury, The dual-tree complex wavelet transform, IEEE Signal Process. Mag. 22 (6) (2005) 123–151.
[45] L.A. Ray, R.R. Adhami, Dual tree discrete wavelet transform with application to image fusion, in: 2006 Proceeding of the Thirty-Eighth Southeastern Symposium on System Theory, IEEE, 2006, pp. 430–433.
[46] P. Müller, B. Vidakovic, Bayesian Inference in Wavelet-Based Models, Vol. 141, Springer Science & Business Media, 2012.
[47] I. Daubechies, Ten Lectures on Wavelets, SIAM, 1992.
[48] G. Lee, R. Gommers, F. Waselewski, K. Wohlfahrt, A. O'Leary, PyWavelets: A Python package for wavelet analysis, J. Open Source Softw. 4 (36) (2019) 1237.
[49] F. Cotter, Uses of Complex Wavelets in Deep Convolutional Neural Networks (Ph.D. thesis), University of Cambridge, 2020.
[50] D. Hendrycks, K. Gimpel, Gaussian error linear units (gelus), 2016, arXiv preprint arXiv:1606.08415.
[51] Y. Meyer, Wavelets: Algorithms & Applications, SIAM (Society for Industrial and Applied Mathematics), Philadelphia, 1993.
[52] G.J. Lord, C.E. Powell, T. Shardlow, An Introduction To Computational Stochastic PDEs, Vol. 50, Cambridge University Press, 2014.