Academic Journal of Science and Technology
ISSN: 2771-3032 | Vol. 13, No. 2, 2024

Feasibility Study of Solving Oil-Water Two-Phase Flow Equations Using Fourier Neural Operator

Rong Zhong

School of Petroleum Engineering, Xi'an Shiyou University, Xi'an, Shaanxi, 710065, China

Abstract: In petroleum reservoir engineering, deep learning shows promising potential for solving partial differential equations (PDEs) in reservoir numerical simulations. However, traditional neural network-based methods for PDE solutions are often limited, typically requiring retraining for different equations. This study introduces the Fourier Neural Operator (FNO) approach, incorporating the fast Fourier transform mechanism to facilitate operator learning. We conduct a feasibility analysis of using FNO for solving two-dimensional oil-water two-phase flow PDEs. Through extensive literature review, we summarize the current state and limitations of deep learning in oil-water two-phase flow modeling, highlighting the application and advancements of neural operators for PDE solutions. This study demonstrates the potential of FNO to solve high-dimensional oil-water two-phase flow equations, offering valuable insights into intelligent predictions for complex subsurface reservoirs and presenting new methods and perspectives for further development in reservoir numerical modeling.

Keywords: Deep learning; Neural operators; FNO; Oil-water two-phase flow; Partial differential equations.

1. Introduction

Reservoir numerical simulation is an essential component in petroleum development and production. Traditional reservoir simulation techniques are grounded in physics-based methods that approximate solutions for partial differential equations (PDEs) using finite difference methods (DM), finite element methods (FEM), or boundary element methods (BEM). The finite difference and finite element methods primarily address PDE problems on bounded domains, while boundary element methods are used for unbounded domains. However, for large-scale, high-precision, and high-dimensional problems, these grid-based traditional methods often demand substantial memory and computational resources.

With the rapid advancement of artificial intelligence, machine learning is increasingly applied to reservoir engineering challenges with complex physical interpretations, offering new approaches to solve PDEs governing oil-water two-phase flow. Current mainstream deep learning approaches for PDE solutions are classified into data-driven and physics-constrained methods. The former uses deep learning algorithms to train on datasets derived from numerical simulators, implicitly solving PDEs. However, data-driven methods often overlook the underlying physical laws embedded in the data, resulting in lower stability and accuracy, and limiting their capacity for long-term and effective predictions. Physics-constrained methods, on the other hand, embed physical laws as regularization terms within neural networks, improving accuracy but still facing limitations in handling high-dimensional and nonlinear equations. Both approaches predominantly focus on learning mappings within finite-dimensional Euclidean spaces or between finite sets.Neural operators provide a generalization of neural networks that can learn mappings between infinite-dimensional function spaces, formulating these operators as combinations of linear integral operators and nonlinear activation functions, which enable them to capture the mapping processes for PDE solutions. Among these, the Fourier Neural Operator (FNO) combines universal approximation capability with discretization invariance, allowing it to approximate any given nonlinear continuous operator and to share model parameters across different discretizations within the underlying function space.[1,2]

This study focuses on applying the Fourier Neural Operator (FNO) to the oil-water two-phase flow equations in reservoir simulation. The FNO offers an efficient framework for solving PDEs, overcoming the limitations of traditional data-driven and physics-constrained methods while combining the advantages of these approaches. Specifically, the study investigates the applicability of FNO to solve and invert oil-water two-phase flow PDEs, analyzing the impact of data volume, hyperparameters, and other factors on the solution process. This research provides a feasibility study of FNO's potential to solve oil-water two-phase PDEs and their inversion problems effectively.

2. Current Research Status of Solving Partial Differential Equations with Neural Networks

Traditional methods for solving partial differential equations (PDEs) require grid discretization and nonlinear equation solving, making the process technically challenging, computationally expensive, and often unsuitable for solving inverse problems. Deep learning-based methods for PDEs can perform automatic differentiation in both time and space, effectively addressing nonlinear problems and enabling solutions for more complex, higher-dimensional PDEs.

At present, mainstream neural network-based PDE-solving approaches include data-driven, physics-informed, and neural operator methods. The data-driven approach relies solely on labeled data constraints, while the physics-informed approach combines constraints from labeled data with those from PDEs. The following sections provide a brief overview of data-driven and physics-informed methods, as well as the Fourier Neural Operator (FNO) and its derivatives.

2.1. Data-Driven and Physics-Informed Methods

In 1943, McCulloch and Pitts [4] introduced artificial neural networks and their mathematical models, marking the beginning of neural network research. Since neural networks essentially map inputs to outputs via composite functions, they can be used to approximate continuous nonlinear functions. Automatic differentiation in artificial neural networks, which computes derivatives precisely using the chain rule, can replace complex gradient calculations in PDEs. In 1986, Rumelhart et al. introduced the backpropagation algorithm, laying the groundwork for neural networks in solving PDEs. Wornik et al. [7] demonstrated in 1990 that multilayer neural networks could approximate any function and its derivatives, providing theoretical support for neural networks in solving differential equations. In 1994, Meade et al. introduced differential equations, initial conditions, and boundary conditions into the loss function to minimize residuals, enabling approximate solutions to equations. In 1996, Li [8] demonstrated that hidden layers in neural networks could approximate multivariate polynomials and their derivatives; further, in 1998, Lagaris et al. [9] independently represented initial and boundary conditions, opening new avenues for solving PDEs. In 2001, Aarts and Van [10] used single-hidden-layer feedforward networks to represent differential operators of different orders for training PDE solutions, while in 2005, Ramuhalli et al. [11] combined finite element modules with neural networks, introducing the finite element neural network. Due to the limitations of early multilayer fully connected neural networks, which could only solve simple equations, neural network methods for PDE solutions did not gain significant attention.

In 2018, Long et al. [3] proposed a data-driven neural network (PDE-Net), introducing Euler discretization for time derivatives and approximating differential operators as constrained convolution kernels, allowing the neural network to better approximate PDEs. Zha et al. [12] extended the constrained convolution kernels to three dimensions, and Liu et al. [13] proposed a universal differential equation solver based on fully connected neural networks for solving initial-boundary value problems. Addressing high-dimensional PDEs in data-driven settings, Han and Jentzen et al. [14,15] developed a deep learning-based solver for parabolic PDEs by approximating gradient operators. Recently, Sirignano et al. [16] proposed the Deep Galerkin Method (DGM), a physics-informed neural network fundamentally based on the Galerkin method for second-order differential operator computations. Kani et al. [17] integrated physics-informed constraints with deep residual recurrent neural networks, introducing orthogonal decomposition and discrete empirical interpolation methods to optimize computational complexity in high-fidelity numerical simulations.

In 2019, Raissi et al. [18] introduced Physics-Informed Neural Networks (PINNs), a method that combines data-driven and physics-informed constraints. By constructing residuals from control equations and boundary conditions, PINNs embed physics laws in regularized form within the loss function, enhancing the efficiency and accuracy of equation solutions. Since then, various studies based on PINNs have flourished. Meng et al. [19] proposed the Parareal Physics-Informed Neural Network (PPINN), which divides long-time problems into several independent short-time problems to improve solution efficiency. Jagtap et al. [20] introduced adaptive activation functions to replace conventional ones in PINNs, resulting in enhanced solution efficiency, accuracy, and robustness. Recently, Fraces et al. [21] addressed uncertainty quantification in reservoir engineering by introducing a parametrized Physics-Informed Neural Network (P-PINN), using two-phase oil-water PDEs to validate the model's superior performance.

2.2. Fourier Neural Operator

In 1989, Hornik et al. [25] introduced the concept that multilayer feedforward neural networks could serve as universal function approximators. They demonstrated that, with a sufficient number of hidden units, feedforward neural networks could approximate any Borel-measurable function in a finite-dimensional space to arbitrary accuracy, effectively approximating any continuous nonlinear function. This universal approximation theorem, proven with Fourier transform methods, laid the foundation for the later development of neural operators.

In 2019, Lu et al. [26] proposed the Deep Operator Network (DeepONet), an architecture capable of efficiently learning operators from relatively small datasets. The DeepONet consists of two sub-networks: one encodes the input function, and the other encodes the spatial locations of the output function. Experimental results showed that DeepONets significantly reduced generalization errors, establishing a theoretical basis for further neural operator research. In 2020, Li et al. [27] introduced the Fourier Neural Operator (FNO), a neural operator model with a fixed number of parameters that also meets discretization invariance. Compared to traditional PDE solvers, the FNO operates three orders of magnitude faster while achieving higher accuracy at fixed resolutions. The FNO replaces finite-dimensional linear layers in neural networks with linear operators acting on function spaces. Research further demonstrated that neural operators are universal approximators for continuous operators between Banach spaces, capable of uniformly approximating any continuous operator defined on Banach spaces. Neural operators, including the FNO, are currently the only models known to possess both discretization invariance and universal approximation properties.

Zhang et al. [28] applied FNO to the field of reservoir engineering, developing an FNO-based fuzzy neural network model to solve two-dimensional oil-water two-phase PDEs in subsurface reservoirs. This model uses Fast Fourier Transform (FFT) to extract information from the Fourier space, approximating differential operators and enhancing the model's efficiency, accuracy, and generalization capability. Recently, Du et al. [29] introduced a modified version of FNO, the Global-Local Fourier Neural Operator, designed to predict magnetohydrodynamics (MHD). Their results demonstrated that this new model not only accelerated simulations but also significantly improved predictive capabilities.

3. Overview of the Fourier Neural Operator

3.1. Basic Principle of FNO

The Fourier Neural Operator (FNO) leverages the Fast Fourier Transform (FFT) to compute complex linear integral operators by parameterizing the kernel in Fourier space rather than working directly in the domain DDD. Let xxx denote the kernel integral operator.

Table 1. Solving and Classification of Partial Differential Equations Based on Neural Networks

Categorization	Author (Year)）	Method (Model)	PDE
Data-Driven	[14]Rudy et al.(2017)	Sparse regression	Navier-Stokes equations; diffusion equation; Kuramoto-Sivashinsky equation
	[17]Bar-Sinai et al. (2019)	Subgrid-Scale Modeling	Burgers' Equation
	[20]Kadeethum(2021)	cGAN	Quasi-static or steady-state problems
Physical Constraints	[18]Raissi et al.(2019)	PINN	Burgers equation; Schrodinger equation; Allen–Cahn equation; Navier–Stokes equation.
	[19]Meng et al.(2020)	PPINN	Burgers equation; diffusion–reaction equation.
	[21]Fraces et al.(2023)	P-PINN	Buckley-Leverett problem
Neural Operator	[26]Lu L et al.(2019)	DeepONet	diffusion-reaction system
	[27]Li et al.(2021)	FNO	Burgers equation; Darcy equation; Navier-Stokes equation.
	[28]Zhang et al.(2022)	FNO	2D two--phase PDEs
	[30]Wen et al.(2022)	U-FNO	multi-phase flow problem; Darcy's flow problem.
	[31]Lehmann et al.(2024)	F-FNO	3D Elastic wave equation
	[32]Zhao et al.(2024)	RecFNO	Navier–Stokes equations; 2D steady-state Darcy flow.

$$(\mathcal{K}(a;\phi)v_t)(x) := \int_D \kappa(x,y,a(x),a(y);\phi)v_t(y)dy, \forall x \in D$$

Furthermore, the Fourier transform expressions involved are as follows:

$$(Ff)_j(k) = \int_D f_j(x)e^{-2i\pi\langle x,k\rangle}dx,$$

In order to remove redundancy and enhance the efficiency of solving equations, the data is filtered in Fourier space to eliminate high-order modes while retaining low-order modes. Subsequently, the data is restored to its original dimensional space through the inverse Fourier transform. The expression for the inverse Fourier transform involved in this process is as follows:

$$(F^{-1}f)_j(x) = \int_D f_j(k)e^{2i\pi\langle x,k\rangle}dk$$

Furthermore, from the expressions of the kernel integral operator and the Fourier transform, we can derive:

$$(\mathcal{K}(a;\phi)v_t)(x) = \mathcal{F}^{-1}\left(\mathcal{F}(\kappa_\phi)\cdot\mathcal{F}(v_t)\right)(x), \forall x \in D$$

Consequently, the Fourier integral operator used in this context can be expressed as:

$$(\mathcal{K}(\phi)v_t)(x) = \mathcal{F}^{-1}\left(R_\phi\cdot\mathcal{F}(v_t)\right)(x), \forall x \in D$$

FNO addresses the integral term in operator learning, and combining it with the iterative relationship yields the expression for the established model:

$$v_{t+1}(x) := \sigma\left(Wv_t(x) + (\mathcal{K}(a;\phi)v_t)(x)\right), \cdots \forall x \in D$$

Where, $D \in \mathbb{R}^d$ represents the spatial domain of the PDE, $x \in D$ is a point in the spatial domain, $a \in A = (D;\mathbb{R}^{da})$ denotes the input coefficient function, $u \in U = (D;\mathbb{R}^{du})$ represents the target solution function, D_j denotes the discretization of (a_j, u_j), ϕ represents the parameters of the kernel network k, $t=0,\ldots,T$ indicates the time steps, σ denotes the activation function, and $\mathcal{F}$ and $\mathcal{F}^{-1}$ are the Fourier transform and its inverse, respectively.

The core concept behind FNO is to operate within the Fourier domain to accelerate computation and enhance prediction accuracy. Unlike traditional neural networks that perform convolutions in spatial domains, FNO transforms the input into the frequency domain via Fourier transform, where it learns features more efficiently using the characteristic structure of the frequency domain. This process involves applying FFT to the input signal, capturing information across different frequency components through convolution, and then mapping the results back to the spatial domain using the inverse Fourier transform.

3.2. Advantages of FNO

Compared to non-operator-based numerical methods for solving PDEs, the Fourier Neural Operator offers several advantages:

Discretization Invariance: The Fourier layers in FNO are discretization-invariant, allowing it to learn and evaluate functions discretized in any manner. Since parameters are learned directly in the Fourier space, this effectively projects the physical space into a basis defined by $e^{2\pi i\langle x,k\rangle}$, ensuring clear definition throughout the space. Additionally, the use of FFT reduces computational complexity to near-linear time, which facilitates large-scale computations.

Single Parameter Set Generation, Independent of the Grid: FNO produces a single set of network parameters that are independent of the grid, enabling solutions to be transferred across different grids. Once trained, FNO only requires a forward pass to compute solutions for new instances.

Data-Only Requirement: FNO requires no prior knowledge of the underlying PDE; only data is necessary. By employing FFT, the neural operator can produce efficient numerical algorithms in finite-dimensional environments that parallel convolutional or recurrent neural networks.

High-Efficiency Zero-Shot Super-Resolution: FNO is

the first machine learning approach capable of zero-shot super-resolution modeling of turbulence, operating at speeds three orders of magnitude faster than traditional PDE solvers. At fixed resolutions, FNO achieves higher accuracy than previous learning-based solvers.

4. Factors Affecting FNO Solution Accuracy

When using the Fourier Neural Operator to solve two-phase oil-water flow partial differential equations, several key factors affect the accuracy of the solution: the neural network structure, dataset size, as well as the choice of optimizer and activation function.

4.1. Neural Network Structure

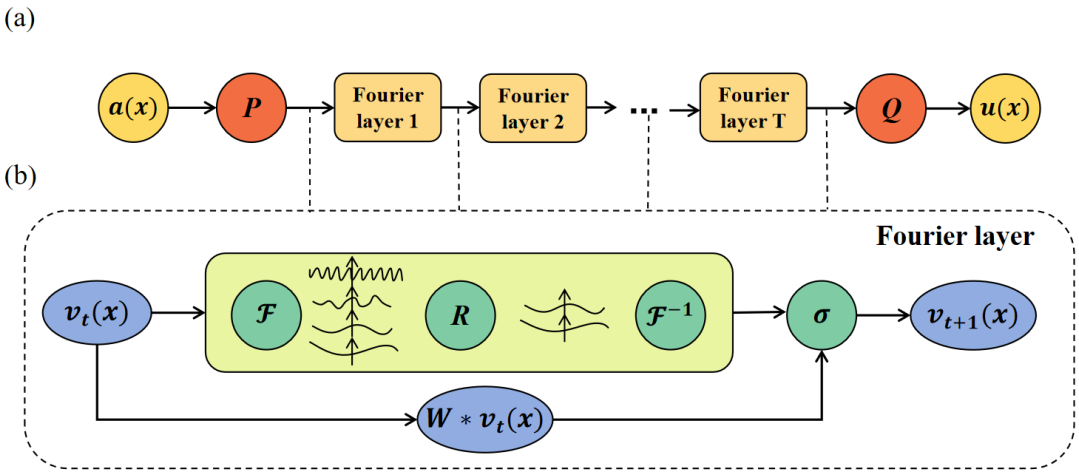

Figure 1. Structure of the Fourier Neural Operator

(a) Overall Neural Operator Structure: The input, denoted as aaa, is first mapped into a higher-dimensional channel space via the neural network. This is followed by applying four integral operator layers and activation functions, then projected back to the target dimension by the network Q, resulting in the output u.

(b) Fourier Layer: Starting with input v, the Fourier transform F is applied. A linear transformation R is then applied to the lower Fourier modes while filtering out higher modes, followed by an inverse Fourier transform F^{-1}. A local linear transformation W is applied at the bottom.

4.2. Choice of Activation Function

The selection of an activation function is critical to enhance the network's fitting capability and ensure effective training. Common activation functions include the Sigmoid, ReLU, and GeLU functions:

Sigmoid Functions: These include S-shaped functions like the Logistic and Tanh functions. The Logistic function, which compresses inputs into the range (0, 1), is continuously differentiable and maps smaller inputs closer to 0 and larger inputs closer to 1. The Tanh function, similarly S-shaped, scales to a range of (−1,1), essentially stretching and shifting the Logistic function.

ReLU (Rectified Linear Unit): ReLU is frequently used in deep neural networks and is characterized by a ramp function that is computationally efficient and biologically plausible. ReLU promotes sparsity, as nearly half of the neurons remain inactive, enhancing computational efficiency. However, inappropriate parameter updates can cause the "dying ReLU" problem, where neurons are permanently inactive.

GeLU (Gaussian Error Linear Unit): The GeLU activation function [Hendrycks et al., 2016] uses a gating mechanism to adjust its output based on a Gaussian cumulative distribution function, producing an S-shaped curve that can be approximated using Logistic or Tanh functions.

For two-phase flow problems, GeLU and ReLU are generally well-suited because they support fast convergence and network stability. GeLU's smoother handling of nonlinear features makes it ideal for complex multi-phase flows, while ReLU effectively supports high-dimensional feature extraction.

4.3. Choice of Optimizer

The optimizer's role is to direct parameter updates in the loss function towards minimizing errors globally or, in some cases, locally. Common optimizers include:

Stochastic Gradient Descent (SGD): A fundamental optimization algorithm, SGD updates model parameters using the gradient of a single (or mini-batch) sample. While computationally efficient, it may converge slowly or become trapped in local minima.

AdaGrad: This algorithm adapts the learning rate for each parameter based on the history of gradients, diminishing the rate for frequently updated parameters. AdaGrad is advantageous for handling sparse data and automatically adjusts learning rates during training.

Adam Optimizer: Adam combines momentum with RMSprop by updating learning rates based on both first-order (mean) and second-order (variance) gradient estimates. It generally provides efficient and stable convergence across varied learning rate settings.

The optimizer choice significantly impacts FNO training convergence and overall effectiveness. Adam and AdamW are commonly used for FNO, as they are particularly effective in frequency-domain operations. Adam's dynamic learning rate adjustment enhances training efficiency, while AdamW's weight decay improves generalization, making FNO solutions

more robust across diverse boundary conditions.

5. Conclusion

This study explores the application and feasibility of Fourier Neural Operators (FNO) in solving oil-water two-phase flow equations. By leveraging Fourier transforms to extract feature information in the frequency domain, FNO can efficiently capture the global characteristics of the flow field, demonstrating a significant computational efficiency advantage compared to traditional numerical methods. Experiments show that FNO exhibits good generalization ability and stability when dealing with complex boundary and initial conditions in multiphase flows. Additionally, benefiting from the combination of the GELU activation function and the Adam optimizer, FNO achieves rapid convergence and high-precision solutions. The results of this study provide new insights and possibilities for the application of FNO in complex fields such as reservoir engineering and flow simulation. Future work will further optimize the network structure to enhance its prediction accuracy in more complex oil-water flow systems.

References

[1] Han Jiangxia, Xue Liang, Liu Yuetian, et al. Solving Oil-Water Two-Phase Flow Equations Using Deep Neural Networks [C]//Beijing Energy Association, China University of Petroleum (Beijing), State Key Laboratory of Petroleum Resources and Prospecting, State Key Laboratory of Heavy Oil Processing, Baidu Intelligent Cloud. Proceedings of the 2022 China Oil and Gas Intelligent Technology Conference - The 5th Petroleum and Petrochemical Artificial Intelligence High-End Forum and the 8th Intelligent Digital Oilfield Open Forum, 2022:11. DOI: 10.26914/c.cnkihy.2022.035114.

[2] Cha Wenshu, Li Daolun, Shen Luhang, et al. A Review of Neural Network-Based Methods for Solving Partial Differential Equations [J]. Chinese Journal of Theoretical and Applied Mechanics, 2022, 54(03): 543-556.

[3] Long Z, Lu Y, Ma X, et al. Pde-net: Learning pdes from data[C]//International conference on machine learning. PMLR, 2018: 3208-3216.

[4] McCulloch W S, Pitts W. A logical calculus of the ideas immanent in nervous activity[J]. The bulletin of mathematical biophysics, 1943, 5: 115-133.

[5] Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. nature, 1986, 323(6088): 533-536.

[6] Baydin A G, Pearlmutter B A, Radul A A, et al. Automatic differentiation in machine learning: a survey[J]. Journal of Marchine Learning Research, 2018, 18: 1-43.

[7] Wornik K, Stinchcombe M, White H. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. Neural Networks, 1990, 3(5): 551-560

[8] Li X. Simultaneous approximations of multivariate functions and their derivatives by neural networks with one hidden layer. Neurocomputing, 1996, 12(4): 327-343

[9] Lagaris I E, Likas A, Fotiadis D I. Artificial neural networks for solving ordinary and partial differential equations[J]. IEEE transactions on neural networks, 1998, 9(5): 987-1000.

[10] Aarts L P, Van Der Veer P. Neural network method for solving partial differential equations[J]. Neural Processing Letters, 2001, 14: 261-271.

[11] Ramuhalli P, Udpa L, Udpa S S. Finite-element neural networks for solving differential equations[J]. IEEE transactions on neural networks, 2005, 16(6): 1381-1392.

[12] Zha W, Zhang W, Li D, et al. Convolution-based model-solving method for three-dimensional, unsteady, partial differential equations[J]. Neural Computation, 2022, 34(2): 518-540.

[13] Liu Z, Yang Y, Cai Q. Neural network as a function approximator and its application in solving differential equations[J]. Applied Mathematics and Mechanics, 2019, 40(2): 237-248.

[14] Han J, Jentzen A. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations[J]. Communications in mathematics and statistics, 2017, 5(4): 349-380.

[15] Han J, Jentzen A, E W. Solving high-dimensional partial differential equations using deep learning[J]. Proceedings of the National Academy of Sciences, 2018, 115(34): 8505-8510.

[16] Sirignano J, Spiliopoulos K. DGM: A deep learning algorithm for solving partial differential equations[J]. Journal of computational physics, 2018, 375: 1339-1364.

[17] Kani J N, Elsheikh A H. Reduced-order modeling of subsurface multi-phase flow models using deep residual recurrent neural networks[J]. Transport in Porous Media, 2019, 126: 713-741.

[18] Raissi M, Perdikaris P, Karniadakis G E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations[J]. Journal of Computational physics, 2019, 378: 686-707.

[19] Meng X, Li Z, Zhang D, et al. PPINN: Parareal physics-informed neural network for time-dependent PDEs[J]. Computer Methods in Applied Mechanics and Engineering, 2020, 370: 113250.

[20] Jagtap A D, Kawaguchi K, Karniadakis G E. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks[J]. Journal of Computational Physics, 2020, 404: 109136.

[21] Fraces C G, Tchelepi H. Uncertainty Quantification for Transport in Porous Media Using Parameterized Physics Informed Neural Networks[C]//SPE Reservoir Simulation Conference?. SPE, 2023: D011S004R003.

[22] Lu L, Jin P, Karniadakis G E. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators[J]. arXiv preprint arXiv:1910.03193, 2019.

[23] Nelsen N H, Stuart A M. The random feature model for input-output maps between banach spaces[J]. SIAM Journal on Scientific Computing, 2021, 43(5): A3212-A3243.

[24] Patel R G, Trask N A, Wood M A, et al. A physics-informed operator regression framework for extracting data-driven continuum models[J]. Computer Methods in Applied Mechanics and Engineering, 2021, 373: 113500.

[25] Hornik K, Stinchcombe M, White H. Multilayer feedforward networks are universal approximators[J]. Neural networks, 1989, 2(5): 359-366.

[26] Lu L, Jin P, Karniadakis G E. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators[J]. arXiv preprint arXiv:1910.03193, 2019.

[27] Li Z, Kovachki N, Azizzadenesheli K, et al. Fourier neural operator for parametric partial differential equations[J]. arXiv preprint arXiv:2010.08895, 2020.

[28] Zhang K, Zuo Y, Zhao H, et al. Fourier neural operator for solving subsurface oil/water two-phase flow partial differential equation[J]. Spe Journal, 2022, 27(03): 1815-1830.

[29] Du Y. Neural operator for accelerating coronal magnetic field computations in bifrost MHD model[J]. 2024.

[30] Wen G, Li Z, Azizzadenesheli K, et al. U-FNO—An enhanced Fourier neural operator-based deep-learning model for multiphase flow[J]. Advances in Water Resources, 2022, 163: 104180.

[31] Lehmann F, Gatti F, Bertin M, et al. 3D elastic wave propagation with a factorized Fourier neural operator (F-FNO)[J]. Computer Methods in Applied Mechanics and Engineering, 2024, 420: 116718.

[32] Zhao X, Chen X, Gong Z, et al. RecFNO: A resolution-invariant flow and heat field reconstruction method from sparse observations via Fourier neural operator[J]. International Journal of Thermal Sciences, 2024, 195: 108619.