Petroleum Science 22 (2025) 4714–4735

Contents lists available at ScienceDirect

Petroleum Science

journal homepage: www.keaipublishing.com/en/journals/petroleum-science

Original Paper

A layer-specific constraint-based enriched physics-informed neural network for solving two-phase flow problems in heterogeneous porous media

Jing-Qi Lin [a,b], Xia Yan [a,b,*], Er-Zhen Wang [c], Qi Zhang [d], Kai Zhang [b,e], Pi-Yang Liu [e], Li-Ming Zhang [a,b]

[a] State Key Laboratory of Deep Oil and Gas, China University of Petroleum (East China), Qingdao, 266580, Shandong, China
[b] School of Petroleum Engineering, China University of Petroleum (East China), Qingdao, 266580, Shandong, China
[c] Petroleum Technology Research Institute of PetroChina, Changqing Oilfield Company, Xi'an, 710000, Shaanxi, China
[d] Department of Civil and Environmental Engineering, University of Macau, Taipa, 999078, Macao, China
[e] Civil Engineering School, Qingdao University of Technology, Qingdao, 266520, Shandong, China

ARTICLE INFO

Article history:
Received 17 January 2025
Received in revised form
13 July 2025
Accepted 13 July 2025
Available online 19 July 2025

Edited by Yan-Hua Sun

Keywords:
Physics-informed learning
Explainable artificial intelligence
Constraint learning
Two-phase flow
Heterogeneous porous media

ABSTRACT

In this study, we propose a constraint learning strategy based on interpretability analysis to improve the convergence and accuracy of the enriched physics-informed neural network (EPINN), which is applied to simulate two-phase flow in heterogeneous porous media. Specifically, we first analyze the layerwise outputs of EPINN, and identify the distinct functions across layers, including dimensionality adjustment, pointwise construction of non-equilibrium potential, extraction of high-level features, and the establishment of long-range dependencies. Then, inspired by these distinct modules, we propose a novel constraint learning strategy based on regularization approaches, which improves neural network (NN) learning through layer-specific differentiated updates to enhance cross-timestep generalization. Since different neural network layers exhibit varying sensitivities to global generalization and local regression, we decrease the update frequency of layers more sensitive to local learning under this constraint learning strategy. In other words, the entire neural network is encouraged to extract more generalized features. The superior performance of the proposed learning strategy is validated through evaluations on numerical examples with varying computational complexities. Post hoc analysis reveals that gradient propagation exhibits more pronounced staged characteristics, and the partial differential equation (PDE) residuals are more uniformly distributed under the constraint guidance. Interpretability analysis of the adaptive constraint process suggests that maintaining a stable information compression mode facilitates progressive convergence acceleration.

1. Introduction

Under the concept of deep learning (LeCun et al., 2015), the potential of neural networks (NNs) is continuously tapped, forming a novel paradigm for addressing a variety of scientific challenges, with impressive advancements exhibited in fields such as computer vision (Krizhevsky et al., 2017) and natural language processing (Vaswani et al., 2017). Modern NNs include multiple foundational architectures; for instance, convolutional neural networks (CNN) (Li et al., 2022) excel at handling grid-structured data such as images and videos, recurrent neural networks (RNN) (Yu et al., 2019) are designed to process data with sequential order, and graph neural networks (GNN) (Zhou et al., 2020) are tailored for data with unstructured adjacency information. The emergence of large language models (Kalyan, 2024) and subsequent multimodal large models (Liang et al., 2024) has ignited the latest wave in artificial intelligence. Concurrently, studies suggest world models (Bardes et al., 2024) represent a promising direction in the future, with one pivotal aspect being AI's ability to internalize physical laws, thereby enabling physically grounded reasoning.

---

* Corresponding author.
E-mail addresses: jsyanxia1989@163.com, 20190025@upc.edu.cn (X. Yan).
Peer review under the responsibility of China University of Petroleum (Beijing).

https://doi.org/10.1016/j.petsci.2025.07.008

Recently, physics-informed machine learning (Karniadakis et al., 2021; Liu et al., 2025; Toscano et al., 2025) has been developed to tackle scientific problems with underlying physical contexts, where methods that utilize physical rules for constraint learning (Daw et al., 2017) are considered to ensure the scientific consistency of the results, thereby enhancing the model's generalization. Raissi et al. (2019) introduced physics-informed neural networks (PINNs) that employ partial differential equations (PDEs) as physical principles and leverage automatic differentiation to construct physics-constrained loss for training, achieving data-free learning of forward processes and enabling the inversion of uncertain parameters from noisy data. Huang et al. (2025) highlights that current research on deep neural networks (DNNs) for solving PDEs continues to face persistent challenges in computational efficiency, generalization capability, interpretability, and handling complex physical systems.

Subsequently, a series of PINN-like variants have been developed to solve different physical problems (Cuomo et al., 2022), and several open-source libraries have been made available to facilitate the construction of physical constraints (Haghighat and Juanes, 2021; Hennigh et al., 2021; Lu et al., 2021). In deeper interdisciplinary research, the physical problems to be addressed may involve greater complexity, such as non-equilibrium behavior (Harbate et al., 2024), intricate wave structures (El Idrissi et al., 2023), and high-gradient dynamics (Zeidan et al., 2019). PINNs have been modified to handle these complex issues, such as integrating diffusion terms in the loss function to address shock waves in two-phase flows (Fuks and Tchelepi, 2020), incorporating expert knowledge during training to align with practical engineering (Wang et al., 2020). Moreover, appropriate discretization methods often play a critical role in enhancing the stability of simulations for complex flow problems (Ouffa et al., 2021). In a previous study (Zhang et al., 2023), we developed a physics-driven approach combining the finite volume method (FVM) (Eymard et al., 2000) and hard constraint boundaries to achieve more accurate PDE constraints in discrete parameter systems of two-phase flow. In subsequent research, we developed the enriched physics-informed neural network (EPINN) (Yan et al., 2024) that incorporates functionalities referring to RNN and GNN to enable the simulation of more complex nonlinear flow systems. However, in current research, NNs are primarily utilized as essential tools, much like they are often treated as universal approximators and criticized as inscrutable black boxes (Alain and Bengio, 2016). Although the neural network incorporates physical meaning through a physics-constrained loss function, its internal mechanisms remain unclear, particularly regarding the critical patterns it may capture and how the physical constraints influence gradient propagation.

Interpretable analysis of neural networks is a crucial topic that aids in understanding and effectively applying them to solve diverse problems (Samek et al., 2021; Shwartz-Ziv, 2022). Katsnelson et al. (2021) suggest that during training, NNs form self-organized critical states characterized by scale invariance, where trainable and non-trainable parameters across layers are driven to a stochastic equilibrium. Equitz and Cover (1991) proposed the rate-distortion theory, suggesting that multilayer NNs process and refine information sequentially, with each hidden layer's output serving as the input for the next, enhancing the construction of higher-level distributed representations. Mehta and Schwab (2014) argued that as deep NNs approach optimal performance, their architecture exhibits increasingly decoupled statistical characteristics across different layers. Combining information-theoretic methods, Tishby and Zaslavsky (2015) believed that the boundary of optimality is computable. Zeiler and Fergus (2014) used visualization techniques to intuitively observe and understand NNs' learning in computer vision tasks, discovering that different layers' learning contents can be discerned with the specific image semantics. Nakkiran et al. (2021) suggest that the “double descent phenomenon” is prevalent during NN training, where network performance initially declines and then improves with increased training epochs once the model complexity is sufficient, extending the variance-bias trade-off theory regarding overfitting. Zhou et al. (2019) argued that effectively learning from noisy information during NN training aids convergence to a position closer to the global optimum. Damian et al. (2022) suggested that learning from noise is a self-stabilizing process with randomness, observable through the fluctuation of performance evaluation metrics. Cohen et al. (2021) believed the optimal state during gradient descent training is at the edge between stability and randomness. The same phenomena were also observed in adaptive optimizers (Cohen et al., 2022), and models were believed to exhibit super-convergence after sufficiently learning from noise (Smith and Topin, 2019). Tishby et al. (2000) introduced the information bottleneck (IB) theory to understand the signal transmission of complex systems, proposing that key feature extraction is an essential source of the NNs' generalization. Researchers have extensively attempted to theoretically interpret the deep learning process in conjunction with IB theory (Goldfeld and Polyanskiy, 2020). Lewkowycz et al. (2020) noted that NN training with gradient descent algorithms undergoes phase transitions, with significant differences between phases. Shwartz-Ziv and Tishby (2017) proposed using the signal-to-noise ratio to represent the fitting-diffusion transition in NN training. Anagnostopoulos et al. (2024) identified the total diffusion phase in PINN training, which exhibits superior convergence of model parameters under a uniform gradient; the emergence of this phase is believed to indicate improved generalization performance. Interpretability methods have seen broad development in various neural networks. Still, few studies attempt deep explanations for physics-constrained problems, especially when these networks offer additional insights from a physical background.

Understanding the learning processes in NNs helps properly employ various domain-specific strategies. For instance, multitask learning (Ruder, 2017) aims to handle multiple tasks simultaneously, enabling NNs to learn shared features to improve generalizability. Transfer learning (Zhuang et al., 2021) seeks to leverage acquired knowledge (or pretrained NNs) to address similar problems. The core idea of continual learning (Wang L. et al., 2024) is to maintain performance on old tasks while learning new ones, thus enhancing global generalizability. In physics-driven methods, most PINN-like approaches can be regarded as multitask learning (Raissi et al., 2019; Fuks and Tchelepi, 2020; Wang et al., 2020, 2022; Haghighat and Juanes, 2021; Hennigh et al., 2021; Lu et al., 2021; Cuomo et al., 2022), where spatial or spatiotemporal sampling forms multiple parallel tasks, constructing a holistic residual for training. Wang S. et al. (2024) suggested that uniform training methods like vanilla PINNs across all time steps might violate temporal causality, and a reasonable learning approach should capture the evolutionary process over time. In transfer learning, Liu et al. (2023) attempt to reuse a trained PINN for new problems with similar physical backgrounds and design a guiding strategy to achieve this progressively. Methods of solving by advancing through time steps (Zhang, 2022; Li et al., 2023; Zhang et al., 2023; Yan et al., 2024) involve basic transfer operations between adjacent time steps, but the method's theoretical basis suggests a lack of global generalization across multiple time steps. In continual learning, Dekhovich et al. (2024) utilize the continual learning strategies, starting with simple PDEs and continuously learning multiple PDEs, thus enabling the NN to develop a more generalized ability. Howard et al. (2024) explore preventing catastrophic forgetting in PINN learning through historical information replay and loss function regularization, thus achieving generalization on multi-fidelity datasets. EPINN is grounded in a well-defined physical framework for two-phase flow solving; certain features of the methods significantly enhance the solving of heterogeneous problems, particularly through an anchored structure that defines spatial geometric relationships, providing a criterion for interpreting these features in a physically meaningful context. Gaining deeper insights into the operational logic of NNs during the learning process would enable improvements to enhance the speed and accuracy of the NN-based simulation.

In this study, we leverage pointwise attributes and physical context to visualize and uncover latent characteristics of EPINN learning. Based on these insights, we propose the EPINN with constrained-layer (EPINN-CL), which mitigates forgetting of historical information by imposing constraints to suppress updates in zones more sensitive to local fitting. This novel method improves NN learning by introducing layer-specific differentiated updates, promoting learning more generalizable features across multiple time steps. The strategy's effectiveness was validated in scenarios of various computational complexity, demonstrating faster convergence speed and higher accuracy. Additionally, we provide insightful post hoc interpretations of the adaptive adjustment processes to elucidate the mechanism of constraint adaptation.

The paper's organization is as follows: Section 2 introduces the physics-driven learning method and provides visual observations and interpretations for homogeneous, heterogeneous, and fracture cases. Section 3 presents a new constraint learning strategy, validating its effectiveness across the three cases and offering post hoc interpretations of the learning process. Section 4 concludes the paper and outlines future research directions.

2. Visualization of EPINN in two-phase flow simulation

In this section, we present the physical foundation of EPINN for two-phase flow learning in porous media. Based on the pointwise attributes of EPINN, we design a visualization method to analyze the latent patterns of layer-specific outputs in homogeneous, heterogeneous, and fracture cases, offering intuitive interpretations of the different functionalities.

2.1. Governing equations of two-phase flow in porous media

We define the physical background as the immiscible two-phase flow of weakly compressible fluids in porous media and consider the complexity introduced by the permeability heterogeneity (Gerritsen and Durlofsky, 2005; Chen et al., 2006; Bear and Cheng, 2010; Chung and Woo, 2011; Abidoye et al., 2015). The mass conservation equation of each phase is (Moortgat and Firoozabadi, 2013)

$$\frac{\partial}{\partial t}\left(\phi\rho_\beta S_\beta\right)+\nabla\cdot\left(\rho_\beta \boldsymbol{v}_\beta\right)=\rho_\beta q_\beta \tag{1}$$

where β indicates fluid phases, $\beta = \mathrm{w}$ for wetting phase and $\beta = \mathrm{n}$ for non-wetting phase; ρ is the fluid density; S is the phase saturation; $\boldsymbol{v}$ is the fluid velocity; q is the source or sink; ϕ is the porosity; t is the time. The flow of each phase in porous media conforms to Darcy's law (Moortgat and Firoozabadi, 2013):

$$\boldsymbol{v}_\beta=-\frac{k_{\mathrm{r}\beta}}{\mu_\beta}k\left(\nabla P_\beta-\rho_\beta g\nabla D\right) \tag{2}$$

where k_r is the relative permeability; k is the absolute permeability; P is the fluid pressure; g is the gravity acceleration; D is the depth; μ is the fluid viscosity. In addition, the following supplementary equations need to be incorporated

$$S_\mathrm{w}+S_\mathrm{n}=1 \tag{3}$$

$$P_\mathrm{c}=P_\mathrm{n}-P_\mathrm{w} \tag{4}$$

where P_c denotes the capillary force, and it usually can be neglected (Yan et al., 2024). Consequently, the pressures of non-wetting and wetting phases are equivalent, i.e., $P_\mathrm{n} = P_\mathrm{w} = P$.

To account for the compressibility of fluid and porous media, we adopt the following constitutive equations

$$\rho_\beta=\rho_{\beta 0}\exp\left[C_\beta(P-P_0)\right] \tag{5}$$

$$\phi=\phi_0\exp\left[C_\mathrm{r}(P-P_0)\right] \tag{6}$$

where subscript 0 denotes the initial state, and C_r and C_β represent the isothermal compressibility of porous media and phase β, respectively. Finally, the initial and boundary conditions (BC) also need to be specified to ensure completeness of the formulation. In this study, we employ the closed boundary conditions throughout numerical examples, and the initial conditions (IC) (i.e., initial pressure and saturation) will be explicitly specified in the subsequent case study sections.

2.2. Numerical schemes for two-phase flow in porous media

The numerical simulation of immiscible two-phase flow in porous media typically employs two schemes: the fully implicit method (Saad and Saad, 2013) and the implicit pressure explicit saturation (IMPES) approach (Chen et al., 2019). While the fully implicit method ensures unconditional stability by solving pressure and saturation equations simultaneously, its high computational cost per time step limits efficiency. In contrast, the IMPES scheme decouples the governing equations, solving pressure implicitly and updating saturation explicitly using the computed pressure field. Although IMPES requires time step restrictions for stability, its reduced computational demand per iteration, simplified implementation, and enhanced stability through CFL condition enforcement (Coats, 2003) make it particularly effective for immiscible two-phase flow simulations (Redondo et al., 2018). In this study, we adopt the IMPES scheme to decouple and sequentially solve the pressure and saturation equations. By resolving pressure first, the saturation update is simplified to an explicit calculation under known pressure conditions. This strategy allows NN to focus solely on pressure prediction throughout the simulation, significantly reducing NN training complexity while improving computational efficiency.

The phase potential is first defined as $\Phi_\beta = P - \rho_\beta g D$. Next, substituting Eq. (2) into Eq. (1) and neglecting the term arising from fluid density gradients (due to the fluid's weak compressibility), Eq. (1) can be approximated as

$$\phi\frac{\partial S_\beta}{\partial t}+S_\beta\phi\left(C_\beta+C_\mathrm{r}\right)\frac{\partial P}{\partial t}=\nabla\cdot\left(\lambda_\beta k\nabla\Phi_\beta\right)+q_\beta \tag{7}$$

where λ denotes the fluid mobility, defined as $\lambda = k_\mathrm{r}/\mu$. Then, the pressure equation required for IMPES scheme is formulated by combining the wetting and non-wetting phase equations, as

$$\phi\left(S_\mathrm{n}C_\mathrm{n}+S_\mathrm{w}C_\mathrm{w}+C_\mathrm{r}\right)\frac{\partial P}{\partial t}=\nabla\cdot\left(\lambda_\mathrm{n}k\nabla\Phi_\mathrm{n}\right)+\nabla\cdot\left(\lambda_\mathrm{w}k\nabla\Phi_\mathrm{w}\right)+q_\mathrm{n}+q_\mathrm{w} \tag{8}$$

Introducing the potential difference $\Phi_\mathrm{c} = \Phi_\mathrm{n} - \Phi_\mathrm{w}$, Eq. (8) can be expressed as

$$\phi(S_\mathrm{n}C_\mathrm{n}+S_\mathrm{w}C_\mathrm{w}+C_\mathrm{r})\frac{\partial P}{\partial t}=\nabla\cdot(\lambda_\mathrm{t}k\nabla\Phi_\mathrm{w})+\nabla\cdot(\lambda_\mathrm{n}k\nabla\Phi_\mathrm{c})+q_\mathrm{n}+q_\mathrm{w} \tag{9}$$

where λ_t represents the total mobility, calculated as the summation of λ_n and λ_w. Now, the two-phase flow simulation can be performed through sequential solutions of Eqs. (9) and (7). First, Eq. (9) is solved implicitly for the current timestep pressure, with saturation from the previous timestep used to compute mobility parameters. The resulting pressure field is then substituted into Eq. (7) to update saturation explicitly. Notably, these governing equations apply to both matrix and fracture domains, with distinct properties assigned to each in fractured porous media.

The FVM-based discretization remains a well-established and prevalent approach for two-phase flow problems due to its inherent conservation properties and robustness in handling discontinuities (Zeidan and El Idrissi, 2025). Therefore, the FVM is employed in this study for equation discretization, and this procedure involves first integrating Eq. (9), subsequently applying the divergence theorem, and implementing a first-order discretization for the time derivative term, ultimately yielding the following residual expression

$$R_i=\frac{\phi(S_\mathrm{n}C_\mathrm{n}+S_\mathrm{w}C_\mathrm{w}+C_\mathrm{r})V_i}{\Delta t}\left(P_i^{t+1}-P_i^t\right)-\sum_{j\in G_i}\left(\lambda_{t,ij+1/2}T_{ij}\left(\Phi_{\mathrm{w},j}^{t+1}-\Phi_{\mathrm{w},i}^{t+1}\right)\right)$$
$$-\sum_{j\in G_i}\left(\lambda_{\mathrm{n},ij+1/2}T_{ij}\left(\Phi_{\mathrm{c},j}^{t}-\Phi_{\mathrm{c},i}^{t}\right)\right)-(q_\mathrm{n}+q_\mathrm{w})V_i \tag{10}$$

where subscripts i and j are element indexes; $ij+1/2$ indicates the intermediate value between two adjacent elements; superscript t is the time step index; R is the residual of the PDE; V is the element volume; Δt is the time step size; G indicates the adjacent elements; λ is explicitly computed with upstream weighting value; q is source or sink, which is approximated using the equivalent model (Peaceman, 1993)

$$q_\beta=\lambda_\beta\frac{2\pi kh}{\ln(r_\mathrm{e}/r_\mathrm{w})+s}\left(P-P_\mathrm{wf}\right) \tag{11}$$

where h is the connection thickness with formation; r_e is the equivalent formation radius; r_w is the wellbore radius; s is the skin factor; P_wf is the bottom hole pressure (BHP). The time step size Δt needs to satisfy the following CFL condition to ensure computational stability (Yan et al., 2016; Redondo et al., 2018)

$$\Delta t\le\frac{\phi_iV_i}{v_i^\mathrm{in}\max\left[f_\mathrm{w}'(s)\right]},\quad f_\mathrm{w}'(s)=\frac{1}{1-s_\mathrm{wc}-s_\mathrm{nr}}\frac{\partial f_\mathrm{w}}{\partial s^*},\ 0\le s\le 1 \tag{12}$$

where v_i^in denotes the gridcell's total influx; S^* denotes the normalized wetting phase saturation; f_w is the fractional flow function; S_wc and S_nr represent irreducible wetting phase and residual non-wetting phase saturations, respectively; T_{ij} is the transmissibility between elements i and j, which is defined as

$$T_{ij}=\frac{A_{ij}k_{ij+1/2}}{d_i+d_j} \tag{13}$$

where A denotes the interface area; d is the vertical distance from the element center to the interface; $k_{ij+1/2}$ is the harmonic mean absolute permeability. In the fractured scenario, the fracture element is characterized by unstructured grids with non-adjacent connections (Losapio and Scotti, 2023), fractures are discretized according to their intersections with the matrix grids, as illustrated in Fig. 1(a). Fig. 1(b) illustrates three types of transmissibility: T_mf (matrix-matrix), T_mf (matrix-fracture), and T_ff (fracture-fracture).

The connection of matrix-matrix and fracture-fracture can be calculated with Eq. (13), the connection between matrix and fracture is calculated as

$$T_\mathrm{mf}=\frac{A_\mathrm{mf}k_\mathrm{mf}}{\langle d\rangle} \tag{14}$$

where A_mf is the fracture contact area in matrix; k_mf is the harmonic mean absolute permeability of matrix and fracture; $\langle d\rangle$ is the average normal distance between matrix and fracture.

Following pressure computation, the wetting phase saturation is explicitly updated via the following equation derived from the discretized form of Eq. (7)

$$S_{\mathrm{w},i}^{t+1}=S_{\mathrm{w},i}^{t}+\frac{\sum_{j\in G_i}\left(\lambda_\mathrm{w}T_{ij}\left(\Phi_{\mathrm{w},j}^{t+1}-\Phi_{\mathrm{w},i}^{t+1}\right)\right)\Delta t}{\phi V_i}+\frac{q_\mathrm{w}\Delta t}{\phi}$$
$$-S_{\mathrm{w},i}^{t}(C_\mathrm{w}+C_\mathrm{r})\left(P_i^{t+1}-P_i^t\right) \tag{15}$$

To sum up, we adopt a numerical framework that achieves first-order accuracy to simulate two-phase flow in porous media by integrating the IMPES method combined with CFL condition, the finite volume spatial discretization, and a first-order temporal discretization.

In this study, we employ NN to solve the pressure equation (Eq. (9)). Specifically, we define the following loss function based on Eq. (10), combined with the mean square error (MSE)

$$L^{t+1}=\frac{1}{N_\mathrm{g}}\sum_{i=1}^{N_\mathrm{g}}\left(R_i\left(P_i^t,\tilde{P}_i^{t+1}\right)\right)^2 \tag{16}$$

where N_g is the number of discrete grids; $\tilde{P}_i^{t+1}$ is the NN output that approximates P_i^{t+1}. The exact value of P_i^{t+1} can be obtained by training NN (minimizing Eq. (16)). The governing equations intrinsically ensure physical consistency, allowing physics-constrained neural network training without labeled data. Section 2.3 details the neural network architecture and training workflow.

2.3. The structure of EPINN

We establish the pressure-to-pressure mapping with EPINN. The mapping of the input layer is

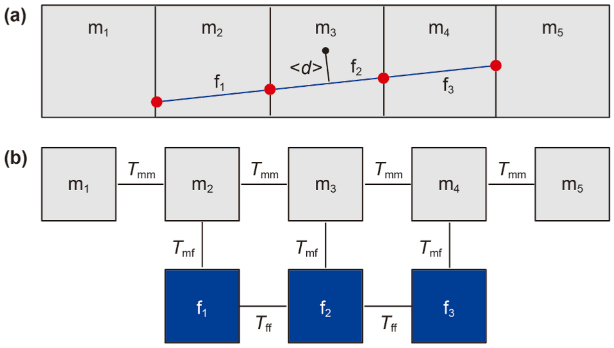

Fig. 1. Schematic of matrix (gray) and fracture (blue) geometry correspondence (**a**) and connection mode (**b**) under the EDFM strategy (Yan et al., 2024).

$$\tilde{P}_1^t = w_1 \cdot P^t + b_1 \tag{17}$$

where w_1 and b_1 are the weight and bias of the input layer. There is a special adjacency-location anchoring structure after the input layer is

$$P_1^t = \gamma\left(\frac{\omega_1 \cdot \tilde{P}_1^t + m_{\text{adj}} - \mu_{\text{B}}}{\sqrt{\sigma_{\text{B}}^2 + \epsilon}}\right) + \beta \tag{18}$$

where ω_1 is a learnable parameter; m_{adj} is the fixed adjacent matrix; μ_{B} is the mean and σ_{B}^2 is the variance of each column in $(\tilde{P}_1^t \cdot \omega_1 + m_{\text{adj}})$; ϵ is a small value to avoid dividing by zero (in this paper, set to 1.0×10^{-5}); γ and β are the learnable weight and bias. The linear mapping of the hidden layer is

$$\tilde{P}_2^t = w_2 \cdot P_1^t + b_2 \tag{19}$$

where w_2 and b_2 are the weight and bias of the hidden layer. The adaptive ReLU (Jagtap et al., 2020) is employed to add nonlinearity as

$$P_2^t = \max\left(0, \alpha \cdot \tilde{P}_2^t\right) \tag{20}$$

where α is a learnable parameter to adjust the activation range adaptively; P_2^t is the activating output. The decoding layer is

$$\tilde{P}_3^t = w_3 \cdot P_2^t + b_3 \tag{21}$$

where w_3 and b_3 are the weight and bias of the hidden layer; $\tilde{P}_2^t$ is the mapping output. The skip connecting and gated updating is

$$\tilde{P}^{t+1} = \omega_2 \cdot P^t + (1 - \omega_2) \cdot \tilde{P}_3^t \tag{22}$$

where ω_2 is a learnable parameter to balance the reference of the previous pressure distribution.

The parameter settings of the numerical model learned by EPINN are consistent with numerical simulations, as detailed in Section 2.3. The EPINN hyperparameter configurations are presented in Appendix A. The training workflow of the physics-constrained EPINN is illustrated in Fig. 2. Specifically, ADAM (Kingma and Ba, 2014) is used as the optimizer to minimize L^{t+1} to a desired threshold (set as $\text{MSE} < 1.0 \times 10^{-18}$ in this study), thus $P^{t+1} = \tilde{P}^{t+1}$. Subsequently, S^{t+1} can be calculated using Eq. (15). The converged P and NN parameters are used as the initial conditions of the next time step. Through multiple iterations, solutions for all discrete time steps are obtained. The mapping process, backpropagation, and gradient descent updates are executed in PyTorch (Paszke et al., 2019) (Version 1.12.1).

2.4. Two-phase flow cases

We designed three physical cases: homogeneous, heterogeneous, and fractured. The reason for choosing these three scenarios is that they generally exemplify the increase in heterogeneity accompanied by a rise in pressure gradients, thus lifting the solving difficulties with more intricate physical field distribution patterns (Chen et al., 2006; Berre et al., 2019; Zhang et al., 2023; Yan et al., 2024). In the subsequent content, the reference solutions all are computed using our validated in-house reservoir simulator (Yan et al., 2016).

The model geometry for the homogeneous (Fig. 3(a)), heterogeneous (Fig. 3(a)), and fractured (Fig. 4(a)) cases is 100 m × 100 m, with a thickness of 1 m. The grid mesh for the homogeneous and heterogeneous cases is 25 × 25 (Fig. 3(b)) (this grid selection was derived from the sensitivity analysis in Appendix B), generating 625 grid elements of 4 m × 4 m; the fractured case incorporates an additional 138 unstructured fracture grids (Fig. 4 (b)). The permeability for the homogeneous case is 50 mD; for the heterogeneous case, it varies from 10 to 120 mD as illustrated in Fig. 5(a); for the fractured case, the matrix permeability is 10 mD, whereas the fracture permeability is 30,000 mD. For all cases, the relative permeability curve is provided in Fig. 5(b), with time step size $\Delta t = 0.01$ d satisfying the CFL stability condition. The injector and producer are centered in the bottom-left and top-right grid cells, respectively. The specific parameter settings for each case are listed in Table 1. We normalized the input pressure to facilitate NN learning; the normalization method is

$$P_{\text{scale}} = \frac{P_{\text{ori}}}{P_{\text{int}}} \tag{23}$$

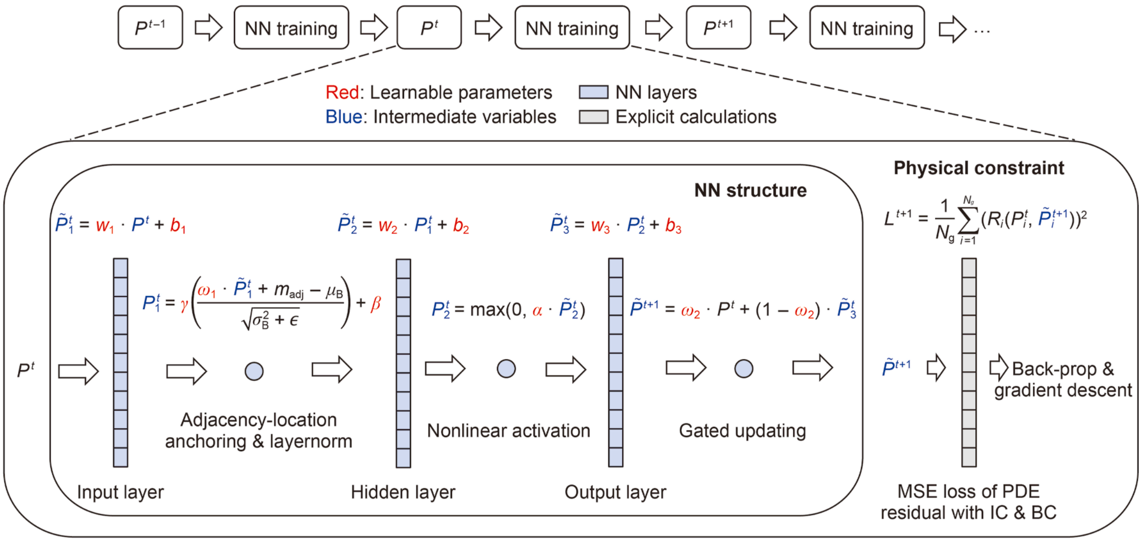

Fig. 2. The procedure of EPINN with physical constraint.

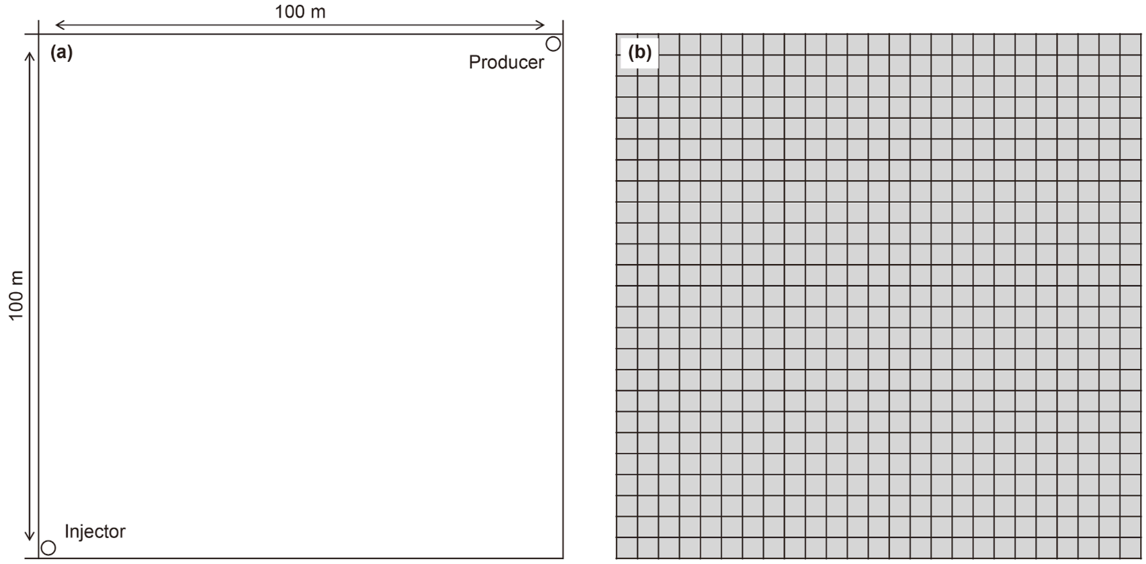

Fig. 3. Schematic of the homogeneous and heterogeneous cases (**a**) and their computational grids (**b**).

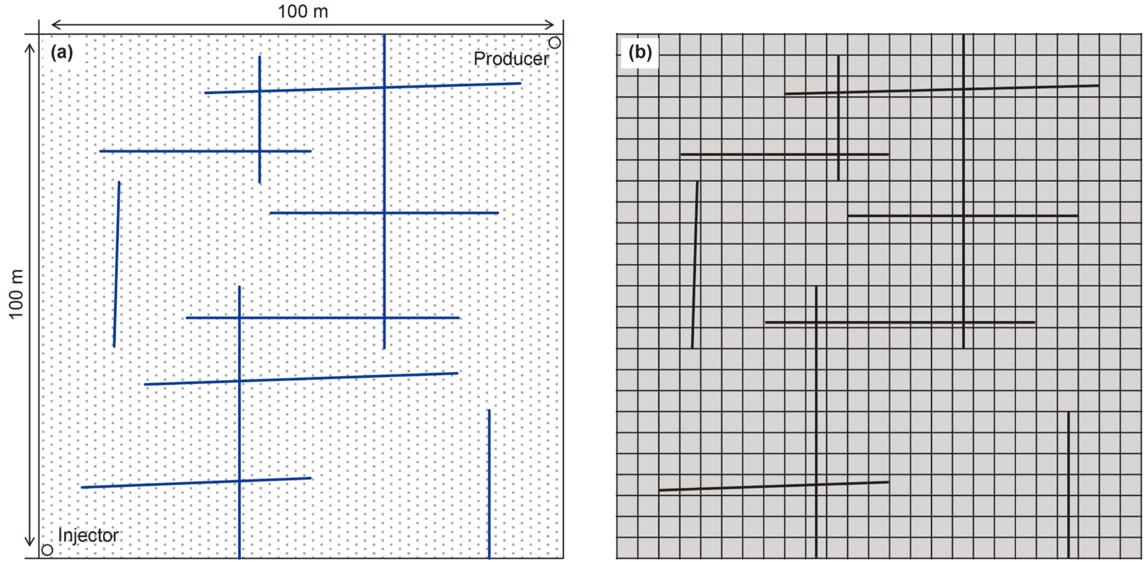

Fig. 4. Schematic of the fractured case (**a**) and its computational grids (**b**).

where P_{ori} is the original pressure; P_{scale} is the scaled dimensionless pressure; P_{int} is the initial formation pressure. The output pressure can be rescaled by multiplying P_{int}.

2.5. Visualization method of EPINN

The "encoder-decoder" architecture is a prevalent framework in NN design (Sutskever et al., 2014; Ronneberger et al., 2015; Vaswani et al., 2017), which maps data into a high-dimensional space to facilitate the manipulation of higher-level features. EPINN utilized this concept to design the architecture. The intricacy of high dimensionality complicates the intuitive comprehension of the importance of the learnable parameters at each layer. Nevertheless, the adjacency-location anchoring of EPINN enables the deduction of the relationship between specific positions in the intermediate outputs and the discrete grid coordinates in the physics model. For Eq. (17), we can deduce that

$$\tilde{P}_1^t[\,:,i] = w_1[\,:\,]\cdot P^t[i] + b_1[i] \tag{24}$$

where $\tilde{P}_1^t[\,:,i]$ represents the i-th column of $\tilde{P}_1^t$; $P_1^t[i]$ represents the i-th element of P_1^t; $w_1[:\,]$ is all elements of the weight term; and $b_1[i]$ is the i-th bias term. From this, we can deduce that each column of $\tilde{P}_1^t$ corresponds to the pressure of each grid location in P_1^t. Likewise, following Eqs. (18)–(21), we can deduce that $P_1^t[\,:,i]$, $\tilde{P}_2^t[\,:,i]$, and $P_2^t[\,:,i]$ contain similar correspondences.

To gain a more intuitive comprehension, we visualized the outputs of layers utilizing the principal component analysis (PCA) (Abdi and Williams, 2010) to extract primary features in $\tilde{P}_1^t$, P_1^t, $\tilde{P}_2^t$, and P_2^t. For matrix A, the singular value decomposition (Sadek, 2012) is

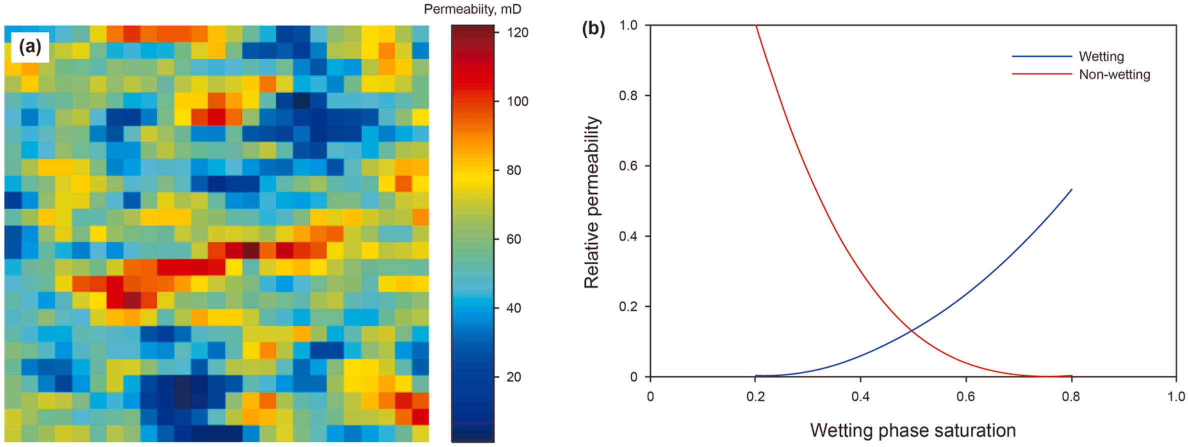

Fig. 5. The permeability distribution in the heterogeneous case (**a**) and relative permeability curve of homogeneous, heterogeneous, and fractured cases (**b**).

Table 1
The parameter setting of different physical cases.

Case	Rock properties			Fluid properties		Schedule
	Permeability	Porosity	Pressure	Wetting phase	Non-wetting phase	
Homogeneous	50 mD	Matrix porosity: 0.25	P_{int}: 25 MPa Matrix compressibility: 1.0×10^{-8} Pa^{-1}	Initial saturation: 0.2 Minimum saturation: 0.2 Density: 1000 kg/m^3 Viscosity: 1.0 mPa·s Compressibility: 1.0×10^{-10} Pa^{-1} Relative permeability: Fig. 5(b)	Initial saturation: 0.8 Minimum saturation: 0.2 Density: 800 kg/m^3 Viscosity: 1.8 mPa·s Compressibility:1.0×10^{-10} Pa^{-1} Relative permeability: Fig. 5(b)	Injection rate: 2.0 m^3/d Production rate: 2.0 m^3/d Wellbore radius: 0.1 m Skin factor: 0
Heterogeneous	Fig. 5(a)					Injection rate: 2.0 m^3/d Production rate: 2.0 m^3/d Wellbore radius: 0.1 m Skin factor: 0
Fracture	Matrix: 10 mD Fracture: 30,000 mD	Matrix porosity: 0.25 Fracture porosity: 1.0 Fracture aperture: 0.01 m				Injection rate: 1.5 m^3/d Production rate: 1.5 m^3/d Wellbore radius: 0.1 m Skin factor: 0

$$A = U{\cdot}S{\cdot}V^{\mathrm{T}} \tag{25}$$

where U and V are orthogonal matrices; and S is a diagonal matrix with singular values. For each singular value σ_i, we compute the variance contribution

$$V_i = \frac{\sigma_i^2}{\sum\limits_{j=1}^{n} \sigma_j^2} \tag{26}$$

We retain the cumulative variance with a variance retention threshold C_{t} (95% in this paper):

$$C_{\mathrm{t}} = \sum_{i=1}^{N_{\mathrm{c}}} V_i \tag{27}$$

The first N_{c} principal components are used to reconstruct the matrix (Abdi and Williams, 2010)

$$\widehat{A} = U_{N_{\mathrm{c}}} S_{N_{\mathrm{c}}} V_{N_{\mathrm{c}}}^{\mathrm{T}} \tag{28}$$

where $U_{N_{\mathrm{c}}}$ and $V_{N_{\mathrm{c}}}$ are the first N_{c} columns of U and V, respectively; and $S_{N_{\mathrm{c}}}$ is submatrix of the top-left $N_{\mathrm{c}} \times N_{\mathrm{c}}$ in S.

2.6. Observations and interpretations of intermediate outputs

In Figs. 6–8, we have the following observations: from P^t to $\tilde{P}_1^t$, each column of $\tilde{P}_1^t$ exhibits internal correlation, and the distinct columns are different in color. From $\tilde{P}_1^t$ to P_1^t, we can observe the construction of the adjacent connection relation in P_1^t with the diagonal matrix forms. From P_1^t to $\tilde{P}_2^t$, although PCA has been applied, the global distribution of $\tilde{P}_2^t$ remains difficult to comprehend intuitively. From $\tilde{P}_2^t$ to P_2^t, P_2^t again exhibits differences between different columns. Significantly, there exists a correlation between columns (or differences between rows), resulting in a grid-like distribution that is different from $\tilde{P}_1^t$. Furthermore, for the fractured case, $\tilde{P}_2^t[625:763,:]$ (corresponding to the unstructured grid for fractures) differs significantly from $\tilde{P}_2^t[1:625,:]$. From P_2^t to $\tilde{P}_3^t$, $\tilde{P}_3^t$ exhibits a distribution analogous to that of P^t.

Since each column of 625 × 625 matrices ($\tilde{P}_1^t$, P_1^t, $\tilde{P}_2^t$, P_2^t) corresponds to an element in the 1 × 625 vector (P^t, $\tilde{P}_3^t$), we reshape $P_1^t[:,1]$, $P_1^t[:,313]$, and $P_1^t[:,625]$ into 25 × 25 matrices, as shown in Fig. (9) (for the fractured case, only the first 625 elements are reshaped). The following observations are obtained: There is an observable construction of non-equilibrium potential. The high-potential areas of the $P_1^t[:,1]$, $P_1^t[:,313]$, and $P_1^t[:,313]$ are located at the lower left corner, central grid, and upper right corner, respectively, aligning with their original spatial coordinates. The potential distribution of homogeneous and heterogeneous cases is similar. The areas of low potential in Fig. 9(m)–(o) correspond with the spatial distribution of fractures in Fig. 4. The locations where fractures intersect correspond exactly to areas

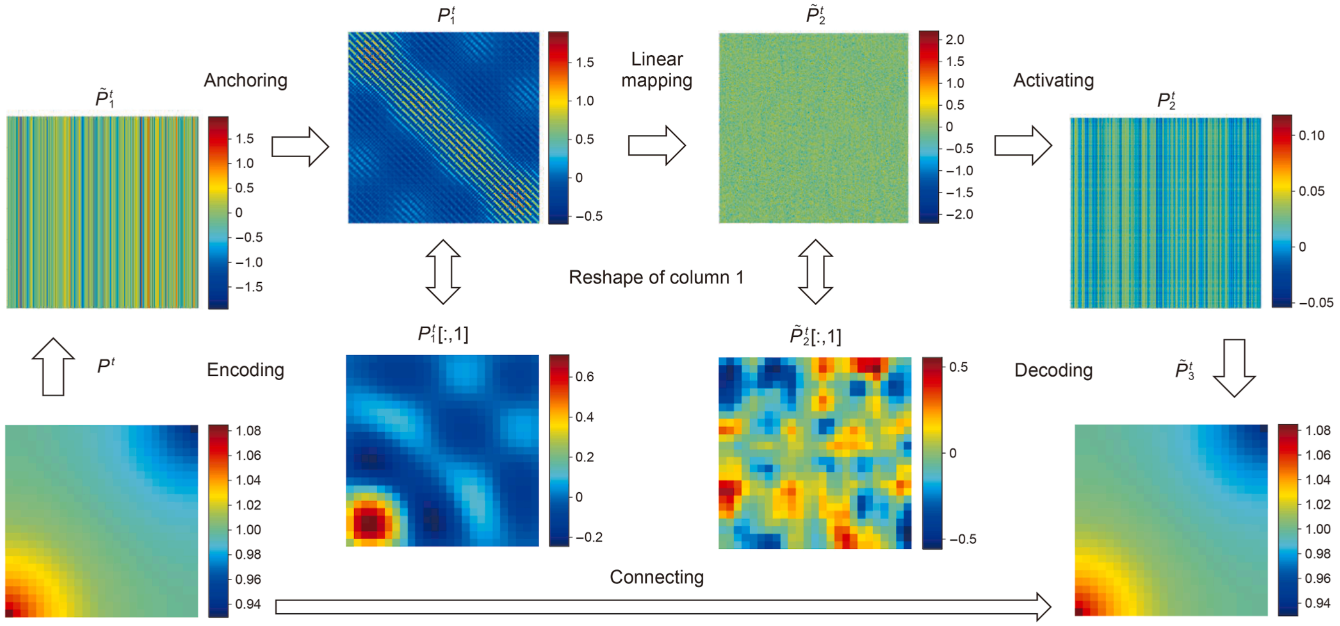

Fig. 6. Visualization of homogeneous case at 100 d.

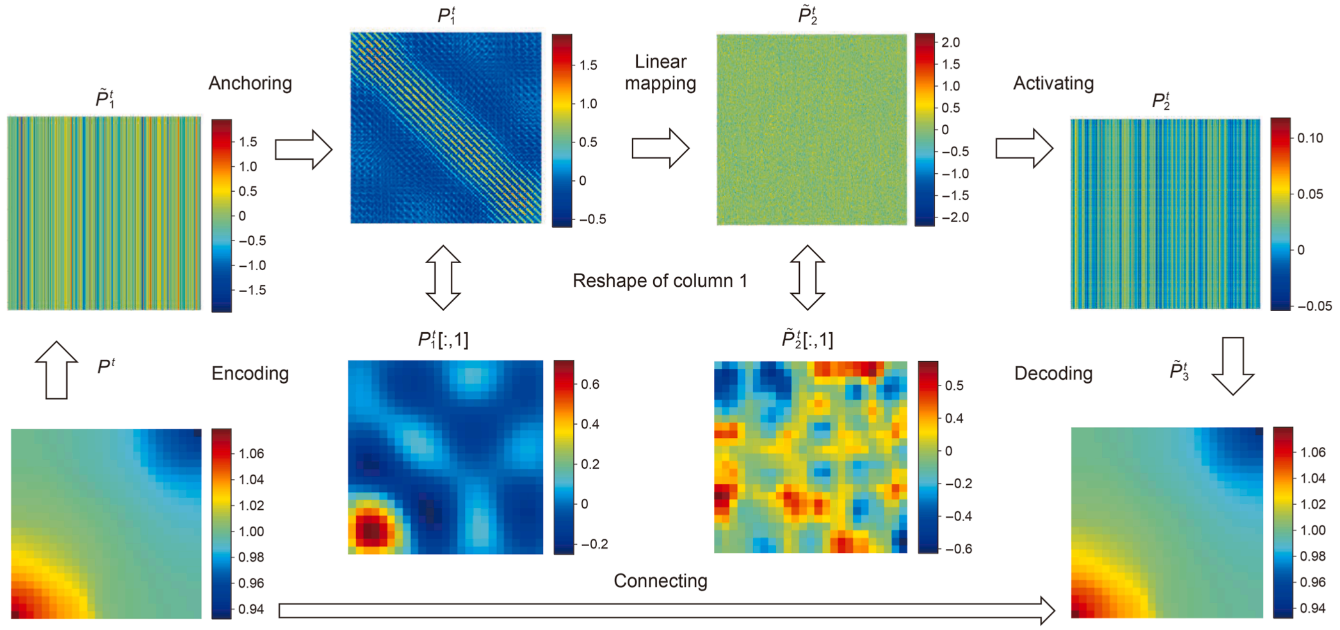

Fig. 7. Visualization of heterogeneous case at 100 d.

of lower potential, which aligns with the typical pressure distribution pattern (fracture intersections usually have lower pressure). We also reshape the $\tilde{P}_2^t[:,1]$, $\tilde{P}_2^t[:,313]$, and $\tilde{P}_2^t[:,625]$ in Fig. 9. It is observable that some high-level spatial features have been extracted. The distribution of the finer features extracted from the homogeneous resembles that of the heterogeneous case.

Based on the observations, we present the following interpretation: The input layer maps the feature of each element to a high-dimensional space with the number of mesh grids. Thereafter, the anchoring operation softly informs the NN of the specific element connections, and a normalization operation is performed (Eq. (18)), akin to methods for accelerating convergence (Ioffe and Szegedy, 2015; Ba et al., 2016; Santurkar et al., 2018), wherein each column is adjusted to reduce covariate shifts and establish a symmetrical, equal form; thus P_1^t is essentially a synthesized form of pressure features and the adjacency information. The subsequent step involves mapping in the high-dimensional space, which captures pressure variations at a finer granularity and is followed by the activation operation to filter out features with low relevance, thus establishing long-range dependencies. A similar positional dependency pattern is also observed in the self-attention mechanism (Shaw et al., 2018; Zhao et al., 2020; Guo et al., 2023), which allows the model to consider the global effects of all elements in the operation process of a specific element

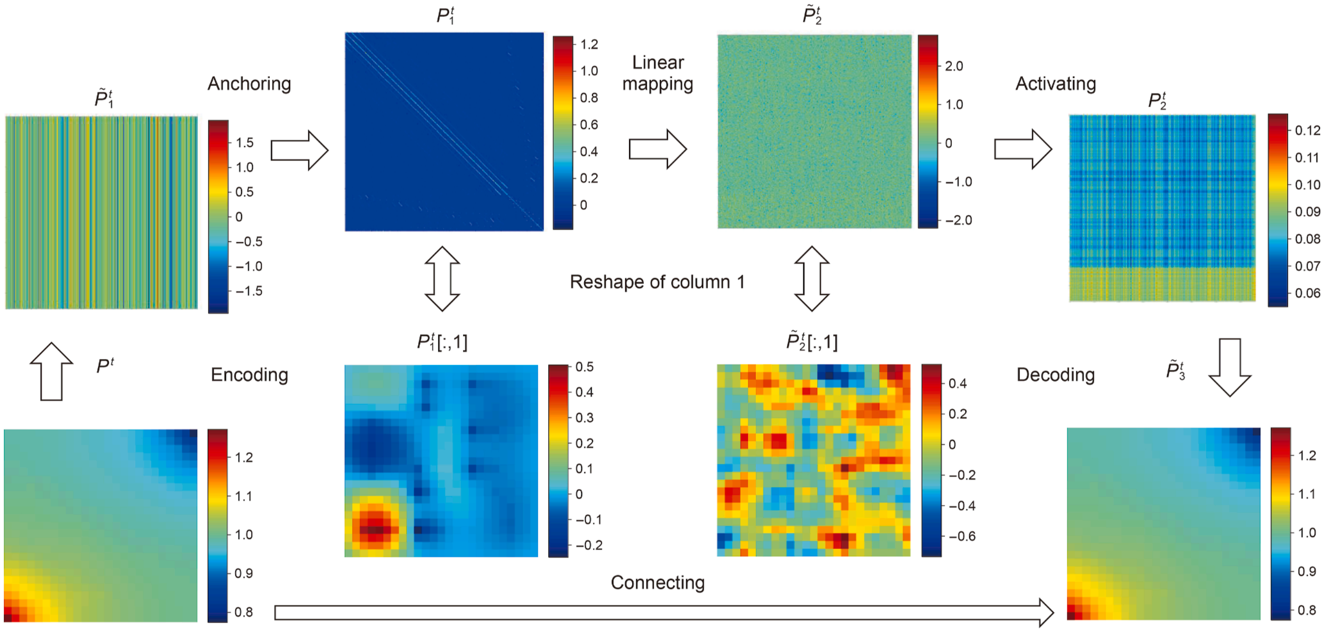

Fig. 8. Visualization of fractured case at 100 d.

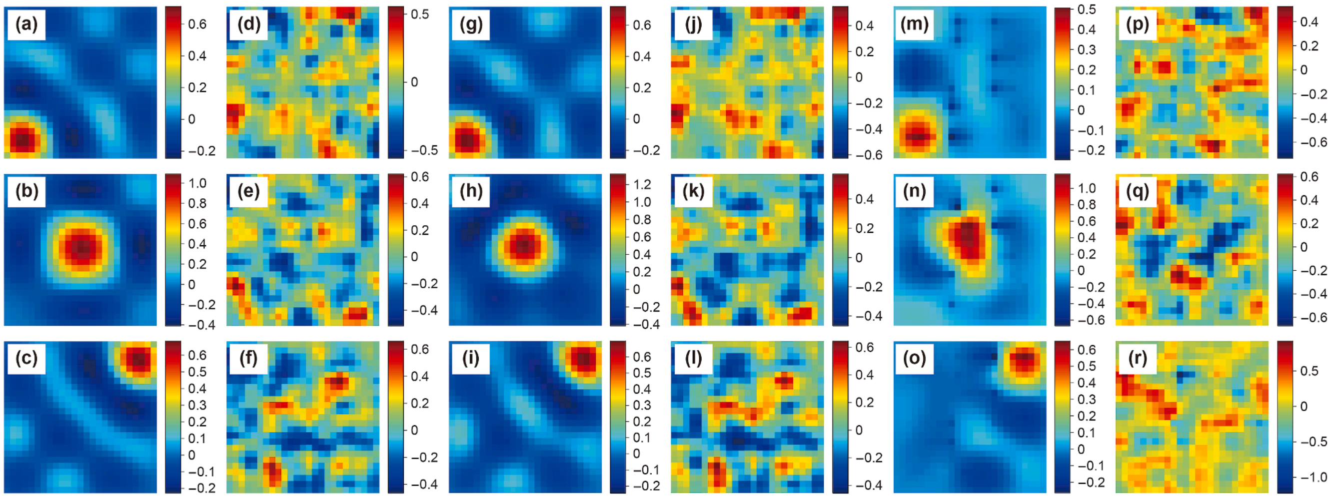

Fig. 9. Reshaped results of the intermediate matrices for homogeneous case (**a**–**f**), heterogeneous case (**g**–**l**), and fractured case (**m**–**r**), with subfigures for each physical problem sequentially representing rows 1, 313, and 625 of the P_1^t, and rows 1, 313, and 625 of the $\tilde{P}_2^t$ at 100 d.

and is regarded as an efficient learning approach. Ultimately, P_2^t is projected back to the original dimension by the output layer, yielding a final result that is a weighted average of $\tilde{P}_3^t$ and P^t.

In summary, different layers of the physics-driven EPINN exhibit varying operational granularities during the learning process and demonstrate relatively decomposed functionalities, resembling the detached characteristics observed in (Equitz and Cover, 1991; Mehta and Schwab, 2014; Zeiler and Fergus, 2014). Moreover, the distinct functionalities of layers combined with the corresponding physical parameters (which exhibit either time-dependent attributes such as pressure features, or static attributes such as geometrical information and adjacency relations) suggest that different layers may vary in scalability to enhance long-term convergence. In other words, while uniform layers' updates enable rapid short-term gradient descent, prioritizing the early training of challenging key modules may improve long-term performance. However, in simulating unsteady two-phase flow problems, EPINN approximates PDE solutions at each time step through uniform parameter updates to achieve stepwise convergence without accounting for long-term effects. In Section 3, we will propose a novel learning strategy to guide EPINN training toward emphasizing scalable components.

3. Layer-specific constraint learning for EPINN

This section introduces a novel layer-specific constraint learning strategy for EPINN, assesses its performance in homogeneous, heterogeneous, and fractured cases, and provides post hoc interpretations of the learning process under the new strategy.

3.1. Constraint learning strategy

As a progressive learning approach across multiple time steps, continual learning strategies appear to be a suitable reference for enhancing EPINN. The objective of continual learning (Lange et al., 2022; Wang et al., 2024) is to achieve gradual improvement in model performance in multiple single-stage learning processes, aiming to satisfy the objectives of each independent process while enhancing global generalization. Similar concepts include sequential learning (McCloskey and Cohen, 1989; Aljundi et al., 2018), incremental learning (Gepperth and Karaoguz, 2016; Chaudhry et al., 2018a), and lifelong learning (Aljundi et al., 2019; Chaudhry et al., 2018b), with implementations being: replay-based approach (Shin et al., 2017; Chaudhry et al., 2018b), where samples of old tasks (or models generated from them) are stored and replayed while learning new tasks; regularization-based approach (Aljundi et al., 2017; Kirkpatrick et al., 2017), which impose soft loss constraints by maintaining a per-parameter sensitivity measurement or directly attaching hard attention masks to update a subset of NN; architecture-based approach (Mallya and Lazebnik, 2018; Serra et al., 2018; Douillard et al., 2022), which carve out distinct effective zones for each task or treat different layers as distinct modules to either share information or fit specific tasks; and reinitialization-based methods, which enhance the performance of underutilized units to improve the model's long-term performance (Dohare et al., 2024).

We propose mitigating catastrophic forgetting by suppressing updates in layers that are more sensitive to local overfitting at each current or recent time step. It should be noted that weights and biases are considered as different zones since they generally execute distinct “scaling” and “shifting” functions (Goodfellow et al., 2018; Williams et al., 2024). This concept of suppression resembles the integration of model decomposition and regularization (Kirkpatrick et al., 2017; Douillard et al., 2022): In multistep learning, each time step is regarded as a subtask, whereas the continuous learning over multiple time steps constitutes a generalized joint task. Different NN layers are considered as different task-sharing or task-specific components. Our objective is to utilize regularization techniques to establish a relatively stable task-sharing component during the gradient descent optimization, thereby enhancing the model's generalization in multistep solving.

Section 2 reveals that different layers have distinct functions and operational granularities; however, accurately determining which layers are more advantageous for long-term performance based on prior knowledge remains difficult. The dynamic quantification of model parameter sensitivity to previous tasks via the fisher information matrix (FIM) (Kirkpatrick et al., 2017) offers a viable method for adaptive adjustment. For learnable parameters θ in EPINN, the elements of the FIM can be calculated with

$$F_{ij} = \mathbb{E}\left[\frac{\partial \log \mathscr{L}(y|x;\theta)}{\partial \theta_i}\frac{\partial \log \mathscr{L}(y|x;\theta)}{\partial \theta_j}\right] \tag{29}$$

where $\mathbb{E}[\cdot]$ is the expectation; $\mathscr{L}(y|x;\theta)$ is the likelihood function for the output y given the x and θ; θ_i and θ_j are i-th and j-th parameters. Eq. (29) can be expressed with the score function as

$$F = \mathbb{E}[s(\theta;x,y)s(\theta;x,y)^{\top}] \tag{30}$$

where the score function $s(\theta)$ is defined as

$$s(\theta) = \frac{\partial \log \mathscr{L}(y|x;\theta)}{\partial \theta} \tag{31}$$

Each time step is considered as a time-sampling point, we can derive the ascertain of sensitivity approximation for all parameters across all time steps is

$$\widehat{F} = \frac{1}{N}\sum_{n=1}^{N} s(\theta;x^n,y^n)s(\theta;x^n,y^n)^{\top} \tag{32}$$

where N is the total number of time steps; and n is the index. Given that the NN employed herein solves incrementally with NN parameters and pressure fields exhibiting slight variations between adjacent time steps (Yan et al., 2024), we adopted a moving average for the most recent N_t time steps to approximate the sensitivity of θ to the loss function of recent N_t time step. Consequently, the FIM at $t+1$ is calculated with

$$\widehat{F}^{t+1} = \frac{1}{N_t}\sum_{n=t-N_t+1}^{t} s\left(\theta^n;x^{n+1},y^{n+1}\right)s\left(\theta^n;x^{n+1},y^{n+1}\right)^{\top} \tag{33}$$

where $t \in \{1,2,3,\ldots,N\}$. When $t < N_t$, only available samples are used, and Eq. (33) is modified as

$$\widehat{F}^{t+1} = \frac{1}{t}\sum_{n=1}^{t} s\left(\theta^n;x^{n+1},y^{n+1}\right)s\left(\theta^n;x^{n+1},y^{n+1}\right)^{\top} \tag{34}$$

Only the diagonal elements of the FIM are considered, and Eqs. (33) and (34) are consolidated into

$$\widehat{F}_{ii}^{t+1} = \frac{1}{\min(t,N_t)}\sum_{n=\max(t-N_t+1,1)}^{t}\left(\frac{\partial \log \mathscr{L}(y^{n+1}|x^{n+1};\theta^n)}{\partial \theta_i^n}\right)^2 \tag{35}$$

where $\widehat{F}_{ii}^{t+1}$ represents the value of the i-th diagonal element (corresponding to the i-th NN parameter). The gradient of each parameter obtained through NN backpropagation reflects its affection for the prediction error. Thus, we consider the approximated FIM is

$$\widehat{F}_{ii}^{t+1} = \frac{1}{\min(t,N_t)}\sum_{n=\max(t-N_t+1,1)}^{t}\left(\frac{\partial L^{n+1}}{\partial \theta_i^n}\right)^2 \tag{36}$$

where L^{n+1} is the loss of NN prediction. We take the fundamental zones as the basic operational units, the average value of the l-th zone is

$$\overline{F}^{t+1,(l)} = \frac{1}{{N_\mathrm{d}}^{(l)}}\sum_{i=1}^{{N_\mathrm{d}}^{(l)}} \widehat{F}_{ii}^{t+1,(l)} \tag{37}$$

where ${N_\mathrm{d}}^{(l)}$ is the number of parameters in l-th zone. By sorting all zones from largest to smallest, we obtain a list S^t. According to the pre-set threshold T_r ($0 \le T_\mathrm{r} \le 1$, set to 0.5 in this paper), the mask is generated as

$$M^{t+1,(l)} = \begin{cases} 0 & \text{if layer } l \text{ is among the top } \lceil T_\mathrm{r} \times N_\mathrm{o} \rceil \text{ layers in } S^t \\ 1 & \text{otherwise} \end{cases} \tag{38}$$

where N_o is the total number of operational zones. Finally, the parameter update rule of EPINN-CL is

$$\theta_i^{t+1,(l)} = \theta_i^{t,(l)} - \widehat{\eta}\cdot M^{t+1,(l)}\cdot \widehat{g}_i^{t,(l)} \tag{39}$$

where $\widehat{\eta}$ is the learning rate; $\widehat{g}$ is the gradient.

The foundation of this constraint learning strategy is essentially a soft prior constraint introduction. It incorporates the observation from Section 2 (that parameters within the same layer serve similar functions and act on different time-dependent and static parameters) to formulate the hypothesis that diverse NN layers exhibit different sensitivities to local learning and global learning. Specifically, we adopt the approach from (Aich, 2021), which uses the squared gradients of the loss function for parameters (approximating the diagonal elements of the FIM) to reflect their sensitivity to the current timestep's solving task. We extend this method to multi-step continuous solving scenarios, employing a moving average (Eq. (36)) to capture and accumulate layer-wise sensitivities over recent short-term periods smoothly. Ultimately, through layer-specific masks (Eq. (39)), we guide the model to suppress updates in layers showing higher short-term sensitivity, thereby facilitating the learning of components with long-term capabilities.

In Fig. 10, for each time step, during the first propagation, the θ^t (Eqs. (17)–(22)) mapping P^t to P_*^{t+1}. Following the computation of R^t (Eq. (10)) and the MSE (Eq. (16)), thus the loss L_*^{t+1} is obtained. The calculated gradient G_*^{t+1} is back-prop through θ^t to update the F^t to F^{t+1}, which is used to generate a new mask M^{t+1} and the masked θ_*^{t+1}, thereby concluding the first propagation (performed only once at the beginning of each time step to update the mask). The rest of propagations' forward-prop to L_*^{t+1} is identical to the first propagation, whereas during back-prop, only the unmasked parameter of θ_*^{t+1} are updated until the convergence criteria is satisfied. Therefore, the zones crucial for maintaining historical memory are suppressed at each time step, guiding the development of generalization ability. Specifically, under constrained learning algorithms, various layers preserve their distinct functional roles in dimensionality adjustment, feature extraction, and information compression during forward propagation mapping. While in backpropagation, updates to layers exhibiting higher sensitivity to local solutions are suppressed. This strategy achieves dual objectives: 1) By limiting updates to gradient-sensitive layers, the algorithm enforces extended training on hard-to-train layers. 2) The suppression mechanism enables knowledge retention and sharing between time steps. This approach inherently incorporates the regularization effect; the NN is guided to develop generalization capabilities across multiple time steps while maintaining physical consistency. In the unsteady two-phase flow, the cooperation between layer-specific functional preservation and selective update suppression enables the model to transcend single-step pressure field approximation. Instead, it evolves into a predictive model with forecasting capabilities. At the onset of each time step, the NN output pressure aligns closer to the convergence threshold, and as the NN is more thoroughly trained, it demonstrates a progressive improvement in convergence. This constraint learning strategy does not directly adopt existing continual learning algorithms that aim to preserve optimal performance at historical time steps. It concentrates on gradually improving subsequent steps through accumulated shared features, emphasizing the progressive enhancement of EPINN's convergence over time. Additionally, this strategy offers a distinct advantage: it allows for an interpretable regulation process based on observing the suppression and activation of different layers, thereby enhancing the understanding of the operational mechanism.

We evaluate EPINN-CL on the homogeneous, heterogeneous, and fractured cases outlined in Section 2.3. In homogeneous and heterogeneous cases (excluding unstructured grids), we also compared the new method with the PICNN (Zhang et al., 2023). Table 2 summarizes the key characteristics of the evaluated methods, including network, input and output, prior embedding configurations, trainable parameter counts, and training strategies. All trainings are conducted on an NVIDIA GeForce RTX 4060 GPU, with each case running for 1000 d (100,000 time steps). The hyperparameter settings of the NNs are provided in Appendix A.

3.2. Convergence speed comparison

In Figs. 11–13, EPINN-CL significantly improved the convergence speed in all cases. For the first 1000 time steps, EPINN-CL exhibited a gradual acceleration in the homogeneous and heterogeneous cases; in the fractured case, EPINN-CL exhibited the fastest speed from the outset. Throughout all 100,000 time steps, the convergence of EPINN-CL exhibited greater stability, with the performance sustained after increasing to a superior speed. Conversely, EPINN exhibited fluctuations in the homogeneous and heterogeneous cases. During suppression and activation processes, the layerNorm weights, α, output layer weights, and output layer biases were consistently suppressed for all cases; ω_1 was also persistently suppressed in the fractured case; the hidden layer weights and ω_1 exhibited dynamic suppression and activation in homogeneous and heterogeneous cases, and there was a complementary relationship between them. It is evident that the NN's performance was unaffected, although only the bias was learned. A recent study also observed a similar phenomenon (Williams et al., 2024). Under the guidance of the constraint, EPINN-CL consistently tended to suppress alpha and the output layer (both weights and biases) entirely in all cases. The α controls the feature filtration threshold; the output layer controls the mapping from $\tilde{P}_2^t$ to $\tilde{P}_3^t$ (Eqs. (20) and (21)), corresponding to the transform from high-dimensional space back to the original space, typically involving feature fusion and aggregation. This position resembles the bottleneck in IB theory. The results suggest that adaptive suppression in specific layers have no negative impact and improved the model's convergence, with EPINN-CL tending to maintain a stable information compression approach.

Furthermore, this fixed compression mode reveals the following characteristics: 1) cross-time-step sharing of information compression modules, 2) enhanced training of feature extraction modules, and 3) latent regularization effects. Specifically, experiments demonstrate that even with fixed parameters in the information compression module, the model maintains

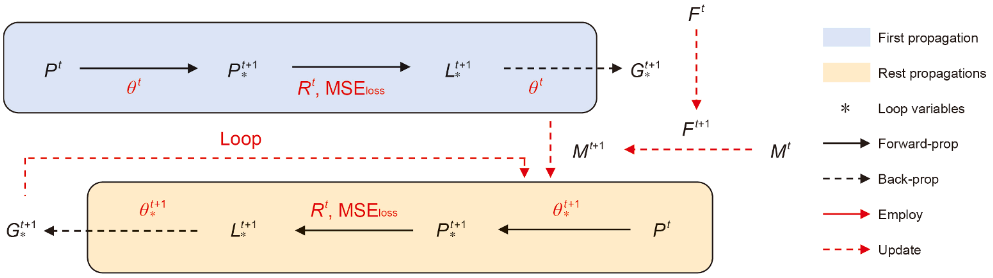

Fig. 10. The constraint learning procedure of EPINN-CL.

Table 2
Comparison of different NN methods.

Method	Network		Input & output	Prior embedding	Trainable parameters	Training strategy
	Function	Module				
PICNN	Feature extraction	Convolutional layer	Pressure field matrix	N/A	Structured grids: 1137325	Constraints: N/A
	Feature extraction	Convolutional layer wit			Unstructured grids: N/A	
	Decoding	Fully connected layer				
EPINN	Encoding	Fully connected layer	Pressure field matrix	Adjacency location	Structured grids: 394379	Constraints: N/A
	Anchoring	Adaptive layer & LayerNorm			Unstructured grids: 586751	
	Feature extraction	Fully connected layer				
	Decoding	Fully connected layer				
	Skip connection	Adaptive layer				
EPINN-CL	Encoding	Fully connected layer	Pressure field matrix	Adjacency location	Structured grids: 394379	Constraint: layer-specific
	Anchoring	Adaptive layer & LayerNorm			Unstructured grids: 586751	
	Feature extraction	Fully connected layer				
	Decoding	Fully connected layer				
	Skip connection	Adaptive layer				

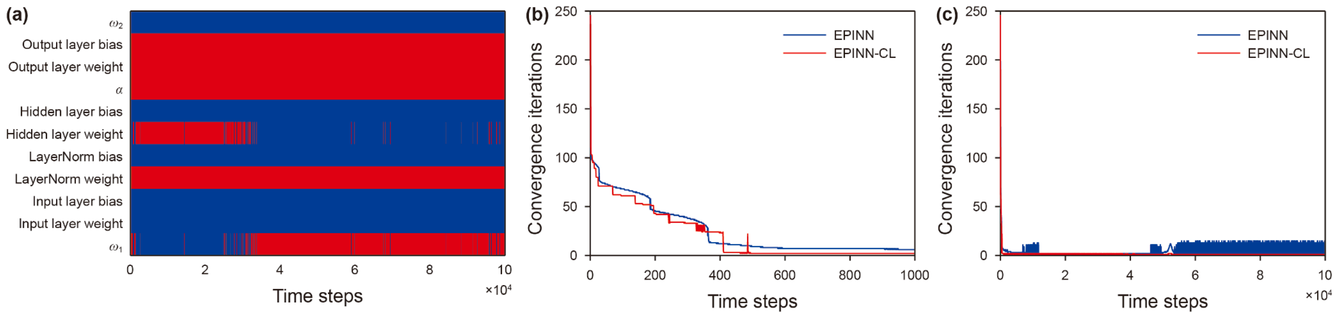

Fig. 11. Adaptive tuning results of EPINN-CL in the homogeneous case. (**a**) Suppression and activation process (red: suppression, blue: activation); (**b**) convergence iterations of the first 1000 time steps; (**c**) convergence iterations of all time steps.

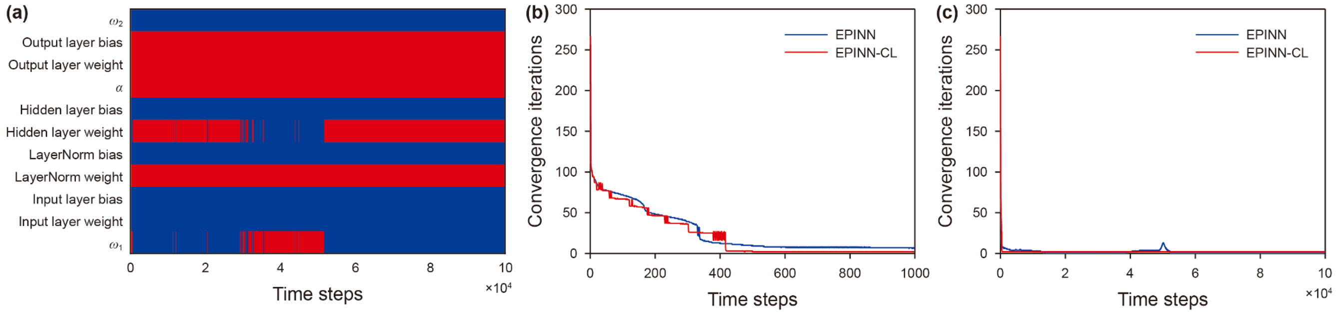

Fig. 12. Adaptive tuning results of EPINN-CL in heterogeneous case. (**a**) Suppression and activation process (red: suppression, blue: activation); (**b**) convergence iterations of the first 1000 time steps; (**c**) convergence iterations of all time steps.

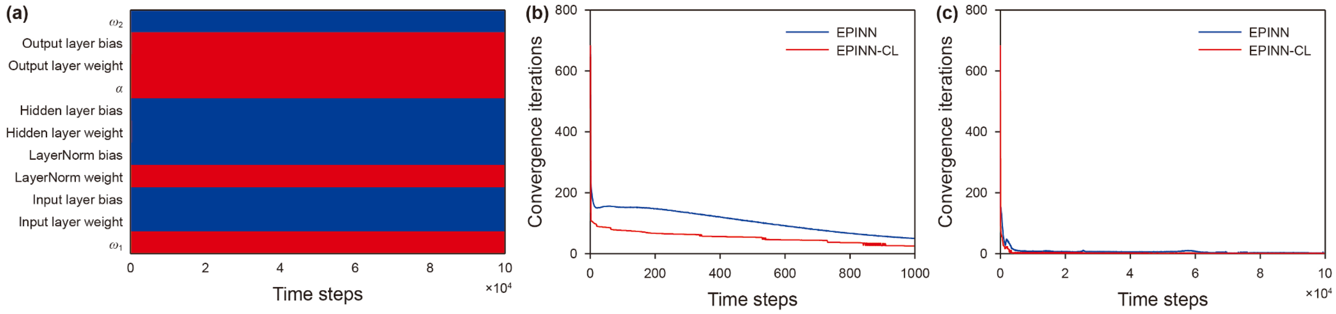

Fig. 13. Adaptive tuning results of EPINN-CL in fractured case. (**a**) Suppression and activation process (red: suppression, blue: activation); (**b**) convergence iterations of the first 1000 time steps; (**c**) convergence iterations of all time steps.

strong performance, indicating the time-independent universality of its functionality. This static component essentially establishes a unified compression criterion across time steps, ensuring that features extracted at different temporal stages adhere to the same specification and preventing feature drift. The constrained strategy reallocates computational resources to prioritize pressure feature extraction layers, allowing them to learn the intrinsic laws of physical fields more thoroughly. This division of labor optimization enables the feature extractor to develop robust capabilities, ensuring the consistent acquisition of feature representations that comply with the compression module's processing standards, even under dynamic variations in input fields. Fixing the compression module imposes implicit prior constraints, forcing the network to seek solutions within a limited parameter space that satisfy both physical laws and compression criteria. This dynamic balancing mechanism aligns with the core idea of regularization (improving generalization by restricting parameter space complexity and balancing training errors with model complexity), thereby mitigating the model's overfitting on short-term patterns observed during training.

3.3. Convergence accuracy comparison

In Fig. 14, all methods achieved high prediction accuracy for BHP of the injector and producer and the wetting phase fraction of the producer; the mean relative error is less than 0.1%. The accuracy of EPINN-CL's physical field is significantly higher than PICNN and EPINN in Figs. 15–17: The maximum error in the pressure predictions of EPINN-CL is under 0.1% in Figs. 15(g), 16(g), and 17(e). The maximum error in the saturation field predictions of EPINN-CL is under 1% in Figs. 15(n), 16(n), and 17(j). Notably, a comparison of the relative errors between EPINN and EPINN-CL (Figs. 15(f, g), 16(f, g) and 17(d, e)) reveals that the model demonstrates superior performance in the upper right corner when guided by the constraint strategy. This suggests that the NN is guided to focus more on the pressure characterization in the producer-nearing location.

3.4. Post hoc interpretation of learning process

In this section, we investigated the training dynamics across multiple steps using relative L_2 error, signal to noise ratio (SNR), and residual entropy (referred to as entropy). The relative L_2 error is utilized to evaluate the statistical accuracy (Anagnostopoulos et al., 2024; Yan et al., 2024)

$$\text{relative } L_2 \text{ error} = \frac{\| P^* - P_{\text{ref}} \|_2}{\| P_{\text{ref}} \|_2} \tag{40}$$

where $\| \cdot \|_2$ is the standard Euclidean norm; P^* is the NN output; and P_{ref} is the reference pressure. The SNR is employed to analyze the phase transition during multistep training (Shwartz-Ziv and Tishby, 2017; Goldfeld and Polyanskiy, 2020; Anagnostopoulos et al., 2024)

$$\text{SNR} = \frac{\frac{1}{N_{\text{P}}} \sum_{i=1}^{N_{\text{P}}} |G_i|}{\sqrt{\frac{1}{N_{\text{P}}-1} \sum_{i=1}^{N_{\text{P}}} \left(|G_i| - \frac{1}{N_{\text{P}}} \sum_{i=1}^{N_{\text{P}}} |G_i| \right)^2}} \tag{41}$$

where N_{P} is the number of parameters; and G_i represents the gradient of the i-th parameter. The SNR numerator represents the signal's strength, while the denominator represents the noise.

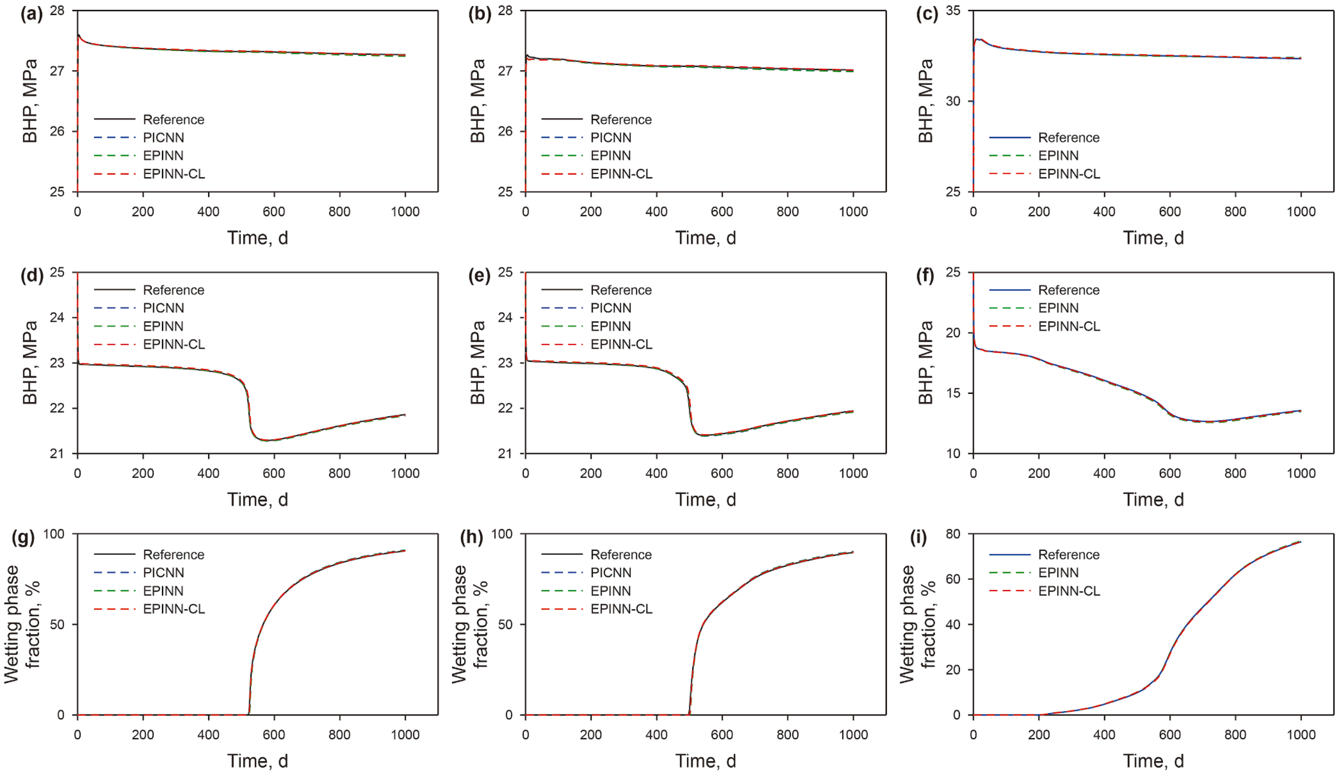

Fig. 14. BHP of injector (**a**–**c**) and producer (**d**–**f**), and wetting phase fraction of producer (**g**–**i**). Left: homogeneous; middle: heterogeneous; right: fractured.

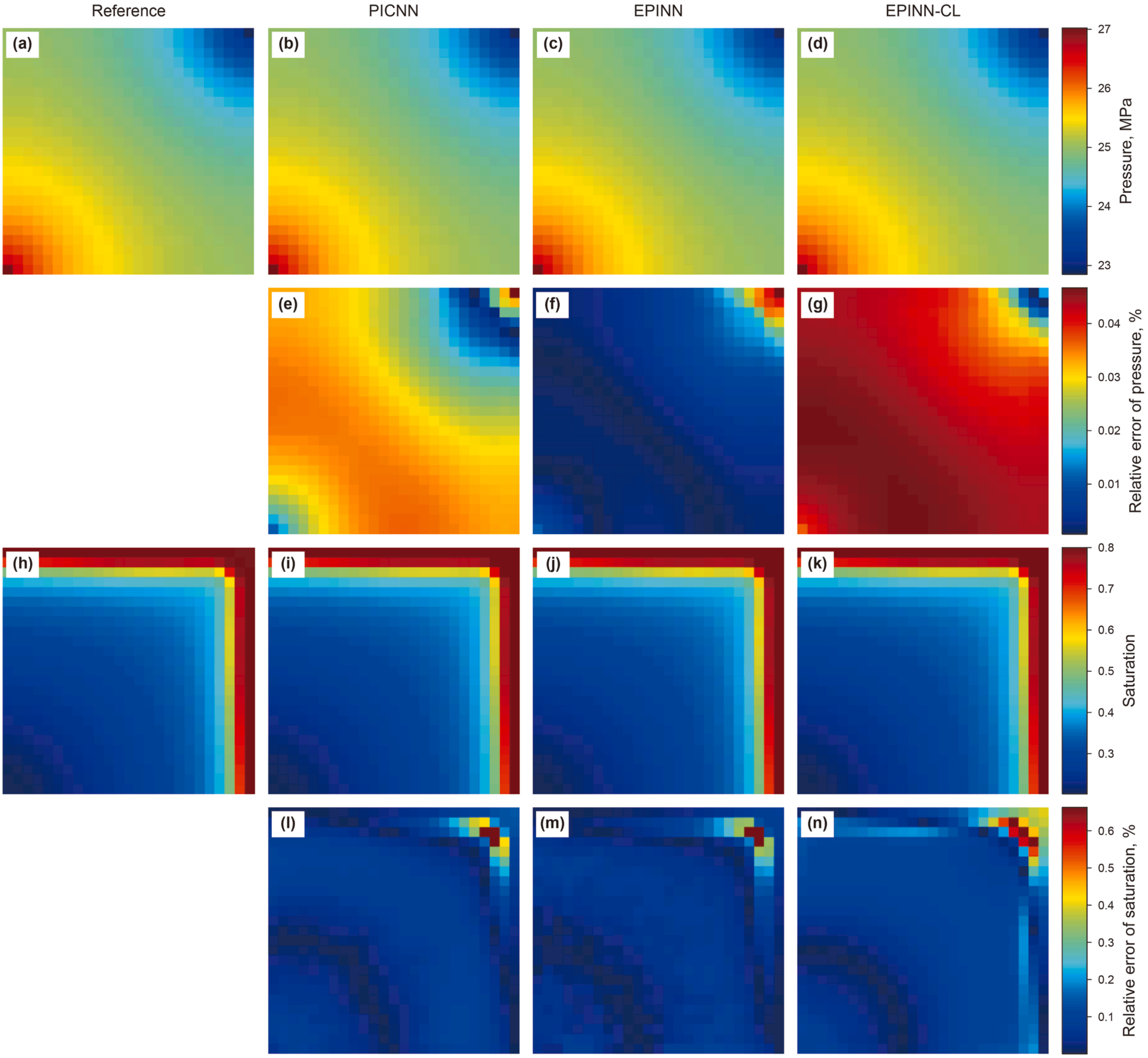

Fig. 15. Pressure distribution (**a**–**d**), relative error of pressure (**e**–**g**), saturation distribution (**h**–**k**), and relative error of saturation (**l**–**n**) in the homogeneous case at 500 d.

Within the same training process, a relatively higher SNR indicates a phase where the signal is relatively stable and the direction of the update is consistent; conversely, a lower SNR denotes a phase of learning from noise with greater randomness (Shwartz-Ziv and Tishby, 2017; Anagnostopoulos et al., 2024). In Shannon's theory, entropy is commonly utilized to assess the quality of information or, in physics, to evaluate the system's degree of disorder or randomness (Lesne, 2014). In this study, we define the Entropy to assess the uniformity of PDE residual

$$\text{Entropy} = -\sum_{i=1}^{N_b}\left(\frac{H[i]}{\sum_{j=1}^{N_b} H[j]}+\epsilon\right)\cdot\ln\left(\frac{H[i]}{\sum_{j=1}^{N_b} H[j]}+\epsilon\right) \tag{42}$$

where $H[i]$ and $H[i]$ represent the frequencies of the i-th and j-th bins, respectively, obtained by using a histogram to count the residuals of all grids at a time step; N_b denotes the total number of bins into which the histogram is divided (in this paper, set to 100); and ϵ represents a very small value (set to 1.0×10^{-10}) to avoid zero. A smaller entropy value suggests that residuals are predominantly clustered in fewer bins, denoting greater uniformity.

Fig. 18 shows that EPINN-CL exhibits the smallest convergence residual, consistent with the results in Fig. 19, where EPINN-CL demonstrates the highest statistical accuracy. Fig. 19 shows that the relative L_2 error for all cases is below 1.0×10^{-3}, combined with a slight increasing trend. This is attributable to the efficiency requirements in experiments (the fixed convergence threshold); increasing the iteration number per time step could mitigate this.

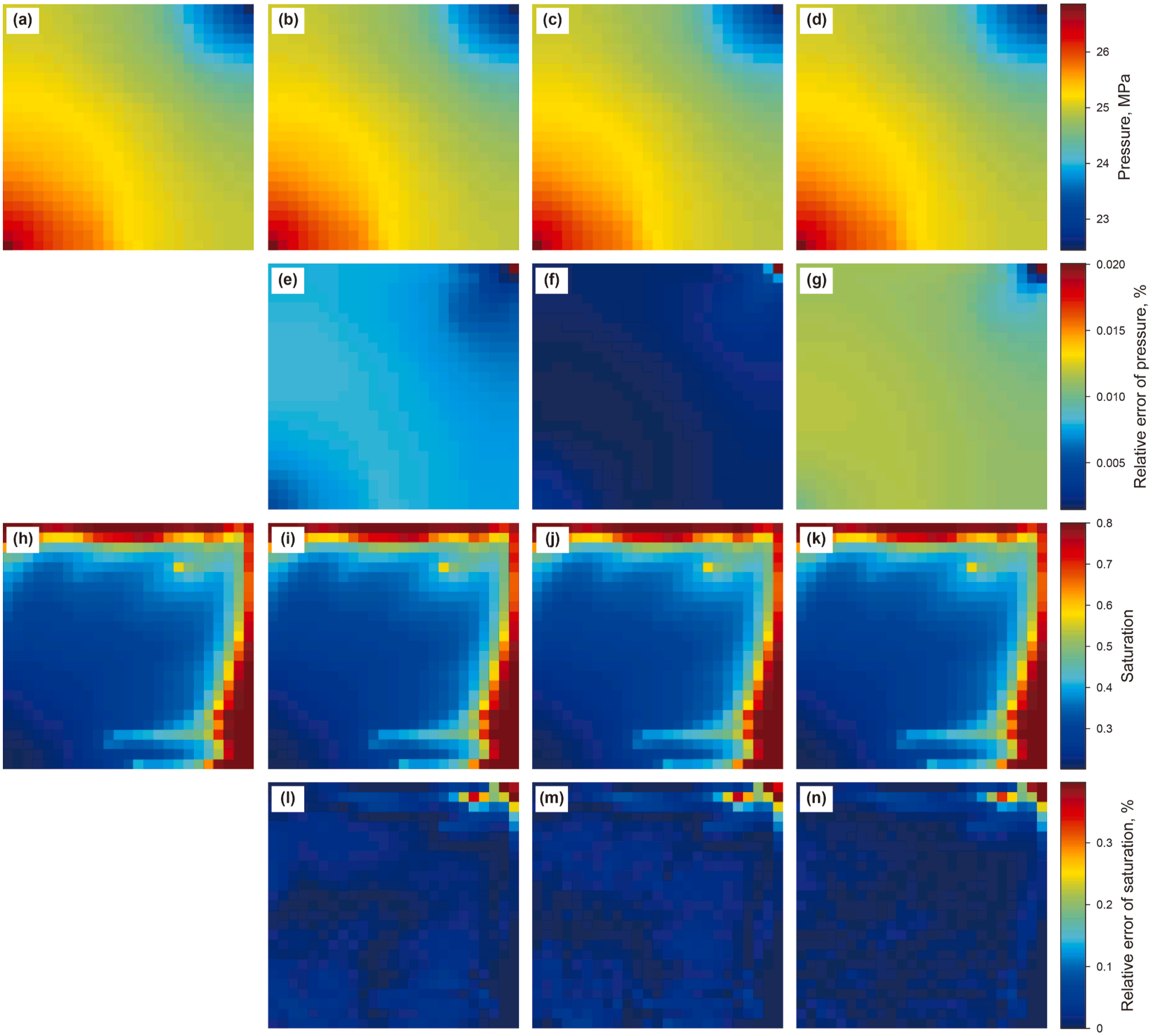

Fig. 16. Pressure distribution (**a**–**d**), relative error of pressure (**e**–**g**), saturation distribution (**h**–**k**), and relative error of saturation (**l**–**n**) in the heterogeneous case at 500 d.

Upon adequate training of the NNs, the activation states tend to be saturated, leading to the emergence of specific phase transition characteristics (Shwartz-Ziv and Tishby, 2017; Lewkowycz et al., 2020; Anagnostopoulos et al., 2024). Fig. 20 illustrates that EPINN-CL demonstrates clear phase boundaries in all cases. The first phase transition (a steep decline in SNR) of EPINN-CL is consistent with the transition to the phase of learning from noise, as interpreted in (Shwartz-Ziv and Tishby, 2017), which is a shift from a “fitting” phase with consistent update directions to a “diffusion” phase with more random updates. In subsequent training, the SNR of EPINN-CL demonstrates an increasing trend for the homogeneous case. It stabilizes at a high value, akin to the “total diffusion” described in (Anagnostopoulos et al., 2024), where this phase's emergence is considered an indicator of superior convergence. Although SNR rises after the first phase transition in more complex heterogeneous and fractured cases, it does not adhere to the standard three-phase characteristic. This is attributed to the intricate loss landscapes, necessitating the navigation of additional local optima to achieve a superior solution. In Fig. 21(a) and (b), the entropy of both EPINN and EPINN-CL is lower than that of PICNN in homogeneous and heterogeneous cases, indicating higher residual uniformity; EPINN initially has lower entropy than EPINN-CL but experiences a rapid decline in performance; conversely, the performance of EPINN-CL gradually improved, ultimately achieving the lowest Entropy. In Fig. 18(c), the uniformity of EPINN-CL's statistical residuals consistently surpasses that of EPINN. Overall, EPINN-CL exhibits superior convergence and more uniform residuals than EPINN and PICNN.

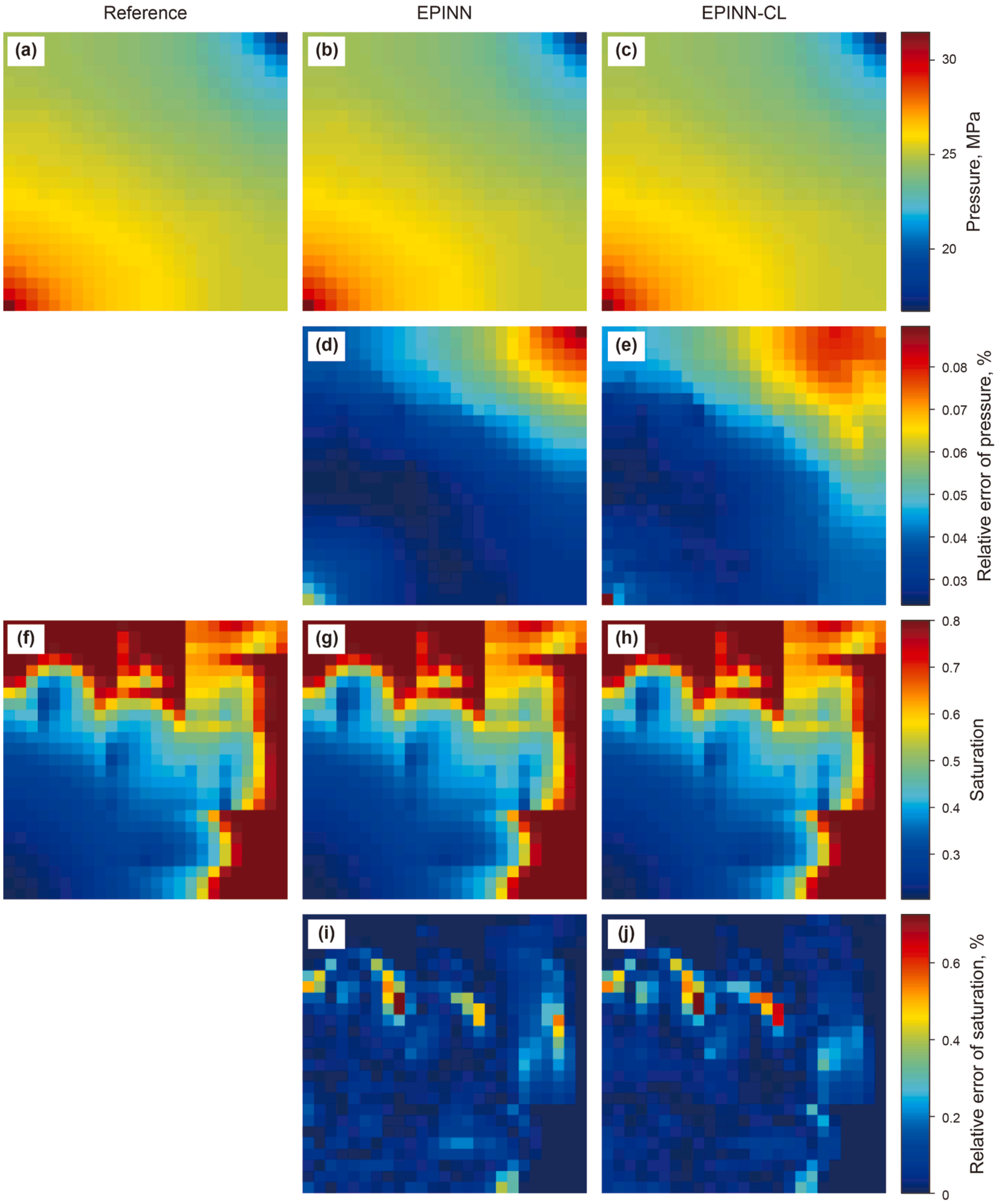

Fig. 17. Pressure distribution (**a**–**c**), relative error of pressure (**d**–**e**), saturation distribution (**f**–**h**), and relative error of saturation (**i, j**) in the fractured case at 500 d.

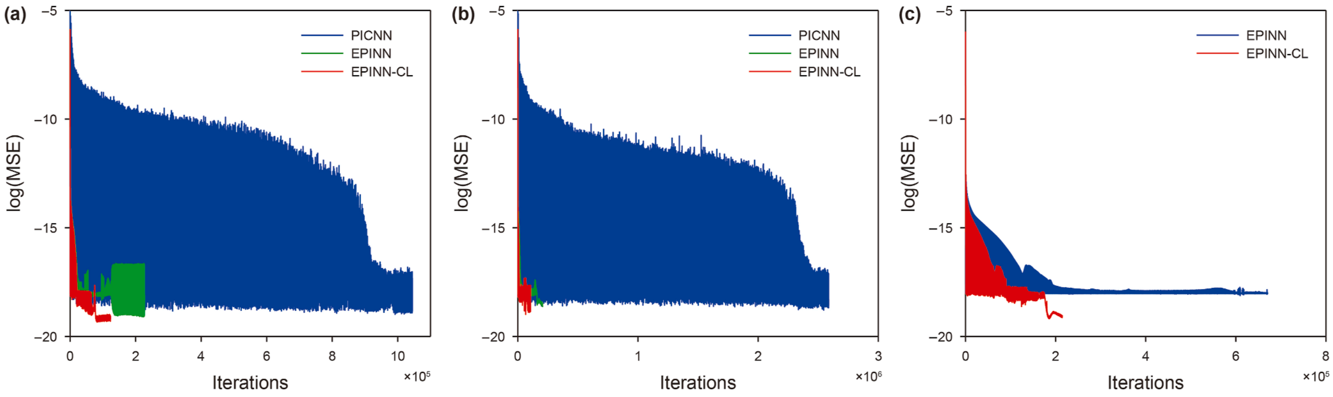

Fig. 18. Change of MSE across all iterations. (**a**) Homogeneous case; (**b**) heterogeneous case; (**c**) fractured case.

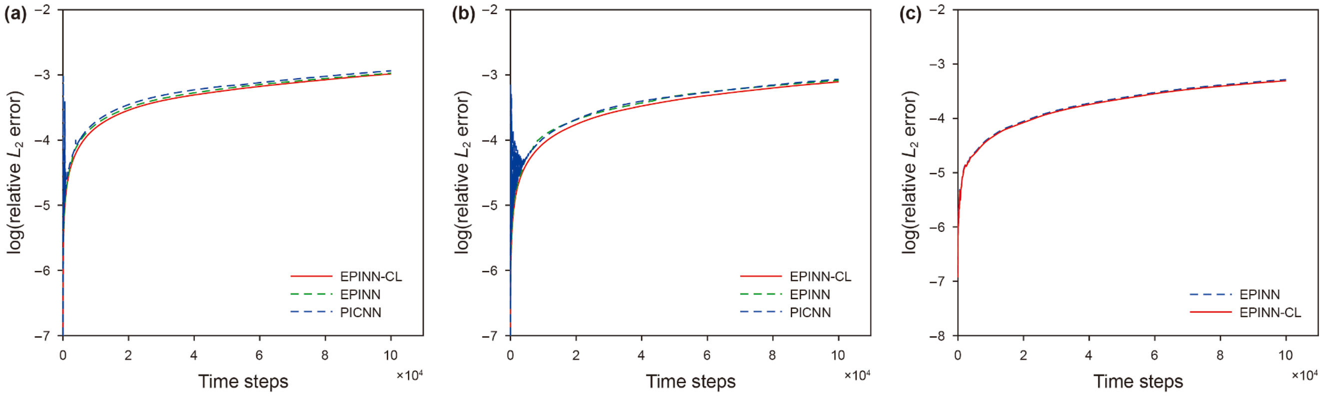

Fig. 19. Change of relative L_2 across all steps. (**a**) Homogeneous case; (**b**) heterogeneous case; (**c**) fractured case.

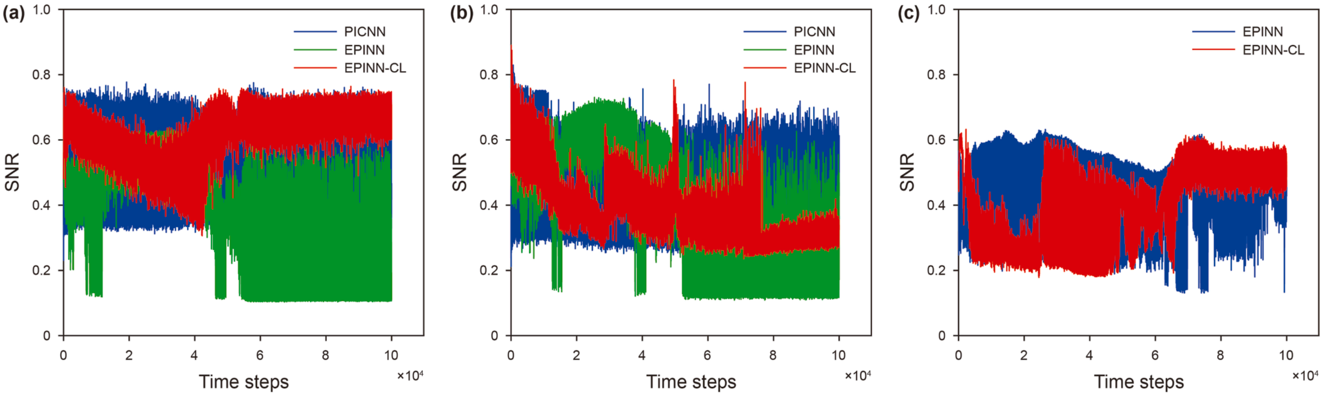

Fig. 20. Change of SNR across all steps. (**a**) Homogeneous case; (**b**) heterogeneous case; (**c**) fractured case.

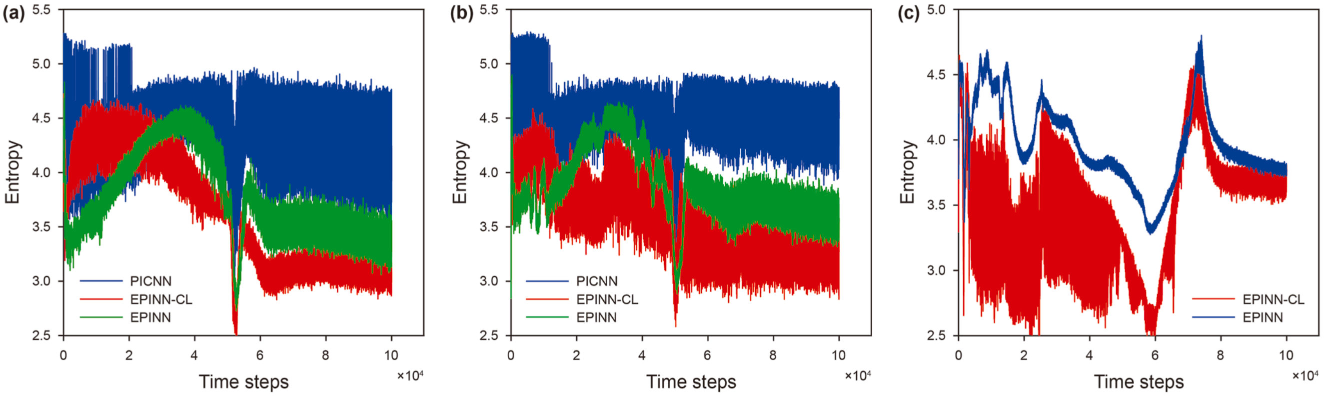

Fig. 21. Change of Entropy across all steps. (**a**) Homogeneous case; (**b**) heterogeneous case; (**c**) fractured case.

4. Conclusions

This study conducts a visualization investigation to explore the learning characteristics of the enriched physics-informed neural network (EPINN) in simulating two-phase flow. The correspondence between the coordinates of intermediate outputs and the original grid locations is deduced, facilitating the recognition of learned physical semantics and the long-range dependencies formed between discrete elements. Furthermore, we interpreted the distinct functionalities of different layers based on the observations: the input and output layers perform dimensionality increase and reduction; the anchoring layer emphasizes spatial connections with the adjacent information; the hidden layer executes finer-grained operations in high-dimensional space and eliminates redundant features with the activation function. Inspired by the exhibition of relatively decomposed functionalities, we developed the EPINN with constrained-layer (EPINN-CL). The novel method introduces layer-specific differentiated updates in neural network (NN) learning to produce a regularization effect, thereby enhancing generalization across time steps. The NN is guided to adaptively suppress updates of zones critical for local fitting to mitigate catastrophic forgetting. Experiments demonstrate the superior performance of EPINN-CL. Post hoc interpretations revealed the more significant phase transition characteristics and improved residual uniformity under the constraint guidance. Notably, we found that EPINN-CL consistently tends to suppress updates of the activation function and the output layer. This suggests maintaining the fixed information compression mode to improve the generalization in multistep solutions.

This study provides an intuitive visualization of the EPINN's underlying processes in solving two-phase flow problems by integrating physical context, thereby enhancing the understanding of different layers at the physical level. This offers a novel perspective for investigating physics correlations in physics-based neural networks, which may inspire future improvements in algorithm and architecture designs. A layer-specific constraint learning strategy is proposed, significantly improving the model's convergence across multiple time steps, thereby enhancing solution speed and accuracy. This advances EPINN-CL's capacity in long-term simulations of heterogeneous and fractured problems, which improves its potential for engineering applications. Still, the current model has limitations requiring improvement: unstable fluctuations exist in the adaptive suppression of critical zones; additional computational costs arise from extra propagations in mask generation; and there are intuitions in the explanations; The discretization procedure includes operational complexity, an inappropriate configuration (e.g., excessive time-step sizes) may introduce convergence problem. In future research, a better threshold determination method or theoretically adequate manual scheme can potentially improve the current method. Specifically, in terms of stability, future work will explore introducing meta-learning modules to enable online threshold fine-tuning or employing Bayesian NNs to quantify and control uncertainties. To address computational resource consumption, we plan to explore “mask freezing” mechanisms or static-mask pretraining methods. For intuition-driven improvements, we will explore quantifying feature-physics correlations (e.g., gradient-based class activation mapping methods) and formulate the suppression process (e.g., lyapunov stability theory). Regarding discretization challenges, our forthcoming research will implement the fully implicit scheme as delineated in Appendix C, aiming to mitigate time-step size limitations. Furthermore, extending the visualization workflow to explore the essence of NN learning in broader physical problems and applying constraint-based methods to similar issues are promising directions for further research.

CRediT authorship contribution statement

Jing-Qi Lin: Writing – original draft, Visualization, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Xia Yan:** Writing – review & editing, Supervision, Project administration, Investigation, Funding acquisition, Conceptualization. **Er-Zhen Wang:** Writing – review & editing, Supervision, Software, Resources, Project administration, Data curation. **Qi Zhang:** Writing – review & editing, Supervision, Project administration, Investigation, Formal analysis. **Kai Zhang:** Writing – review & editing, Supervision, Resources, Investigation, Funding acquisition, Conceptualization. **Pi-Yang Liu:** Writing – review & editing, Validation, Supervision, Resources, Formal analysis. **Li-Ming Zhang:** Writing – review & editing, Validation, Supervision, Resources, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the National Key R&D Program of China (No. 2023YFB4104200), the National Natural Science Foundation of China (Nos. 52474067, 52441411, 52325402, 52034010, and 12131014), the Natural Science Foundation of Shandong Province, China (No. ZR2024ME005), Fundamental Research Funds for the Central Universities (Nos. 25CX02025A and 21CX06031A), and the Youth Innovation and Technology Support Program for Higher Education Institutions of Shandong Province, China (No. 2022KJ070).

Appendix A

The parameter settings of NN architectures are presented in Table A1, “in-ch” and “out-ch” represent the number of input and output channels for the convolutional kernels in PICNN, “in-dim” and “out-dim” denote the input and output dimensions for the fully connected layers. In addition to the parameters listed in Table A1, EPINN and EPINN-CL possess hyperparameters, ω_1, ω_2, and α, with initial values set to 0.01, 0.5, and 0.1, respectively. All hyperparameters have been tuned to represent the general performance of each method.

Table A1
The summary of NN architectures and optimization parameters

Case	Method	NN architecture	Model setting	Training time, s
Homogeneous	PICNN	CNN block1: in-ch = 1, out-ch = 25 CNN block2: in-ch = 25, out-ch = 50 Output: in-dim = 1800, out-dim = 625	Initialization: Xavier Optimizer: Adam Learning rate: 0.01 Maximum iteration for the first time step: 2000 Maximum iteration for all time steps: 2000 Tolerance: 1.0×10^{-18}	5690.93
	EPINN	Input: in-dim = 1, out-dim = 625 Hidden: in-dim = 625, out-dim = 625 Output: in-dim = 625, out-dim = 1		1302.44
	EPINN-CL			1104.12
Heterogeneous	PICNN	CNN block1: in-ch = 1, out-ch = 25 CNN block2: in-ch = 25, out-ch = 50 Output: in-dim = 1800, out-dim = 625		13490.01
	EPINN	Input: in-dim = 1, out-dim = 625 Hidden: in-dim = 625, out-dim = 625 Output: in-dim = 625, out-dim = 1		1519.68
	EPINN-CL			1389.65
Fractured	EPINN	Input: in-dim = 1, out-dim = 763 Hidden: in-dim = 763, out-dim = 763 Output: in-dim = 763, out-dim = 1		3400.28
	EPINN-CL			2501.08

Appendix B

A grid convergence study (Zeidan et al., 2007) is supposed to be conducted to examine the convergence and stability of the proposed method. The NN training process rigorously enforces physics-informed constraints through identical numerical discretization schemes as the numerical simulator, thereby establishing that NN-based PDE solutions fundamentally constitute successive approximations of the accurate numerical solutions through iterative gradient descent optimization. Consequently, we designate the solution obtained from sufficiently refined meshes as the benchmark, and regard the convergence rate of numerical solutions as the theoretical convergence rate. Computational results indicate that when the model grid resolution reaches 55 × 55, further refinement yields a relative error between two model predictions of less than 0.1%. Therefore, the 55 × 55 grid configuration was adopted as the benchmark. Under the condition of maintaining the model geometry size at 100 m × 100 m, we constructed models with grid resolutions of 5 × 5, 7 × 7, …, 35 × 35 on the homogeneous case. This is a commonly adopted approach, as the heterogeneous case would introduce permeability upscaling uncertainties and fractured case would involve fracture location matching discrepancies (both potentially introducing significant errors). For each model configuration, we recorded the pressure and saturation values at the central grid block at 500 d. The variation trends of pressure and saturation prediction errors with increasing grid resolution are presented in Fig. B1. Numerical results exhibit progressively improved accuracy through successive grid refinements, ultimately demonstrating convergence towards the benchmark solution.

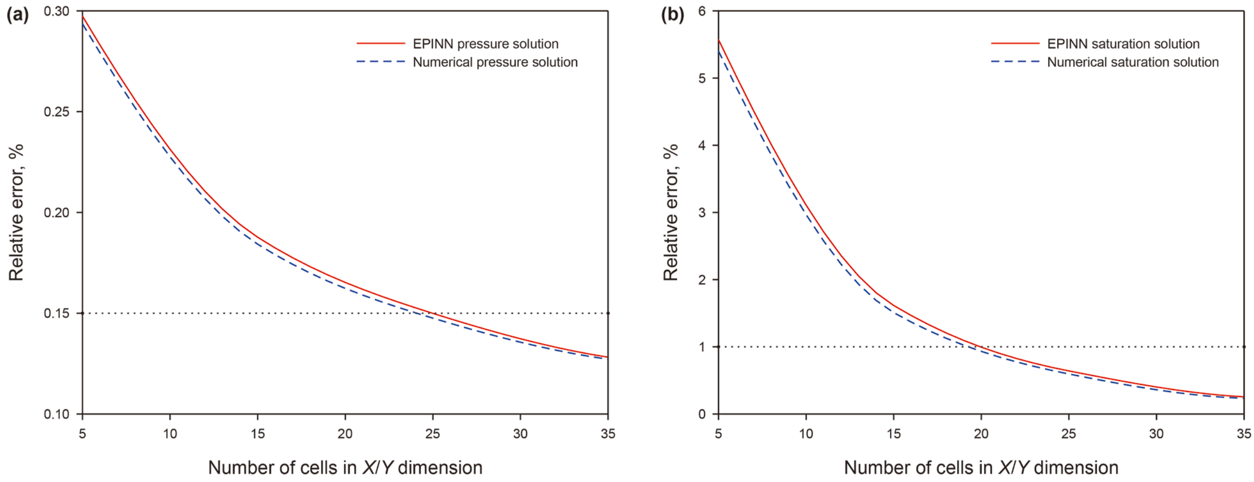

Fig. B1. Grid convergence study for the saturation and pressure at 500 d. (**a**) Pressure; (**b**) saturation.

Notably, the convergence rate discrepancy observed in Fig. B1 between the EPINN solution and the numerical benchmark solution originates from the preset convergence threshold (MSE < 1.0×10^{-18}) in the neural network training protocol. This threshold was selected after computational efficiency-accuracy tradeoff (the numerical reference solution yields an MSE magnitude of approximately 1.0×10^{-20}). At a spatial discretization resolution of 25 × 25 grid nodes, the current model demonstrates sufficient performance: pressure field relative error remains below 0.15%, while saturation field relative error stays under 1%. Although further reducing the convergence threshold could theoretically enhance the NN's prediction precision, such refinement would necessitate additional computational resource allocation.

Appendix C

This appendix provides a brief introduction to solving the two-phase flow problem in porous media under a fully implicit scheme using EPINN-CL, including: FVM-based physics constraint construction, neural network architecture design, and layer-specific constraint design. First, physical constraints are constructed by discretizing Eq. (7) using FVM

$$\begin{aligned}&\frac{\phi V_i}{\Delta t}\left(S_{\mathrm{n}i}^{t+1}-S_{\mathrm{n}i}^{t}\right)+\frac{S_{\mathrm{n}}\phi(C_{\mathrm{n}}+C_{\mathrm{r}})V_i}{\Delta t}\left(P_i^{t+1}-P_i^{t}\right)\\&\quad=\sum_{j\in G_i}\left(\lambda_{\mathrm{n},ij+1/2}T_{ij}\left(\Phi_{\mathrm{n},j}^{t+1}-\Phi_{\mathrm{n},i}^{t+1}\right)\right)+q_{\mathrm{n}}V_i\end{aligned} \tag{C1}$$

$$\begin{aligned}&\frac{\phi V_i}{\Delta t}\left(S_{\mathrm{w}i}^{t+1}-S_{\mathrm{w}i}^{t}\right)+\frac{S_{\mathrm{w}}\phi(C_{\mathrm{w}}+C_{\mathrm{r}})V_i}{\Delta t}\left(P_i^{t+1}-P_i^{t}\right)\\&\quad=\sum_{j\in G_i}\left(\lambda_{\mathrm{w},ij+1/2}T_{ij}\left(\Phi_{\mathrm{w},j}^{t+1}-\Phi_{\mathrm{w},i}^{t+1}\right)\right)+q_{\mathrm{w}}V_i\end{aligned} \tag{C2}$$

In the fully implicit scheme, both pressure and saturation require iterative solving. Thus, NN-based solvers must simultaneously predict both variables. Based on Eqs. (C1) and (C2), the residual terms for the non-wetting and wetting phases are derived as

$$\begin{aligned}R_{\mathrm{n}i}=&\frac{\phi V_i}{\Delta t}\left(\tilde{S}_{\mathrm{n}i}^{t+1}-S_{\mathrm{n}i}^{t}\right)+\frac{S_{\mathrm{n}}\phi(C_{\mathrm{n}}+C_{\mathrm{r}})V_i}{\Delta t}\left(\tilde{P}_i^{t+1}-P_i^{t}\right)\\&-\sum_{j\in G_i}\left(\lambda_{\mathrm{n},ij+1/2}T_{ij}\left(\Phi_{\mathrm{n},j}^{t+1}-\Phi_{\mathrm{n},i}^{t+1}\right)\right)-q_{\mathrm{n}}V_i\end{aligned} \tag{C3}$$

$$\begin{aligned}R_{\mathrm{w}i}=&\frac{\phi V_i}{\Delta t}\left(\tilde{S}_{\mathrm{w}i}^{t+1}-S_{\mathrm{w}i}^{t}\right)+\frac{S_{\mathrm{w}}\phi(C_{\mathrm{w}}+C_{\mathrm{r}})V_i}{\Delta t}\left(\tilde{P}_i^{t+1}-P_i^{t}\right)\\&-\sum_{j\in G_i}\left(\lambda_{\mathrm{w},ij+1/2}T_{ij}\left(\Phi_{\mathrm{w},j}^{t+1}-\Phi_{\mathrm{w},i}^{t+1}\right)\right)-q_{\mathrm{w}}V_i\end{aligned} \tag{C4}$$

where $\tilde{S}$ and $\tilde{P}$ denote the NN-predicted saturation and pressure, respectively. After computing residuals using Eqs. (C3) and (C4), the mean squared error (MSE) loss is defined as

$$L^{t+1} = \frac{1}{2N_g} \sum_{i=1}^{N_g} \left((R_{ni})^2 + (R_{wi})^2 \right) \tag{C5}$$

Consistent with the IMPES approach, initial conditions are incorporated into the NN's input for the first timestep. Boundary conditions are enforced during residual computation in Eqs. (C3) and (C4) by replacing predicted values at boundary locations with observed values in S^* and P^*.

For neural network architecture design and constrained learning, the dual-net architecture (Fig. C1) represents a well-suited solution. Independent network blocks are designed to predict pressure and saturation separately. While forward predictions remain decoupled, backpropagation facilitates interaction between the networks. Notably, unlike the IMPSE solver that employs a single unbranched backpropagation path, more refined constraint learning becomes feasible through this architecture. For instance, generating distinct masks for each network followed by synchronized parameter updates proves effective, as pressure and saturation fields typically exhibit divergent evolutionary patterns requiring separate feature representations. Specifically, as illustrated in Fig. C1, during the initial forward propagation at each timestep, P^t and S^t are independently fed into the dual-branch EPINN blocks to obtain P_*^{t+1} and S_*^{t+1}, followed by computation of R^t (Eqs. (C3) and (C4)) and the MSE loss (Eq. (C5)). Subsequently, a backpropagation operation is performed through both the pressure branch network θ_P^t and saturation branch network θ_S^t. This updates their respective masks M_P^t and M_S^t to M_P^{t+1} and M_S^{t+1}, thereby transforming the original parameters θ_P^t and θ_S^t into masked versions $\theta_{P,*}^{t+1}$ and $\theta_{S,*}^{t+1}$. During subsequent learning iterations, parameter updates will continue based on the masked NN until meeting convergence criteria for the current timestep.

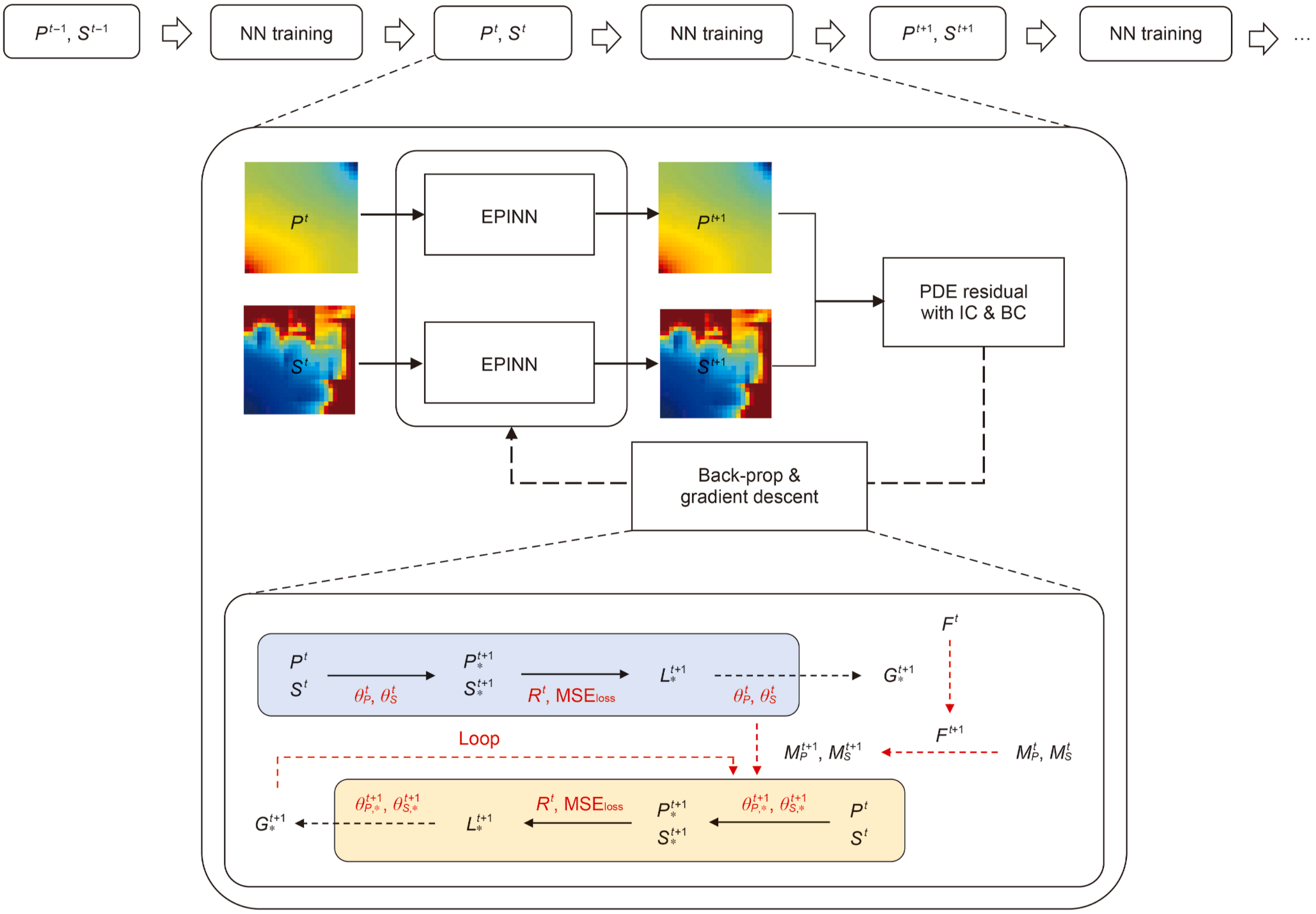

Fig. C1. The procedure of EPINN-CL-based pressure and saturation prediction.

In this work, we primarily investigate two-phase flow in porous media using FVM-based discretization. We will continue to explore the application of this methodology to broader physical problems, including complex two-phase flow governed by other equations (e.g., shock-bubbles, Mach 3 flow). The fundamental framework of constraint learning proposed in this study can be extended to these broader problems. After completing physics-informed constraint construction and NN architecture design, similar layer-specific constraint design principles could be adopted. However, this process should integrate the practical variation patterns of physical quantities. Strategically designed mask generation based on the inherent physical characteristics of different systems could achieve enhanced integration.

References

Abdi, H., Williams, L.J., 2010. Principal component analysis. WIREs Comput. Statist. 2 (4), 433–459. https://doi.org/10.1002/wics.101.

Abidoye, L.K., Khudaida, K.J., Das, D.B., 2015. Geological carbon sequestration in the context of two-phase flow in porous media: A review. Crit. Rev. Environ. Sci. Technol. 45 (11), 1105–1147. https://doi.org/10.1080/10643389.2014.924184.

Aich, A., 2021. Elastic weight consolidation (EWC): Nuts and bolts. https://doi.org/10.48550/arXiv.2105.04093.

Alain, G., Bengio, Y., 2016. Understanding intermediate layers using linear classifier probes. https://doi.org/10.48550/arXiv.1610.01644.

Aljundi, R., Chakravarty, P., Tuytelaars, T., 2017. Expert gate: Lifelong learning with a network of experts. CVPR 2017. https://doi.org/10.1109/CVPR.2017.753.

Aljundi, R., Babiloni, F., Elhoseiny, et al., 2018. Memory aware synapses: Learning what (not) to forget. ECCV 2018. https://doi.org/10.1007/978-3-030-01219-9_9.

Aljundi, R., Rohrbach, M., Tuytelaars, T., 2019. Selfless sequential learning. ICLR 2019. https://doi.org/10.48550/arXiv.1806.05421.

Anagnostopoulos, S.J., Toscano, J.D., Stergiopulos, N., Karniadakis, G.E., 2024. Learning in PINNs: Phase transition, total diffusion, and generalization. https://doi.org/10.48550/arXiv.2403.18494.

Ba, J., Kiros, J.R., Hinton, G.E., 2016. Layer normalization. https://doi.org/10.48550/arXiv.1607.06450.

Bardes, A., Garrido, Q., Ponce, J., et al., 2024. Revisiting feature prediction for learning visual representations from video. https://doi.org/10.48550/arXiv.2404.08471.

Bear, J., Cheng, A.H.-D., 2010. Modeling Groundwater Flow and Contaminant Transport. Springer, Dordrecht. https://doi.org/10.1007/978-1-4020-6682-5.

Berre, I., Doster, F., Keilegavlen, E., 2019. Flow in fractured porous media: A review of conceptual models and discretization approaches. Transport Porous Media 130 (1), 215–236. https://doi.org/10.1007/s11242-018-1171-6.

Chaudhry, A., Dokania, P.K., Ajanthan, T., et al., 2018a. Riemannian walk for incremental learning: Understanding forgetting and intransigence. ECCV 2018. https://doi.org/10.1007/978-3-030-01252-6_33.

Chaudhry, A., Ranzato, M.A., Rohrbach, M., et al., 2018b. Efficient lifelong learning with A-GEM. ICLR 2019. https://doi.org/10.48550/arXiv.1812.00420.

Chen, H., Kou, J., Sun, S., et al., 2019. Fully mass-conservative IMPES schemes for incompressible two-phase flow in porous media. Comput. Methods Appl. Mech. Eng. 350, 641–663. https://doi.org/10.1016/j.cma.2019.03.023.

Chen, Z., Huan, G., Ma, Y., 2006. Computational methods for multiphase flows in porous media. SIAM, Philadelphia. https://doi.org/10.1137/1.9780898718942.

Chung, H.P., Woo, L., 2011. Modeling void formation and unsaturated flow in liquid composite molding processes: A survey and review. J. Reinforc. Plast. Compos. 30 (11), 957–977. https://doi.org/10.1177/0731684411411338.

Cohen, J.M., Kaur, S., Li, Y., et al., 2021. Gradient descent on neural networks typically occurs at the edge of stability. ICLR 2021. https://doi.org/10.48550/arXiv.2103.00065.

Coats, K.H., 2003. IMPES stability: The CFL limit. SPE J. 8 (3), 291–297. https://doi.org/10.2118/85956-PA.

Cohen, J.M., Ghorbani, B., Krishnan, S., et al., 2022. Adaptive gradient methods at the edge of stability. https://doi.org/10.48550/arXiv.2207.14484.

Cuomo, S., Cola, V.S.D., Giampaolo, F., et al., 2022. Scientific machine learning through physics-informed neural networks: Where we are and what's next. J. Sci. Comput. 92 (3), 88. https://doi.org/10.1007/s10915-022-01939-z.

Damian, A., Nichani, E., Lee, J.D., 2022. Self-stabilization: The implicit bias of gradient descent at the edge of stability. ICLR 2023. https://doi.org/10.48550/arXiv.2209.15594.

Daw, A., Karpatne, A., Watkins, W., et al., 2017. Physics-guided neural networks (PGNN): An application in lake temperature modeling. In: Knowledge Guided Machine Learning. Chapman and Hall/CRC, pp. 353–372. https://doi.org/10.48550/arXiv.1710.11431.

Dekhovich, A., Sluiter, M.H.F., Tax, D.M.J., et al., 2024. iPINNs: Incremental learning for physics-informed neural networks. Eng. Comput. 41, 389–402. https://doi.org/10.1007/s00366-024-02010-1.

Douillard, A., Ramé, A., Couairon, G., et al., 2022. DyTox: Transformers for continual learning with dynamic token expansion. CVPR 2022. https://doi.org/10.1109/CVPR52688.2022.00907.

Dohare, S., Hernandez-Garcia, J.F., Lan, Q., et al., 2024. Loss of plasticity in deep continual learning. Nature 632, 768–774. https://doi.org/10.1038/s41586-024-07711-7.

El Idrissi, A.Q., da Silva, E.G., Zeidan, D., 2023. Numerical studies of two-phase water hammer flows using Godunov methods. Ann. Nucl. Energy 182, 109578. https://doi.org/10.1016/j.anucene.2022.109578.

Equitz, W.H.R., Cover, T.M., 1991. Successive refinement of information. IEEE Trans. Inf. Theor. 37 (2), 269–275. https://doi.org/10.1109/18.75242.

Eymard, R., Gallouët, T., Herbin, R., 2000. Finite volume methods. Handb. Numer. Anal. 7, 713–1018. https://doi.org/10.1016/S1570-8659(00)07005-8.

Fuks, O., Tchelepi, H.A., 2020. Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. Journal of Machine Learning for Modeling and Computing 1 (1), 19–37. https://doi.org/10.1615/JMachLearnModelComput.2020033905.

Gepperth, A., Karaoguz, C., 2016. A bio-inspired incremental learning architecture for applied perceptual problems. Cogn. Comput. 8 (5), 924–934. https://doi.org/10.1007/s12559-016-9389-5.

Gerritsen, M.G., Durlofsky, L.J., 2005. Modeling fluid flow in oil reservoirs. Annu. Rev. Fluid Mech. 37, 211–238. https://doi.org/10.1146/ANNUREV.FLUID.37.061903.175748.

Goldfeld, Z., Polyanskiy, Y., 2020. The information bottleneck problem and its applications in machine learning. IEEE J. Select. Areas Inform. Theory 1 (1), 19–38. https://doi.org/10.1109/JSAIT.2020.2991561.

Goodfellow, I., Bengio, Y., Courville, A., 2018. Deep learning. Genet. Program. Evolvable Mach. 19, 305–307. https://doi.org/10.1007/s10710-017-9314-z.

Guo, M.H., Liu, Z.N., Mu, T.J., et al., 2023. Beyond self-attention: External attention using two linear layers for visual tasks. IEEE Trans. Pattern Anal. Mach. Intell. 45 (5), 5436–5447. https://doi.org/10.1109/TPAMI.2022.3211006.

Haghighat, E., Juanes, R., 2021. SciANN: A Keras/TensorFlow wrapper for scientific computations and physics-informed deep learning using artificial neural networks. Comput. Methods Appl. Mech. Eng. 373, 113552. https://doi.org/10.1016/j.cma.2020.113552.

Harbate, F., Nouh, I., Mohammed, S., 2024. High-order relaxation methods for nonequilibrium two-phase flow equations. Int. J. Numer. Methods Heat Fluid Flow 34 (3), 1399–1423. https://doi.org/10.1108/HFF-06-2023-0344.

Hennigh, O., Narasimhan, S., Nabian, M.A., et al., 2021. NVIDIA SimNet™: An AI-accelerated multi-physics simulation framework. ICCS 2021. https://doi.org/10.1007/978-3-030-77977-1_36.

Howard, A., Fu, Y., Stinis, P., 2024. A multifidelity approach to continual learning for physical systems. Mach. Learn.: Sci. Technol. 5 (2), 025042. https://doi.org/10.1088/2632-2153/ad45b2.

Huang, S., Feng, W., Tang, C., et al., 2025. Partial differential equations meet deep neural networks: A survey. IEEE Transact. Neural Networks Learn. Syst. https://doi.org/10.1109/TNNLS.2025.3545967.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. ICML 2015 37, 448–456. https://doi.org/10.48550/arXiv.1502.03167.

Jagtap, A.D., Kawaguchi, K., Karniadakis, G.E., 2020. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. J. Comput. Phys. 404, 109136. https://doi.org/10.1016/j.jcp.2019.109136.

Kalyan, K.S., 2024. A survey of GPT-3 family large language models including ChatGPT and GPT-4. Natural Language Processing Journal 6, 100048. https://doi.org/10.1016/j.nlp.2023.100048.

Karniadakis, G.E., Kevrekidis, I.G., Lu, L., et al., 2021. Physics-informed machine learning. Nature Reviews Physics 3 (6), 422–440. https://doi.org/10.1038/s42254-021-00314-5.

Katsnelson, M.I., Vanchurin, V., Westerhout, T., 2021. Self-organized criticality in neural networks. https://doi.org/10.48550/arXiv.2107.03402.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. https://doi.org/10.48550/arXiv.1412.6980.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., et al., 2017. Overcoming catastrophic forgetting in neural networks. Proc. Natl. Acad. Sci. 114 (13), 3521–3526. https://doi.org/10.1073/pnas.1611835114.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60 (6), 84–90. https://doi.org/10.1145/3065386.

Lange, M.D., Aljundi, R., Masana, M., et al., 2022. A continual learning survey: Defying forgetting in classification tasks. IEEE Trans. Pattern Anal. Mach. Intell. 44 (7), 3366–3385. https://doi.org/10.1109/TPAMI.2021.3057446.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436–444. https://doi.org/10.1038/nature14539.

Lesne, A., 2014. Shannon entropy: A rigorous notion at the crossroads between probability, information theory, dynamical systems and statistical physics. Math. Struct. Comput. Sci. 24 (3), e240311. https://doi.org/10.1017/S0960129512000783.

Lewkowycz, A., Bahri, Y., Dyer, E., et al., 2020. The large learning rate phase of deep learning: The catapult mechanism. https://doi.org/10.48550/arXiv.2003.02218.

Li, D., Lv, S., Zha, W., et al., 2023. A nonlinear solver based on residual network for seepage equation. Eng. Appl. Artif. Intell. 126, 106850. https://doi.org/10.1016/j.engappai.2023.106850.

Li, Z., Liu, F., Yang, W., et al., 2022. A survey of convolutional neural networks: Analysis, applications, and prospects. IEEE Transact. Neural Networks Learn. Syst. 33 (12), 6999–7019. https://doi.org/10.1109/TNNLS.2021.3084827.

Liang, Z., Xu, Y., Hong, Y., et al., 2024. A survey of multimodel large language models. In: 3rd International Conference on Computer. Association for Computing Machinery, New York. https://doi.org/10.1145/3672758.3672824.

Liu, J., Meng, Y., Fitzsimmons, M., et al., 2025. Physics-informed neural network lyapunov functions: PDE characterization, learning, and verification. Automatica 175, 112193. https://doi.org/10.1016/j.automatica.2025.112193.

Liu, Y., Liu, W., Yan, X., et al., 2023. Adaptive transfer learning for PINN. J. Comput. Phys. 490, 112291. https://doi.org/10.1016/j.jcp.2023.112291.

Losapio, D., Scotti, A., 2023. Local embedded discrete fracture model (LEDFM). Adv. Water Resour. 171, 104361. https://doi.org/10.1016/j.advwatres.2022.104361.

Lu, L., Meng, X., Mao, Z., et al., 2021. DeepXDE: A deep learning library for solving differential equations. SIAM Rev. 63 (1), 208–228. https://doi.org/10.1137/19M1274067.

Mallya, A., Lazebnik, S., 2018. PackNet: Adding multiple tasks to a single network by iterative pruning. CVPR 2018. https://doi.org/10.1109/CVPR.2018.00810.

McCloskey, M., Cohen, N.J., 1989. Catastrophic interference in connectionist networks: The sequential learning problem. Psychol. Learn. Motiv. 24, 109–165. https://doi.org/10.1016/S0079-7421(08)60536-8.

Mehta, P., Schwab, D.J., 2014. An exact mapping between the variational renormalization group and deep learning. https://doi.org/10.48550/arXiv.1410.3831.

Moortgat, J., Firoozabadi, A., 2013. Three-phase compositional modeling with capillarity in heterogeneous and fractured media. SPE J. 18 (6), 1150–1168. https://doi.org/10.2118/159777-PA.

Nakkiran, P., Kaplun, G., Bansal, Y., et al., 2021. Deep double descent: Where bigger models and more data hurt. J. Stat. Mech. Theor. Exp. 2021 (12), 124003. https://doi.org/10.1088/1742-5468/ac3a74.

Ouffa, S., Zeidan, D., Seba, D., 2021. The WAF scheme for the isentropic drift-flux model of compressible two-phase flows. Comput. Fluid 229, 105091. https://doi.org/10.1016/j.compfluid.2021.105091.

Paszke, A., Gross, S., Massa, F., et al., 2019. PyTorch: An imperative style, high-performance deep learning library. https://doi.org/10.48550/arXiv.1912.01703.

Peaceman, D.W., 1993. Representation of a horizontal well in numerical reservoir simulation. SPE Adv. Technol. 1 (1), 7–16. https://doi.org/10.2118/21217-PA.

Raissi, M., Perdikaris, P., Karniadakis, G.E., 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. Comput. Phys. 378, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045.

Redondo, C., Rubio, G., Valero, E., 2018. On the efficiency of the IMPES method for two phase flow problems in porous media. J. Petrol. Sci. Eng. 164, 427–436. https://doi.org/10.1016/j.petrol.2018.01.066.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. MICCAI 2015. https://doi.org/10.48550/arXiv.1505.04597.

Ruder, S., 2017. An overview of multi-task learning in deep neural networks. https://doi.org/10.48550/arXiv.1706.05098.

Saad, B., Saad, M., 2013. Study of full implicit petroleum engineering finite-volume scheme for compressible two-phase flow in porous media. SIAM J. Numer. Anal. 51 (1), 716–741. https://doi.org/10.1137/120869092.

Sadek, R.A., 2012. SVD based image processing applications: State of the art, contributions and research challenges. https://doi.org/10.48550/arXiv.1211.7102.

Samek, W., Montavon, G., Lapuschkin, S., et al., 2021. Explaining deep neural networks and beyond: A review of methods and applications. Proc. IEEE 109 (3), 247–278. https://doi.org/10.1109/JPROC.2021.3060483.

Santurkar, S., Tsipras, D., Ilyas, A., Madry, A., 2018. How does batch normalization help optimization? NeurIPS 2018. https://doi.org/10.48550/arXiv.1805.11604.

Serra, J., Suris, D., Miron, M., 2018. Overcoming catastrophic forgetting with hard attention to the task. ICML 2018. https://doi.org/10.48550/arXiv.1801.01423.

Shaw, P., Uszkoreit, J., Vaswani, A., 2018. Self-attention with relative position representations. NAACL 2018. https://doi.org/10.48550/arXiv.1803.02155.

Shin, H., Lee, J.K., Kim, J., 2017. Continual learning with deep generative replay. NeurIPS 2017. https://doi.org/10.48550/arXiv.1705.08690.

Shwartz-Ziv, R., 2022. Information flow in deep neural networks. https://doi.org/10.48550/arXiv.2202.06749.

Shwartz-Ziv, R., Tishby, N., 2017. Opening the black box of deep neural networks via information. https://doi.org/10.48550/arXiv.1703.00810.

Smith, L.N., Topin, N., 2019. Super-convergence: Very fast training of neural networks using large learning rates. Artificial Intelligence and Machine Learning for Multi-domain Operations Applications 11006, 369–386. https://doi.org/10.1117/12.2520589.

Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. https://doi.org/10.48550/arXiv.1409.3215.

Tishby, N., Zaslavsky, N., 2015. Deep learning and the information bottleneck principle. 2015 IEEE Information Theory Workshop. https://doi.org/10.1109/ITW.2015.7133169.

Tishby, N., Pereira, F.C., Bialek, W., 2000. The information bottleneck method. https://doi.org/10.48550/arXiv.physics/0004057.

Toscano, J.D., Oommen, V., Varghese, A.J., 2025. From PINNs to PIKANs: Recent advances in physics-informed machine learning. Machine Learning for Computational Science and Engineering 1 (15). https://doi.org/10.1007/s44379-025-00015-1.

Vaswani, A., Shazeer, N., Parmar, N., et al., 2017. Attention is all you need. https://doi.org/10.48550/arXiv.1706.03762.

Wang, L., Zhang, X., Su, H., et al., 2024. A comprehensive survey of continual learning: Theory, method and application. IEEE Trans. Pattern Anal. Mach. Intell. 46 (8), 5362–5383. https://doi.org/10.1109/TPAMI.2024.3367329.

Wang, N., Zhang, D., Chang, H., 2020. Deep learning of subsurface flow via theory-guided neural network. J. Hydrol. 584, 124700. https://doi.org/10.1016/j.jhydrol.2020.124700.

Wang, N., Chang, H., Zhang, D., 2022. Surrogate and inverse modeling for two-phase flow in porous media via theory-guided convolutional neural network. J. Comput. Phys. 466, 111419. https://doi.org/10.1016/j.jcp.2022.111419.

Wang, S., Sankaran, S., Perdikaris, P., 2024. Respecting causality for training physics-informed neural networks. Comput. Methods Appl. Mech. Eng. 421, 116813. https://doi.org/10.1016/j.cma.2024.116813.

Williams, E., Payeur, A., Ryoo, A.H.-W., et al., 2024. Expressivity of neural networks with random weights and learned biases. ICLR 2025. https://doi.org/10.48550/arXiv.2407.00957.

Yan, X., Huang, Z., Yao, J., et al., 2016. An efficient embedded discrete fracture model based on mimetic finite difference method. J. Petrol. Sci. Eng. 145, 11–21. https://doi.org/10.1016/j.petrol.2016.03.013.

Yan, X., Lin, J., Wang, S., 2024. Physics-informed neural network simulation of two-phase flow in heterogeneous and fractured porous media. Adv. Water Resour. 189, 104731. https://doi.org/10.1016/j.advwatres.2024.104731.

Yu, Y., Si, X., Hu, C., 2019. A review of recurrent neural networks: LSTM cells and network architectures. Neural Comput. 31 (7), 1235–1270. https://doi.org/10.1162/neco_a_01199.

Zeiler, M.D., Fergus, R., 2014. Visualizing and understanding convolutional networks. Computer Vision. ECCV 2014. https://doi.org/10.1007/978-3-319-10590-1_53.

Zeidan, D., El Idrissi, A.Q., 2025. A TVD WAF scheme based on an accurate Riemann solver to simulate compressible two-phase flows. Int. J. Numer. Methods Heat Fluid Flow 35 (1), 414–445. https://doi.org/10.1108/HFF-03-2024-0214.

Zeidan, D., Romenski, E., Slaouti, A., et al., 2007. Numerical study of wave propagation in compressible two-phase flow. Int. J. Numer. Methods Fluid. 54 (4), 393–417. https://doi.org/10.1002/fld.1404.

Zeidan, D., Bähr, P., Farber, P., et al., 2019. Numerical investigation of a mixture two-phase flow model in two-dimensional space. Comput. Fluid 181, 90–106. https://doi.org/10.1016/j.compfluid.2018.12.013.

Zhang, Z., 2022. A physics-informed deep convolutional neural network for simulating and predicting transient Darcy flows in heterogeneous reservoirs without labeled data. J. Petrol. Sci. Eng. 211, 110179. https://doi.org/10.1016/j.petrol.2022.110179.

Zhang, Z., Yan, X., Liu, P., et al., 2023. A physics-informed convolutional neural network for the simulation and prediction of two-phase Darcy flows in heterogeneous porous media. J. Comput. Phys. 477, 111919. https://doi.org/10.1016/j.jcp.2023.111919.

Zhao, H., Jia, J., Koltun, V., 2020. Exploring self-attention for image recognition. https://doi.org/10.48550/arXiv.2004.13621.

Zhou, J., Cui, G., Hu, S., et al., 2020. Graph neural networks: A review of methods and applications. AI Open 1, 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001.

Zhou, M., Liu, T., Li, Y., et al., 2019. Toward understanding the importance of noise in training neural networks. ICML 2019. https://doi.org/10.48550/arXiv.1909.03172.

Zhuang, F., Qi, Z., Duan, K., et al., 2021. A comprehensive survey on transfer learning. Proc. IEEE 109 (1), 43–76. https://doi.org/10.1109/JPROC.2020.3004555.