Physics Informed Kolmogorov-Arnold Network for Two-Phase Flow Model with Experimental Data

Daulet Kalesh[1], Timur Merembayev[1], Sagyn Omirbekov[2], and Yerlan Amanbek[1(✉)]

[1] Department of Mathematics, School of Sciences and Humanities, Nazarbayev University, Kabanbay batyr 53, Astana, Kazakhstan
[2] National Laboratory Astana, Nazarbayev University, Kabanbay batyr 53, Astana 010000, Kazakhstan
{daulet.kalesh,timur.merembayev,sagyn.omirbekov, yerlan.amanbek}@nu.edu.kz

Abstract. The recently introduced neural network structure, known as the Kolmogorov-Arnold Network (KAN), provides greater interpretability and accuracy compared to ordinary Multi-Layer Perceptrons (MLPs) due to its use of spline functions. The primary goal of this research is to propose a novel Physics-Informed Neural Network (PINN) architecture based on KAN for solving the Buckley-Leverett equation, aiming to study its performance versus a conventional MLP-based PINN.

We conducted a comparative analysis between the KAN and MLP architectures. We found that they both achieved comparable accuracy. However, the KAN's training process is approximately twice as time-consuming as that of the MLP. When validating with experimental data, the KAN architecture showed enhanced robustness when trained with a hybrid optimizer approach that combines the Adam and L-BFGS optimizers rather than using L-BFGS alone. Moreover, increasing the number of layers and neurons in the KAN model improved its predictions, nearly matching the MLP-based PINN results for modeling the laboratory experiment.

The results highlight the promising potential of KAN-based PINNs, especially when optimized architectures and hybrid training approaches are used for modeling fluid dynamics in porous media.

Keywords: Physics-Informed Neural Networks · Kolmogorov-Arnold Network · Hybrid Optimization · Buckley-Leverett equation · Porous Media Flow

1 Introduction

Kolmogorov-Arnold Networks (KANs) are a class of neural networks inspired by the Kolmogorov-Arnold representation theorem, which posits that any multivariate continuous function can be expressed as a sum of continuous functions

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026
O. Gervasi et al. (Eds.): ICCSA 2025 Workshops, LNCS 15888, pp. 447–460, 2026.
https://doi.org/10.1007/978-3-031-97596-7_30

of one variable [14]. One of the key advantages of KANs is their ability to place learnable activation functions on the edges of the network graph rather than at the nodes, as is typical in Multi-Layer Perceptrons (MLPs).

KANs are effective in various tasks, including function regression and partial differential equations (PDEs) solving [13,14]. They have also been demonstrated to be more efficient and accurate than traditional MLPs in these tasks (reference). For example, KANs have been used to approximate complex functions with greater interpretability and accuracy, making them valuable for scientific applications [13].

The authors [15,18] used a Fourier-based Kolmogorov-Arnold Network module, complemented by two dropout strategies. Empirical results demonstrate notable gains in performance, underscoring that a carefully chosen nonlinear feature transformation—particularly one that remains tractable to train can yield better recommendations than "light" graph network variants that omit such transformations altogether. FourierKAN is a well-validated approach that enriches user-item representations while preserving training stability, offering a promising new direction.

However, KANs also have some limitations. For instance, they can be computationally expensive to train, especially for large-scale datasets. Additionally, KANs may not be as widely adopted as more conventional models such as CNNs or RNNs [4,19].

A novel approach to solving PDEs through Physics-Informed Neural Networks (PINN) with KAN was introduced by Wang et al. [17] regarded as Physics-Informed Kolmogorov-Arnold Network (PIKAN). MLP-based PINNs were shown to have struggled with heterogeneous and multiscale problems, as well as handling complex boundary conditions [3]. The authors evaluated PIKAN with various PDE forms, such as strong, energy, and inverse, and assessed its performance for multi-scale, heterogeneous, and non-linear problems. PIKAN resulted in higher accuracy and faster convergence for most PDEs, showcasing better alignment with Finite Element Method (FEM). Despite all of this, it struggles with complex geometric PDEs and is sensitive to grid size due to the risk of overfitting.

The authors [1] applied PIKAN to the Buckley-Leverett equation in the context of waterflooding in oil reservoirs. The research paper demonstrates that PIKANs achieve accuracy comparable to PINNs but experience greater fluctuations during training, indicating areas for further optimization. Additionally, they did not consider PIKAN's time efficiency compared with PINN. Nonetheless, their study established the feasibility and potential of PIKANs in simulating fluid dynamics within porous media, highlighting their advantages for simulations and potential for broader applications in reservoir modeling and history matching.

In fluid dynamics, modeling two-phase flow in porous media is essential in various engineering and scientific applications, particularly in the oil and gas industry. Understanding and accurately simulating such processes is critical for optimizing enhanced oil recovery, designing effective CO_2 sequestration, under-

standing groundwater contamination, and analyzing fluid behavior in subsurface environments. These simulations also support cost-effective field planning through simulations and decision-making. While conventional numerical methods such as Finite Volume or Finite Difference schemes, especially when GPU-accelerated, can be more computationally efficient and stable, they often require fine meshes and strict adherence to the CFL condition for explicit schemes. Recent advances in machine learning, such as PINNs, particularly when integrated with KAN, offer an alternative mesh-free solution that inherently avoids such restrictions. Moreover, PINNs provide the advantage of integrating underlying physics with experimental or simulation data, enhancing flexibility and enabling data-driven modeling where conventional methods may fall short. This supports better decision-making capabilities in oil field operations and reservoir engineering.

In this paper, we investigate the PINN with various architectures, including KAN and MLP, for the two-phase flow problem. Initially, we evaluate the PINN-KAN model by conducting a comparative analysis with MLP predictions and along with the analytical solution of the Buckley-Leverett equation. The verified KAN model is further validated for the laboratory experimental data, where we explore the impact of various optimization techniques and modifications to the depth and complexity of the KAN architecture. The computational performance and accuracy of the models are reported for fluid dynamics phenomena in a porous media.

2 Problem

In this study, we consider a problem that involves two-phase fluid flow in porous media. The problem is regarded as the Buckley-Leverett equation involving two immiscible and incompressible fluid phases (e.g., oil (o) and water(w)) with neglected gravity and capillarity. The governing equation for this case is a hyperbolic conservation law PDE, which is defined as follows:

$$\frac{\partial u(x,t)}{\partial t} + \frac{\partial f(u(x,t))}{\partial x} = 0. \tag{1}$$

where $u(x,t)$ is the displacing phase (water) saturation at position x and time t. The fractional flow function $f(u)$ describes the relationship between saturation and the volumetric flux. For this study, we consider the fractional flow with non-convex case given by:

$$f(u) = \frac{u}{u^2 + \frac{(1-u)^2}{M}}, \tag{2}$$

where $M = \frac{\mu_o}{\mu_w}$ is the mobility ratios of viscosities oil μ_o and water μ_w. The task is to solve for $u(x,t)$ given initial and boundary conditions:

$$u(x,0) = 1 \quad u(0,t) = 0 \tag{3}$$

Additionally, the equation can be analytically solved by a method of characteristics (MOC) [10]. However, solving this problem might be challenging due to the nonconvex nature, which often can result in multiple shock saturations and discontinuities [11]. To address these issues, a vanishing viscosity method can be used for these kinds of hyperbolic PDEs, modulated by a small viscosity coefficient ϵ:

$$\frac{\partial u(x,t)}{\partial t} + \frac{\partial f(u(x,t))}{\partial x} = \epsilon \frac{\partial^2 u(x,t)}{\partial x^2}. \tag{4}$$

By incorporating the diffusion term, the hyperbolic PDE becomes parabolic. This stabilizes the numerical solution by smoothing sharp fronts and discontinuities while maintaining the system's important physical features.

3 Methods

3.1 PINN

Deep neural networks, such as MLP, were used in this paper. which typically apply sequences of linear and nonlinear functions [5,8]. We denote MLP which consists the input layer $\mathbf{X} = [x, t]$ taking space x and time t, L hidden layers N_l and the output layer N_L which all defined as follows:

$$\mathcal{N}_0(\mathbf{X}) = \mathbf{X} \in \mathbb{R}^{d_{in}} \tag{5}$$

$$\mathcal{N}_l(\mathbf{X}) = \sigma(\mathbf{W}_l \mathcal{N}_{l-1}(\mathbf{X}) + \mathbf{b}_l), \quad \text{for} \quad l = 1, \ldots, L-1, \tag{6}$$

$$\mathcal{N}_L(\mathbf{X}) = \mathbf{W}_L \mathcal{N}_{L-1}(\mathbf{X}) + \mathbf{b}_L \in \mathbb{R}^{d_{out}}. \tag{7}$$

where weights $\mathbf{W}_l \in \mathbb{R}^{N_l \times N_{l-1}}$ and biases $\mathbf{b}^l \in \mathbb{R}^{N_l}$ connect the input $\mathbb{R}^{d_{in}}$ and output $\mathbb{R}^{d_{out}}$ spaces with activation function σ.

Based on the above MLP definitions, we aimed to solve $u(x,t)$ of our PDE, which results in PINNs, the modern neural network capable of estimating physics problems. We define the solution as follows:

$$\hat{u}_\theta(x,t) \approx u(x,t). \tag{8}$$

with set of series of weights and biases $\theta = \{W_1, W_2, \ldots, W_L, b_1, b_2, \ldots, b_L\}$,. To ensure that the learned solution satisfies the governing PDE, we define the residual loss of PDE r_θ, incorporating a diffusion term [6] to address challenges with saturation fronts:

$$r_\theta(\mathbf{X}) = (\hat{u}_\theta)_t + f(\hat{u}_\theta)_x - \epsilon(\hat{u}_\theta)_{xx} \tag{9}$$

The goal of PINN is to approximate $\hat{u}_\theta(x,t)$ by minimizing the following loss function with mean squared error (MSE) during the learning process:

$$\mathcal{L}(\theta) = \mathcal{L}_\mathcal{U}(\theta) + \mathcal{L}_\mathcal{R}(\theta), \tag{10}$$

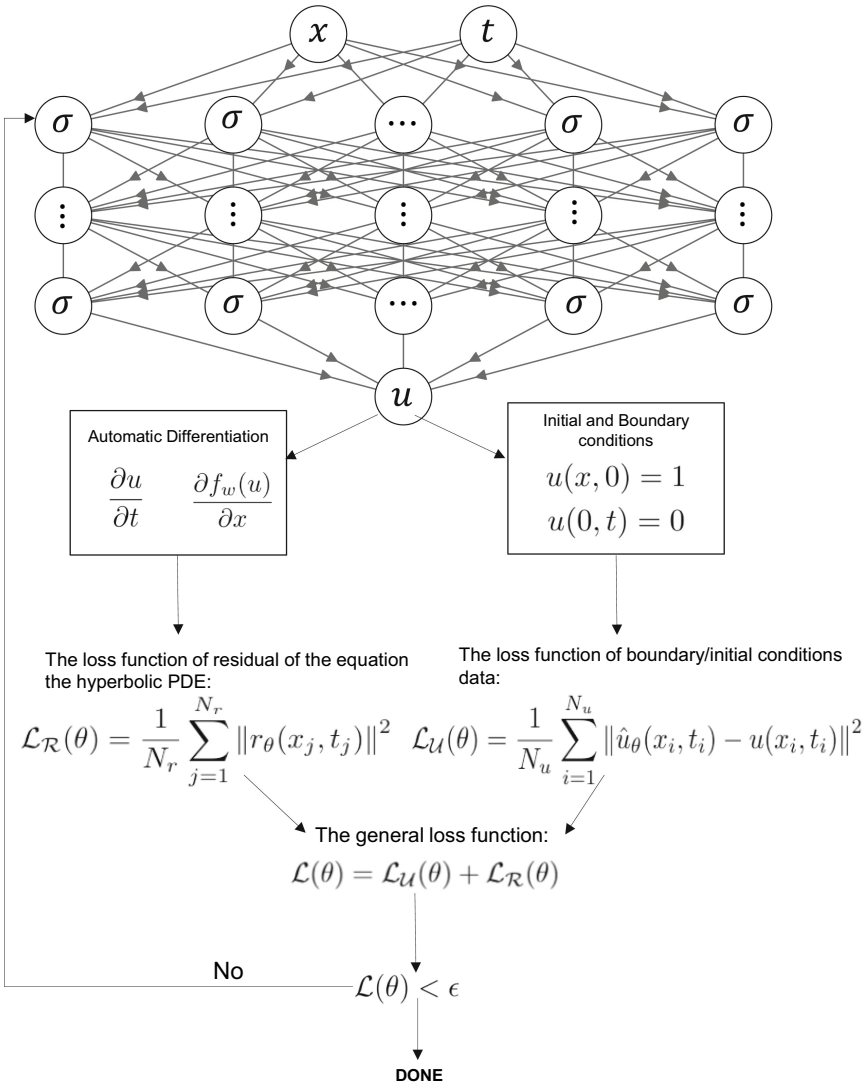


Fig. 1. A schematic representation of the PINN architecture based on the MLP architecture.

where $\mathcal{L}_{\mathcal{U}}(\theta) = \frac{1}{N_u}\sum_{i=1}^{N_u} \|\hat{u}_\theta(x_i,t_i) - u(x_i,t_i)\|^2$ – boundary and initial conditions' loss function, $\mathcal{L}_{\mathcal{R}}(\theta) = \frac{1}{N_r}\sum_{j=1}^{N_r} \|r_\theta(x_j,t_j)\|^2$. – the loss function of the residual of the PDE. Figure 1 demonstrates the full scheme of PINN based on MLP architecture.

PINNs are considered unsupervised machine learning aiming to solve well-posed PDEs and their boundary data. It can be achieved through automatic differentiation (AD) of the *Tensorflow* library which computes the necessary partial derivatives to enforce the residual of the PDE $r_\theta(x,t)$ and updates the network parameters $\theta^* = \arg\min_\theta \mathcal{L}(\theta)$ using the gradient-based limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) optimizer.

In implementing PINN under MLP architecture, a fully connected feed-forward neural network was used containing 8 hidden layers with 20 neurons per layer, each with the activation function tanh. The weights were initialized through the Glorot normal initializer [7]. The network uses $N_u = 300$ boundary and initial condition points and $N_f = 10000$ collocation points distributed within the domain generated by Latin Hypercube Sampling.

3.2 KAN

KAN offers a compelling alternative to conventional MLP. While MLPs trace their foundations to the universal approximation theorem, KANs draw on the Kolmogorov-Arnold representation theorem [13,14]. Formally, a KAN can be written as:

$$KAN = f(x) = \sum_{q=1}^{2n+1} \Phi_q\left(\sum_{p=1}^{n} \phi_{q,p}(x_p)\right) \tag{11}$$

where each $\phi_{q,p}$ is a univariate function that processes the input variable x_p, and Φ_q is another univariate function that acts on these transformed inputs. According to the Kolmogorov-Arnold theorem, the "inner" set of functions corresponds to a KAN layer with $n_{in} = n$ and $n_{out} = 2n + 1$, while the "outer" set corresponds to a KAN layer with $n_{in} = 2n + 1$ and $n_{out} = n$. Thus, the Kolmogorov-Arnold representation is achieved by composing these two layers. A practical enhancement in KAN involves the introduction of a basis function $b(x)$, making the activation function $\phi(x)$ the sum of $b(x)$ and an additional spline component:

$$\phi(x) = w(b(x) + spline(x)) \tag{12}$$

where $spline(x)$ is typically expressed as a linear combination of B-splines with trainable coefficients. This formulation increases flexibility and expressive power while preserving a manageable framework for function approximation. In Fig. 2, the complete scheme of PINN based on KAN architecture is shown.

Despite its expressive strengths, standard KAN can be challenging to train because of its spline-based formulation, which runs counter to the goal of devising a more straightforward yet effective replacement for the feature transformation component. In a research paper [13], the author introduces FastKAN, an efficient re-implementation of Kolmogorov-Arnold Networks (KANs) that significantly improves computational speed without sacrificing accuracy. The core innovation lies in approximating the traditionally used 3-order B-spline basis with Gaussian radial basis functions (RBFs) [12]. The RBF network is defined as:

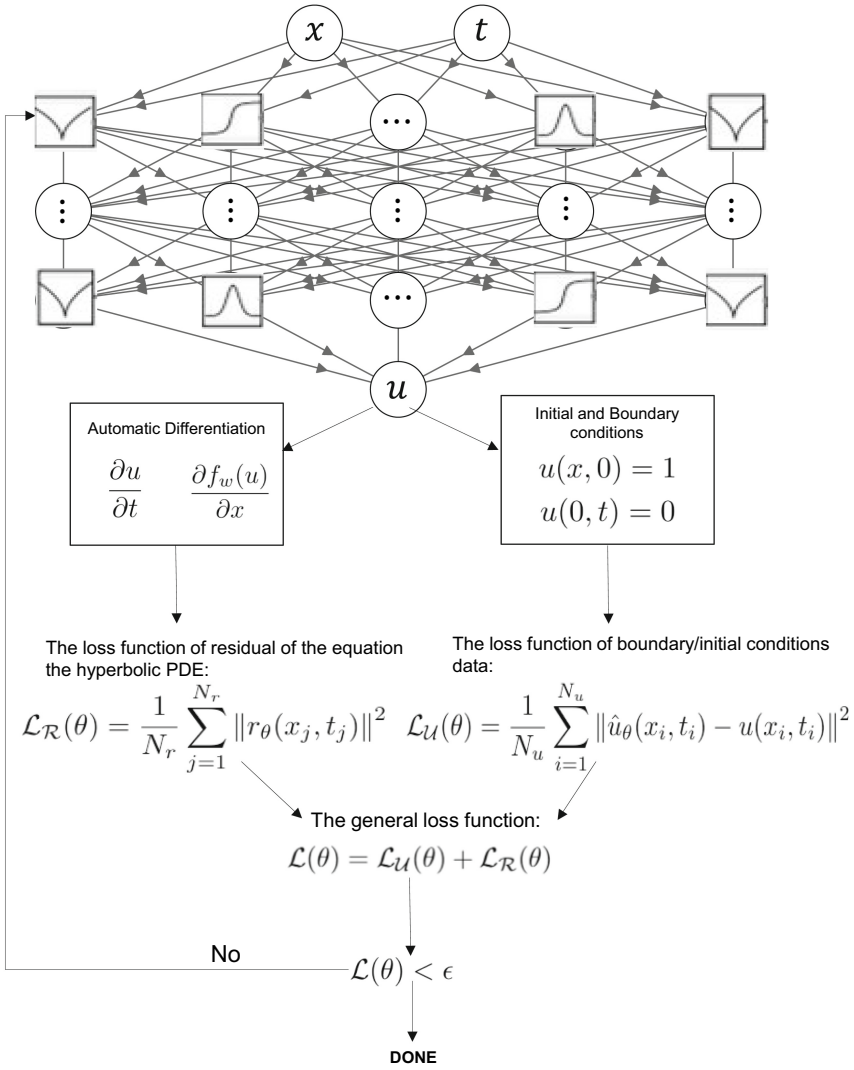


Fig. 2. A schematic representation of the PINN architecture based on the KAN architecture.

$$f(x) = \sum_{i=1}^{N} w_i \phi(||x - c_i||) \tag{13}$$

where w_i are weights and ϕ is a radial basis function. And $\phi(x) = exp(-\frac{r^2}{2h^2})$, where r is the radial distance and h is the spread of a function.

This simplification reduces complexity and computational overhead, achieving over three times the speed of prior KAN implementations. Additionally, the study conclusively shows that KANs can be viewed as a specific type of RBF network with fixed centers.

To implement KAN-PINN, the boundary and collocation points remained consistent with those used in MLP. Instead of employing 8 hidden layers as in the MLP, the KAN-PINN utilized 5 KAN-layers with 5 spline neurons in each, based on observations from [17]. However, instead of Adam optimizer from the observation [17], the KAN-PINN model was trained under L-BFGS optimizer as MLP for comparison purposes.

4 Results

4.1 Assesment of KAN and MLP

This section presents the numerical and graphical results obtained using PINN under MLP and KAN architectures to solve the Buckley-Leverett equation. Includes comparative analyzes of the accuracy and performance of these models.

Figure 3 shows comparisons of the MLP and KAN models for different diffusion coefficients ($\epsilon = 0.01, 0.025, 0.05$) together with the exact solution without diffusion at various time steps. We took the mobility ratio $M = 2$, considering oil to be twice as viscous as water. For all cases, both models closely approximate the exact solution, though some deviation near the saturation front can be observed. As ϵ varies, both architectures show robustness to changes in the diffusion coefficient. However, KAN exhibits slightly better stability as ϵ increases.

In Table 1, we calculated the root mean square error (RMSE) for the PINN MLP and KAN models across all time instances in Fig. 3, quantifying the deviation of models' predictions from the exact solution. The MLP outperforms KAN slightly at $\epsilon = 1 \times 10^{-2}$, but, in general, the KAN achieved comparable accuracy to the MLP. Regarding the training time, KAN models required approximately 2–3 mins, whereas MLP models were trained in about 1–1.5 min. The results in a speedup factor in the range of 2.39–2.56 indicate that MLP learns roughly twice as fast as KAN. The training time difference may be attributed to the simplicity of activation functions in MLP compared to the more complex spline functions in KAN, despite the fact KAN employed fewer parameters and fewer parameters than MLP.

Table 1. RMSE values and training times for MLP and KAN at different ϵ values.

ϵ	1×10^{-2}	2.5×10^{-2}	5×10^{-2}
RMSE_{KAN}	0.092	0.085	0.100
RMSE_{MLP}	0.067	0.084	0.107
Training Time (T_{KAN})	170 s	193 s	149 s
Training Time (T_{MLP})	71 s	80 s	58 s
Speedup ($T_{\text{KAN}}/T_{\text{MLP}}$)	2.39	2.41	2.56

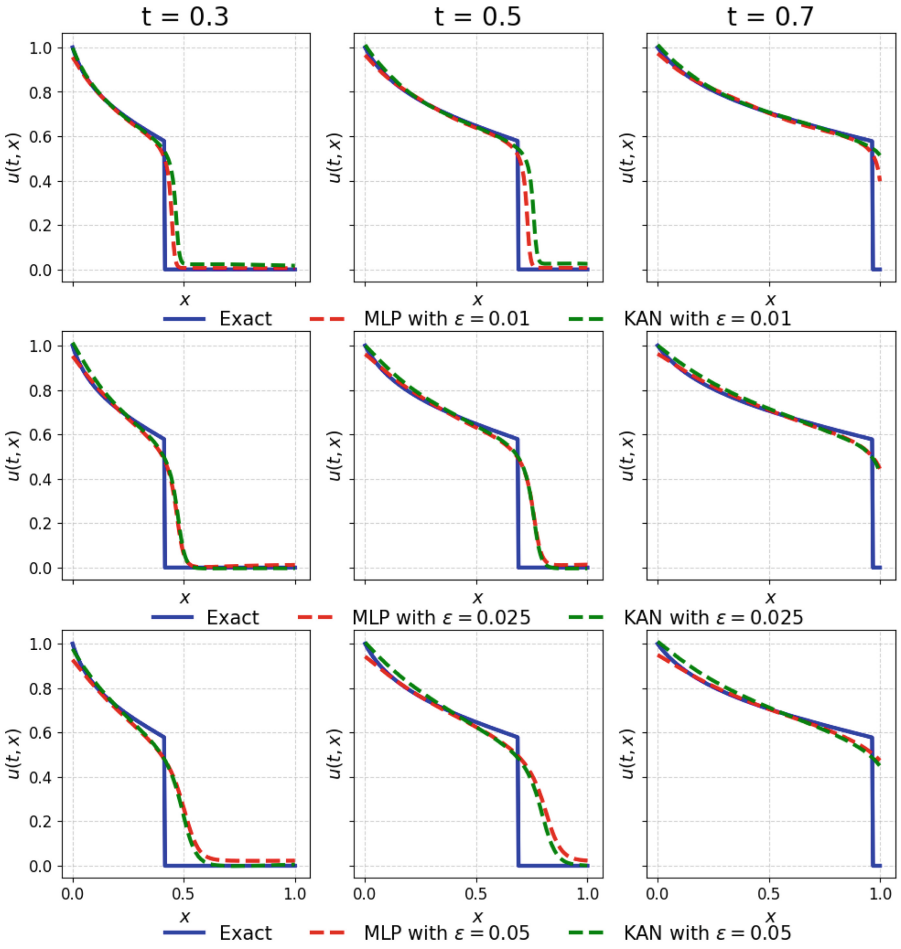


Fig. 3. Predictions of MLP and KAN for different values of diffusion coefficient ϵ with an exact solution without diffusion.

4.2 Optimization Impact on KAN for the Laboratory Experiment

In the following sections, we demonstrate the capabilities of KAN architecture in modeling laboratory experiments. The experimental study was conducted by Omirbekov et al. [16], where displacement of a Dense Non-Aqueous Phase Liquid (DNAPL) by a polymer solution was investigated [21]. We gathered all measured parameters of interacted fluids from the study, including viscosities, boundary conditions to adjust the mobility ratio $M = \frac{\mu_{\text{dnapl}}}{\mu_{\text{polymer}}}$ for PINN models where $\mu_{\text{dnapl}} = 4.47$ and $\mu_{\text{polymer}} = 200$ [20]. This resulted in the mobility ratio $M = 0.0235$, which shows the significant contrast between the viscosities of the DNAPL and polymer phases.

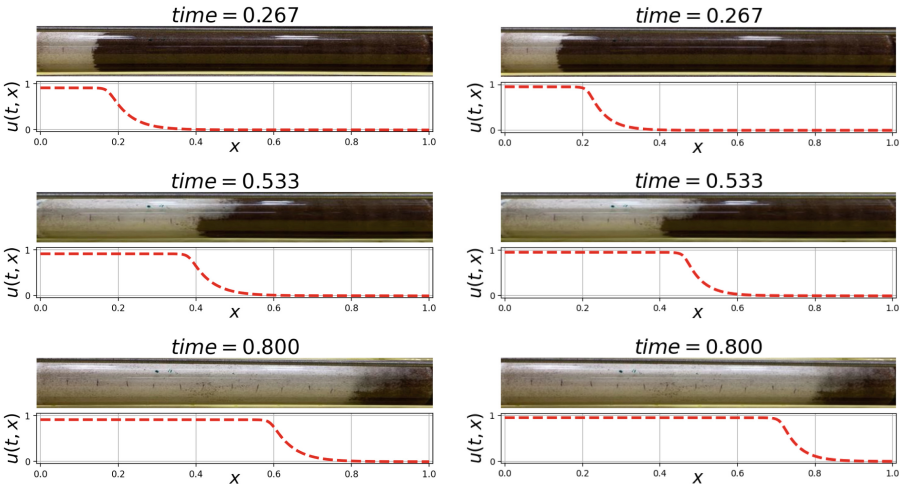


Fig. 4. Comparison of KAN optimization methods at $\epsilon = 3.75 \times 10^{-2}$: (4a) KAN at $\epsilon = 3.75 \times 10^{-2}$ with L-BFGS; (4b) KAN at $\epsilon = 3.75 \times 10^{-2}$ with hybrid optimizers (Adam with L-BFGS).

After several investigations of the diffusion coefficient for the experiment, we determined that $\epsilon = 3.75 \times 10^{-2}$ is optimal. The KAN training parameters, collocation, and boundary points remained the same.

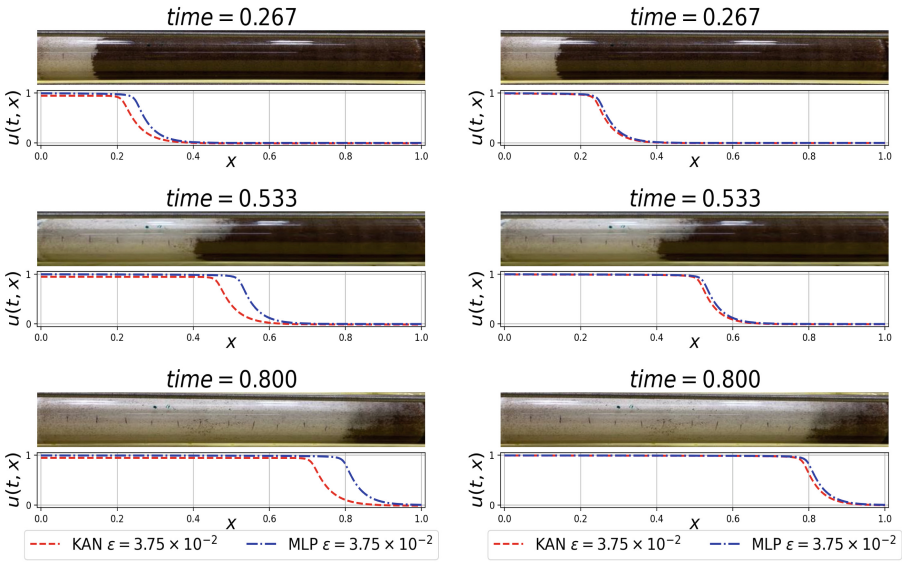
In Fig. 4a, predictions of the KAN model with the laboratory data using L-BFGS optimization were demonstrated at three various time steps. The training loss reached 1.23×10^{-2}. Although the model seemingly captured the saturation front at the first time step ($t = 0.267$), its performance failed at later time steps, indicating that the model may not be fully trained or optimized.

To address the issue, we utilized a hybrid optimization strategy by combining Adam with L-BFGS [2]. Specifically, the KAN model was first trained using Adam for 5000 epochs, then switching to L-BFGS for fine-tuning. Figure 4b shows the results of the hybrid approach. It can be remarked that the hybrid optimizer significantly enhances the performance of the model, especially in capturing the sharp fronts across all time steps. The final training achieved 5.19×10^{-3}, which is notably lower than the one obtained with only L-BFGS.

4.3 Modeling KAN and MLP for the Experimental Data

The section compares the PINN performance of KAN models with MLP models for the laboratory experiment, which is focused on the impact of structural differences in KAN architectures.

The obtained KAN model with a hybrid approach was compared with the MLP model with the same diffusion coefficient ($\epsilon = 3.75 \times 10^{-2}$) and the opti-



(a) KAN [2, 5, 5, 5, 5, 5, 1] vs. MLP (b) KAN [2, 8, 8, 8, 8, 8, 8, 1] vs. MLP

Fig. 5. Comparison of KAN architectures with different layer sizes against MLP at $\epsilon = 3.75 \times 10^{-2}$.

mizer. The comparison is shown in Fig. 5a, where the initial KAN architecture was [2, 5, 5, 5, 5, 5, 1], a structure roughly adapted from the architectural explorations in recent work on PI-KANs [17]. Both models demonstrated comparable results at the first and second time steps ($t = 0.267, t = 0.533$), though the approximation of the KAN at the last time step was lagging than the MLP.

To further investigate the influence of architecture depth, we evaluated another KAN model with the structure [2, 8, 8, 8, 8, 8, 8, 1], also trained with a hybrid optimizer. This deeper architecture was chosen empirically to assess the impact of increased representational capacity on the KAN's performance for this problem. As shown in Fig. 5b, increasing the number of layers and neurons in the KAN structure significantly improved KAN performance, resulting in an approximation identical to the MLP across all time steps. The MLP architecture's depth was kept consistent throughout these comparisons to isolate the effect of the KAN's structural variations.

Another observation was made by comparing the KAN model trained under constant diffusion ($\epsilon = 3.75 \times 10^{-2}$) with the MLP model with a spatially dependent diffusion function ($\epsilon(x) = 0.04x^2 + 0.05x + 0.0015$) adopted from our previous work [9] to evaluate the KAN's performance against it. Figure 6 illustrates this comparison along with laboratory experiment images across three time steps where the KAN is structured with [2, 5, 5, 5, 5, 5, 1] layers. As shown in the figure, KAN's saturation shock approximations are smoother due to its fixed dif-

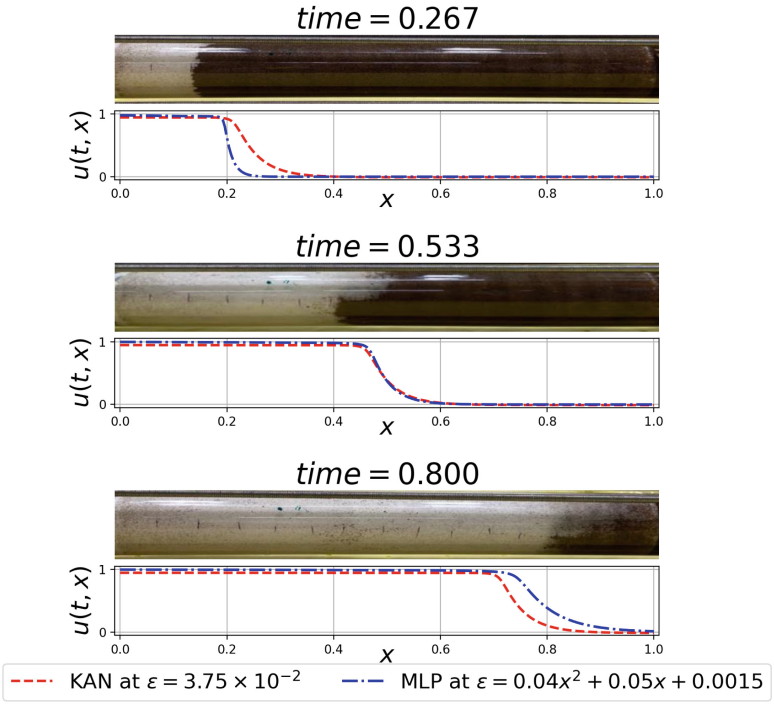


Fig. 6. Comparison of KAN [2, 5, 5, 5, 5, 5, 1] with constant diffusion ($\epsilon = 3.75 \times 10^{-2}$) and MLP with spatially dependent diffusion function ($\epsilon(x) = 0.04x^2 + 0.05x + 0.0015$).

fusion coefficient. Despite this, it closely matches the gradient pattern predicted by MLP. The MLP captures more details, especially with its varying diffusion function. The KAN model, however, performs well even with simpler assumptions, showcasing the trade-off between the MLP's details and the KAN model's efficiency, which makes KAN a practical option for its simplicity and accuracy.

5 Conclusion

This research explored the effectiveness of PINNs with Multi-Layer Perceptron (MLP) and Kolmogorov-Arnold Networks(KAN) architectures in solving the Buckley-Leverett equation for two-phase flow in porous media, validated by the laboratory experimental data.

Comparative analysis revealed that while MLP slightly outperformed KAN regarding training speed, the KAN architecture achieved comparable accuracy and demonstrated improved robustness under varied diffusion conditions. Furthermore, employing a hybrid optimization strategy combining Adam and L-BFGS significantly enhanced KAN's accuracy, especially in capturing sharp saturation fronts observed experimentally rather than using L-BFGS alone.

Additionally, increasing the depth and complexity of the KAN architecture notably improved its predictive accuracy, achieving parity with the MLP results across all tested scenarios. The comparison between the initially architectured KAN model with the constant diffusion coefficient and the MLP model with the spatially dependent diffusion function highlights that while the MLP provides more detailed predictions with varying diffusion, the KAN model demonstrates robust performance and generalization under simpler assumptions, showcasing its effectiveness even without complex spatial variability. These findings underscore the potential of KAN, especially when implemented with optimized network architectures and hybrid training methods, for modeling complex fluid dynamics in porous media.

Acknowledgments. The authors acknowledge the support of the research grants no. AP19575428 and AP22784980, from the Ministry of Science and Higher Education of the Republic of Kazakhstan. The authors appreciate the support of the Nazarbayev University Faculty Development Competitive Research Grant (NUFDCRG), Grant No. 20122022FD4141.

References

1. Rao, X., Liu, Y., He, X., Hoteit, H.: Physics-informed Kolmogorov-Arnold networks (PIKANs) for solving the Buckley-Leverett equation in waterflooding reservoirs. In: SPE Reservoir Simulation Conference, p. D012S001R009. SPE (2025). https://doi.org/10.2118/223891-MS
2. Almajid, M.M., Abu-Al-Saud, M.O.: Prediction of porous media fluid flow using physics informed neural networks. J. Petrol. Sci. Eng. **208**, 109205 (2022)
3. Baydaulet, U., Omarova, P., Merembayev, T., Yedilkhan, A.: Modeling siltation of river channels using the physics-informed neural networks method and numerical simulation. Eng. Sci. **33**, 1296 (2024)
4. Bodner, A.D., Tepsich, A.S., Spolski, J.N., Pourteau, S.: Convolutional Kolmogorov-Arnold networks. arXiv preprint arXiv:2406.13155 (2024)
5. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control Sig. Syst. **2**(4), 303–314 (1989)
6. Fuks, O., Tchelepi, H.A.: Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. J. Mach. Learn. Model. Comput. **1**(1), 19–37 (2020)
7. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pp. 249–256. JMLR Workshop and Conference Proceedings (2010)
8. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
9. Kalesh, D., Merembayev, T., Omirbekov, S., Amanbek, Y.: Application of physics-informed neural networks for two-phase flow model with variable diffusion and experimental validation. Results Eng. **26**, 105439 (2025)
10. Lax, P.D.: Hyperbolic systems of conservation laws and the mathematical theory of shock waves. SIAM (1973)

11. LeVeque, R.J.: Finite Volume Methods for Hyperbolic Problems, vol. 31. Cambridge university press (2002)
12. Li, Z.: Kolmogorov-Arnold networks are radial basis function networks. arXiv preprint arXiv:2405.06721 (2024)
13. Liu, Z., Ma, P., Wang, Y., Matusik, W., Tegmark, M.: KAN 2.0: Kolmogorov-Arnold networks meet science. arXiv preprint arXiv:2408.10205 (2024)
14. Liu, Z., et al.: KAN: Kolmogorov-Arnold networks. arXiv preprint arXiv:2404.19756 (2024)
15. Mehrabian, A., Adi, P.M., Heidari, M., Hacihaliloglu, I.: Implicit neural representations with fourier kolmogorov-arnold networks. arXiv preprint arXiv:2409.09323 (2024)
16. Omirbekov, S., et al.: Experimental study of DNAPL displacement by a new densified polymer solution and upscaling problems of aqueous polymer flow in porous media. J. Contam. Hydrol. **252**, 104120 (2023)
17. Wang, Y., et al.: Kolmogorov–Arnold-informed neural network: a physics-informed deep learning framework for solving forward and inverse problems based on kolmogorov–arnold networks. Comput. Methods Appl. Mech. Eng. **433**, 117518 (2025). https://doi.org/10.1016/j.cma.2024.117518
18. Xu, J., et al.: Fourierkan-GCF: fourier kolmogorov-arnold network–an effective and efficient feature transformation for graph collaborative filtering. arXiv preprint arXiv:2406.01034 (2024)
19. Yang, X., Wang, X.: Kolmogorov-arnold transformer. arXiv preprint arXiv:2409.10594 (2024)
20. Omirbekov, S., Zhaidarbek, B., Amanbek, Y.: Effect of fluid rheology on sand production: Comprehensive study. In: ARMA US Rock Mechanics/Geomechanics Symposium, p. D042S062R010. ARMA (2024)
21. Zhumabek, M., et al.: Stability and rheological characterization of colloidal gas aphrons: influence of xanthan gum and sodium dodecyl sulfate. Discover Appl. Sci. **7**(5), 1–18 (2025)