Advancing Petroleum Engineering Solutions: Integrating Physics-Informed Neural Networks for Enhanced Buckley-Leverett Model Analysis

A PREPRINT

Jingjing Zhang Department of Petroleum Engineering Texas A&M University blingbling1996@tamu.edu Ulisses Braga-Neto Department of Electrical & Computer Engineering Texas A&M University ulisses@tamu.edu

Eduardo Gildin Department of Petroleum Engineering Texas A&M University egildin@tamu.edu

April 22, 2024

ABSTRACT

Physics-Informed Neural Networks (PINNs) integrate physical principles - typically mathematical models expressed as differential equations - into the machine learning (ML) processes to guarantee the physical validity of ML model solutions. This approach has gained traction in science and engineering for modeling a wide range of physical phenomena, such as wave propagation, fluid dynamics, and turbulence. Nonetheless, the effectiveness of PINNs in solving nonlinear hyperbolic partial differential equations (PDEs), is found challenging due to discontinuities inherent in such PDE solutions. While previous research has focused on advancing training algorithms, our study highlights that encoding precise physical laws into PINN framework suffices to address the challenge. By coupling well-constructed governing equations into the most basic, simply structured PINNs, this research tackles both data-independent solution and data-driven discovery of the Buckley-Leverett equation, a typical hyperbolic PDE central to modeling multi-phase fluid flow in porous media. Our results reveal that vanilla PINNs are adequate to solve the Buckley-Leverett equation with superior precision and even handling more complex scenarios including variations in fluid mobility ratios, the addition of a gravity term to the original governing equation, and the presence of multiple discontinuities in the solution. This capability of PINNs enables accurate, efficient modeling and prediction of practical engineering problems, such as water flooding, polymer flooding, inclined flooding, and CO_2 injection into saline aquifers. Furthermore, the forward PINN framework with slight modifications can be adapted for inverse problems, allowing the estimation of PDE parameters in the Buckley-Leverett equation from observed data. Sensitivity analysis demonstrate that PINNs remain effective under conditions of slight data scarcity and up to a 5% data impurity. Remarkably, vanilla PINNs can learn more than one parameters of the Buckley-Leverett concurrently: both the mobility ratio and the gravity term.

This research demonstrates the versatility and robustness of PINNs in their most elemental form for solving and discovering nonlinear hyperbolic PDE models that exhibit intricate solution behaviors, focusing on their applications on real-world engineering challenges. We provide insights into the construction of governing equations for PINNs to integrate which is generally applicable for other physical processes governed by nonlinear hyperbolic PDEs.

Keywords Physics-informed neural networks, Multi-phase fluid transport, Inverse problems, Gravity effect, Water flooding, CO_2 injection

1 Introduction

For the past decades, machine learning (ML) has undergone a transformative evolution, extending its influence beyond domains such as computer vision [Krizhevsky et al., 2012], natural language processing [Vaswani et al., 2017], and pattern recognition [Braga-Neto, 2020]. ML has emerged as a powerful complement or an alternative to traditional analytical and numerical simulation tools in the fields of science and engineering. ML algorithms automatically decipher solutions by analyzing extensive input data, providing swift and direct outcomes. However, they are not without limitations. ML methods rely on extensive datasets, which can be scarce in scientific and engineering fields. Furthermore, ML models struggle when making predictions outside their training data range, and the modeled results may lack physical realism due to the purely data-driven nature.

In addressing these limitations, Physics-Informed Machine Learning (PIML) methods have emerged as a powerful bridge between data-driven and physics-based approaches. PIML integrates fundamental physical principles into ML training through five primary approaches: 1) feeding physically plausible synthetic data as inputs; 2) postprocessing to filter out non-physical solutions; 3) initializing model parameters via transfer learning from simpler, physics-compliant tasks; 4) customizing neural network architectures to adhere to specific physical constraints; and 5) encoding governing physical laws into the model's loss function [Latrach et al., 2023]. PIML has found applications in subsurface energy, particularly in the oil and gas sector, such as geoscience data interpretation, autonomous directional drilling, production forecasting, reservoir characterization, and Carbon Capture, Utilization, and Storage (CCUS) simulations [Wang and Chen, 2023, Latrach et al., 2023].

Among the array of PIML methodologies, Physics-Informed Neural Network (PINN), introduced by Raissi et al. in 2019, stands out for the most explicit integration of physics, thereby gaining widespread adoption. This method preserves the physical integrity by incorporating governing equations into the neural network's training loss function. These equations act as informative priors, steering the model towards physically plausible solutions. PINNs excel in both data-independent solving and data-driven discovery of governing equations. In forward or inference problems, neural networks solve the governing equations by minimizing the loss on governing PDE residues, as well as losses on initial and boundary conditions. In inverse or identification problems, PINNs employ observed data to unearth unknown parameters within the governing equations, all while adhering to the constraints imposed by PDE residues[Raissi et al., 2019, Fraces et al., 2020].

This paper focuses on the application of PINNs to address a two-phase fluid transport problem mathematically represented by a nonlinear hyperbolic PDE, known as the Buckley-Leverett equation [1942]. This equation models the mass conservation of two-phase displacement processes, such as water displacing oil and CO₂ displacing brine, playing a pivotal role in subsurface hydrocarbon reservoirs and carbon sequestration studies. Its solution involves a shock wave and a rarefaction wave connected by a sharp transition. This inherent complexity and nonlinearity of the Buckley-Leverett equation make analytical or numerical solutions, using methods like finite difference or finite element, challenging. Prior efforts have been made to apply PINNs to solve the Buckley-Leverett problems and encountered difficulties, with Fuks and Tchelepi [2020] pointing to the solution's discontinuity as a key challenge for vanilla (basic form) PINNs. As a response, enhancement methods have been proposed including the introduction of artificial viscosity terms to the equation, attention-based mechanisms, and convex hull construction on the flux function to aid the problem. Adding an artificial viscosity or diffusion term transforms the Buckley-Leverett equation from hyperbolic to parabolic, approximating the exact solution as the viscosity coefficient nears zero [Fuks and Tchelepi, 2020, Fraces et al., 2020, Coutinho et al., 2023]. However, this approach can lead to a smoothed shock front that mimics the diffused shock front seen with traditional numerical methods' truncation or discretization errors, thereby diminishing the strength of PINNs. Attention-based mechanisms offer a way for neural networks to focus on specific data segments, adjusting the 'attention' level to different elements in the sequence Rodriguez-Torrado et al. [2022], Diab et al. [2022]. Nevertheless, attention-based PINNs may complicate both the implementation and computational demands of PINNs. Alternatively, Welge's construction imposes the Rankine-Hugoniot condition and Oleinik entropy condition to avoid the non-physical, multi-valued solutions caused by the original flux function. By constructing a convex hull for the flux function, it establishes a criterion for shock propagation, resulting in a sharp, physically plausible shock front Welge [1952], Fraces and Tchelepi [2021], Latrach et al. [2023].

This study emphasizes the necessity of providing an accurate and physically meaningful governing equation for PINNs to achieve solutions that align with physical reality. We underscore the essential role of construction of governing equations, positioning it not merely as an auxiliary tool but as an indispensable element in tackling hyperbolic PDEs, such as the Buckley-Leverett equation. In this study, we first replicate Fraces' work by training the Buckley-Leverett

equation with the constructed flux function using a vanilla PINN. We then extend the investigation by introducing the gravity term into the equation, which poses a more complex PDE to solve. Next, we evaluate the sensitivity of PINNs' performance to varying parameters within the Buckely-Leverett equation including the mobility ratio and dip angle, and further solve a more challenging situation where two discontinuities exist in the solution due to the mutual solubility of the displacing and displaced phases. Our results show the proficiency of standard PINNs in managing a diverse array of situations, from water flooding and polymer flooding to inclined flooding and CO_2 injection into saline aquifers. On the other hand, this work delves into the application of PINNs to the inverse problems, aiming to estimate the parameters of the Buckley-Leverett PDE from observed data. Starting with the mobility ratio as a learnable parameter, we examine the impact of variations in the volume and quality of observed data on PINNs' effectiveness. Furthermore, we conduct a two-parameter training exercise, identifying both the mobility ratio and gravity term. Throughout these endeavors, vanilla PINNs demonstrate great performance, showing their robustness and versatility in data-drive discovery of the Buckley-Leverett equation.

The remainder of this paper is organized as follows: Section two provides an introduction to the training algorithm of PINNs, including basic concepts and overall workflow; Section three details the physical law, including the derivation of Buckley-Leverett equation, derivation of the flux function, construction of convex hull, and the variant for modeling CO_2 -brine displacement; Section four presents the results of the implementation of forward and inverse PINN training. Finally, we summarize the key findings and discuss their implications for future research.

2 Physics-informed neural networks

Physics-Informed Neural Networks (PINNs) rely on a structured and systematic workflow, as illustrated in Fig. 1, to model the evolution of systems with spatio-temporal dynamics. This workflow involves a sequence of key steps aimed at approximating the solution u(x, t). Initially, an Artificial Neural Network (ANN) processes input data sampled across spatial and temporal domains, providing a preliminary solution estimate. Following this, the automatic differentiation (AD) algorithm is utilized to compute the derivatives of the NN output with respect to its input coordinates and model hyperparameters, which are critical in calculating the loss function and in updating the model parameters, respectively. Through this iterative process of minimizing the composite loss, the model's weights and biases are continually optimized, progressively refining the NN's solution \hat{u} , and gradually converging towards the exact solution u. The criteria for stopping the training process can be based on reaching a user-defined loss threshold or a maximum number of iterations. The upcoming subsections will provide a detailed examination of each element within the PINN framework, dissecting the workflow and its key components for a comprehensive understanding.



Figure 1: Forward physics-informed neural network workflow.

2.1 Neural network (NN)

The neural network, inspired by the structure and functioning of the human brain's neural networks, consists of input layers, multiple hidden layers, and output layers. These layers are composed of interconnected processing units, known as neurons, which execute computations and pass information by summing the weighted inputs they receive and then applying an activation function to the result. Neural networks have been shown to be universal function approximators

[Hornik et al., 1989], capable of representing a wide range of complex relationships between inputs and outputs. A neural network with L layers and n_l neurons in the *i*-th layer is defined as

$$y_{i}^{l} = \sigma \left(\sum_{j=0}^{n_{l}-1} w_{ij}^{l} y_{j}^{l-1} + b_{i}^{l} \right), \quad i = 1, \dots, n_{l}$$
(1)

for l = 1, 2..., L, α is the activation function, w_{ij}^l and b_i^l are weights and biases, respectively, y_j^{l-1} are the outputs of the previous layer. For the first hidden layer y_j^{l-1} is a vector as a function of temporal and spatial inputs. The outputs of the last layer are the solution approximated by NN. The training process consists of finding the minimum of a loss function by optimizing a along with the weights and biases.

Neural networks are primarily classified into three categories: Aritificial Neural Networks (ANNs), Convolutional Neural Networks (CNNs), and Recurrent Neural Networks (RNNs). For the purposes of PINNs, fully-connected ANNs are typically favored due to their straightforward architecture, efficacy, and adaptability in mimicking complex functions. A variety of tools and frameworks support the construction of such models, including TensorFlow's Keras API, TFLearn API, and PyTorch's FastAI. In our research, we employ the keras.models.Sequential() method from TensorFlow's Keras API to develop the PINN model.

2.2 Automation Differentiation(AD)

Automatic Differentiation (AD) is a vital technique in various machine learning and scientific computing applications as it enables the computation of numerical derivatives for arbitrarily complex functions or programs. AD achieves this by systematically applying the chain rule of calculus to break down the derivative of a composite function into the derivatives of its constituent functions [Baydin et al., 2018]. This approach allows for the calculation of derivatives with machine precision and minimal computational overhead, offering excellent asymptotic efficiency. While derivatives can be computed to high orders as needed, the focus is primarily on gradients or Hessians, as demonstrated in Fig. 1 with first-order derivatives, $\frac{\partial \hat{u}}{\partial x}$ and $\frac{\partial \hat{u}}{\partial t}$, serving as examples. By employing the AD algorithm, PINNs gain the ability to accurately compute exact derivatives. This capability proves valuable as it enables PINNs to bypass common issues associated with traditional numerical methods, such as truncation errors and discretization errors, leading to an enhancement in performance.

Several open-source machine learning frameworks have developed readily accessible tools for the implementation of AD. These tools include the 'tf.GradientTape()' function of TensorFlow (2.x), the 'autograd' package of PyTorch, and the JAX library. In the course of this study, the 'tf.GradientTape()' is employed to compute gradients of the solution with respect to time and space for PDE residues, as well as gradients of the loss concerning model hyperparameters (weights and biases) for PINN training.

2.3 Integration of physical laws

Physical phenomena are characterized by mathematical expressions frequently in the form of PDEs. These equations involve unknown functions and their partial derivatives concerning independent variables like time and space. Certain PDEs present unique challenges for numerical solutions, attributed to factors such as pronounced nonlinearity, the prevalence of convection effects, or the presence of shocks.

A generalized form of a PDE can be written as follows:

$$u_t + \mathcal{N}[u] = 0$$
, where $x \in \Omega \subset \mathbb{R}^d, t \in [0, T]$ (2)

In this equation, $N[\cdot]$ is a nonlinear differential operator, Ω is a subset of \mathbb{R}^d , and u(x,t) represents the exact solution to be determined. During the machine learning process, this solution can be approximated by the NN modeled value $\hat{u}(x,t)$, achieved through the minimization of a composite loss function defined as follows:

Loss function
$$= \mathcal{L}_{PDE} + \mathcal{L}_{IC} + \mathcal{L}_{BC}$$
 (3)

It comprises three essential components: the loss associated with the governing PDE residual \mathcal{L}_{PDE} , the loss related to initial conditions \mathcal{L}_{IC} , and the loss concerning boundary conditions \mathcal{L}_{BC} . \mathcal{L}_{PDE} serves as a regularization mechanism penalizing deviations from the governing equation, as given by

$$\mathcal{L}_{PDE} = \frac{1}{N_r} \sum_{i=1}^{N_r} \left| \widehat{u}_t \left(x_r^{\ i}, t_r^{\ i} \right) + \mathcal{N} \left[\widehat{u} \left(x_r^{\ i}, t_r^{\ i} \right) \right] \right|^2 \tag{4}$$

Here, $\{x_r^i, t_r^i\}_{i=1}^{N_r}$ are data points sampled within the domain of time and space, referred to as collocation points. \mathcal{L}_{IC} and \mathcal{L}_{BC} regulate the solution to adhere to the initial condition $u(x_0, 0)$ and boundary conditions $u(x_b, x_b)$, as given by Eq.5 and Eq.6, respectively.

$$\mathcal{L}_{IC} = \frac{1}{N_0} \sum_{i=1}^{N_0} \left| \hat{u} \left(x_0^{i}, 0 \right) - u \left(x_0^{i}, 0 \right) \right|^2 \tag{5}$$

$$\mathcal{L}_{BC} = \frac{1}{N_b} \sum_{i=1}^{o} \left| \widehat{u} \left(x_b^{\ i}, t_b^{\ i} \right) - u \left(x_b^{\ i}, t_b^{\ i} \right) \right|^2 \tag{6}$$

in which $\{x_0{}^i, t_0{}^i\}_{i=1}^{N_0}$ denote the initial condition data points and $\{x_b{}^i, t_b{}^i\}_{i=1}^{N_b}$ represent the boundary condition data points. Typically, the number of collocation points greatly exceeds the number of initial or boundary condition points. The composite loss function effectively quantifies the disparity between the exact solution and the approximation provided by the NN. It serves as a guiding metric for adjusting the weights and biases of the NN through specific optimization algorithms.

PDEs like Eq.6 can be categorized into three types: parabolic, elliptic, or hyperbolic, each modeling different physical phenomena. Elliptic PDEs typically describe steady-state scenarios, whereas parabolic PDEs are used to model diffusive processes that exhibit gradual solution gradients. Hyperbolic PDEs, like the Buckley-Leverett equation, are characterized by their ability to simulate wave propagation at finite speeds, often resulting in the formation of shocks or discontinuities. These discontinuities present challenges for solving the equations. A more detailed exploration of their characteristics and methods of solution will be undertaken in Section 3.

2.4 Forward and inverse problems

Fig. 1 and our discussion so far introduce the idea of solving PDEs to predict future behaviors without using any labeled data, known as forward problems. Actually PINNs offer a versatile framework for addressing both forward and inverse problems with minimal modification in code [Fraces et al., 2020, Cuomo et al., 2022]. The inverse problems shift the focus towards learning the underlying solution map and identifying uncertain parameters of governing equations, such as rock and fluid properties in this study. The training for inverse problems involves iterative refinement of both model hyperparameters and specific PDE parameters to align the model outputs with observed data. This iterative process distinguishes forward problems as a form of unsupervised learning, whereas inverse problems align more closely with supervised learning paradigms. The loss function for inverse PINNs comprised of two components: the mean square error associated with PDE residues and observed data \mathcal{L}_{data} , as demonstrated in Eq. 7.

$$Loss function = \mathcal{L}_{PDE} + \mathcal{L}_{data}$$
(7)

 \mathcal{L}_{data} quantifies the difference between the model-predicted solutions and the observed data as, Eq. 8 describes.

$$\mathcal{L}_{\text{data}} = \frac{1}{N_s} \sum_{i=1}^{N_s} \left| \hat{u} \left(x_s^{\ i}, t_s^{\ i} \right) - u \left(x_s^{\ i}, t_s^{\ i} \right) \right|^2 \tag{8}$$

 $\{x_s^i, t_s^i\}_{i=1}^{N_s}$ are observed or labeled data points. Ideally, the number of labeled data points is comparably large as the number of collocation points.

3 Multi-phase fluid flow model

This study focuses on multi-phase fluid flow dynamics, particularly focusing on processes where water displaces oil and CO_2 displaces brine. In water flooding, water injection into a reservoir displaces the oil left unrecovered by primary depletion and thus improves oil recovery. By adding a polymer to the water, the viscosity of the displacing phase is increased, thereby boosting displacement efficiency and recovery rates in what is known as polymer flooding. Given that water and oil are almost immiscible, these two processes are aptly described by the original Buckley-Leverett equation. However, for processes that feature mutual solubility between the displacing and displaced phases, like CO_2 displacing brine, the Buckley-Leverett model requires modification. This process, injecting CO_2 in its supercritical state into subsurface saline aquifers, can reduce greenhouse gas emissions and combat climate change [Green et al., 1998].

To deepen the understanding of the physical principles involved, this section starts with the derivation of the original Buckley-Leverett equation and its flux function for water flooding modeling. We will then expound on the reasons why these equations are inadequate as governing equations and consequently introduce Welge's method to construct more precise governing equations for PINN training. Following this, we will detail the modified Buckley-Leverett equation for CO_2 injection modeling.

3.1 Derivation of the Buckley-Leverett model

In 1942, Buckley and Leverett formulated a fundamental equation to characterize the transport of two immiscible fluids in the porous medium based on mass conservation and Darcy's law. The mass conservation equation states that the difference in water mass entering and exiting a control volume is equal to the accumulated change in water mass within the control volume of length Δx over a specific time period Δt , as Fig.2 displays. It is mathematically described by Eq.9.



Figure 2: Mass Conservation in a finite volume.

$$\underbrace{\rho_w q_{w,x} \Delta t - \rho_w q_{w,x+\Delta x} \Delta t}_{\text{in - out}} = \underbrace{(\rho_w \phi S_w \Delta V)_{t+\Delta t} - (\rho_w \phi S_w \Delta V)_t}_{\text{accumulation}}$$
(9)

where q_w is the volumetric water rate, S_w denotes the water saturation, and α represents the angle of flow deviation from the horizontal plane. Positive α indicates up dip flow while negative α indicates down dip flow. Under the assumption of incompressibility for both fluids and rocks (constant water density ρ_w and rock porosity ϕ), the equation simplifies to:

$$\frac{q_{w,x} - q_{w,x+\Delta x}}{\Delta x} = A\phi \frac{S_{w,t+\Delta t} - S_{w,t}}{\Delta t}$$
(10)

in which A is the cross-sectional area perpendicular to the flow direction. As Δx and Δt approach zero, the conservation law takes the form of a partial differential equation:

$$A\phi \frac{\partial S_w}{\partial t} + \frac{\partial q_w}{\partial x} = 0 \tag{11}$$

Introducing f_w , the fractional flow or flux function, which is the ratio of the water flow rate to the total flow rate $(q_t = q_w + q_o)$, yields:

$$f_w = \frac{q_w}{q_t} = \frac{q_w}{q_w + q_o} \tag{12}$$

Incorporating q_w from this fractional flow into Eq.11 results in:

$$A\phi \frac{\partial S_w}{\partial t} + q_t \frac{\partial f_w}{\partial x} = 0 \tag{13}$$

To further simplify, we introduce dimensionless time t_D and length x_D :

$$\frac{\partial S_w}{\partial t_D} + \frac{\partial f_w}{\partial x_D} = 0 \tag{14}$$

where,

$$x_D = \frac{x}{L}, t_D = \frac{q_t t}{AL\phi} = \frac{\text{volume injected}}{\text{pore volume}}$$
(15)

And L is the total length in the direction of fluid flow. The initial and boundary conditions of Eq.14 are as follows:

$$S_w (x_D, t_D = 0) = S_{wc}$$

$$S_w (x_D = 0, t_D) = 1 - S_{or}$$
(16)

3.2 Derivation of the flux function

In Eq. 14, the flux function or fractional flow f_w is a function of water saturation S_w . As Darcy's law states, the fluid flow rate in porous media is directly proportional to the pressure gradient and the medium's permeability, and inversely proportional to fluid viscosity. Accordingly, the flow rates of water and oil phases can be described by Eq.17 and Eq.18.

$$q_w = -\frac{kk_{rw}A}{\mu_w} \left(\frac{dp_w}{dx} + \rho_w g \sin\alpha\right) \tag{17}$$

$$q_o = -\frac{kk_{ro}A}{\mu_o} \left(\frac{dp_o}{dx} + \rho_o g \sin\alpha\right) \tag{18}$$

in which μ_w and μ_o are the viscosities of water and oil, respectively. k is the absolute permeability of rock. g is the gravitational constant. The pressure difference between the oil and water phase is defined by a capillary pressure p_c ($p_c = p_o - p_w$) and the density difference of two phases is written as $\Delta \rho = \rho_w - \rho_o$. With these definitions and by subtracting Eq.18 from Eq.17, we derive:

$$\frac{k_{ro}}{\mu_o} f_w q_t - \frac{k_{rw}}{\mu_w} \left(1 - f_w\right) q_t = \frac{k k_{ro} k_{rw} A}{\mu_o \mu_w} \left(\frac{dp_c}{dx} - \Delta \rho g \sin \alpha\right)$$
(19)

Rearranging this equation yields the full expression for fractional flow:

$$f_w = \frac{1 + \frac{kk_{ro}A}{q_t\mu_o} \left(\frac{dp_c}{dx} - \Delta\rho g\sin\alpha\right)}{1 + \frac{k_{ro}\mu_w}{k_{rw}\mu_o}}$$
(20)

Usually p_c is sufficiently small to be ignored, so Eq.20 becomes

$$f_w = \frac{1 - \frac{kk_{ro}A}{q_t\mu_o}\Delta\rho g\sin\alpha}{1 + \frac{k_{ro}\mu_w}{k_{rw}\mu_o}}$$
(21)

The relative permeabilities for water (k_{rw}) and oil (k_{ro}) describe the effective permeability of the medium to each fluid in the presence of both. They are typically modeled using the Corey-Brook equation [Brooks and Corey, 1966], with S being the effective water saturation:

$$S = \frac{S_w - S_{wc}}{1 - S_{wc} - S_{or}}$$
(22)

$$k_{rw} = k_{rw}^{0} S^{n_{w}} k_{ro} = k_{ro}^{0} (1 - S)^{n_{0}}$$
(23)

 k_{rw}^0 is the maximum value of k_{rw} occurring at the lower endpoint of the water saturation profile (connate water saturation (S_{wc}) and k_{ro}^0 is the maximum value of k_{ro} occurring at the upper endpoint of water saturation profile (residual oil saturation $1 - S_{or}$). Fig.3 illustrates an example of k_{rw} and k_{ro} profiles as functions of water saturation. The mobility ratio M, as defined by Eq.24, quantifies the relative mobility of two phases. Typically, the displacing



Figure 3: Relative permeability profiles for water and oil .

phase (water) is much more mobile than the displaced phase (oil), resulting in a value of M lower than 1. In the case of polymer flooding, where the viscosity of the displacing phase is increased through polymer addition, the value of M can approach 1.

$$M = \frac{\lambda_w}{\lambda_o} = \frac{kk_{rw}^0 / \mu_w}{kk_{ro}^0 / \mu_o} = \frac{k_{rw}^0 \mu_o}{k_{ro}^0 \mu_w}$$
(24)

Additionally, a gravity number N is introduced to quantify the effect of gravity on fluid flow velocity, expressed as the ratio of gravity to viscous forces, as Eq. 25 shows.

$$N = \frac{kk_{ro}^0 A \Delta \rho g}{q_t \mu_o} \tag{25}$$

With these definitions, f_w is ultimately expressed as:

$$f_w = \frac{1 - (1 - S)^{n_o} N \sin \alpha}{1 + \frac{(1 - S)^{n_o}}{M S^{n_w}}}$$
(26)

3.3 Construction of convex hull

For enhanced clarity and understanding, Eq.14 is hereafter simplified as follows:

$$\frac{\partial u}{\partial t} + \frac{df(u)}{du}\frac{\partial u}{\partial x} = 0, \quad t\in[0,\infty], x\in[0,1]$$
(27)

Here, u(x,t) symbolizes water saturation in the water flooding scenario or gas saturation in the CO₂ injection scenario. A typical flux function f(u) curve is depicted with a dashed line in Fig.4a. The derivative $\frac{df(u)}{du}$ represents the propagation velocity for a specific saturation level u. Using this velocity to determine how far a certain saturation level travels over time enables the construction of a saturation profile, as plotted by the dashed curve in Fig. 4b. However, this profile is physically implausible because the water saturation is triple-valued at a single location. This irrationality originates from the non-convex nature of the flux function, where the velocity $\frac{df(u)}{du}$ initially increases with increasing saturation, peaks, and subsequently decreases, causing higher saturation levels to overtake lower ones and forming a shock front [Dake, 1983]. Because of the shock, the mathematical expression of the Buckley-Leverett problem in Eq.26, assuming continuity and differentiability of u, fails to accurately represent the dynamics ahead the shock, in the saturation range of $[u_f, u_l]$, Eq.26 remains valid. Here, u_f represents the front saturation, while u_l and u_r represent the initial states on the left and right sides of the shock front, respectively. To develop a fully valid solution for the Buckley-Leverett



Figure 4: Flux function (left) and resultant saturation profile (right).

problem, we introduce the concept of Riemann problems, which are hyperbolic conservation laws accompanied by piecewise initial conditions such as Eq. 28.

$$u(x,0) = \begin{cases} u_l, & x \le 0\\ u_r, & x > 0 \end{cases}$$
(28)

The Buckley-Leverett model with its non-convex flux function is a classic example of a Riemann problem, where the left state is $u_l = 1 - S_{or}$ and the right state is $u_r = S_{wc}$. In numerical analysis, the impose of Rankine-Hugoniot jump condition and the E-condition of Oleinik on the flux function help to select the unique and proper solution for Riemann problems: [LeVeque and Leveque, 1992]. The E-condition of Oleinik, as shown in Eq.29, ensures that the rarefaction wave trailing the shock does not surpass it, thereby adhering to the second law of thermodynamics, which mandates that entropy in an isolated system should not decrease:

$$\frac{f(u) - f(u_r)}{u - u_r} \le \frac{df(u_f)}{du_f} \le \frac{f(u) - f(u_l)}{u - u_l}$$
(29)

The Rankine-Hugoniot jump condition equates the flow rates of the displacing fluid on either side of the shock, ensuring the conservation of mass across the shock front. It helps determine the speed at which a shock wave moves through the medium, as shown in Eq.30.

$$\frac{df(u_f)}{du_f} = \frac{f(u_l) - f(u_r)}{u_l - u_r}$$
(30)

From points at $(u_r, 0)$ to $(u_f, f(u_f))$, the original f(u) curve is replaced by a straight line segment to represents a shock jumping from $u = u_r$ to $u = u_f$, forming a convex hull [Welge, 1952]. These two constraints ensure that the solutions are not only mathematically consistent with the conservation laws but also physically realistic. With these constraints, the flux function can be constructed as solid lines in Fig.4a, which can be mathematically expressed as:

$$f(u) = \begin{cases} (u - u_r) \times \frac{f(u_l)}{u_l - u_r}, & \text{if } u_r \le u \le u_f \\ \frac{1 - (u_l - u)^2 N \sin \alpha}{1 + \frac{(u_l - u)^2}{M(u - u_r)^2}}, & \text{if } u_f < u \le u_l \end{cases}$$
(31)

In this equation, $n_0 = n_w = 2$. The solution can be can be obtained accordingly, as shown by the solid line in Fig. 4b. This construction correctly presents that when pure water is pumped into a 1D oil reservoir, at a production well (x = 1), one obtains pure oil until the water front arrives, followed by a mixture of oil and water with increasing water cut as time goes on [Araujo et al., 2020].

3.4 Two-shock BL model for CO₂ injection

Beyond the assumptions of incompressibility of rock and fluids, steady flow, and negligible capillary pressure, the original Buckley-Leverett model is based on the premise of strict immiscibility between two phases. Modifications by Noh et al. 2007, Burton et al. 2009, and Azizi et al.2013 have adapted the BL model to include a retardation factor, capturing the partial solubility between CO_2 and brine during displacement. This adjustment accounts for a two-phase, two-component system where phase properties remain constant, regardless of composition. The H₂O-saturated gas is subject to hydrodynamic trapping while the CO_2 -saturated aqueous phase represents the solubility trapping.

The displacement of CO₂ injection involves three flow regions: pure CO₂ region (I), fresh brine region (II), and two-phase region (J) where CO₂ component can dissolve in each other's phases. As illustrated in Fig.5, these regions are separated by the leading shock and the trailing shock, with the saturation being S_{g1} and S_{g2} , respectively. The



Figure 5: Schematic of a miscible gas-water displacement. Two saturation shocks divide the medium into three regions.

velocities of two shocks can be determined by constructing the fractional flow curve and calculated by Eq.32 and Eq. 33 [Noh et al., 2007]. Specifically, the tangent line representing leading front speed spans from point $(D_{I \rightarrow II}, D_{I \rightarrow II})$ to $(f(S_{g1}), S_{g1})$, as shown in Fig.6. Its slope depends on CO₂'s solubility in the aqueous phase. Similarly, the tangent line representing trailing front speed extends from $(D_{II \rightarrow J}, D_{II \rightarrow J})$ to $(f(S_{g2}), S_{g21})$. The slope depends on water's solubility in the gaseous phase. The disappearance of the trailing shock, represented by a zero slope, occurs when water solubility in gas is zero, with $s_{g2} = 1 - s_{wr}$. The different speeds of these two shocks, indicated by the slope values, reveal that the leading shock advances faster than the trailing one.

$$v_{\text{leading}} = \left. \frac{df(u)}{du} \right|_{S_{a1}} = \frac{f(S_{g1}) - D_{\text{I} \to \text{II}}}{S_{g1} - D_{\text{I} \to \text{II}}}$$
(32)

$$v_{\text{trailing}} = \left. \frac{df(u)}{du} \right|_{S_{g2}} = \frac{f\left(S_{g2}\right) - D_{\text{II} \to J}}{S_{g2} - D_{\text{II} \to J}}$$
(33)

The retardation factors D, defined by the phase concentrations in the different regions as Eq.34 and Eq.35 formulated, represent interphase mass transfer (mutual solubility) of CO₂. The concentration of a component in each phase, and consequently the retardation factors, are affected by temperature, pressure, and salinity. $D_{I \rightarrow II}$ and $D_{II \rightarrow J}$, located on the line with a unit slope through the origin, correspond to conditions in pure brine (initial) and pure CO₂ (injection),



Figure 6: Construction of the flux function for miscible gas-water displacement.

respectively. Usually, the solubility of CO_2 in water is markedly higher than that of H_2O in the gas phase.

$$D_{I \to II} = \frac{C_{CO_2,a}^{II}}{C_{CO_2,a}^{II} - C_{CO_2,g}^{II}}$$
(34)

$$D_{\mathrm{II} \to J} = \frac{C_{\mathrm{CO}_{2,a}}^{\mathrm{II}} - C_{\mathrm{CO}_{2,g}}^{J}}{C_{\mathrm{CO}_{2,a}}^{\mathrm{IO}} - C_{\mathrm{CO}_{2,g}}^{\mathrm{II}}}$$
(35)

This approach allows for a comprehensive flux function representation: for gas saturation below s_{g1} and greater than s_{g2} , leading and trailing tangent lines indicative of mutual solubility substitute the original f(u). Within the saturation range of $[S_{g1}, S_{g2}]$, the original flux function remain valid.

4 Implementation and training results

This section presents the results of applying the PINN framework to address both forward and inverse problems. The neural network structure utilized here is simple and straightforward, featuring an input layer with two neurons (for spatial and temporal inputs), eight hidden layers with 20 neurons each, and a single-neuron output layer (for the solution). Model hyperparameters are initialized using the Xavier method and optimized with the Adam optimizer [Kingma and Ba, 2014]. The entire implementation was carried out with TensorFlow 2.x.

4.1 Forward problems

The objective of forward training is to solve the Buckley-Leverett equation by minimizing a composite loss function that includes residual loss, initial condition loss, and boundary condition loss. The training dataset consisted of 10,000 collocation points within the solution domain, along with 300 data points to enforce the initial condition and another 300 data points to enforce the boundary condition, all generated using the Latin Hypercube Sampling (LHS) method. No labeled data is used for this process. The maximum iteration number is set at 20,000.

Initially, the hyperbolic tangent (tanh) function was chosen as the activation function across all layers. This setup, however, led to unphysical results where the solution values fell below zero at the shock front—contrary to the expectation that water saturation levels should remain within the [0, 1] interval. To remedy this, we transitioned to using the sigmoid function for the output layer's activation, ensuring the solution values were constrained within the appropriate range. This change led to a more accurate solution map as displayed in Fig.7.

Subsequent subsections will explore the results from forward training for the base scenario, examine the sensitivity of PINN performance to various fluid mobility, incorporate gravity into the governing equation, and finally tackle a two-shock Buckley-Leverett model.

4.1.1 Base case

For the base case, we exclude the influence of gravity and use a unit mobility ratio (M). Osher's method is utilized to compute analytical solutions, serving as benchmarks for evaluating the PINN predictions[Ketcheson et al., 2020]. The



Figure 7: 3D visualization of Buckley-Leverett solutions: comparing tanh (left) and Sigmoid (right) activation functions for the output layer.

progression of the training is illustrated in Fig.8, displaying how the solution's profile changes over distance at specific time intervals (0.1, 0.4, and 0.9) through various stages of iteration (200, 1000, 5000, and 20000). By the approximately 5000th iteration, the trained solution closely aligns with the exact solution. The ultimate L_2 -norm error of the PINN solution is 4.55% and an L_2 -norm loss is calculated at 1.36E-6, according to Eq.3.



Figure 8: Evolution of solution profiles during forward PINN training for the base case.

To assess the flexibility of standard PINNs in different scenarios, we explored four additional cases. The details of these cases, including their specific parameters, losses, and errors, are compiled in Table 1. They were categorized into two groups for a detailed sensitivity analysis.

4.1.2 Sensitivity analysis on fluid mobility

The first set of cases (base, M1, M2) investigates the impact of mobility ratio. M influences the speed that water front travels and how sharp the front is. The base case uses a unity M to reflect equal mobility between the displacing (water) and displaced (oil) phases. Yet, real-world scenarios, such as water and polymer flooding, often present varied mobility ratios of two fluids. We examine this by setting M = 0.1 for the M1 case and M = 10 for the M2 case.

Table 1: Summary of Forward PINN Training Cases.							
Case	Μ	Gravity Term	Error	Loss			
base	1	0	0.04546	1.36E-06			
M1	0.1	0	0.01408	7.18E-06			
M2	10	0	0.02164	4.86E-06			
N1	1	-3	0.01229	1.19E-06			
N2	1	3	0.01243	2.59E-06			

As summarized in Table 1, cases M1 and M2 yields errors and losses comparable to the base case, indicating the proficiency of standard PINNs in resolving Buckley-Leverett models across different \$M\$ values. This is further evidenced by a side-by-side 2D comparison of analytical and PINN solutions in Fig. 9, where the difference between left (analytical solution) and (PINN solution) right figures is not noticeable. The figure uses color gradations to depict saturation changes over time and space, with cooler hues for higher water saturation and warmer ones for higher oil saturation. The shock front is marked by the transition between these color zones. It can be observed that a smaller M leads to a higher front saturation u_f and a delayed breakthrough time of water t_{bt} (the value of t when x = 1). The influence of M on the front saturation and breakthrough time can also be observed in Fig.10. On the flux



Figure 9: Analytical vs. PINN solution profiles: comparative 2D views for cases with M=0.1(first row), M=1 (second row), and M=10 (third row).

function profile, the point where the dashed line (original fractional flow) intersects with the solid line (constructed fractional flow) signifies the front saturation, with values of 0.30, 0.71, and 0.95 for the three cases, respectively. Additionally, Fig.10b calculates oil recovery factors by integrating the oil rate (1 - u) over time at the producer's location (x = 1). Its slope represents oil production rate. Oil is produced at a constant rate across all cases until the water breakthrough occurs at different times: 0.463 for the low-mobility-ratio case, 0.828 for the base case, and 0.976 for the high-mobility-ratio case. Upon breakthrough, the water cut at the producer jumps from 0 to u_f and continues to rise as flooding advances through the reservoir. A high mobility ratio leads to an early breakthrough, resulting in considerable oil being left unrecovered—an unfavorable scenario. A Moderate mobility ratio delays the breakthrough and improves sweep efficiency. A low mobility ratio causes a late breakthrough, enabling nearly complete oil recovery, marking it as the most advantageous scenario. Another thing to notice in Fig.10b is that oil recovery is predicted until a dimensionless time of 1.2, extending beyond the training data's range of [0,1]. The prediction is visually represented by the blue region in the figure, demonstrating the remarkable ability of PINNs to extrapolate or forecast once the underlying physics is well-learned by the model.



Figure 10: (a) Fractional flow curves and (b) oil recoveries of cases with M=0.1, 1, and 10.

4.1.3 Sensitivity analysis on gravity

The second set of cases (base, N1, N2) incorporates the gravity term $(N \sin \alpha)$ into the Buckley-Leverett equation, adding complexity to the flux function. This gravity term influences the shock front's propagation speed and its definition. The value of the gravity term, as Eq.25 defines, is mainly subject to reservoir's inclination angle of the reservoir (α), rock permeability, the density difference between water and oil, and the flow rate. A positive α indicates upward flooding, whereas a negative α suggests downward flooding. We explore $N \sin \alpha$ values ranging from -3 to 3.

Results from the N1 and N2 cases, detailed in Table 1, indicate small errors and losses of PINN training, demonstrating PINNs' effectiveness in solving gravity-incorporated Buckley-Leverett models. Fig. 11 reinforces this point through a two-dimensional comparison, affirming PINNs' precision in reflecting the comprehensive solution landscape for these scenarios. A negative gravity term results in faster water breakthrough and reduced front saturation, compared to a positive one.

Moreover, the negative gravity effect alters the initial condition of the displacement process. Due to the phase density difference of oil and water, the saturation distribution undergoes changes immediately after water injection starts. As a result, the modeling of down dip flooding cases requires careful adjustment of new boundary conditions from the original value of $1 - S_{or}$. Fig.12 examines closer into gravity's impact on the shape of the fractional flow curve and oil recovery. In scenarios with steeply downward-dipping reservoirs, the value of f(u) may exceed one, indicating a counter-current flow where oil moves upward and water moves downward. This condition promotes water flow but restricts oil production. Conversely, up dip flooding impairs water flow, resulting in a high front saturation u_f and low moving speed. Therefore, we see that positive gravity term leads to more oil displacement at breakthrough, albeit occurring later in Fig.12b. Late t_{bt} is more favorable because displacement efficiency tends to be poor after breakthrough. The system's future solution behaviors are predicted until t=1.2.

4.1.4 Two-shock Buckley-Leverett model

Until now, vanilla PINNs have managed to model the Buckley-Leverett equation featuring a single shock in the solution. Yet, if PINNs are capable of tackling scenarios with two discontinuities in the governing equation remains unknown. To explore this, the adapted Buckley-Leverett equation that models a gas-displacing-water process is additionally trained by PINNs. The data we use is referred from Noh et al. 2007, including retardation factors at -0.45 ($D_{I\rightarrow II}$) and 1.05 ($D_{II\rightarrow J}$) to account for the solubility of CO₂ in aqueous phase and H₂O component in gaseous phase. The viscosities for reservoir brine and injected CO₂ were set at 0.548 cp and 0.189 cp, respectively, with connate water saturation at 0.25 and residual gas saturation at 0. The reservoir conditions are 50 °C and 5000 kPa. The flux function is constructed in the same way as 6.

The training process is displayed in Fig.13. We see that PINN handles two discontinuities very well and the leading shock travels much faster than the trailing shock. The final training loss is achieved at 3.28E-7 and the error margin is 3.90%. The PINN trained solution map is compared with the analytical solution in Fig. 14 and warmer colors indicate areas of high gas saturation, whereas cooler colors signify zones of high water saturation for this case. The gas saturation at the leading (S_{g1}) and training shocks (S_{g2}) are identified as 0.37 and 0.56, respectively. By calculating the derivative of flux function at S_{g1} and S_{g2} , we can obtain the traveling velocities of two gas fronts. Assuming CO₂ is injected into a reservoir as Fig.15 shows for 30 years at a rate of 4 cubic meter per day (reservoir condition), we can



Figure 11: Analytical vs. PINN solution profiles: 2D comparison for cases with $N \sin \alpha$ =-3 (first row), $N \sin \alpha$ =0 (second row), and $N \sin \alpha$ =3 (third row).



Figure 12: (a) Fractional flow curves and (b) oil recoveries of cases with *M*=-3, 0, and 3.

project the expansion of the CO₂ plume. The distance the leading shock spreads is:

$$x|_{S_{g_2}} = \left. \frac{df(u)}{du} \right|_{S_{g_2}} \times t_D \times L = 985.4 \,\mathrm{m}$$
 (36)

The distance the trailing shock reaches is:

$$x|_{S_{g_2}} = \left. \frac{df(u)}{du} \right|_{S_{g_2}} \times t_D \times L = 69.7 \,\mathrm{m}$$
 (37)

4.2 Inverse problems

Inverse problems leverage observed (labeled) data to unravel the properties and dynamics of physical systems. Within the framework of PINNs, inverse training is designed to determine the hidden parameters in the governing equations, thereby



Figure 13: Evolution of solution profiles during forward PINN training for the two-shock case.



Figure 14: Analytical vs. PINN solution profiles: 2D comparison for the two-shock case.

enabling the comprehensive prediction of the system's behavior over space and time, u(x, t). The training procedure and setups for inverse PINNs mirror those of forward PINNs, including neural network architecture, initialization method, and optimization strategies. However, a key difference lies in the composition of the loss function, which includes only the observed data error and PDE residual error, as defined by Eq. 7 for inverse problems. Initial and boundary conditions are unknown in these scenarios. In addition to L_2 -norm error and loss, a parameter error is used to evaluate the inverse PINN training performance, which is quantified as:

parameter error =
$$\frac{|\operatorname{param}_{\operatorname{ture}} - \operatorname{param}_{\operatorname{estimated}}|^2}{|\operatorname{param}_{\operatorname{ture}}|^2}$$
(38)

It is noteworthy that the constructed fractional flow is not predefined before inverse training begins. Instead, f(u) is dynamically constructed during each iteration using Eq. 31, with the front saturation calculated by Eq.30. The iterative refinement ensures that the PDE parameters and their corresponding constructions are updated to align with observed data.

The following subsections will delve into the outcomes of inverse training within the base scenario that focuses on a singular learnable parameter. We will scrutinize how variations in the quantity and quality of the sampling data impact PINN performance. Subsequently, we will address the complexities of learning with two parameters in the context of the Buckley-Leverett model.



Figure 15: Schematic of a rectangular 1D flow field (Modified from Noh et al., 2007).

4.2.1 Base case

For the base case, the focus is on learning the mobility ratio (M), with gravity effects momentarily set aside. We assembled a dataset consisting of 10,000 collocation points alongside 10,000 labeled data points, gathered through Latin Hypercube Sampling (LHS). This method ensures broad coverage across both time and spatial domains, aiming for thorough characterization of the system's dynamics.

The progression of loss, error, and parameter error throughout the training process of the base case is recorded in Fig. 16. We save the best model at an iteration wherein both the parameter error and solution error exhibit reductions in comparison to their preceding values. Eventually, at iteration 11416, the model precisely predicts M to be 1.000000119 (true M=1), achieving an L_2 -norm error of 1.18% and a loss of 6.17E-05. Expanding our analysis, we adjust M values



Figure 16: Inverse training of base case: evolution of loss, error, and parameter error.

for training and compiled the results in Table 2. For instance, the MM1 case yields an estimated M value of 0.9999997 (true M = 0.1) and for case MM2, the NN model estimates M to be 9.99998665 (true M = 10). The inverse PINN training undertaken in these cases consistently produces commendable results regarding parameter error, solution error, and the loss function.

To facilitate a more in-depth investigation on PINNs' performance with various sampling strategies of labeled data, six more cases are performed. The outcomes of these experiments, detailed in Table 2, are categorized into two groups for analysis.

4.2.2 Sensitivity analysis on sampling size

In scenarios with abundant input data, such as the base, MM1, and MM2 cases, PINNs demonstrate exceptional training performance. However, considering the frequent scarcity of data in scientific and engineering contexts, it's crucial to explore how PINN effectiveness fluctuates with varying sizes of labeled data. For this purpose, we experiment PINN

1	Table 2. Summary of inverse i non training Cases (One Learnable i arameter).							
Case	Labeled Data	Μ	Param Error	Error	Loss			
base	10000	1	1.42E-14	0.01182	6.17E-05			
MM1	10000	0.1	3.56E-15	0.01428	9.08E-05			
MM2	10000	10	1.78E-12	0.01846	8.66E-05			
D1	1000	1	6.96E-13	0.02306	2.23E-06			
D2	100	1	1.42E-14	0.03730	6.67E-06			
D3	10	1	3.55e-15	0.11830	4.73E-07			
C1	1% noise	1	8.88E-14	0.00997	1.41E-04			
C2	3% noise	1	1.28E-13	0.01333	9.63E-04			
C3	5% noise	1	4.30E-13	0.02476	2.74E-03			

Table 2: Summary of Inverse PINN Training Cases (One Learnable Parameter).

performance using smaller datasets in the D1, D2, and D3 cases, which utilize 1,000, 100, and 10 labeled data points, respectively. An illustration of the model's learning from 100 labeled data points is shown in Fig. 17. The results from



Figure 17: PINN solution map learnt from 100 labeled data.

these cases, as presented in Table 2, validate that our method for selecting the optimal model consistently facilitates reliable parameter estimation across various datasets. Furthermore, the loss values recorded for these PINN models remain modest.. Nonetheless, as depicted in Fig. 18, the clarity in distinguishing between high and low saturation regions diminishes with smaller sample sizes, indicating that solution prediction accuracy degrades as the number of sampling points decreases. Training with as few as 10 data points leads to a significant decline in predictive performance, with the loss rate jumping to 11.8% - an increase by an order of magnitude when compared to the base case's loss.

4.2.3 Sensitivity analysis on sampling purity

Except the impact of sampling size, we also consider data purity as a potential determinant in the performance of inverse PINN training. Given that real-world data collection frequently encounters the challenge of noise or impurities, we assess the resilience of PINNs against such imperfections by introducing Gaussian noise at varying intensities of 1% (C1 case), 3% (C2 case), and 5% (C3 case) to the pristine base case data The corresponding training outcomes detailed in Table 2.

It can be seen that PINNs successfully estimate the unknown parameter M despite slight data corruption. For better visualization, errors and losses of four cases in this group are plotted in Fig. 19. In the noise-free base case, the error and loss stand at 1.18% and 6.17E-05, respectively. The introduction of 1% noise slightly alters the results to an error of 1.00% (lower than that of the base case, attributable to sampling variability) and a loss of 1.41E-04. At a 3% noise level, the error and loss are 1.33% and 9.63E-4, respectively. The 5% noise level, the most corrupted, results in an error of 2.48% and a loss of 2.74E-03. Although the overall trend does indicate that increasing corruption impairs PINN solution accuracy, the error and loss values, even at 5% noise, remain relatively modest. The tolerance of PINN training to minor noise interference enhances their applicability in real-world scenarios where data often comes with inherent inaccuracies.



Figure 18: Analytical vs. PINN solution profiles: 2D comparison for various data sampling sizes.



Figure 19: PINN prediction accuracy relative to data purity level.

4.2.4 Two learnable parameters

While vanilla PINNs shows promising performance for one-parameter inverse problems, this subsection further examines their capability to address Buckley-Leverett model with two unknown parameters in the PDE: the mobility ratio and the gravity term. Similar to the one-parameter case, 10000 labeled data and 10000 collocation points are used as inputs.

The journey of training with two parameters proved to be considerably more complex, marked by fluctuations in losses and errors. Three individual optimizers for the model hyperparameters as well as two parameters and careful adjustment of the learning rate for each of them are required. The training duration doubles compared to the one-parameter case.

Through careful adjustments, we achieve a stable convergence with the final error and loss being 1.43% and 2.79E-6, respectively. The training process is displayed in Fig.20. PINN estimates M to be 0.999979854 (true value = 1) and the gravity term to be -1.00098896 ((true value = -1)).



Figure 20: Evolution of solution profiles during inverse PINN training for the two-parameter case.

5 Discussion and conclusion

This research leverages the state-of-the-art Physics-Informed Neural Network (PINN) techniques to tackle petroleum engineering problems, demonstrating several advantages PINNs offer over traditional approaches. For example, PINNs operate independently of labeled data for forward problems and excel in extrapolation or prediction capabilities, suppressing other machine learning methods. In addition, compared to numerical methods, the meshless characteristic of PINNs eliminates the necessity for fine grid blocks to track shock front movements, thereby avoiding the discretization errors associated with numerical simulation. However, it is essential to recognize that PINNs are not designed to replace but to augment traditional simulation methods. Currently, PINN techniques are in the early stages of development and face challenges in modeling physics in non-uniform spatial domains (e.g., reservoir heterogeneity). Future endeavors will aim to address these challenges and extend PINN applications to two-dimensional and even three-dimensional models, broadening their impact in engineering and scientific research fields. With these in mind, we conclude the key findings of this study as follows:

- 1. The success of PINN training the Buckley-Leverett model critically depends on the proper construction of a convex hull for the original flux function. Such construction imposes precise physical constraints to the solutions, eliminating the issue of multiple saturation values at a single location related to the original fractional flow. This strategy is generally applicable for addressing other nonlinear hyperbolic PDEs that exhibit discontinuities in their solutions. With the incorporation of accurate governing equations, the PINN training framework is very straightforward, utilizing lightweight neural networks, basic activation functions, and simple optimization methods, with solution accuracy guaranteed.
- 2. Vanilla PINNs efficiently solve the Buckley-Leverett equation across varying fluid mobility ratios and gravity terms without relying on labeled data, identifying that lower mobility ratios and up-inclined reservoirs are favorable for a delayed water breakthrough and thus higher oil recoveries in water-displacing-oil processes.
- 3. Vanilla PINNs demonstrate the ability to resolve not only a single saturation shock for a water-displacing-oil process, but also manage two shocks for a semi-miscible gas-displacing-water process. TThe presence of an additional discontinuity, stemming from the mutual solubility between the displacing and displaced phases, does not detract from the PINNs' effectiveness. PINNs provide a valuable tool to gauge the spread of a CO₂ plume, which is critical for the energy industry's efforts towards achieving net-zero emissions.

- 4. With the utilization of observed data, inverse PINNs are able to precisely identify the hidden parameter in the governing equation (mobility ratio only). The constraints from governing equations reduce the dependence of inverse PINNs on labeled data. Our sensitivity analysis reveals that PINNs demonstrate resilience to data impurities of up to 5% and cope well with moderate data shortages.
- 5. Inverse PINNs have the capability to identify multiple parameters within the Buckley-Leverett equation (both mobility ratio and gravity term), enabling the comprehensive mapping of the entire solution space. This is achieved through meticulous adjustments of learning rates for individual optimizers concerning model hyperparameters and two learnable parameters.

6 Abbreviations

ML Machine Learning

PIML Physics-Informed Machine Learning

CCUS Carbon Capture, Utilization, and Storage

PINN Physics-Informed Neural Network

PDE Partial Differential Equation

BL Buckley-Leverett

NN Neural Network

ANN Aritificial Neural Network

CNN Convolutional Neural Network

RNN Recurrent Neural Networks

AD Automation Differentiation

LHS Latin Hypercube Sampling

References

- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- Ulisses Braga-Neto. Fundamentals of pattern recognition and machine learning. Springer, 2020.
- Abdeldjalil Latrach, Mohamed Lamine Malki, Misael Morales, Mohamed Mehana, and Minou Rabiei. A critical review of physics-informed machine learning applications in subsurface energy systems. *arXiv preprint arXiv:2308.04457*, 2023.

Hai Wang and Shengnan Chen. Insights into the application of machine learning in reservoir engineering: Current developments and future trends. *Energies*, 16(3):1392, 2023.

Vivek Kesireddy, Georgy Kompantsev, Sheelabhadra Dey, Eduardo Gildin, Enrique Z Losoya, and Narendra Vishnumolakala. Maximizing efficiency of deep-reinforcement learning agents in autonomous directional drilling with hyperparameter optimization. In *SPE/AAPG/SEG Unconventional Resources Technology Conference*, page D031S063R004. URTEC, 2023.

- Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.
- Cedric G Fraces, Adrien Papaioannou, and Hamdi Tchelepi. Physics informed deep learning for transport in porous media. buckley leverett problem. *arXiv preprint arXiv:2001.05172*, 2020.
- Se E Buckley and MCi Leverett. Mechanism of fluid displacement in sands. *Transactions of the AIME*, 146(01): 107–116, 1942.
- Olga Fuks and Hamdi A Tchelepi. Limitations of physics informed machine learning for nonlinear two-phase transport in porous media. *Journal of Machine Learning for Modeling and Computing*, 1(1), 2020.

- Emilio Jose Rocha Coutinho, Marcelo Dall'Aqua, Levi McClenny, Ming Zhong, Ulisses Braga-Neto, and Eduardo Gildin. Physics-informed neural networks with adaptive localized artificial viscosity. *Journal of Computational Physics*, page 112265, 2023.
- Ruben Rodriguez-Torrado, Pablo Ruiz, Luis Cueto-Felgueroso, Michael Cerny Green, Tyler Friesen, Sebastien Matringe, and Julian Togelius. Physics-informed attention-based neural network for hyperbolic partial differential equations: application to the buckley–leverett problem. *Scientific reports*, 12(1):7557, 2022.
- Waleed Diab, Omar Chaabi, Wenjuan Zhang, Muhammad Arif, Shayma Alkobaisi, and Mohammed Al Kobaisi. Data-free and data-efficient physics-informed neural network approaches to solve the buckley–leverett problem. *Energies*, 15(21):7864, 2022.
- Henry J Welge. A simplified method for computing oil recovery by gas or water drive. *Journal of Petroleum Technology*, 4(04):91–98, 1952.
- Cedric G Fraces and Hamdi Tchelepi. Physics informed deep learning for flow and transport in porous media. In SPE Reservoir Simulation Conference?, page D011S006R002. SPE, 2021.
- Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.
- Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018.
- Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.
- Don W Green, G Paul Willhite, et al. *Enhanced oil recovery*, volume 6. Henry L. Doherty Memorial Fund of AIME, Society of Petroleum Engineers ..., 1998.
- Royal Harvard Brooks and Arthur T Corey. Properties of porous media affecting fluid flow. *Journal of the irrigation and drainage division*, 92(2):61–88, 1966.
- Laurence Patrick Dake. Fundamentals of reservoir engineering. Elsevier, 1983.
- Randall J LeVeque and Randall J Leveque. Numerical methods for conservation laws, volume 214. Springer, 1992.
- Isamara LN Araujo, Panters Rodríguez-Bermúdez, and Yoisell Rodríguez-Núñez. Numerical study for two-phase flow with gravity in homogeneous and piecewise-homogeneous porous media. *TEMA (São Carlos)*, 21:21–41, 2020.
- Myeong Noh, Larry Wayne Lake, Steven Lawrence Bryant, and A Araque-Martinez. Implications of coupling fractional flow and geochemistry for co2 injection in aquifers. *SPE Reservoir Evaluation & Engineering*, 10(04):406–414, 2007.
- McMillan Burton, Navanit Kumar, and Steven L Bryant. Co2 injectivity into brine aquifers: Why relative permeability matters as much as absolute permeability. *Energy Procedia*, 1(1):3091–3098, 2009.
- Ehsan Azizi and Yildiray Cinar. Approximate analytical solutions for co2 injectivity into saline formations. *SPE Reservoir Evaluation & Engineering*, 16(02):123–133, 2013.
- Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- David I Ketcheson, Randall J LeVeque, and Mauricio J Del Razo. *Riemann problems and Jupyter solutions*, volume 16. SIAM, 2020.