Hindawi
Geofluids
Volume 2022, Article ID 1781388, 10 pages
https://doi.org/10.1155/2022/1781388

Research Article

PINN-Based Method for Predicting Flow Field Distribution of the Tight Reservoir after Fracturing

Jun Pu,[1,2] **Wenfang Song**,[1,2] **Junlai Wu**,[1,2] **Feifei Gou**,[1,2] **Xia Yin**,[1,2] **and Yunqian Long**[3]

[1]*State Key Laboratory of Shale Oil and Gas Enrichment Mechanisms and Effective Development, Beijing 100083, China*
[2]*Sinopec Exploration & Production Research Institute, Beijing 100083, China*
[3]*School of Petrochemical Engineering & Environment, Zhejiang Ocean University, Zhoushan, Zhejiang 316022, China*

Correspondence should be addressed to Yunqian Long; longyunqian@163.com

Received 13 October 2021; Accepted 19 March 2022; Published 6 April 2022

Academic Editor: Jinze Xu

The physical-informed neural network (PINN) model can greatly improve the ability to fit nonlinear data with the incorporation of prior knowledge, which endows traditional neural networks with interpretability. Considering the seepage law in the tight reservoir after hydraulic fracturing, a model based on PINN and two-dimensional seepage physical equations was proposed, which can effectively predict the flow field distribution of the tight reservoir after fracturing. Firstly, the dataset was obtained based on physical and numerical models of the tight reservoirs developed by volume fracturing. Furthermore, coupling the neural networks and the two-dimensional unsteady seepage equation, a PINN model was developed to predict the flow field distribution of the tight reservoir. Finally, a systematic study was performed concerning the noise corruption levels, training iterations, and training sample size that affect the prediction results of PINN models. Besides, a comparison between PINN and traditional deep neural networks (DNN) was presented. The results show that the DNN model was not only sensitive to noisy data but also more vulnerable to overfitting as the training iterations increase. In addition, the prediction accuracy cannot be guaranteed when the samples are inadequate (<500). In contrast, the PINN model was less affected by noise and training iterations and thus indicates greater stability. Moreover, the PINN model outperforms the DNN model in the case of inadequate samples attributing to prior knowledge. This study confirms that the adopted PINN model can provide algorithmic support for the accurate prediction of flow field distribution of the tight reservoirs.

1. Introduction

Tight reservoirs have become the focus of unconventional oil and gas exploration and development. At the end of the "12th Five-Year Plan," the major oil fields have been explored for the large-scale development of tight oil reservoirs. During the "13th Five-Year Plan" period, the tight oil reservoirs will become an important substitute energy source in China [1–4]. And the hydraulic fracturing technology has been widely utilized in the development of the tight reservoirs [5]. Hydraulic fracturing is an advanced stimulation technological process accompanied by increasing lateral horizontal well drilling and the injection of fracturing fluid under high pressure [6]. However, the prediction of flow field distribution of the tight reservoir after fracturing is mostly based on the traditional simulation method, which is time-consuming and inefficient. Besides, given the anisotropy and heterogeneity of the reservoir after fracturing [7], data acquisition and characterization of rock and fluid properties are very difficult. In addition, there are many uncertainties in each step of simulation, which can greatly affect the prediction of reservoir production performance.

Currently, artificial intelligence techniques and machine learning (ML) methods play an important role in various industries which are more efficient and fast compared to the traditional method [8–10]. The ML also provides a new

sight to predict the flow field distribution of the tight reservoir. However, it is difficult to obtain sufficient data in some practical projects, resulting that machine learning cannot effectively learn the nonlinear relationships between data. In addition, most of the practical engineering is based on mathematical-physical laws, which are crucial to the machine learning models, as prior knowledge can constrain the training process of machine learning and reveal the implicit relationships between data. In contrast, machine learning models do not ensure that the fundamental physical laws relevant to practical engineering are met if they rely on data alone.

This idea of physics-constrained learning was already proposed in the late 90s to solve classic differential equations [11–13]. Using physical constraints to regularize the DNN training has also been investigated in [14–17]. Since Raissi et al. proposed physical-informed neural network (PINN) [18, 19], this model and its variations have been widely used to solve partial differential equations [20–22] and practical engineering problems. Wang et al. [23] proposed the theory-guided neural network (TgNN) model, which is a variant of PINN, and applied the model to the groundwater flow problem. Experiments show that TgNN has higher accuracy, better reliability, and scalability than the ordinary deep neural network (DNN) [24]. Jagtap et al. [25] applied PINN to the solution of nonlinear conservation laws by decomposing the study region into multiple nonoverlapping subdomains and setting an independent PINN for each subdomain to increase flexibility. This method can also improve the parallelism of the model, thus effectively reducing the training cost. Karimpouli and Tahmasebi [26] applied PINN to a one-dimensional time-varying seismic wave equation solution problem. The results showed that the method is more accurate for inversion of velocity and density. Mao et al. [27] used PINN to approximate the Euler equation for high-speed aerodynamic flow and solved the positive and inverse problems for one- and two-dimensional regions. He et al. [28] applied PINN to the parameter and state estimation of subsurface transport problems. The results indicated that the method has high prediction accuracy and can invert the required parameters more accurately. Rao et al. [29] applied PINN to the numerical simulation of steady-state and transient laminar flow at a low Reynolds number. It was shown that this method had great potential for high precision fluid flow simulation. Goswami et al. [30] used PINN to solve the brittle fracture problem. Meng et al. [31] decomposed a long-term problem into many independent short-term problems while using multiple PINN models for prediction. The experiments demonstrated that the method can significantly improve the speed of neural network. Sun et al. [32] conducted numerical experiments using PINN for several internal flows related to hemodynamic applications, and the results showed that the method was in good agreement with the numerical simulation results. Kissas et al. [33] employed PINN to predict the arterial blood pressure based on the MRI data and the conservation laws.

This paper focuses on the mechanism and application of the PINN model in predicting the flow field distribution of the tight reservoir after fracturing. The dataset was firstly obtained based on physical and numerical models of the tight reservoirs developed by volume fracturing. Then, the neural networks and the two-dimensional unsteady seepage equation are coupled. In addition, a PINN model was developed to predict the flow field distribution of the tight reservoir. Finally, a systematic study was performed to demonstrate the effect of the noise corruption levels, training iterations, and training sample size. Besides, a comparison between PINN and traditional DNN was presented.

2. PINN Model

The PINN model is built based on the DNN model, and the prior information is encoded into the DNN model by modifying the loss function, which can constrain the training process of the model and finally achieve a reasonable and accurate prediction result.

The workflow of the PINN model is shown in Figure 1.

Taking the general nonlinear partial differential equation as an example, assume the partial differential equation is represented as follows:

$$u_t + N[u, \lambda] = 0, x \in \Omega, t \in [0, t_T], \tag{1}$$

where u is the solution of the partial differential equation, $N[u, \lambda]$ is the nonlinear operator with parameters $\in$, Ω is a subset of R^D, and t_T is the ending point.

First, build a deep neural network model as the base model and then modify the loss function. The loss function is defined in two parts: one part of the error derived from the true and predicted values, which is calculated by mean square error (MSE). The MSE formula is expressed as follows:

$$\text{MSE} = \frac{\sum_{i=1}^{N} (y_i - y_i)^2}{N}, \tag{2}$$

where y_i is the predicted values, y_i is the true values, and N is the number of sample.

The other part of the error is derived from the residuals generated by the partial differential equation. The partial differential equation is given by the following:

$$f(x, t; \theta, \lambda) = \frac{\partial}{\partial t} n(x, t; \theta) + N[n(x, t; \theta), \lambda]. \tag{3}$$

There are four methods to calculate derivatives in the process of training neural networks: (1) manual differentiation (MD), (2) numerical differentiation (ND), (3) symbolic differentiation (SD), and (4) automatic differentiation (AD) [34–37].

The MD algorithm means solving the gradient formula manually and then putting the actual values into the formula and calculating them. The ND algorithm approximates the solution results by finite difference methods. The SD algorithm refers to converting the problem into a specialized mathematical symbolic problem and then implementing the differentiation using algebraic software. These three

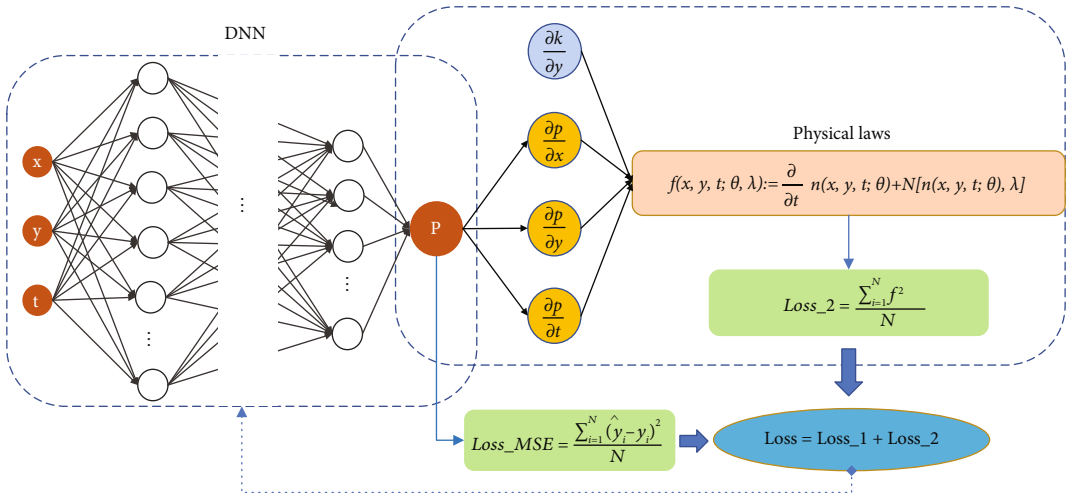


Figure 1: Framework of the PINN model.

methods are easy to encode and suitable for simple functions. For complex functions, these three methods are time-consuming and have poor application performance. Therefore, the AD algorithm was proposed. The AD algorithm first decomposes the complex functions into a series of elementary operations. The symbolic differentiation is then applied to atomic operations such as constants, exponential functions, logarithmic functions, and trigonometric functions using computational graphs. The partial derivatives of the function are calculated at each successive node and then calculated according to the chain rule. In this way, the derivative value of any complex function can be calculated. It can be seen that the AD algorithm has the advantages of accurate calculation, high applicability, and flexibility.

3. Application of PINN on the Fractured Flow Field of Tight Oil

3.1. Physical Model. Unconventional reservoirs have low porosity and permeability due to the heterogeneity of the formation [38, 39], which can be increased by volume fracturing, thus increasing the production of unconventional reservoirs.

In this paper, the fractured reservoir is simplified into three parts: the matrix zone, the transition zone, and the fractured zone. It is assumed that the studied reservoir is horizontal, homogeneous, and isotropic. The fluid is assumed to be a single-phase, homogeneous, and weakly compressible Newtonian fluid. It is also assumed that no special physical or chemical phenomena occur during the seepage process.

The physical model [40] is shown in Figure 2.

The basic differential equation for two-dimensional unsteady seepage flow is calculated as follows:

$$\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2} = \frac{1}{\eta}\frac{\partial p}{\partial t}, \tag{4}$$

where $\eta = K/\phi\mu C_t$, η (Pa · s) is the hydrodynamic viscosity, $K(\mathrm{m}^2)$ is the absolute permeability, ϕ is the porosity at a certain pressure, and $C_t(1/\mathrm{Pa})$ is the rock compression coefficient.

Define the residual formation of the governing equation as follows:

$$f := \frac{1}{\eta}\frac{\partial p}{\partial t} - \left(\frac{\partial^2 p}{\partial x^2} + \frac{\partial^2 p}{\partial y^2}\right). \tag{5}$$

Assuming the absolute permeability K as a function about y, the residual function can be written as follows:

$$f := \frac{\phi\mu C_t}{K(y)}\frac{\partial p(t,x,y)}{\partial t} - \frac{\partial^2 p(t,x,y)}{\partial x^2} - \frac{\partial^2 p(t,x,y)}{\partial y^2}. \tag{6}$$

The residual function of the governing equation is then defined as follows:

$$\mathrm{Loss\_2} := \frac{\sum_{i=1}^{N} f^2}{N}. \tag{7}$$

Finally, the loss function of the PINN model is defined as follows:

$$\mathrm{Loss} = \mathrm{Loss\_1} + \mathrm{Loss\_2}. \tag{8}$$

Assuming that the distribution from the volume fractured zone to the matrix zone is in accordance with the S-shaped curve, Equation (9) is used to characterize the permeability distribution, and it can be obtained by the following steps: first, we plot the S-shaped curve and set the endpoint values which, respectively, indicate the equivalent permeability of the modified area and the matrix permeability at both ends of the curve, and then, we use curve fitting

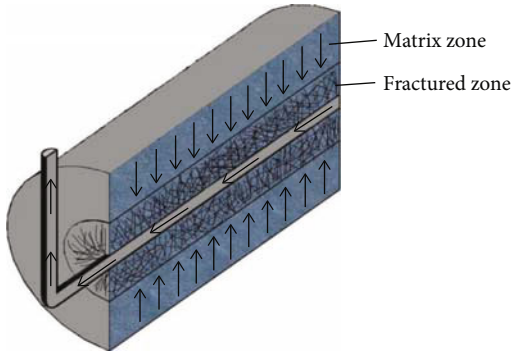


Figure 2: Division of flow field around fractured horizontal well in tight oil.

methods to get the fitting formula of S-shaped curve.

$$y = \left(-\frac{a/(b+e^{cx})}{d} + d\right) + \left|\min\left(-\frac{a/(b+e^{-cx})}{d} + d\right)\right|, \quad (9)$$

where a, b, c, and d are the S-curve fitting coefficients, which can be assigned according to the actual situation.

In this paper, a is $1.982 \cdot 10^{-5}$, b is $1.941 \cdot 10^{-7}$, c is 0.08466, and d is 10. Consider the studied zone of the horizontal well as a cylinder, x is the radius of the circle of the cylinder cross-section, and y is the absolute permeability.

The permeability of the flow field around a fractured horizontal well in tight oil is shown in Figure 3 where $K1$ denotes the permeability of the main fractured zone, $K2$ means the permeability of the transition zone, and $K3$ is the permeability of the matrix zone.

Assuming that the motion of the fluid follows the linear Darcy flow, the process can be described as follows:

$$\frac{\partial}{\partial t}(\phi\rho) + \nabla \cdot (\rho u) = Q_m, \quad (10)$$

$$u = -\frac{K}{\mu}\nabla p, \quad (11)$$

where ϕ is the porosity, ρ is the fluid density (kg/m^3), Q_m is the mass of the fluid, u is the velocity, K denotes the absolute permeability (m^2), μ is the dynamic viscosity of the fluid (Pa·s), and p is the pressure (Pa).

3.2. Data Preparation. The numerical model is designed based on the physical model. And the porous media and the underground water flow module in COMSOL are used for a two-dimensional transient study to simulate the development of horizontal wells fractured in infinite discrete fractured reservoirs. Specifically, a two-dimensional axial-symmetric model is used. The horizontal coordinate indicates the horizontal well length with a value of 1500 m, and the vertical coordinate indicates the radius of the fractured zone and matrix zone with value of 150 m and 50 m, respectively. The horizontal axis is the symmetry axis of the constructed physical model.

The numerical model is shown in Figure 4.

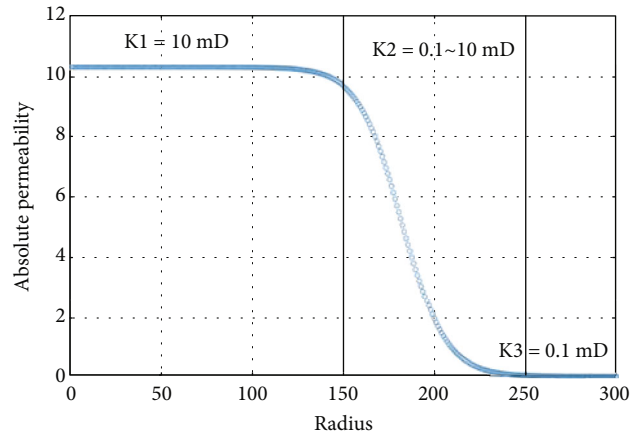


Figure 3: Permeability of the flow field around a fractured horizontal well in tight oil.

Because of the poor fluid flow capacity of the reservoir, the outer boundary condition is considered to be a no-flow boundary.

The boundary conditions and initial conditions and specific parameters are presented in Table 1.

Considering the 50-day variation of the flow field, every 0.5 days is considered as a time step; thus, the time steps are a total of 101. The simulation data are generated by calculating time t, position coordinates x and y, and flow field distribution value p. Time t is $101 * 1$ dimensions; position coordinates x and y values are 255×1 dimensions; flow field pressure value p is generated at different time steps and different coordinate values; thus, the dimension is 255×101.

The generated simulation data are used as the dataset for machine learning. It can be seen that the dimensions of t, x, y, and p are different. In order to make them consistent, the time (t) is expanded to 255×101 dimensions by row, which means that there are 101 time points under each different location coordinate; the location coordinates (x, y) are also expanded to 255×101 dimensions by column, which means that there are 255 sets of coordinate values under each different time step. Thus, the dimensions of t, x, y, and p are all 255×101 and can be used as the dataset for the PINN model.

3.3. Model Setup. In this paper, there are 3 input parameters for the PINN model, which are time (t), spatial coordinates (x and y), and the output parameter which is the flow field distribution (the pressure value, p). There are all 25755 samples.

In machine learning, the widely used activation functions are the sigmoid function, the tanh function, and the ReLU function. The tanh function converges faster than the sigmoid function and can effectively prevent the model from the fluctuation of gradient descent. The ReLU function is prone to cause the cases where weights cannot be updated during training process while tanh function cannot cause this phenomenon.

As a result, the tanh function is chosen as the activation function of the neural network, and it can be described as

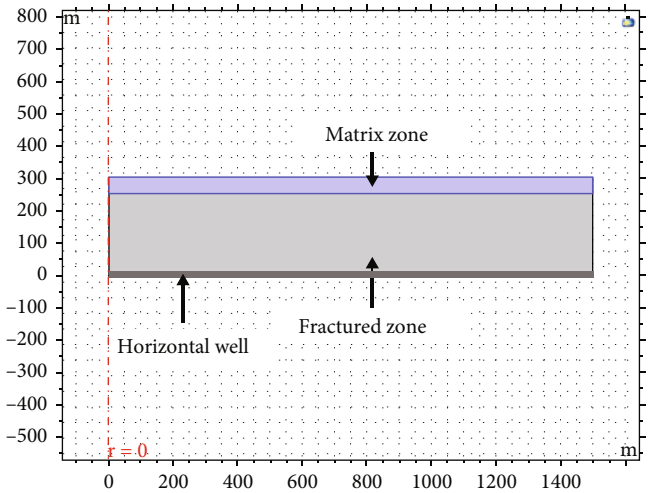


Figure 4: Numerical model of the flow field around a fractured horizontal well in tight oil.

Table 1: Specific parameters of the numerical model.

Parameters	Value
Horizontal well length	1500 m
Radius of fractured zone	250 m
Radius of matrix zone	50 m
Petroleum density	860 kg/m^3
Dynamic viscosity	$1.27 \cdot 10^{-3}$Pa · s
Initial reservoir pressure	25 MPa
Bottomhole flow pressure	15 MPa
Initial porosity	0.1
Rock compression coefficient	$8 \cdot 10^{-4}$MPa^{-1}
Time range	50 days
Absolute permeability of the matrix zone	0.1 mD
Absolute permeability of the fractured zone	10 mD

follows:

$$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \tag{12}$$

Adam's algorithm [41] is used as the optimization algorithm for gradient descent. Adam's algorithm is an optimization of the gradient descent algorithm and Adagrad algorithm. The learning rate of each parameter can be dynamically adjusted using the first-order moment estimation and second-order moment estimation of the gradient. It makes the learning rate has a determined range for each iteration and thus is suitable for most nonconvex optimization problems [42, 43]. The specific operations are given by the following:

$$m_i = \mu * m_{i-1} + (1 - \mu) * g_t, \tag{13}$$

$$n_i = \nu * n_{i-1} + (1 - \nu) * g_t^2, \tag{14}$$

Table 2: Specific setting of the PINN model.

Hyperparameters	Value
Hidden layers	8
Neurons in each hidden layer	20
Neurons in input layer	3
Neurons in input layer	1
Activation function	tanh function
Optimization algorithm	Adam

$$m_i = \frac{m_i}{1 - \mu^i}, \tag{15}$$

$$n_i = \frac{n_i}{1 - \nu^i}, \tag{16}$$

$$\Delta\theta_i = -\frac{m_i}{\sqrt{n_i} + \varepsilon} \cdot \eta, \tag{17}$$

where m_i and n_i are the first-order moment estimates and second-order moment estimates of the gradient, respectively, m_i and n_i are the corrections to m_i and n_i, approximated as unbiased estimates of the expectation, μ is the momentum factor, g_t is the gradient, ν is the decayed learning rate, i is the timestep, $\Delta\theta_i$ is the variation of parameters, ε is a constant approximating zero, avoiding the denominator to be zero, and η is the step.

Therefore, the setting of hyperparameters of the PINN model is presented in Table 2.

4. Model Validation and Analysis of Influencing Factors

4.1. Model Validation. In this section, the performance of the modified PINN model is tested. The accuracy of the PINN model is compared with the simulation and the DNN model in Figure 5.

It can be seen from Figure 5 that the predicted results of the PINN model are greatly consistent with the simulated values. Besides, compared to the baseline model (DNN model), the PINN model has higher accuracy.

In addition, it is demonstrated from Figure 6 that both the DNN model and PINN model can converge quickly. However, by comparison, the PINN model outperforms the DNN model when the epoch is small.

As a result, the PINN model has a marvelous performance in the prediction of the flow field distribution of the tight reservoir after fracturing effectively and converges more quickly and can be evaluated further in the following studies.

4.2. The Effect of Noise Corruption Levels. In this paper, we measure the predictive ability of the PINN model by the decision coefficient R^2.

R^2 is calculated by the formula as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - y_i)^2}{\sum_{i=1}^{N}(\overline{y_i} - y_i)^2}, \tag{18}$$

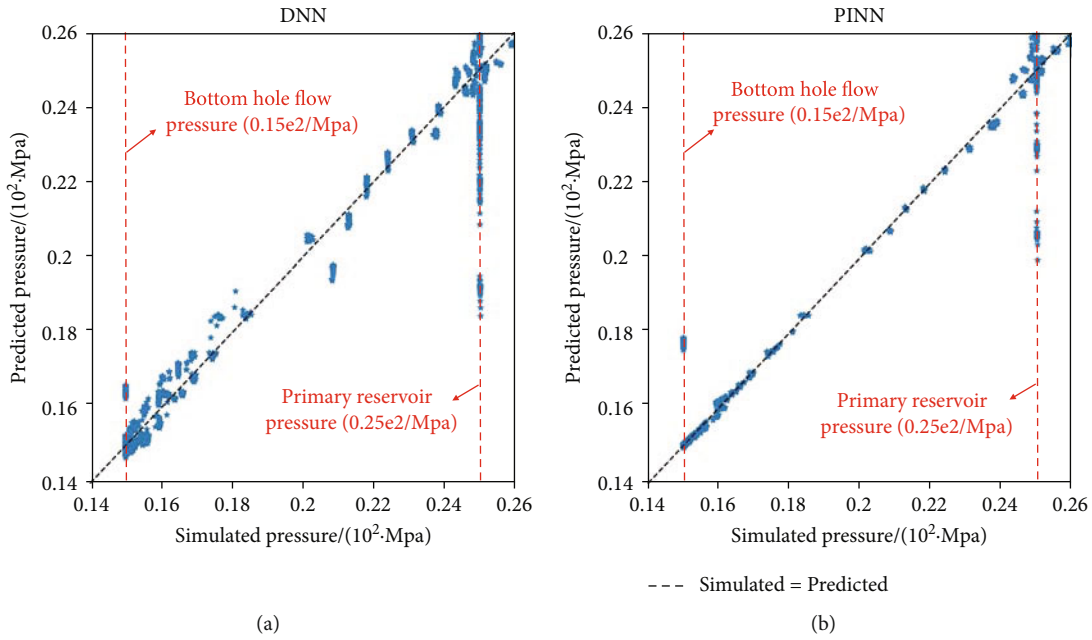


Figure 5: Cross-plot of the DNN and PINN models.

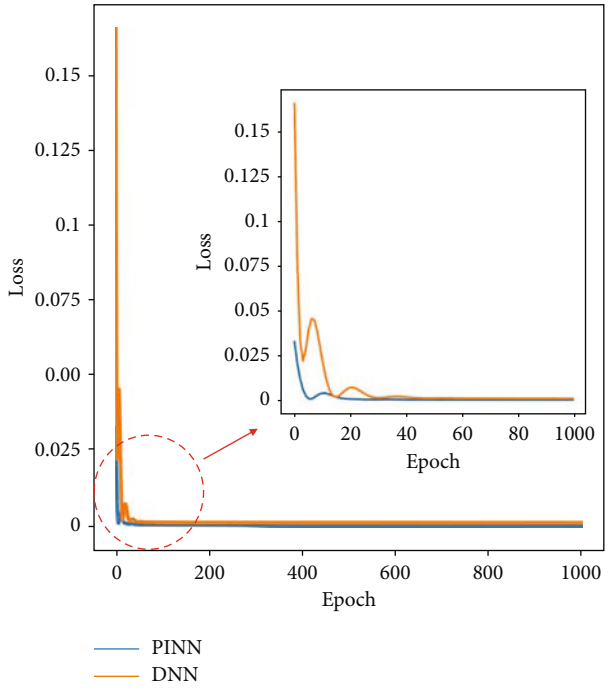


Figure 6: Convergence of the DNN and PINN models.

where y_i is the true value, y_i is the predicted value, and $\overline{y_i}$ is the mean value.

In this section, we measure the predictive ability of the PINN model; at the same time, the accuracy of the predictions of the DNN model and the PINN model is compared.

The data obtained in practical engineering are usually the data with noise, and the prediction accuracy of the PINN varies under different levels of noise corruption. Therefore, it is necessary to evaluate the effect of the noise corruption levels.

The formula to add noise is represented by the following:

$$P(t,x,y) = P(t,x,y) + a \times \text{std}(P) \times \varepsilon, \tag{19}$$

where $P(t,x,y)$ is the pressure value at the coordinates (x,y) at the moment t. a is the noise level. std(P) is the standard deviation of the pressure value. ε is a random number.

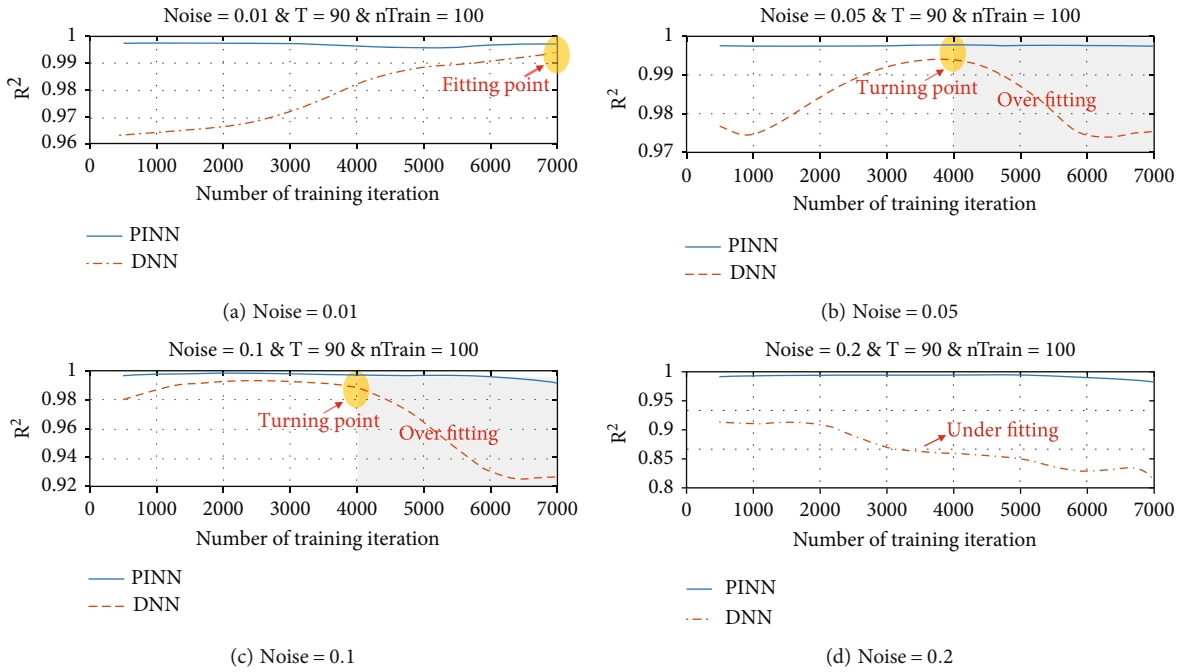
The noise corruption levels are set to 0.01, 0.05, 0.1, and 0.2. And then, the variation of the model prediction ability of PINN with the training iterations under these four levels of noise corruption is analyzed.

The results are shown in Figure 7 where "nTrain" denotes the training samples and "nIter" denotes the training iterations.

Figure 7 represents the accuracy of the DNN and the PINN models when the sample size is 100, T is 90, and the noise levels are 0.01, 0.05, 0.1, and 0.2, respectively.

As can be seen from Figure 7(a), when the noise level is 0.01, the prediction accuracy of the DNN model increases with the increase of the training iterations. When the training iteration is below 1000, the prediction accuracy of DNN can only reach 96.2%, while when the training number is increased to 7000, the prediction accuracy of the DNN model increases to more than 99%. At this time, the model has a high precision which is considered to have reached the best-fitting point. In contrast, the PINN model is less affected with the noise level of 0.01 and continually keeps to be a high accuracy.

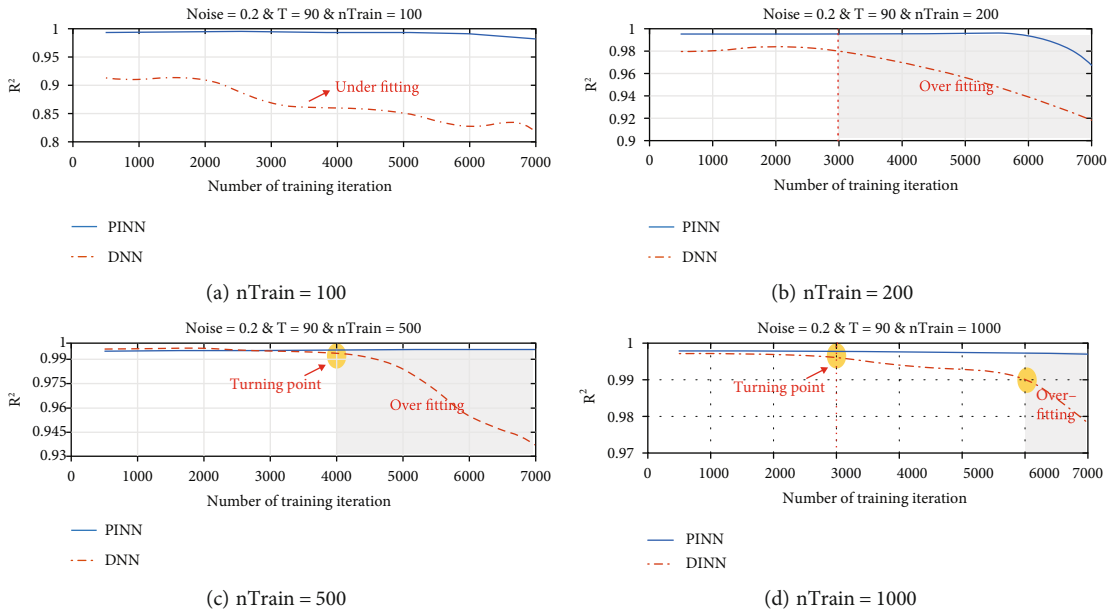
Figures 7(b) and 7(c) show the prediction results of the DNN and PINN models when the noise is 0.05 and 0.1, respectively. It can be seen that the prediction accuracy of the DNN model increases with the training iteration increases when the training iterations are below 4000. However, once the training iterations exceed 4000, the accuracy decreases sharply. Thus, the point of 4000 is called the



(a) Noise = 0.01 (b) Noise = 0.05

(c) Noise = 0.1 (d) Noise = 0.2

Figure 7: The effect of noise corruption levels.



(a) nTrain = 100 (b) nTrain = 200

(c) nTrain = 500 (d) nTrain = 1000

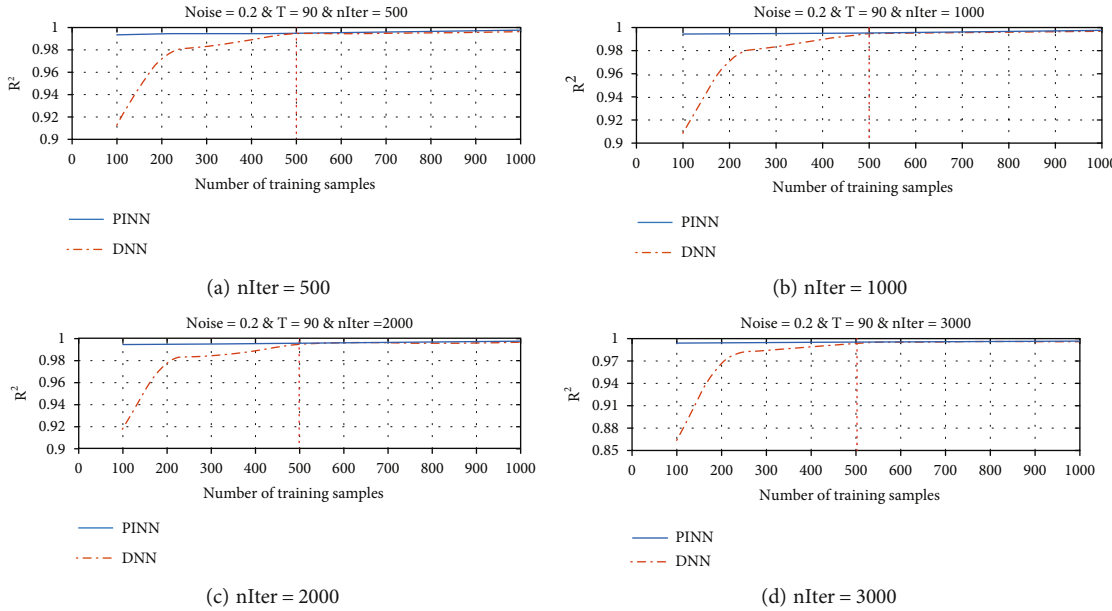
Figure 8: The effect of training iterations.

turning point. This phenomenon is due to the overfitting caused by the small amount of training data and too many training iterations. For PINN, the model has been at a high prediction accuracy and is more robust, which once again indicates that the noise data has less influence on the prediction ability of the PINN model again.

Figure 7(d) shows the effect of the noise level of 0.2. It should be noted that the accuracy of the DNN model has been decreasing and is below 90% which is called underfitting. In contrast, the PINN model can always maintain high prediction accuracy (>95%).

In general, compared with the traditional DNN model, the PINN model is not sensitive to noisy data and yields remarkable accuracy, indicating more robustness.

4.3. The Effect of Training Iterations. In addition, different neural network models have their optimal hyperparameter settings. In this way, the prediction accuracy of the models under different settings needs to be considered. The hyperparameters that affect the accuracy of the PINN model are mainly the training iterations and the sample size.



(a) nIter = 500

(b) nIter = 1000

(c) nIter = 2000

(d) nIter = 3000

Figure 9: The effect of sample size.

In this paper, we evaluate the prediction accuracy of the DNN model and the PINN model with different training iterations when the noise level is 0.2 and T is 90.

The result is shown in Figure 8 where "nTrain" denotes the sample size and "nIter" denotes the training iterations.

Figure 8 presents the prediction results with the training iterations for nTrain = 100, nTrain = 200, nTrain = 500, and nTrain = 1000, respectively.

As can be seen from Figure 8(a), when the training sample is 100, the prediction accuracy of the DNN model is mostly below 90%, which is underfitting. While the accuracy of the PINN model is about 99%, which has a better prediction ability.

If the number of samples is 200 (Figure 8(b)), the R^2 of the DNN model can achieve about 98% when the training iteration is less than 3000, but subsequently decreases and overfitting, while the R^2 of the PINN model is always about 99% until the training iterations are more than 6000.

As can be seen from Figure 8(c), when nTrain = 500 and the training iterations are less than 4000, the difference between the DNN model and the PINN model is not much, both are above 99%. After passing the turning point (nIter = 4000), the PINN model maintains high accuracy, while the DNN model shows an overfitting phenomenon.

When the training sample size is 1000 (Figure 8(d)), the turning point advanced to nIter = 3000. Before the turning point is reached, the accuracies of the DNN and PINN models are comparable and both are above 99%. Nevertheless, once passing the turning point, the accuracy of the DNN model starts to fall off and eventually overfitting.

In general, the DNN model is more vulnerable to overfitting as the training iteration increases, whereas the PINN model is less affected and demonstrated greater stability.

4.4. The Effect of Sample Size. In this paper, we investigate the predictive ability of DNN and PINN models with different sample sizes when the noise level is 20% and T is 90.

The results are displayed in Figure 9 where "nTrain" denotes the number of training samples and "nIter" denotes the training iterations.

Figure 9 represents the prediction results with the number of training samples at nIter = 500, nIter = 1000, nIter = 2000, and nIter = 3000, respectively.

The general trend shows increased prediction accuracy of the DNN model as the number of samples increased. When the number of training samples is greater than 500, the prediction accuracies of the DNN and PINN models do not differ much, both are above 99%. However, the PINN model outperforms the DNN model in the case of inadequate samples (nTrain < 500).

Commonly, the practical data are not easy to acquire resulting in an insufficient sample. As a result, the PINN model can be an effective alternative.

5. Conclusion

The physical-informed neural network (PINN) model can greatly improve the ability to fit nonlinear data and make the traditional neural networks interpretable. In this paper, the partial differential equation characterizing the seepage law of tight oil is added to the loss function of the PINN model. In addition, the PINN model is successfully utilized to approximate the high-dimensional complex partial differential equation by applying the automatic differentiation method, which improved the prediction efficiency and accuracy of the model.

By comparison, it is found that the DNN model is more sensitive to noisy data than the PINN model. When the noise corruption level is less than 0.1, the prediction accuracy of both models is above 95%. But as the number of

training iterations increases, the DNN model is more prone to overfitting, while the PINN model has a certain capability to avoid. When the noise corruption is up to 0.2, the DNN model has poor prediction accuracy while the PINN model yields remarkable accuracy (over 95%). It is indicated that the PINN model is less affected by noise and training iterations. Moreover, when the sample size is small (<500), the DNN model cannot achieve great accuracy, while the PINN model can fully learn the relationship among data and attain high accuracy. The comparison results show that the PINN model surpasses the traditional DNN model.

However, due to the poor generalization capability, the use of the PINN model is limited.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

We gratefully acknowledge the Natural Science Foundation of Zhejiang Province (LY19A020004) and the Zhoushan Science and Technology Project (2019C21028).

References

[1] S. Wang, F. Lai, K. Wang, Z. Li, and H. Wang, "Development favorable area and productivity potential evaluation method of a tight oil reservoir," *Geofluids*, vol. 2021, 14 pages, 2021.

[2] G. G. Wu, H. Fang, and Z. Han, "Growth features of measured oil initially in place & gas initially in place during the 12th five-year plan and its outlook for the 13th five-year plan in China," *Acta Petrolei Sinica*, vol. 37, no. 9, pp. 1145–1151, 2016.

[3] Y. Luo, Z. Yang, Z. Tang, S. Zhou, J. Wu, and Q. Xiao, "Longitudinal reservoir evaluation technique for tight oil reservoirs," *Geofluids*, vol. 2019, 8 pages, 2019.

[4] H. Chu, X. Liao, Z. Chen, and W. J. John Lee, "Rate-transient analysis of a constant-bottomhole-pressure multihorizontal well pad with a semianalytical single-phase method," *SPE Journal.*, vol. 25, no. 6, pp. 3280–3299, 2020.

[5] G. Hui, S. Chen, Y. Wang, and F. Gu, "The effect of hydraulic-natural fracture networks on the waterflooding development in a multilayer tight reservoir: case study," *Geofluids*, vol. 2021, 15 pages, 2021.

[6] J. Chen, M. H. Al-Wadei, R. C. M. Kennedy, and P. D. Terry, "Hydraulic fracturing: paving the way for a sustainable future?," *Journal of Environmental and Public Health*, vol. 2014, 10 pages, 2014.

[7] Z. Q. Fan, Z.-H. Jin, and S. E. Johnson, "Modelling petroleum migration through microcrack propagation in transversely isotropic source rocks," *Geophysical Journal International*, vol. 190, no. 1, pp. 179–187, 2012.

[8] H. Song, S. Du, R. Wang, J. Wang, Y. Wang, and C. Wei, "Potential for vertical heterogeneity prediction in reservoir basing on machine learning methods," *Geofluids*, vol. 2020, 12 pages, 2020.

[9] S. Du, R. Wang, C. Wei et al., "The connectivity evaluation among wells in reservoir utilizing machine learning methods," *IEEE Access*, vol. 8, pp. 47209–47219, 2020.

[10] Q. Zhang, C. Wei, Y. Wang, S. Du, Y. Zhou, and H. Song, "Potential for prediction of water saturation distribution in reservoirs utilizing machine learning methods," *Energies*, vol. 12, no. 19, p. 3597, 2019.

[11] H. Lee and I. S. Kang, "Neural algorithm for solving differential equations," *Journal of Computational Physics*, vol. 91, no. 1, pp. 110–131, 1990.

[12] I. E. Lagaris, A. Likas, and D. I. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Transactions on Neural Networks*, vol. 9, no. 5, pp. 987–1000, 1998.

[13] I. E. Lagaris, A. C. Likas, and D. G. Papageorgiou, "Neural-network methods for boundary value problems with irregular boundaries," *IEEE Transactions on Neural Networks*, vol. 11, no. 5, pp. 1041–1049, 2000.

[14] R. Sharma, A. B. Farimani, J. Gomes, P. Eastman, and V. Pande, "Weakly-supervised deep learning of heat transport via physics informed loss," arXiv preprint arXiv:1807.11374.

[15] M. A. Nabian and H. Meidani, "Physics-informed regularization of deep neural networks," arXiv preprint arXiv:1810.05547.

[16] K. Xu and E. Darve, "The neural network approach to inverse problems in differential equations," arXiv preprint arXiv:1901.07758.

[17] J. R. Holland, J. D. Baeder, and K. Duraisamy, "Towards integrated field inversion and machine learning with embedded neural networks for RANS modeling," in *in: AIAA Scitech 2019 Forum*, San Diego, California, 2019.

[18] M. Raissi, P. Perdikaris, and G. 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*, vol. 378, no. 378, pp. 686–707, 2019.

[19] M. Raissi, Z. Wang, M. S. Triantafyllou, and G. E. Karniadakis, "Deep learning of vortex-induced vibrations," *Journal of Fluid Mechanics*, vol. 861, pp. 119–137, 2019.

[20] S. Karumuri, R. Tripathy, I. Bilionis, and J. Panchal, "Simulator-free solution of high-dimensional stochastic elliptic partial differential equations using deep neural networks," *Journal of Computational Physics*, vol. 404, p. 109120, 2020.

[21] J. Siepmann, Y. Sun, and F. DeJaco, "Deep neural network learning of complex binary sorption equilibria from molecular simulation data," *Chemical Science*, vol. 10, no. 16, pp. 4377–4388, 2019.

[22] J. Berg and K. Nyström, "Data-driven discovery of PDEs in complex datasets," *Journal of Computational Physics*, vol. 384, pp. 239–252, 2019.

[23] N. Wang, D. Zhang, H. Chang, and H. Li, "Deep learning of subsurface flow via theory-guided neural network," *Journal of Hydrology*, vol. 584, article 124700, 2020.

[24] F. Scarselli and A. C. Tsoi, "Universal approximation using feedforward neural networks: a survey of some existing methods, and some new results," *Neural Networks*, vol. 11, no. 1, pp. 15–37, 1998.

[25] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis, "Conservative physics-informed neural networks on discrete domains for conservation laws: applications to forward and inverse

problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 365, article 113028, 2020.

[26] S. Karimpouli and P. Tahmasebi, "Physics informed machine learning: seismic wave equation," *Geoscience Frontiers*, vol. 11, no. 6, pp. 1993–2001, 2020.

[27] Z. Mao, A. D. Jagtap, and G. E. Karniadakis, "Physics-informed neural networks for high-speed flows," *Computer Methods in Applied Mechanics and Engineering*, vol. 360, article 112789, 2020.

[28] Q. He, D. Barajas-Solano, G. Tartakovsky, and A. M. Tartakovsky, "Physics-informed neural networks for multiphysics data assimilation with application to subsurface transport," *Advances in Water Resources*, vol. 141, article 103610, 2020.

[29] C. Rao, H. Sun, and Y. Liu, "Physics-informed deep learning for incompressible laminar flows," *Theoretical & Applied Mechanics Letters*, vol. 10, no. 3, pp. 207–212, 2020.

[30] S. Goswami, C. Anitescu, S. Chakraborty, and T. Rabczuk, "Transfer learning enhanced physics informed neural network for phase-field modeling of fracture," *Theoretical and Applied Fracture Mechanics*, vol. 106, article 102447, 2020.

[31] X. Meng, D. Zhang, and G. Karniadakis, "PPINN: parareal physics-informed neural network for time-dependent PDEs," *Computer Methods in Applied Mechanics and Engineering*, vol. 370, article 113250, 2020.

[32] L. Sun, H. Gao, S. Pan, and J.-X. Wang, "Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data," *Computer Methods in Applied Mechanics and Engineering*, vol. 361, article 112732, 2020.

[33] G. Kissas, Y. Yang, E. Hwuang, W. R. Witschey, J. A. Detre, and P. Perdikaris, "Machine learning in cardiovascular flows modeling: predicting arterial blood pressure from non-invasive 4D flow MRI data using physics-informed neural networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 358, article 112623, 2020.

[34] B. Merriënboer, O. Breuleux, and A. Bergeron, "Automatic differentiation in ML: where we are and where we should be going," in *32nd Conference on Neural Information Processing Systems (NIPS 2018)*, Montréal, Canada, 2018.

[35] C. C. Margossian, "A review of automatic differentiation and its efficient implementation," *WIREs Data Mining Knowledge Discovery.*, vol. 9, no. 4, article e1305, 2019.

[36] P. Adam, G. Sam, and C. Soumith, "Automatic differentiation in PyTorch," in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA, USA, 2017.

[37] A. Baydin, "Automatic differentiation in machine learning: a survey," *Journal of Machine Learning Research*, vol. 18, pp. 1–43, 2018.

[38] J. Wang, H. Song, W. Zhu, Y. Wang, and J. Killough, "Flow characteristics and a permeability model in nanoporous media with solid-liquid interfacial effects," *Interpretation*, vol. 5, no. 1, p. SB1-SB8, 2017.

[39] J. Wang, H. Song, and Y. Wang, "Investigation on the micro-flow mechanism of enhanced oil recovery by low-salinity water flooding in carbonate reservoir," *Fuel*, vol. 266, article 117156, 2020.

[40] H. Song and Z. Li, "A novel approach for modeling gas flow behaviors in unconventional reservoirs with nanoporous media," in *International Petroleum Technology Conference (IPTC, 2015)*, Doha, Qatar, 2015.

[41] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," arXiv preprint arXiv: 1412.6980.

[42] S. Ruder, "An overview of gradient descent optimization algorithms," 2017, arXiv:1609.04747v2 cs.LG.

[43] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.