Petroleum Science 19 (2022) 264–278

Contents lists available at ScienceDirect

Petroleum Science

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

Original Paper

A deep reinforcement learning (DRL) based approach for well-testing interpretation to evaluate reservoir parameters

Peng Dong [a], Zhi-Ming Chen [a, b, *], Xin-Wei Liao [a], Wei Yu [b]

[a] State Key Laboratory of Petroleum Resources and Prospecting, China University of Petroleum Beijing (CUP), Beijing, 102249, China
[b] University of Texas at Austin, Austin, 78731, Texas, USA

ARTICLE INFO

Article history:
Received 6 April 2021
Accepted 26 September 2021
Available online 1 October 2021

Edited by Yan-Hua Sun

Keywords:
Well testing
Deep reinforcement learning
Automatic interpretation
Parameter evaluation

ABSTRACT

Parameter inversions in oil/gas reservoirs based on well test interpretations are of great significance in oil/gas industry. Automatic well test interpretations based on artificial intelligence are the most promising to solve the problem of non-unique solution. In this work, a new deep reinforcement learning (DRL) based approach is proposed for automatic curve matching for well test interpretation, by using the double deep Q-network (DDQN). The DDQN algorithms are applied to train agents for automatic parameter tuning in three conventional well-testing models. In addition, to alleviate the dimensional disaster problem of parameter space, an asynchronous parameter adjustment strategy is used to train the agent. Finally, field applications are carried out by using the new DRL approaches. Results show that step number required for the DDQN to complete the curve matching is the least among, when comparing the naive deep Q-network (naive DQN) and deep Q-network (DQN). We also show that DDQN can improve the robustness of curve matching in comparison with supervised machine learning algorithms. Using DDQN algorithm to perform 100 curve matching tests on three traditional well test models, the results show that the mean relative error of the parameters is 7.58% for the homogeneous model, 10.66% for the radial composite model, and 12.79% for the dual porosity model. In the actual field application, it is found that a good curve fitting can be obtained with only 30 steps of parameter adjustment.

1. Introduction

Well test analysis plays an important role in understanding the characteristics of wellbore, reservoir and boundary, especially in the discovery and development of some important oil and gas fields (Yao and Ge, 2011; Chen et al., 2018; Mohammed et al., 2020). One of the most important means of well test interpretation is the type curve match. In the early days, curve matching was performed based on the chart (Earlougher, 1977; Horne, 1995). The parameters are obtained by moving the measured curve to find the most consistent theoretical curve on the chart. Since the number of curves on the chart is limited, this method will cause large errors. At present, manual parameter adjustment assisted by computer software become the primary method for curve matching (Bourdet, 2002). However, the human bias, non-unique solutions, and low efficiency caused by artificial interpretation cannot adapt to the rapid development of oil and gas fields (AlMaraghi and El-Banbi, 2015).

In recent decades, with the improvement of calculation technology and test instrument precision, many optimization algorithms have been gradually applied to the automatic matching of well test curve. Among them, nonlinear regression is the classical method, and the least square method represented by Levenberg-Marquardt method is the most common solution method (Nanba and Horne, 1992; Dastan, 2010; Dastan and Horne, 2011). However, the method is greatly affected by the initial value, and the parameters obtained are locally optimal. Therefore, some global optimization algorithms, such as genetic algorithm and particle swarm optimization algorithm, are also applied to the automatic matching of well test curve (Guyaguler et al., 2001; Gomez et al., 2014; Awotunde, 2015). But the global optimization algorithm has the problem of low efficiency.

With the great progress made in computer science in recent years, artificial intelligence (AI) algorithms have been widely used in the oil and gas industry (Zhu et al., 2019; Liu et al., 2020; Gao

* Corresponding author.
E-mail addresses: zhimingchn@cup.edu.cn, zhimingchn@utexas.edu (Z.-M. Chen).

https://doi.org/10.1016/j.petsci.2021.09.046

et al., 2021; Huang et al., 2021). Meanwhile, AI algorithms are also tried for automatic well test interpretation. Al-Kaabi and Lee (1990) firstly deployed artificial neural network (ANN) in well test model recognition. They used a number of independent ANN to calculate the probability that the curve fell into each type of well test model. Adibifard et al. (2014) trained an ANN to automatically identify well test curve parameters. The coefficients interpolated by Chebyshev polynomial on pressure derivative data are used as the input of ANN to improve the accuracy of parameter identification. Li et al. (2020) used the convolutional neural network to train the deep learning model for identifying curve parameters by taking the complete curve as input, without manually extracting curve features. In addition, the application of deep learning on well-test interpretation for identifying pressure behavior and characterizing reservoirs was also performed (Dong et al., 2021). However, training neural network requires the acquisition or synthesis of a large number of samples, and the interpretability of the parameter inversion results is weak. Sometimes unrealistic interpretation results are prone to appear and the reasons cannot be known (Zhang and Zhu, 2018). To make the parameter inversion results more interpretable and reasonable, this work proposes for the first time a robust automatic well test curve matching method based on reinforcement learning (RL).

RL is inspired by relevant principles of animal psychology (Sutton and Barto, 2018). By imitating the trial-and-error mechanism of humans or animals, the agent can interact with the environment and learn the mapping relationship between state and behavior to obtain the maximum cumulative expected return. Compared with supervised learning algorithms, RL algorithms have the potential to achieve results beyond human performance through active learning, exploration and exploitation. Because of this, RL tends to be a harder learning task. Among RL algorithms, Q learning is one of the most popular (Gao et al., 2020). Recently, the original DRL algorithm naive DQN and its improved algorithm DQN, which combines Q learning with deep neural network, have been introduced and applied into Atari games to achieve automatic control at or beyond the human level (Mnih et al., 2013, 2015). However, these two algorithms lead to overoptimistic value estimates, so van Hasselt et al. (2016) proposed DDQN algorithm to alleviate this problem. DDQN algorithm has been successfully used for battery energy storage system, power system, and stock trading (Arulkumaran et al., 2017; Bui et al., 2019; Shi et al., 2021). These studies show that the DDQN algorithm can avoid the agent trapped into local optimization and is suitable for the environment with large state space.

At present, only limited work about RL has been done in the oil and gas industry. Hourfar et al. (2019) used the DRL algorithm to optimize reservoir water injection. By allowing the agent to dynamically adjusting the water injection rate, higher NPV can be obtained than the traditional optimization method. Miftakhov et al. (2020) use reservoir pressure and water saturation distribution as direct observations to train agents to optimize injection and production parameters. Li and Misra (2020) transformed the history matching problem into a continuous decision problem, and adapted the reservoir permeability using a RL algorithm to achieve automatic history fitting. Guevara et al. (2018) used RL to optimize the gas injection strategy in steam-assisted gravity drainage process. The field test results show that the RL method increases the NPV by at least 30% and reduces the calculation cost by more than 60%. Unfortunately, although the RL has lots of advantages, few work about RL has been done in well test interpretation.

In this study, the DDQN algorithm based agent learns to match the well test data in optimal number of steps by iteratively adjusting the parameters of well test model. This work represents, to our knowledge, the first application of DRL approach for automatically match the well test curves. The new proposed method has a fast speed and reliable results in automatic well test interpretation, which is of great significance to improve the repeatability of well test interpretation.

2. Theory

2.1. Reinforcement learning

The reinforcement learning (RL) is an important paradigm of machine learning, whose goal is to find an optimal strategy to obtain the largest cumulative expected return by train an agent. Markov Decision Process (MDP) is a basic theoretical framework to solve the problem of RL. Within this framework, machines that learn and implement decisions are called **agent**. Anything outside the agent that interacts with it is called the **environment**. In the interaction process, the agent observes the **state** s of the current environment and chooses the **action** an under a certain **policy** π, the environment responds to the action, and the **new state** s' and **reward** r will be fed back to the agent. Therefore, assuming that starting from the initial state s_0, executing the MDP will result in a sequence, $s_0, a_0, r_0, s_1, a_1, r_1, \ldots, s_n, a_n, r_n$.

The agent's job is to optimize the policy for taking action to maximize the cumulative expected return (Sutton and Barto, 2018). The return at step t is the sum of the discount rewards $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$, where $\gamma \in [0,\ 1]$ is discount rate, which determines the present value of future rewards. In RL, it is the most important method to train agents for solving MDP problem based on action value function $Q_\pi(s,a)$. $Q_\pi(s,a)$ represents the expected return on the action a taken in accordance with policy π at state s, as in Eq. (1).

$$Q_\pi(s,a) = \mathbb{E}_\pi[[G_t|s,a], \tag{1}$$

$Q_\pi(s,a)$ calculates the value of the action in a certain state. Simply speaking, $Q_\pi(s,a)$ expresses how good it is for an agent to be in a certain state (Sun, 2020). Therefore, the optimal strategy is based on the optimal value of the action. Specifically, when the optimal action value function $Q^*(s,a) = \max_\pi Q_\pi(s,a)$ is obtained, the optimal policy $\pi^*(s) = \arg\max_{a \in A} Q^*(s,a)$ is to pick the action corresponding to the maximum $Q^*(s,a)$ in each state. In general, $Q^*(s,a)$ can be solved by Bellman optimality equation (Sutton and Barto, 2018), as Eq. (2), which shows the relationship between the current optimal action value function and the subsequent optimal action value function.

$$Q^*(s,a) = \mathbb{E}_{s'}\left[\left[r + \gamma \max_{a'} Q^*(s',a')\middle|s,a\right]\right], \tag{2}$$

where s' is the next state reached after taking action a, and a' is the action taken in the next state. As long as iterates on Eq. (2), it will eventually converge to the optimal action value function $Q^*(s,a)$ (Watkins and Dayan, 1992).

However, the action value function described above can only represent discrete states. For a continuous state space, like the well test curve, the function approximator, denoted as $Q(s,a;\theta) \approx Q^*(s,a)$, must be used to estimate the action value function. $Q(s,a;\theta)$ is usually designed as a neural network (NN) with weights θ, as shown in Fig. 1, which method is called deep reinforcement learning (DRL). This study uses the DDQN algorithm, a value based DRL algorithm, proposed by van Hasselt et al. (2016) to train the agent for curve matching, and the details will be discussed in Section 3.

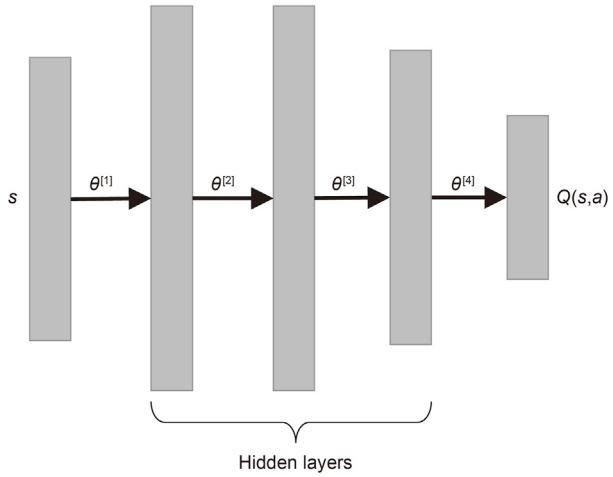

Fig. 1. Schematic diagram of NN function approximator. Wherein, the input layer accepts the state s consisting of the target and the predicted derivative pressure curve, and the output layer returns the action value $Q(s,a)$.

2.2. Well test model

In this work, three classical well test models are used to construct an environment for training agents respectively, and the performance of agents in different environments is evaluated to verify the generalization ability of the proposed method. The three well test models are: homogeneous model, dual-porosity model, and radial composite model. To facilitate analysis, the parameters are dimensionless and are defined in Appendix 1. This section briefly introduces these three well test models.

2.2.1. Homogeneous model

It is assumed that a production well in homogeneous infinite formation is affected by wellbore storage effect and skin effect. According to the solution of Lee et al. (2003), the dimensionless bottom hole pressure p_{wD} is a function of t_D, C_D, and S, which can be denoted as follows:

$$p_{wD} = f(t_D; C_D, S) \tag{3}$$

where C_D is the dimensionless wellbore storage, S is the skin factor, t_D is the dimensionless time. Therefore, the required inversion parameters are C_D, S. A typical log-log plots of bottomhole pressure and its derivative curve is shown in Fig. 2. The derivative curve type used in this work is Bourdet pressure derivative curve (Bourdet et al., 1984).

2.2.2. Dual-porosity model

It is assumed that there are two types of pore media in the formation: the fracture system, which is the fluid flow channel, and the matrix rock system, which is the fluid reservoir space. According to the solution of Lee et al. (2003), the p_{wD} is a function of t_D, C_D, S, ω, and λ, which can be expressed as follows:

$$p_{wD} = f(t_D; C_D, S, \omega, \lambda) \tag{4}$$

with

$$\omega = \frac{(\phi C_t)_f}{(\phi C_t)_f + (\phi C_t)_m}$$

$$\lambda = a r_w^2 \frac{k_m}{k_f}$$

where ω is the storativity ratio, which represents the storage capacity of the fracture; λ is the inter-porosity flow factor, which represents the communication between the fracture and the matrix; ϕ is porosity; C_t is the total compressibility, MPa^{-1}; a is the parameter characteristic of the system geometry; r_w is the well radius, m; k is permeability, mD; Subscript m represents matrix; Subscript f represents fracture. Therefore, the parameters need to be inverted are C_D, S, ω, and λ. A typical log-log plots of pressure and its derivative curve is shown in Fig. 3.

2.2.3. Radial composite model

It is assumed that there are two seepage zones with different physical parameters in the formation and there is no additional pressure drop at the interface of the two zones. According to the solution of Chu and Shank (1993), The p_{wD} is a function of t_D, C_D, S, M, and R_i, which can be denoted as follows:

$$p_{wD} = f(t_D; C_D, S, M, R_i) \tag{5}$$

with

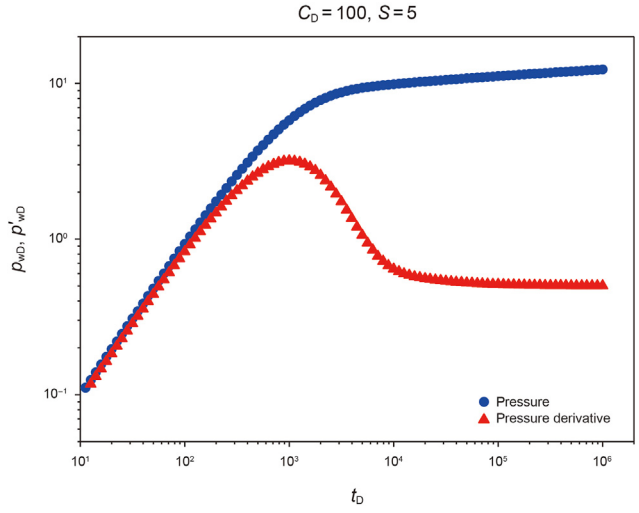

Fig. 2. The typical well test curves for homogeneous model.

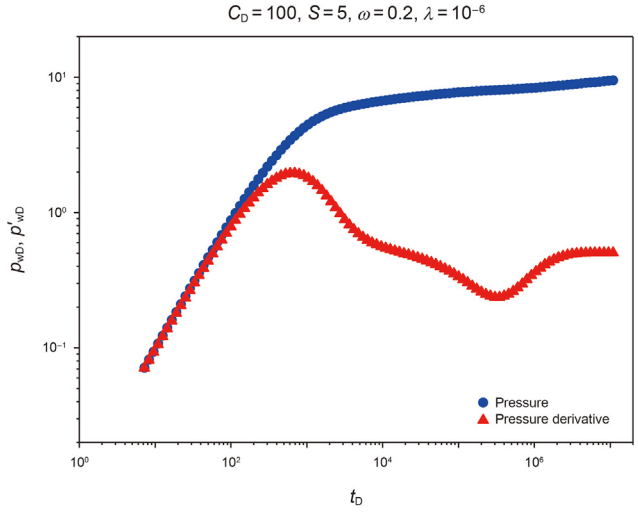

Fig. 3. The typical well test curves for dual-porosity model.

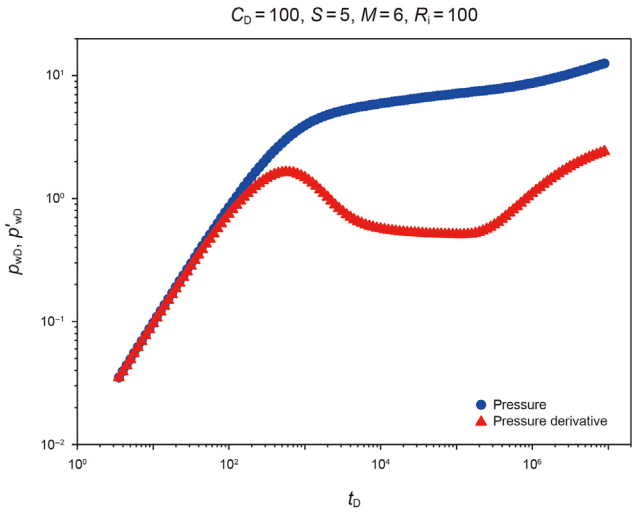

Fig. 4. The typical well test curves for radial composite model.

$$M = \frac{(k/\mu)_{ir}}{(k/\mu)_{er}}$$

where M is mobility ratio; R_i is inner zone radius, m; μ is viscosity, mPa·s; Subscript ir represents inner zone; Subscript er represents outer zone. Therefore, the parameters to be inverted are C_D, S, M, and R_i. A typical log-log plots of pressure and its derivative curve is shown in Fig. 4.

3. Method

Based on concepts of DRL, the well test curve matching process can be regarded as an MDP. Therefore, the automated well test curve matching process can be achieved by applying DRL algorithm. Fig. 5 is a schematic diagram of automatic curve matching process based on MDP. In this process, the target and predicted pressure derivative curve are considered as state s_t at first. Here, the target curve $p'_{wD,target}$ comes from the measured curve, and the predicted curve p'_{wD} is calculated from the well test model. Following that, the agent provides the action a_t of parameter adjustment and updates the policy based on the reward r_t. Next, the environment based on the well test (WT) model receives action a_t and then updates the parameters of the WT model. Finally, the environment outputs new states s_{t+1} and rewards r_{t+1} back to the agent. In this way, the agent continuously interacts with the environment to obtain more rewards, which will make the prediction parameters continuously update to the target parameters to complete the curve matching.

In the remainder of this section, we will detail the agent and environment of the study in turn.

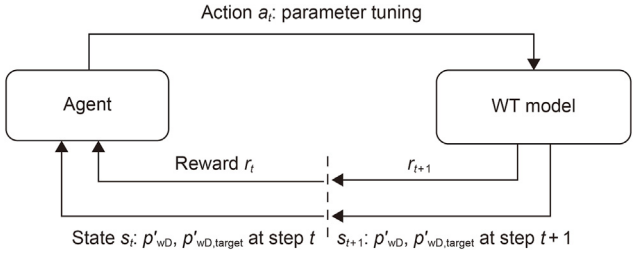

Fig. 5. Schematic diagram of agent interacting with the environment in MDP. The environment is encapsulated by the well test model (WT model).

3.1. Agent

This section will introduce the double deep Q-network (DDQN) algorithm used to train the agent, the policy used when the agent interacts with the environment, and the discrete mode of action space to avoid the curse of dimensionality in the parameter adjustment process.

3.1.1. DDQN algorithm

As noted earlier, in this work the double deep Q-network (DDQN) (van Hasselt et al., 2016) algorithm is considered. DDQN is a value-based DRL algorithm that can be used with continuous state spaces. DDQN uses the policy based on the online Q-network $Q(s,a;\theta)$ to select actions that interact directly with the environment, and the target Q-network $Q(s,a;\theta')$ to evaluate the selected actions separately, as shown in Fig. 6. Separating the action selection from the evaluation can alleviate the overly optimistic estimate of the value of the action and improves the accuracy of the action value estimation. In addition, to improve the utilization efficiency of data, the transition tuples at timestep t, $e_t = (s_t, a_t, r_t, s'_t)$ generated by the interaction between the agent and the environment is stored in a buffer called experience replay memory $\mathscr{D}_t = \{e_1, e_2, ..., e_t\}$ (Mnih et al., 2015), as Fig. 6. Therefore, by extracting experiments randomly in $\mathscr{D}$, it is possible to use minibatch samples to update these two Q-networks. More precisely, the loss function, as Eq. (6), is used to updates the weights θ of online Q-networks, while the weights θ' of target Q-network is a delayed copy of the online Q-networks, which is to copy after a certain number of training. The complete training process is discussed in Section 3.3.

$$Loss(\theta) = \mathscr{E}_{(s,a,r,s')\sim\mathscr{U}(\mathscr{D})}\left[\left[\left(r + \gamma Q\left(s', a_{pred}{}'; \theta'\right) - Q(s,a;\theta)\right)^2\right]\right], \tag{6}$$

with

$$a_{pred}{}' = \underset{a}{\text{argmax}} Q(s', a; \theta),$$

where θ is the weight of online Q-networks, θ' is the weight of target Q-networks, γ is discount factor. The notation $(s,a,r,s') \sim \mathscr{U}(\mathscr{D})$ represents a minibatch samples are uniformly sampled from the replay memory.

Once the training of the agent is completed, the online Q-networks is used for prediction. For a particular state, the online Q-networks will provide a Q-value for each of the possible actions for that state. Finally, the action is determined by the policy in terms of Q-value, which will be discussed in detail later.

3.1.2. Policy

When agents interact with the environment, an important problem is to trade-off exploration and exploitation. Exploitation

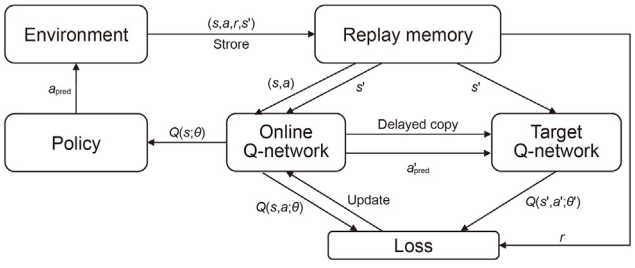

Fig. 6. Schema of the DDQN model in training process.

Table 1
Action spaces in three environments based on well test models.

Well test model	Parameters	Parameter bounds	Action space	Action index	Reward weight
Homogeneous	C_D	50–5000	$[+\Delta C_D, -\Delta C_D]$	1/2	1
	S	0–10	$[+\Delta S, -\Delta S]$	3/4	1
Dual porosity	C_D	50–5000	$[+\Delta C_D, -\Delta C_D]$	1/2	1
	S	0–10	$[+\Delta S, -\Delta S]$	3/4	1
	ω	0.02–0.6	$[+\Delta\omega, -\Delta\omega]$	5/6	2
	λ	1.0×10^{-8}–1.0×10^{-5}	$[+\Delta\lambda, -\Delta\lambda]$	7/8	1.5
Radial composite	C_D	50–5000	$[+\Delta C_D, -\Delta C_D]$	1/2	1
	S	0–10	$[+\Delta S, -\Delta S]$	3/4	1
	M	0.1–10	$[+\Delta M, -\Delta M]$	5/6	1.5
	R_i, m	30–300	$[+\Delta R_i, -\Delta R_i]$	7/8	2

means that the agent takes the most valuable action it estimates every time, and exploration means that the agent randomly takes actions to explore the environment. Proper exploration will help the agent find better policy and improve the accuracy of the action value estimation. However, the agent cannot simultaneously exploration and exploitation in the same action selection. Therefore, the selection of action needs to obey a certain probability distribution to make exploration and exploitation alternate. In Section 5.1, we discussed the influence of different policies on the training process. The results show that Boltzmann exploration policy (Derhami et al., 2008) can better balance the exploration and exploitation of agents in the curve matching task. In the Boltzmann exploration policy, the probability distribution of the agent's action selection is determined by Eq. (7).

$$\pi(a|s) = \frac{e^{Q(s,a)/\tau}}{\sum_a e^{Q(s,a)/\tau}}, \tag{7}$$

where $\tau \in (0, 1]$ is the temperature parameter, which controls the certainty of the action.

3.1.3. Curse of dimension

The DRL algorithm based on value function $Q(s,a)$ needs to clarify the actions that an agent can perform, which means that the action space of the agent is discrete. Therefore, when it is applied to a high-dimensional continuous action space, such as well test curve matching problem, the continuous action needs to be discrete. However, this will lead to huge combinatorial increase in the number of actions with the number of well-test parameter dimensions. For example, for the homogeneous model, it is assumed that the parameters to be inverted are dimensionless well storage coefficient C_D and skin coefficient S. If the parameter space is discretized into 100 intervals, the action space of the two parameters C_D and S will reach 100^2. For the agent, the value of this 100^2 action needs to be calculated, that is, the output dimension of the Q network is 100^2. Generally, if the number of parameters to be inverted is N and the number of parameter discrete spaces is n_d, then the total action space to be considered is $\prod_{d=1}^{N} n_d$. When the parameter precision requirement is high or the number of parameters is large, the agent will not be able to handle the resulting huge action space. To alleviate the problem, an asynchronous parameter tuning method is proposed. To be more specific, the agent only adjusts one well test parameter at a time, and the action for each parameter is only set to increase and decrease the value of the parameter, and the step length of the increase or decrease is set to a fixed value. Obviously, the total action space that needs to be considered is only $2N$ based on this approach. This allows the current discrete action DRL algorithm to be applied to well test curve matching task. In this work, three typical well test models are used to verify the proposed method. The parameter range and action space of each well test model are shown in Table 1. In addition, the reward weights are given for different parameters to make the agent pay more attention to those parameters that are important in reflecting the reservoir characteristics, as shown in Table 1. In the next section, we will discuss how these weights are used.

For the action step length, it can be set as follows.

a) The well test model parameters ψ are uniformly distributed in the interval:

$$\psi_i \leftarrow \psi_i \pm \Delta_i, \tag{8}$$

with

$$\Delta_i = \frac{\psi_{i\max} - \psi_{i\min}}{n_d}.$$

b) The sensitivity of the double log curve to the parameter values in different ranges is different. We hope to have smaller step length in more sensitive areas. Therefore, some well test model parameters ψ are exponentially distributed in the interval:

$$\psi_i \leftarrow \psi_i \times 10^{\Delta_i} \text{ or } \psi_i \leftarrow \psi_i / 10^{\Delta_i}, \tag{9}$$

with

$$\Delta_i = \frac{\log_{10}\psi_{i\max} - \log_{10}\psi_{i\min}}{n_d}.$$

In this work, the S is set to uniformly distributed, and other parameters are set to exponentially distributed. n_d is set to 100.

3.2. Environment

After the agent takes an action, its state will change and the environment will generate feedback (reward). The environment is constructed based on the test well model and is designed for the agents to interact with it for curve matching. In an environment, the state and reward provided to the agent are two basic elements. For the state, we define it as in Eq. (10).

$$s = \left[\log_{10} p'_{wD}(\psi), \log_{10} p'_{wD,target}(\psi_{target})\right], \tag{10}$$

where p'_{wD} is the predicted dimensionless pressure derivative curve, and $p'_{wD,target}$ is the target dimensionless pressure derivative curve. It should be noted that to ensure the good versatility of the trained agent, the parameters and pressure data are all dimensionless. In the training process, $p'_{wD,target}$ comes from the theoretical curve. In the inference process, $p'_{wD,target}$ comes from the

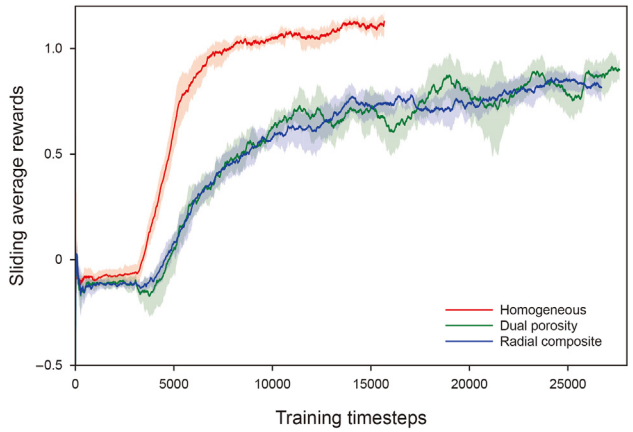

Fig. 7. DDQN performance on different well test models. The results are obtained by running 5 random simulations. The darker line shows the median over 5 random simulations and the shaded area shows 95% confidence interval.

measured curve. In addition, $p'_{\mathrm{wD,target}}$ needs to be interpolated to ensure consistency with p'_{wD} in the time dimension. For the reward, we give the definition as in Eq. (11).

$$r=\begin{cases} w(i) & \text{if } err^{s}-err^{s'}>0 \\ -1.1\times w(i) & \text{if } err^{s}-err^{s'}\le 0 \\ -2 & \text{if out of scope} \\ 10 & \text{if } err^{s}-err^{s'}\le \delta \end{cases}, \tag{11}$$

where $w(i)$ is the reward weight of the ith well test model parameter, and its value is shown in Table 1. The interpretation results of important or insensitive parameters can be improved by setting different reward weights. δ is the maximum error when curve matching is completed. The definition of err is as follows:

$$err=\sum\left|\psi-\psi_{\mathrm{target}}\right|, \tag{12}$$

Equation (12) indicates that when the agent takes an action, the parameter error becomes smaller and the reward will be positive. Otherwise, the reward will be negative. In addition, negative rewards need to be greater than positive rewards, which helps the agent reach its goal faster and reduce unnecessary actions (Wiewiora, 2003).The quality of reward design significantly affects the performance of the agent. Therefore, in Section 5.2, we discuss the influence of different reward design methods on the results.

3.3. Training process and implementation details

Algorithm 1 is the training process for the proposed method to accomplish automated pressure derivative curve matching. The process starts with an initial target pressure derivative curve at the beginning of each episode. We trained the agent to perform a total of 200 episodes of curve matching. The termination condition of each episode is that the curve matching is completed or the number of timesteps exceeds 300. To allow the agent to fully explore the environment, the agent's action in the first ten episodes are randomly sampled. After completing the exploration, the agent enters the training state.

Algorithm 1. The training process of an agent in this work

Algorithm 1 The training process of an agent in this work
Initialize replay memory D to capacity 10^5
Initialize $Q(s,a;\theta)$ with random weights θ
Initialize $Q(s,a;\theta')$ with weights θ'
for episode =1 **to** max episode **do**
  Reset the environment and obtain s
  while $\Delta err>\delta$ and $T<$ max timesteps **do**:
    if episode > 10 **then**:
      Sample random minibatch of transitions $(s;a;r;s')$ from D
      Calculate the $Loss(\theta)$,
      Run the Adam algorithm on $Loss(\theta)$ to update θ
    Every 1000 timesteps reset $\theta'=\theta$
      Under state s, select action a according to Boltzmann exploration policy
    else:
      Randomly select the action a
    end if
    Execute action a in environment, obtain reward r, curves s', and error change Δerr
    Store transition $(s;a;r;s')$ in D
      Set $s'=s$
  end while
end for

In this work, the Adam algorithm (Kingma and Ba, 2014) was

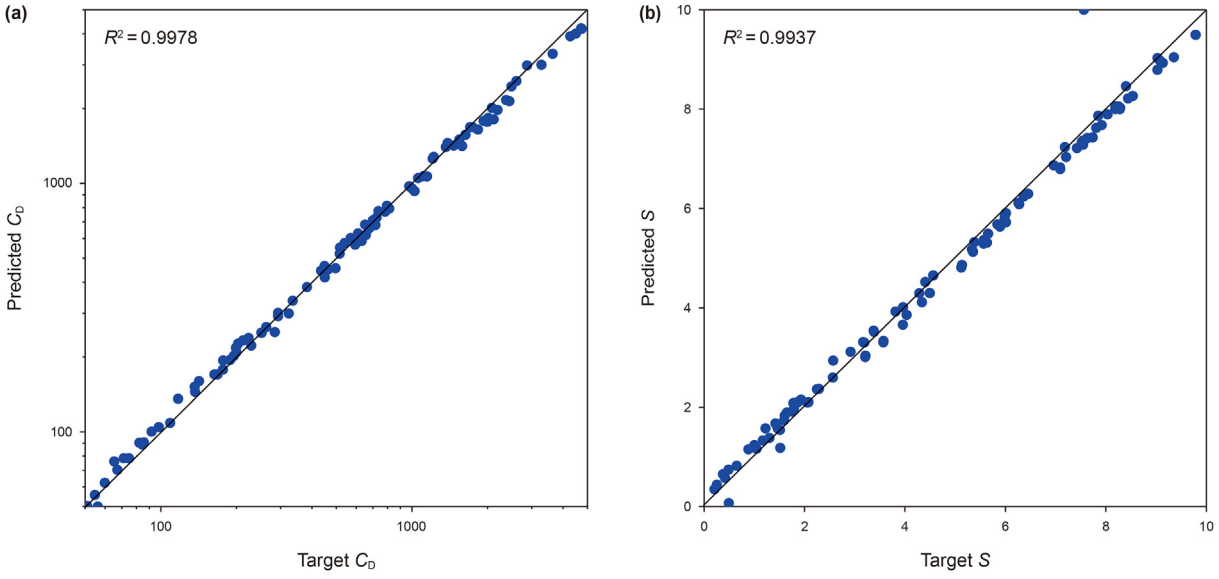

Fig. 8. The result of 100 times curve matching of the agent on the homogeneous model.

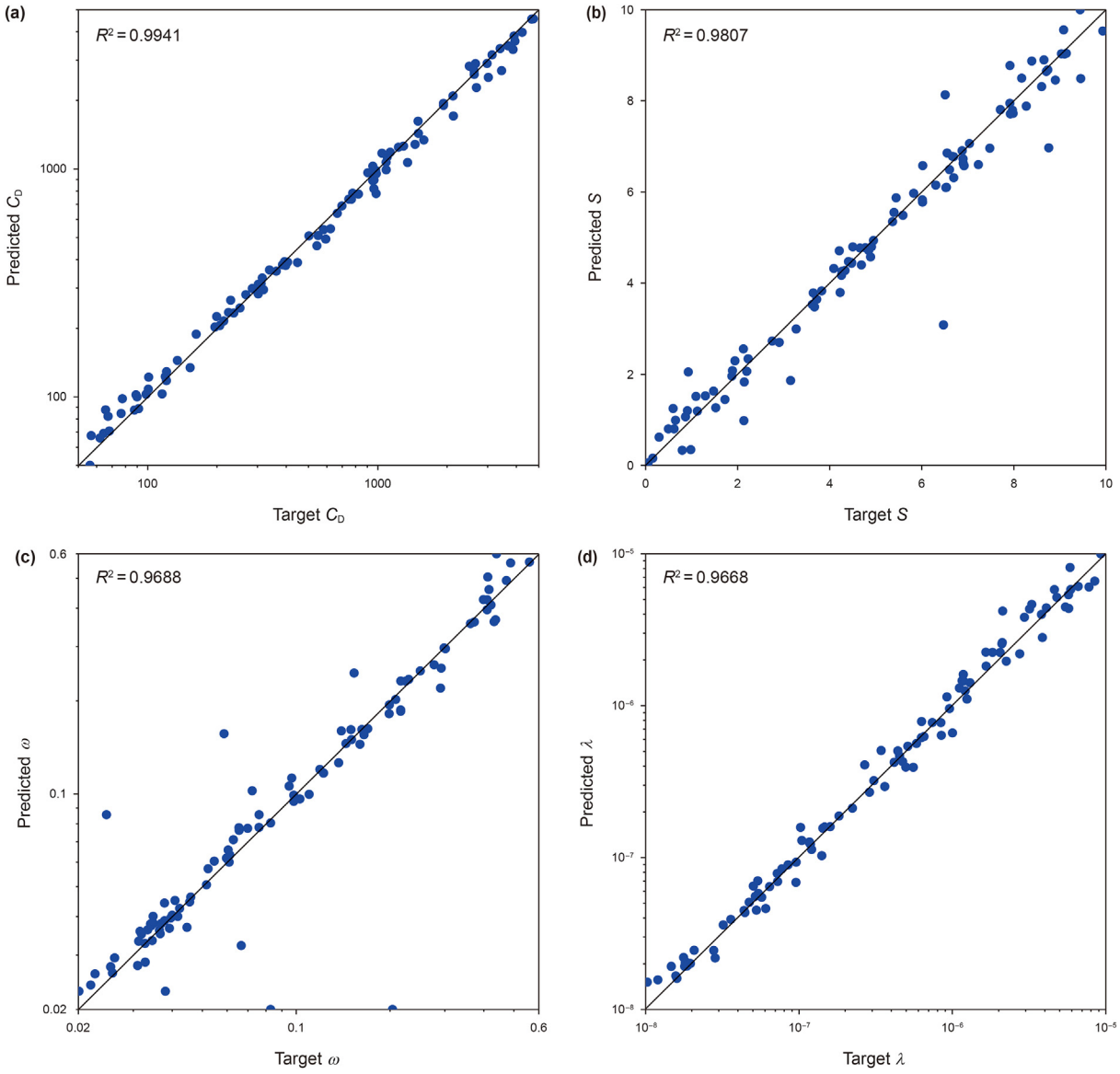

Fig. 9. The result of 100 times curve matching of the agent on the dual porosity model.

used to updates the weight θ of the online Q-network by minimizing the loss function in Eq. (6) with a learning rate of 0.0001. The weight θ' of target Q-network was updated every 1000 timesteps by the delayed copy. The discount factor γ in Eq. (6) is set to 0.99. The capacity of the replay memory is 10^5, and the minibatch size is set to 128. The temperature parameter τ in Eq. (7) is 1 during the training and 10 during the inference.

The Q-network used in this work is a neural network composed of three hidden layers, and the number of neurons in each layer is 500, 500, 300, respectively. The hidden layer activation function is ReLU, while the output layer is not activated. The input layer takes the state with a feature number of 80 as input, and the output layer gets the Q-value for each action. The output dimension is determined by the environment. In this work, the output dimension based on the homogeneous model environment is 4, and others environment is 8. The agent is implemented based on PyTorch and trained on NVIDIA 1060Ti graphic processing unit. The average training time was 26 min, and the average time for each curve matching was only 2.86 s. In addition, the raw data, due to the data noise from human factors, measuring instruments, and production conditions, which will affect the results of automatic parameter inversion. In this work, wavelet threshold denoising method (Valencia et al., 2016) is used to denoise the original pressure and derivative data.

4. Method verification

4.1. Result verification

To show the reliability of DDQN, Fig. 7 compares the performance of DDQN on different well test models. To avoid the influence of randomness on the results (Mania et al., 2018; Henderson et al., 2018), we perform 5 random simulations and draw the training curve with 95% confidence interval. The results show that the agent trained based on the DDQN algorithm takes the fewest timesteps to complete the curve matching and obtains the highest reward in the homogeneous model. However, for the dual porosity and radial composite model, the agent has similar performance in both. This indicates that the number of parameters to be inverted, that is, the size of the action space, has a greater impact on the

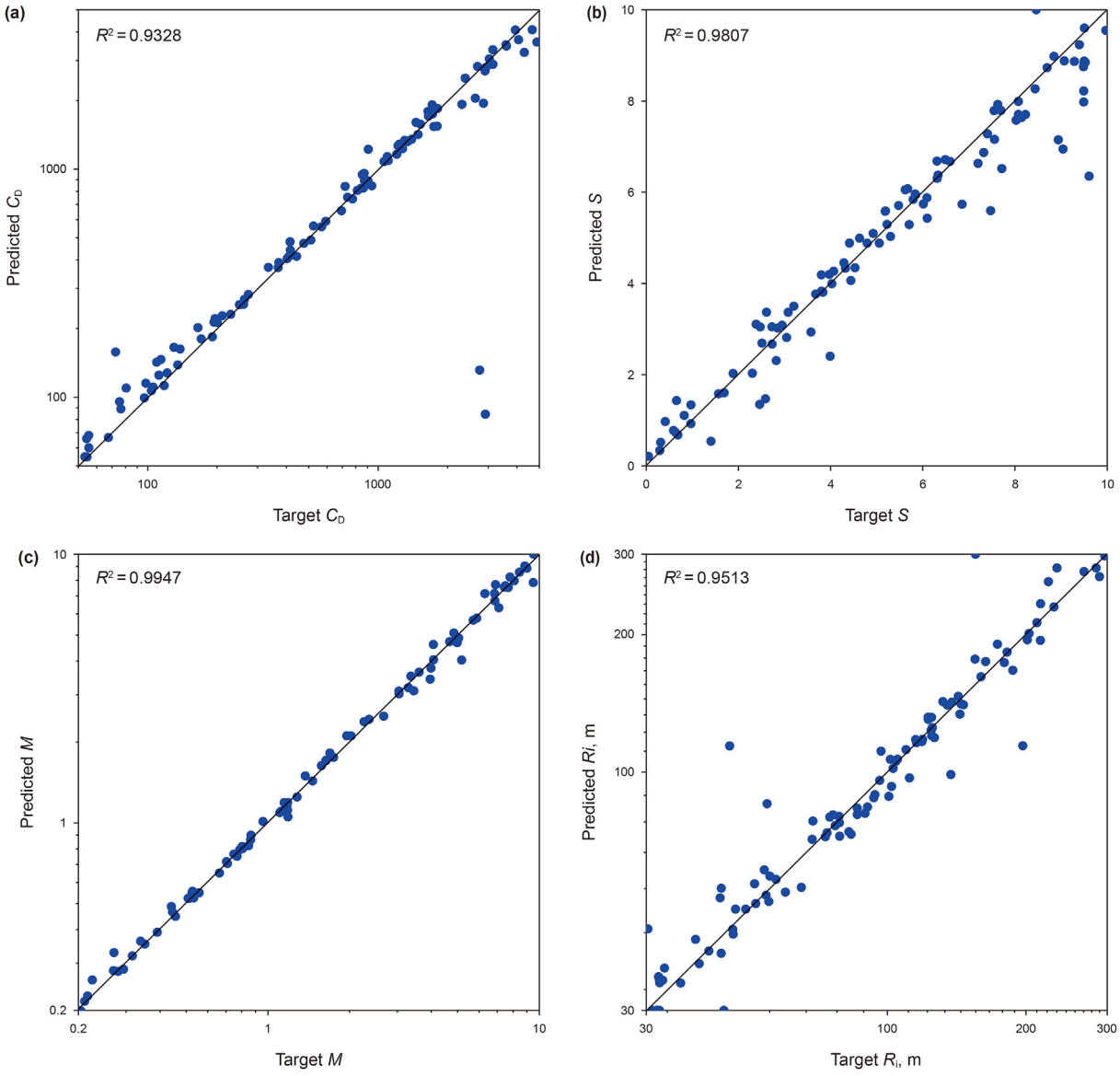

Fig. 10. The result of 100 times curve matching of the agent on the radial composite model.

performance of the agent than the well test model itself.

In order to verify the reliability of the proposed method, the trained agent based on DDQN algorithm was asked to complete 100 times of curve matching on different well testing models, and the error of the finally obtained parameters was analyzed. In the inference process, the agent only selects actions and does not update the parameters of its action value network Q. The parameter inversion results on the three well test models are shown in Figs. 8–10. The results show that the predicted parameters have a pronounced correlation with the actual parameters (considering R^2), which proves that the trained agent are able to invert the curve parameters accurately. Furthermore, we can observe that the accuracy of parameter inversion on the homogeneous model is higher than the other two models, which indicates that the parameter inversion error increases correspondingly with its number.

We performed further statistical analysis of the errors in the parameter inversions. Table 2 is the statistical indicators used and their calculation formulas. The statistic results of parameter errors are reported in Table 3. Concretely, the mean of relative error (MRE) of the parameters is 7.58% for the homogeneous model, 10.66% for the radial composite model, and 12.79% for the dual porosity model. In addition, the median is substantially lower than the mean, as Table 3, which indicates that the accuracy of the parameter inversion is quite high without considering the effect of extreme values. Standard deviation of relative error reflects the effect of parameter sensitivity on the inversion results. We can observe that the standard deviation of relative error of the skin factors (S) is generally larger due to its low sensitivity. Moreover, mean absolute error (MAE) and root mean square error (RMSE) are calculated to visualize the magnitude of the error, as shown in Table 3. Finally, coefficient of determination R^2 are calculated to synthetically evaluate the performance of the parameter inversion. The results show that the agents perform well in these three models, and the error of parameter inversion and the parameter inversion results balance accuracy and stability.

4.2. Result comparison

In this section, to further show the advantages of DDQN, we compare its results with other two DRL algorithms and three classic

Table 2
The calculation formula of evaluation index. K is the number of inferences.

Evaluation index	Calculation formula
Mean of relative error, %	$\frac{1}{K}\sum_{i}^{K}\left\|\frac{\psi_i-\widehat{\psi}_i}{\psi_i}\right\|$
Median of relative error, %	Median of $\left\|\frac{\psi_i-\widehat{\psi}_i}{\psi_i}\right\|$
Std. of relative error	$\sqrt{\frac{1}{K}\sum_{i}^{K}(e_{r,i}-\overline{e}_r)^2}$, where $e_{r,i}=\left\|\frac{\psi_i-\widehat{\psi}_i}{\psi_i}\right\|$
Mean absolute error	$\frac{1}{K}\sum_{i}^{K}\|\psi_i-\widehat{\psi}_i\|$
Root mean square error	$\sqrt{\frac{1}{K}\sum_{i}^{K}(\psi_i-\widehat{\psi}_i)^2}$
R^2	$1-\sum_{i}^{K}(\psi_i-\widehat{\psi}_i)^2/\sum_{i}^{K}(\psi_i-\overline{\psi})^2$

machine learning (ML) algorithms.

The other two DRL algorithms are Naive DQN (Mnih et al., 2013), DQN (Mnih et al., 2015). Fig. 11 shows the single-step rewards obtained by the Naive DQN (Mnih et al., 2013), DQN (Mnih et al., 2015), and DDQN algorithms when performing curve-fitting training on the homogeneous, dual porosity, and radial composite well test model. Fig. 11 indicates that DDQN algorithm finally converges to a stable reward value on all well test models and the reward is the highest with the smallest variance. The Naïve DQN single-step reward has been hovering at 0, indicating that a good strategy cannot be learned. The performance of DQN lies between these two algorithms. The training results show that the DDQN algorithm is more suitable for the problem of automatic curve matching.

Three alternative supervised ML algorithms are: artificial neural network (ANN), random forest (RF), support vector regression (SVR) with multiple output. In this work, the input of ML algorithm is set to the measured pressure derivative curve with a feature number of 40, and the output is set to the unknown parameters of

Table 3
Error statistical of parameter inversion results.

Well test model	ψ	Mean of relative error, %	Median of relative error, %	Std. of relative error	Mean absolute error	Root mean square error	R^2
Homogeneous model	C_D	5.48	4.63	3.95	63.62	129.19	0.9978
	S	9.69	3.67	16.09	0.21	0.32	0.9937
Dual porosity model	C_D	7.59	5.97	6.74	76.53	149.60	0.9941
	S	11.83	4.82	20.07	0.33	0.55	0.9807
	ω	15.29	8.43	31.76	0.017	0.036	0.9688
	λ	16.42	10.54	15.21	2.54×10^{-7}	5.35×10^{-7}	0.9668
Radial composite model	C_D	11.37	5.11	18.47	152.63	448.38	0.9328
	S	16.35	6.30	39.94	0.4466	0.6852	0.9807
	M	4.75	2.89	5.32	0.13	0.28	0.9947
	R_i	10.17	6.07	19.09	9.68	20.95	0.9513

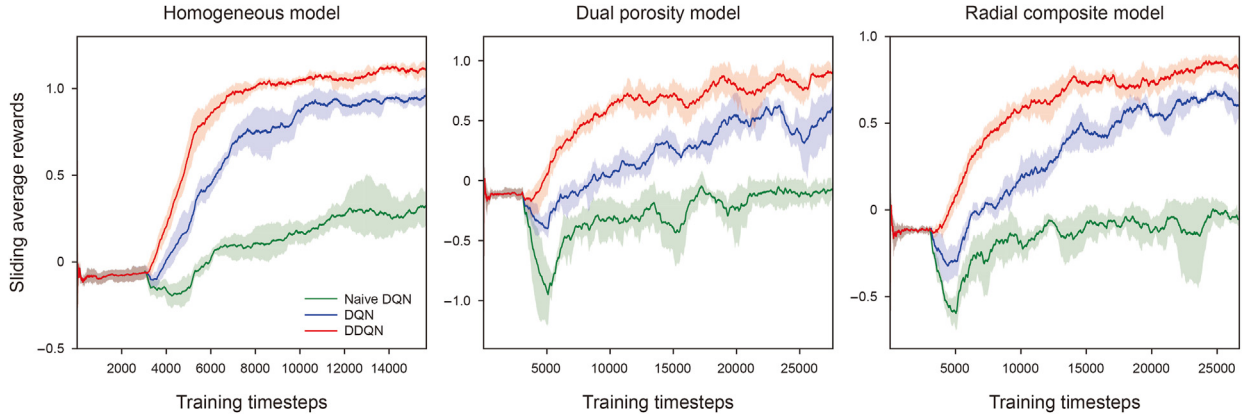

Fig. 11. Performance curves of different agents on different well test models.

Table 4
Performance scores for all models.

		Mean of relative error, %	Median of relative error, %	Std. of relative error	R^2
Homogeneous model	ANN	8.09	**3.08**	26.90	0.9925
	SVR	18.64	16.43	9.25	0.9336
	RF	**4.97**	4.11	**4.62**	**0.9979**
	DDQN	7.58	4.15	10.02	0.9958
Dual porosity model	ANN	18.07	7.64	32.45	0.9289
	SVR	21.69	13.48	46.13	0.9025
	RF	19.65	**6.71**	47.65	0.9590
	DDQN	**10.66**	7.44	**18.45**	**0.9776**
Radial composite model	ANN	18.21	7.11	52.81	0.9316
	SVR	18.59	14.95	**13.34**	0.9207
	RF	16.24	5.44	41.44	0.9392
	DDQN	**12.79**	**5.09**	20.71	**0.9649**

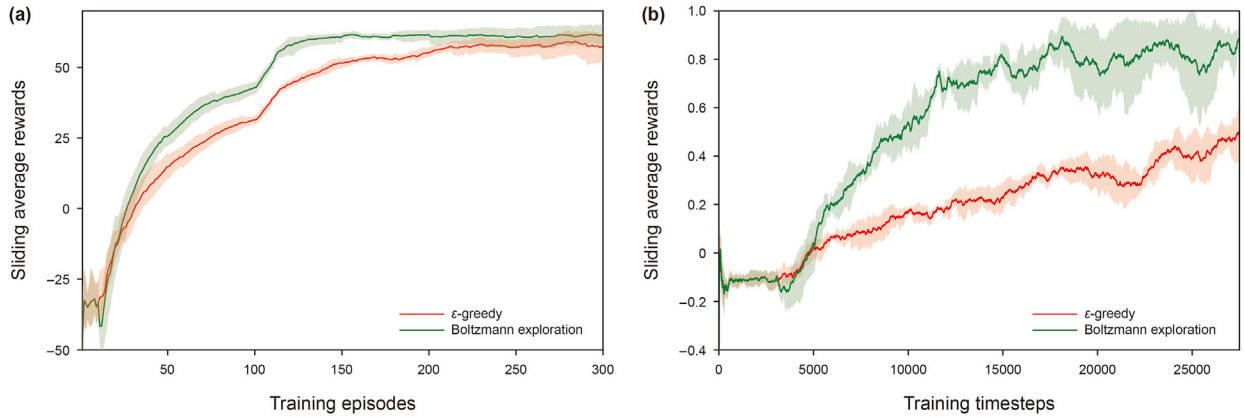

Fig. 12. Performance curves of different policies. (**a**) Cumulative reward curve for each episode, and (**b**) single-step reward curve.

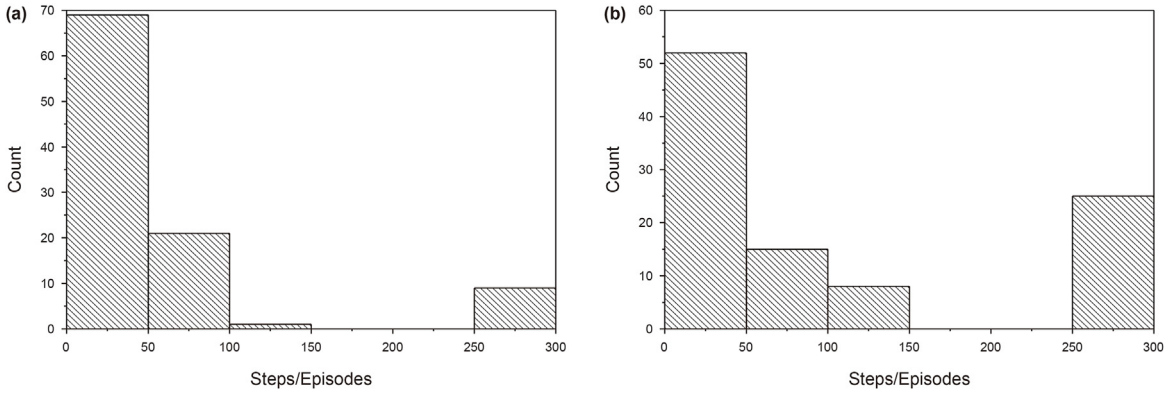

Fig. 13. Histograms of steps required to complete curve matching 100 times by different policies. (**a**) Boltzmann exploration policy, and (**b**) ε-greedy policy.

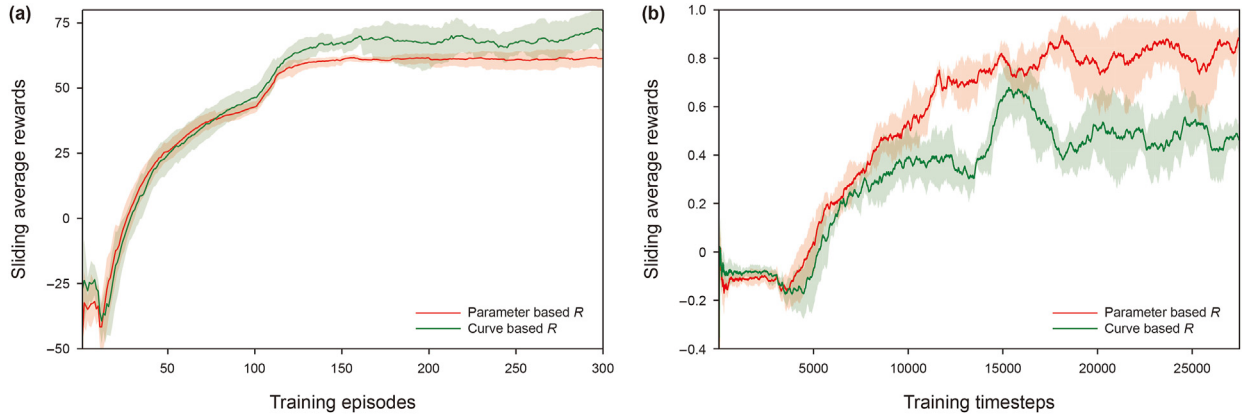

Fig. 14. Performance curves of different reward designs. (**a**) Cumulative reward curve for each episode, and (**b**) single-step reward curve.

the well test model. The sample size of training set and validation set are 900 and 100, respectively.

Table 4 shows the comparison between the inference results of DDQN and the prediction results of ML algorithm in the validation set. The indicator in Table 4 are averages of all parameters for each well test model. We can observe that, for the homogeneous model, RF has obtained the best parameter inversion results (considering R^2), and DDQN has obtained the suboptimal results. For dual porosity model and radial composite model, DDQN achieved the optimal results, followed by RF, ANN and SVR. When the unknown parameters of the well test model are few (homogeneous model), the ML algorithm generally performs well. However, when the well test model becomes complex, the parameter inversion results of DDQN will be more robust, which can be observed from the fluctuation of Mean of Relative Error in different well test models.

5. Results and discussion

5.1. Impact of policy

In this section, we compare the effects of ε-greedy policy and bBoltzmann exploration policy, two commonly used exploration

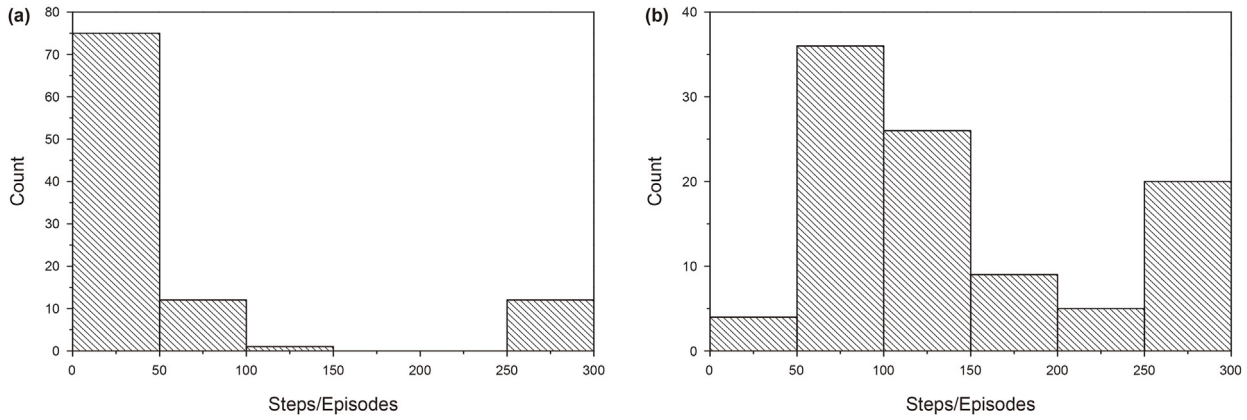

Fig. 15. Histograms of steps required to complete curve matching 100 times by different reward designs. (**a**) Parameter-based reward design, and (**b**) curve-based reward design.

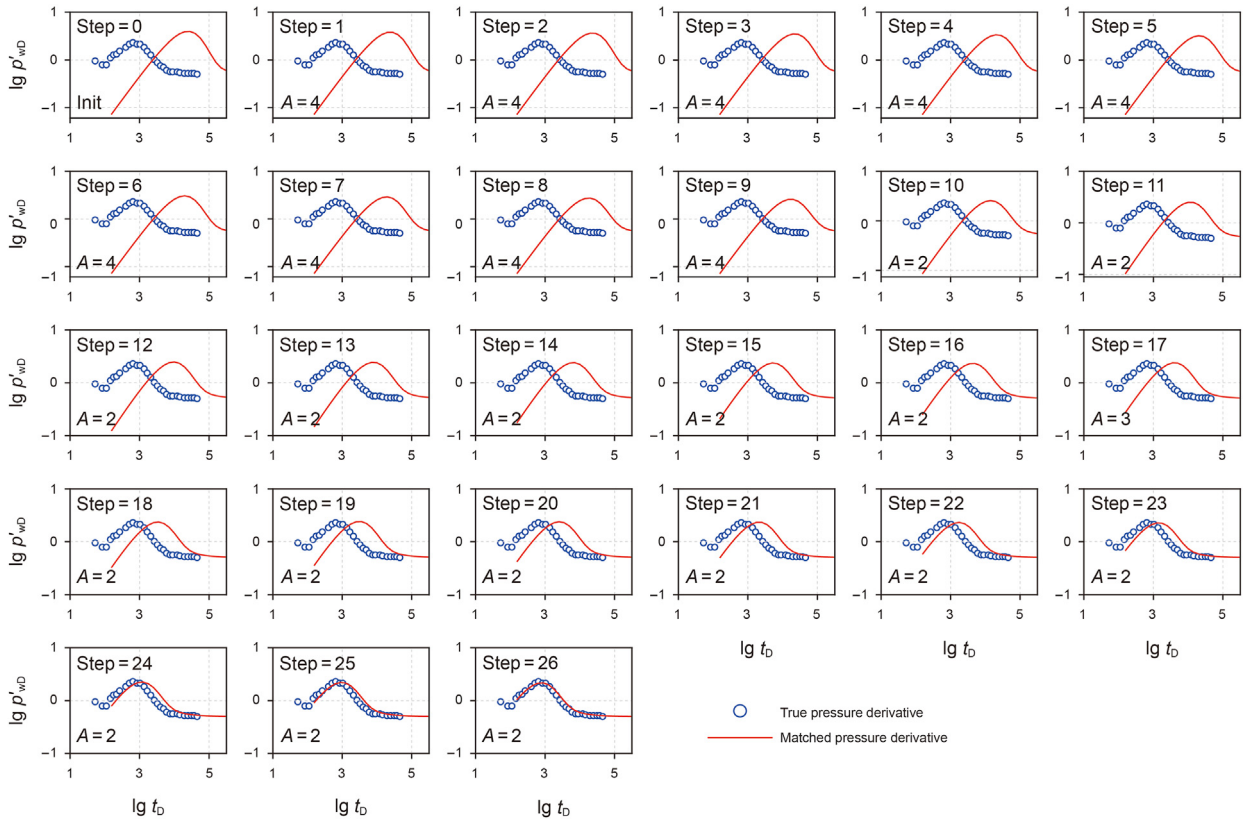

Fig. 16. Visualization of the step-by-step parameter adjustment process in Example 1. *A*: action index.

and balance methods, on the performance of the agent's well test curve matching. Among them, the action probability distribution of the ε-greedy policy is shown in Eq. (13).

Fig. 12 presents the training process under different policies. The results show that the Boltzmann exploration policy will enable the agent to obtain higher cumulative rewards and single-step rewards. In addition, the agent under Boltzmann exploration policy has fewer steps when completing curve matching, as shown in Fig. 13.

This is because Boltzmann exploration policy is more suitable for a highly certain environment such as well test curve matching tasks. Besides, the use of Boltzmann exploration policy allows the agent to be biased towards exploration in the early stages of training and towards exploitation in the later stages. Therefore, the action certainty of the agent using Boltzmann exploration policy will improve with training, and converge to a better strategy.

$$\pi(a|s)=\begin{cases}1-\varepsilon+\dfrac{\varepsilon}{|\mathscr{A}(s)|} & \text{if } a=A^*\\ \dfrac{\varepsilon}{|\mathscr{A}(s)|} & \text{if } a\neq A^*\end{cases}, \tag{13}$$

Table 5
Case 1 parameter inversion result.

	C_D	S
Manual matching	87.3	2.63
Automatic interpretation	79.0	2.64
Relative error, %	9.51	0.38

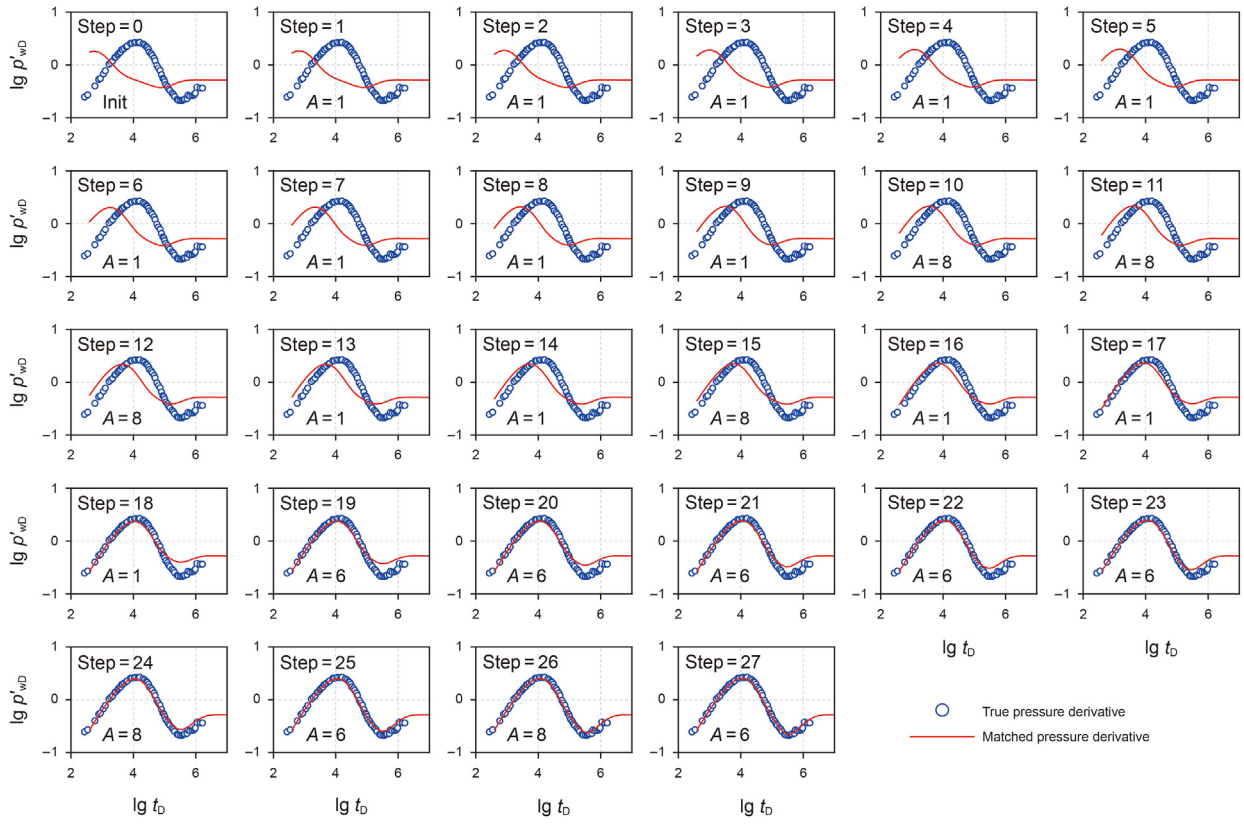

Fig. 17. Visualization of the step-by-step parameter adjustment process in Example 2. *A*: action index.

Table 6
Case 2 parameter inversion result.

	C_D	S	ω	λ
Manual matching	1809	1.10	0.125	7.76×10^{-7}
Automatic interpretation	2032	0.86	0.130	6.40×10^{-7}
Relative error, %	12.33	21.82	4.00	17.53

where A^* is the optimal action, $a \in \mathscr{A}(s)$, $|\mathscr{A}(s)|$ is the number of actions, ε is the probability of taking a random action.

5.2. Reward design

In this section, we discuss the impact of two different reward design methods on the agent performance in the task of automatic curve matching. In RL, the design of reward function will significantly affect the training effect of the agent (Ng et al., 1999; Laud, 2004). Therefore, it is very important to find a reward function suitable for the current environment for the agent. For curve matching problems, an intuitive reward function design method is based on the error between the target curve and the prediction curve, as in Eq. (14). However, the reward design based on curve error is not a good practice. In fact, when the agent takes the action of parameter adjustment, the reduction of the error between the target curve and the prediction curve sometimes does not mean that the parameter error is reduced. As a result, this ambiguous reward design causes the agent's training to oscillate or diverge. Correspondingly, a more accurate reward design method is to directly use the error between the target curve parameters and the prediction curve parameters. So, we adopt this method to design the reward function, as shown in Eq. (12).

$$err = \sum \left| p'_{\mathrm{wD}} - p'_{\mathrm{wD,target}} \right|, \tag{14}$$

Fig. 14 compares the training process of the two reward design methods. The training process is based on Algorithm 1 and the reward value is still as shown in Eq. (11). It is shown that when the agent uses the curve error-based reward design, the cumulative reward is more, as in Fig. 14(a), but the single-step reward is lower, as in Fig. 14(b). This indicates that the agent has taken an action that can be rewarded but does not help accomplish the goal. Moreover, Fig. 15 compares the histograms of steps required to complete curve matching 100 times by these two methods. The results indicate that setting the reward based on the parameter error will complete the curve matching task faster.

6. Field application

To further show the practicality of the proposed automatic matching method, case studies were carried out with 3 typical wells as examples. To make the agent suitable for reservoirs and wellbore under different conditions, the data used is dimensionless.

6.1. Example 1

Example 1 is an exploration well from the Dagang Oilfield. The permeability is interpreted as 5.4 mD. The well was produced 222 min before the test at a rate of 13.8 m^3/d, shut in 621 min during the test, and a good build-up pressure curve was measured. The pressure derivative curve of this well shows the characteristics of a homogeneous reservoir. After dimensionless, the agent trained on the homogeneous model is used to automatically fit the curve.

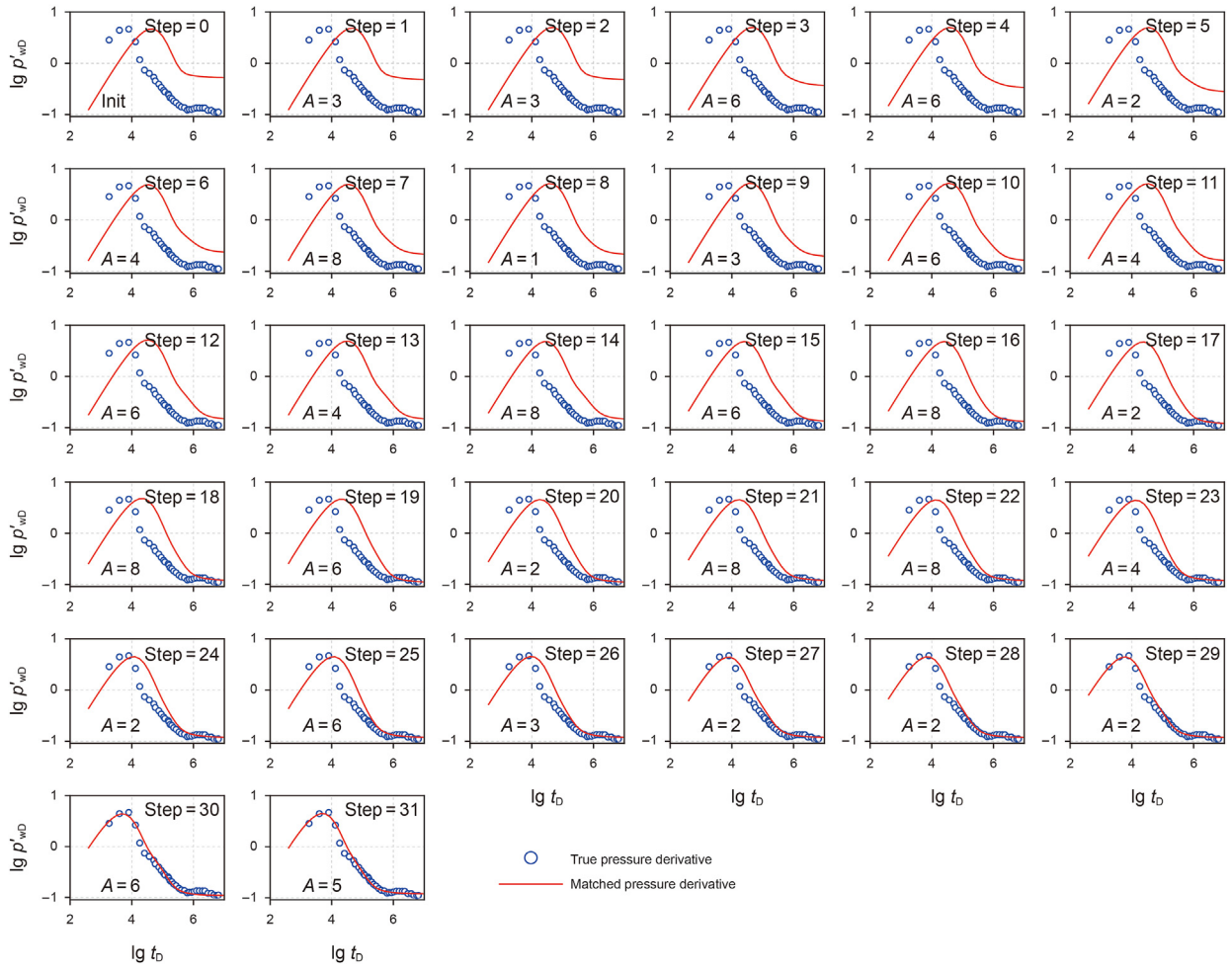

Fig. 18. Visualization of the step-by-step parameter adjustment process in Example 3. *A*: action index.

After 26 steps of parameter adjustments, the agent completed the matching process. To understand the curve matching process of the agent, the parameter adjustment action is shown in Fig. 16 step by step. Finally, the automatic interpretation result is $S = 2.64$, $C_D = 79$. In curve matching process, the agent prefers to adjust C_D with obvious features first, and then adjust S. Table 5 compares the results of automatic matching and computer-assisted manual matching, and the average relative error between them is 4.9%. It is shown that the automatic fitting method has the potential to reach the level of manual interpretation.

6.2. Example 2

Example 2 is a well test data from the Sichuan Basin. The well is a high-pressure gas well, and the reservoir has the characteristics of dual porosity. The permeability is interpreted as 0.23 mD. Due to the extremely low permeability of the reservoir, radial flow characteristics still did not appear after 934 h of shut-in testing. After dimensionless, the agent trained on dual porosity model is used to automatically fit the curve. The agent completed the curve matching by adjusting the parameters 27 times, as shown in Fig. 17. The result of automatic interpretation is $S = 0.86$, $C_D = 2032$, $\omega = 0.13$, $\lambda = 6.4 \times 10^{-7}$. The results show that in the case of insufficient test time, the agent can also fit the curve well. Table 6 compares the results of automatic matching and computer-assisted manual matching, and the average relative error between them is 13.92%. The error in Case 2 is higher than in Case 1, but still acceptable.

Table 7
Case 3 parameter inversion result.

	C_D	S	M	R_i
Manual matching	315.5	7.8	0.23	44.3
Automatic interpretation	300.0	8.0	0.23	40.0
Relative error, %	4.91	2.56	0.00	9.71

6.3. Example 3

Example 3 is a gas well in the Tarim Basin. The well was tested for 72 h of shut-in pressure build-up, and a pressure build-up curve has been obtained. The pressure derivative curve dropped at the end of the test, showing the characteristics of a composite formation. After dimensionless, the agent trained on radio composite model is used to automatically fit the curve. The agent completed the curve matching by gradually adjusting the parameters, as shown in Fig. 18. The result of automatic interpretation is $S = 8$, $C_D = 300$, $M = 0.23$, $R_i = 40$ m. The interpretation results show that the skin of the well is large. It turns out that there is pollution around the well and there is serious sanding problem. 4.3. Table 7 compares the results of automatic matching and computer-assisted manual matching, and the average relative error between

them is 4.30%. The result of automatic interpretation in this case is very close to the result of manual interpretation, which proves the practical value of the proposed method.

7. Conclusions

In this work, we successfully applied DRL to the task of automatically interpreting well test data. In the automatic interpretation process, the agent interacts in an environment encapsulated based on the well test model to learn how to adjust the parameters to match the well test curve. By testing the performance of the DRL algorithms on different well test models, the following key conclusions are drawn:

(1) By making the agent adjust the curve parameters asynchronously, the dimensioning disaster was alleviated, and DDQN algorithm was successfully used in the automatic curve matching task on different well test models. Using DDQN algorithm to perform 100 curve matching tests on three well test models, the results show that the mean relative error of the parameters is 7.58% for the homogeneous model, 10.66% for the radial composite model, and 12.79% for the dual porosity model.
(2) Comparing the performance of Naïve DQN, DQN, DDQN algorithms on the homogeneous model, radial composite model, and dual porosity model, it is shown that the agent based on the DDQN algorithm obtains the highest cumulative reward on these three well test models. In addition, compared with the supervised ML algorithm, DDQN has the least fluctuation of evaluation index on different well test models, which reflects its robustness in curve matching.
(3) The experimental results show that the use of parameter-based reward design can achieve better training results. In addition, Boltzmann exploration policy is more suitable for agents to balance exploration and exploitation on curve matching tasks.
(4) In the three field case tests, the agent completed the curve matching within 30 steps. By visualizing the process of the step-by-step parameter tuning, it was verified that the agent learned the correct strategy.

Acknowledgements

This work received funding support from National Natural Science Foundation of China (52074322), Beijing Natural Science Foundation (3204052), Science Foundation of China University of Petroleum, Beijing (No. 2462018YJRC032), and National Major Project of China (2017ZX05030002-005). The authors sincerely thank the colleagues at State Key Laboratory of Petroleum Resources for their helpful support.

Appendix 1. Dimensionless definitions

Dimensionless bottom hole pressure	$P_{wD} = \frac{kh\Delta p}{1.842 \times 10^{-3} \times q\mu B}$, h is the formation thickness, q is the production rate, B is the volume factor.
Dimensionless wellbore storage	$C_D = \frac{C}{2\pi\phi c_t h r_w^2}$,
Dimensionless time	$t_D = \frac{3.6kt}{\phi\mu c_t r_w^2}$,

References

Adibifard, M., Tabatabaei-Nejad, S., Khodapanah, E., 2014. Artificial neural network (ANN) to estimate reservoir parameters in naturally fractured reservoirs using well test data. J. Petrol. Sci. Eng. 122, 585–594. https://doi.org/10.1016/j.petrol.2014.08.007.
Al-Kaabi, A.U., Lee, W.J., 1990. An artificial neural network approach to identify the well test interpretation model: applications. In: SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers. https://doi.org/10.2118/20552-MS.
AlMaraghi, A.M., El-Banbi, A.H., 2015. Automatic reservoir model identification using artificial neural networks in pressure transient analysis. In: SPE North Africa Technical Conference and Exhibition. Society of Petroleum Engineers. https://doi.org/10.2118/175850-MS.
Arulkumaran, K., Deisenroth, M.P., Brundage, M., Bharath, A.A., 2017. Deep reinforcement learning: a brief survey. IEEE Signal Process. Mag. 34 (6), 26–38. https://doi.org/10.1109/MSP.2017.2743240.
Awotunde, A.A., 2015. Estimation of well test parameters using global optimization techniques. J. Petrol. Sci. Eng. 125, 269–277. https://doi.org/10.1016/j.petrol.2014.11.033.
Bourdet, D., 2002. Well Test Analysis: the Use of Advanced Interpretation Models. Elsevier.
Bourdet, D., Alagoe, A., Pirard, Y.M., 1984. New type curves aid analysis of fissured zone well tests. World Oil 198 (5).
Bui, V., Hussain, A., Kim, H., 2019. Double deep Q-learning-based distributed operation of battery energy storage system considering uncertainties. IEEE Transactions on Smart Grid 11 (1), 457–469. https://doi.org/10.1109/TSG.2019.2924025.
Chen, Z., Liao, X., Sepehrnoori, K., Yu, W., 2018. A semianalytical model for pressure-transient analysis of fractured wells in unconventional plays with arbitrarily distributed discrete fractures. SPE J. 23 (6), 2041–2059. https://doi.org/10.2118/187290-PA.
Chu, W., Shank, G.D., 1993. A new model for a fractured well in a radial, composite reservoir (includes associated papers 27919, 28665 and 29212). SPE Form. Eval. 8 (3), 225–232. https://doi.org/10.2118/20579-PA.
Dastan, A., 2010. A New Look at Nonlinear Regression in Well Testing. Ph.D. Dissertation. Stanford University.
Dastan, A., Horne, R., 2011. Robust well-test interpretation by using nonlinear regression with parameter and data transformations. SPE J. 16 (3), 698–712. https://doi.org/10.2118/132467-PA.
Derhami, V., Majd, V.J., Ahmadabadi, M.N., 2008. Fuzzy Sarsa learning and the proof of existence of its stationary points. Asian J. Contr. 10 (5), 535–549. https://doi.org/10.1002/asjc.54.
Dong, P., Chen, Z., Liao, X., Yu, W., 2021. Application of deep learning on well-test interpretation for identifying pressure behavior and characterizing reservoirs. J. Petrol. Sci. Eng. 109264. https://doi.org/10.1016/j.petrol.2021.109264.
Earlougher, R.C., 1977. Advances in Well Test Analysis, 5. Henry L. Doherty Memorial Fund of AIME, New York.
Gao, Z., Gao, Y., Hu, Y., Jiang, Z., Su, J., 2020. Application of deep Q-network in portfolio management. In: 2020 5th IEEE International Conference on Big Data Analytics (ICBDA). IEEE, pp. 268–275. https://doi.org/10.1109/ICBDA49040.2020.9101333.
Gao, Z., Liu, M., Dang, W., Cai, Q., 2021. A novel complex network-based deep learning method for characterizing gas–liquid two-phase flow. Petrol. Sci. 18 (1), 259–268. https://doi.org/10.1007/s12182-020-00493-3.
Gomez, S., Camacho, R., Vásquez, M., Ramos, G., Castillo, N.D., Mesejo, J., 2014. Well test characterization of naturally fractured vuggy reservoirs, with a global optimization method. Offshore Technology Conference-Asia. In: Offshore Technology Conference. Society of Petroleum Engineers. https://doi.org/10.4043/24762-MS.
Guevara, J.L., Patel, R.G., Trivedi, J.J., 2018. Optimization of steam injection for heavy oil reservoirs using reinforcement learning. In: SPE International Heavy Oil Conference and Exhibition. Society of Petroleum Engineers. https://doi.org/10.2118/193769-MS.
Guyaguler, B., Horne, R.N., Tauzin, E., 2001. Automated reservoir model selection in well test interpretation. In: SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers. https://doi.org/10.2118/71569-MS.
Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., Meger, D., 2018. Deep reinforcement learning that matters. In: Proceedings of the AAAI Conference on Artificial Intelligence. https://ojs.aaai.org/index.php/AAAI/article/view/11694.
Horne, R.N., 1995. Modern Well Test Analysis. Petroway Inc, p. 926.
Hourfar, F., Bidgoly, H.J., Moshiri, B., Salahshoor, K., Elkamel, A., 2019.

A reinforcement learning approach for waterflooding optimization in petroleum reservoirs. Eng. Appl. Artif. Intell. 77, 98–116. https://doi.org/10.1016/j.engappai.2018.09.019.
Huang, W.L., Gao, F., Liao, J.P., Chuai, X.Y., 2021. A deep learning network for estimation of seismic local slopes. Pet. Sci. 18, 92–105. https://doi.org/10.1007/s12182-020-00530-1.
Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. https://arxiv.org/abs/1412.6980.
Laud, A.D., 2004. Theory and Application of Reward Shaping in Reinforcement Learning. University of Illinois at Urbana-Champaign.
Lee, J., Rollins, J.B., Spivey, J.P., 2003. Pressure transient testing (eBook). In: SPE Textbook Series, 9. Society of Petroleum Engineers.
Li, H., Misra, S., 2020. Reinforcement learning based automated history matching for improved hydrocarbon production forecast. Appl. Energy 116311. https://doi.org/10.1016/j.apenergy.2020.116311.
Li, D., Liu, X., Zha, W., Yang, J., Lu, D., 2020. Automatic well test interpretation based on convolutional neural network for a radial composite reservoir. Petrol. Explor. Dev. 47 (3), 623–631. https://doi.org/10.1016/S1876-3804(20)60079-9.
Liu, X., Zhou, L., Chen, X., Li, J., 2020. Lithofacies identification using support vector machine based on local deep multi-kernel learning. Petrol. Sci. 17 (4), 954–966. https://doi.org/10.1007/s12182-020-00474-6.
Mania, H., Guy, A., Recht, B., 2018. Simple random search provides a competitive approach to reinforcement learning arXiv preprint arXiv:1803.07055. https://arxiv.org/abs/1803.07055.
Miftakhov, R., Al-Qasim, A., Efremov, I., 2020. Deep reinforcement learning: reservoir optimization from pixels. In: International Petroleum Technology Conference. https://doi.org/10.2523/IPTC-20151-MS.
Mnih, V., Kavukcuoglu, K., Silver, D., et al., 2013. Playing atari with deep reinforcement learning arXiv preprint arXiv:1312.5602. https://arxiv.org/abs/1312.5602.
Mnih, V., Kavukcuoglu, K., Silver, D., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533. https://doi.org/10.1038/nature14236.
Mohammed, I., Olayiwola, T.O., Alkathim, M., Awotunde, A.A., Alafnan, S.F., 2020. A review of pressure transient analysis in reservoirs with natural fractures, vugs and/or caves. Petrol. Sci. 18, 154–172. https://doi.org/10.1007/s12182-020-00505-2.
Nanba, T., Horne, R.N., 1992. An improved regression algorithm for automated well-test analysis. SPE Form. Eval. 7 (1), 61–69. https://doi.org/10.2118/18161-PA.
Ng, A.Y., Harada, D., Russell, S., 1999. Policy Invariance under Reward Transformations: Theory and Application to Reward Shaping. ICML, pp. 278–287.
Shi, Y., Li, W., Zhu, L., Guo, K., Cambria, E., 2021. Stock trading rule discovery with double deep Q-network. Applied Soft Computing. https://doi.org/10.1016/j.asoc.2021, 107320107320.
Sun, A.Y., 2020. Optimal carbon storage reservoir management through deep reinforcement learning. Appl. Energy 278, 115660. https://doi.org/10.1016/j.apenergy.2020.115660.
Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: an Introduction. MIT press.
Valencia, D., Orejuela, D., Salazar, J., Valencia, J., 2016. Comparison analysis between rigrsure, sqtwolog, heursure and minimaxi techniques using hard and soft thresholding methods. In: 2016 XXI Symposium on Signal Processing, Images and Artificial Vision (STSIVA). IEEE, pp. 1–5. https://doi.org/10.1109/STSIVA.2016.7743309.
van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double Q-learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. https://ojs.aaai.org/index.php/AAAI/article/view/10295.
Watkins, C.J., Dayan, P., 1992. Q-learning. Mach. Learn. 8 (3–4), 279–292.
Wiewiora, E., 2003. Potential-based shaping and Q-value initialization are equivalent. J. Artif. Intell. Res. 19, 205–208. https://doi.org/10.1613/jair.1190.
Yao, Y., Ge, J., 2011. Characteristics of non-Darcy flow in low-permeability reservoirs. Petrol. Sci. 8 (1), 55–62. https://doi.org/10.1007/s12182-011-0115-3.
Zhang, Q., Zhu, S., 2018. Visual interpretability for deep learning: a survey. Frontiers of Information Technology & Electronic Engineering 19 (1), 27–39. https://doi.org/10.1631/FITEE.1700808.
Zhu, L., Zhang, C., Zhang, C., Zhang, Z., Wang, X., 2019. Forming a new small sample deep learning model to predict total organic carbon content by combining unsupervised learning with semisupervised learning. Appl. Soft Comput. 83, 105596. https://doi.org/10.1016/j.asoc.2019.105596.