Original Article

Deep Q-learning to globally optimize a *k*-D parameter search for medical imaging

Hongmei Zhang[1], Songshi Liang[2], Luke A. Matkovic[3], Shadab Momin[3], Kai Wang[4], Xiaofeng Yang[2], Michael F. Insana[4]

[1]Key Laboratory of Biomedical Information Engineering of Ministry of Education, School of Life Science and Technology, Xi'an Jiaotong University, Xi'an, China; [2]Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, China; [3]Department of Radiation Oncology and Winship Cancer Institute, Emory University, Atlanta, GA, USA; [4]Beckman Institute for Advanced Science and Technology, Department of Bioengineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA

Contributions: (I) Conception and design: H Zhang, MF Insana; (II) Administrative support: H Zhang; (III) Provision of study materials or patients: H Zhang; (IV) Collection and assembly of data: S Liang, K Wang; (V) Data analysis and interpretation: H Zhang, S Liang, K Wang; (VI) Manuscript writing: All authors; (VII) Final approval of manuscript: All authors.

Correspondence to: Hongmei Zhang, PhD. Key Laboratory of Biomedical Information Engineering of Ministry of Education, School of Life Science and Technology, Xi'an Jiaotong University, 28 Xianning West Road, Xi'an 710049, China. Email: claramei@mail.xjtu.edu.cn; Michael F. Insana, PhD. Beckman Institute for Advanced Science and Technology, Department of Bioengineering, University of Illinois at Urbana-Champaign, 2108 Everitt Laboratory, 1406 West Green Street, Urbana, IL 61801, USA. Email: mfi@illinois.edu.

Background: Estimation of the global optima of multiple model parameters is valuable for precisely extracting parameters that characterize a physical environment. This is especially useful for imaging purposes, to form reliable, meaningful physical images with good reproducibility. However, it is challenging to avoid different local minima when the objective function is nonconvex. The problem of global searching of multiple parameters was formulated to be a *k*-D move in the parameter space and the parameter updating scheme was converted to be a state-action decision-making problem.

Methods: We proposed a novel Deep Q-learning of Model Parameters (DQMP) method for global optimization which updated the parameter configurations through actions that maximized the Q-value and employed a Deep Reward Network (DRN) designed to learn global reward values from both visible fitting errors and hidden parameter errors. The DRN was constructed with Long Short-Term Memory (LSTM) layers followed by fully connected layers and a rectified linear unit (ReLU) nonlinearity. The depth of the DRN depended on the number of parameters. Through DQMP, the *k*-D parameter search in each step resembled the decision-making of action selections from 3^k configurations in a *k*-D board game.

Results: The DQMP method was evaluated by widely used general functions that can express a variety of experimental data and further validated on imaging applications. The convergence of the proposed DRN was evaluated, which showed that the loss values of six general functions all converged after 12 epochs. The parameters estimated by the DQMP method had relative errors of less than 4% for all cases, whereas the relative errors achieved by Q-learning (QL) and the Least Squares Method (LSM) were 17% and 21%, respectively. Furthermore, the imaging experiments demonstrated that the imaging of the parameters estimated by the proposed DQMP method were the closest to the ground truth simulation images when compared to other methods.

Conclusions: The proposed DQMP method was able to achieve global optima, thus yielding accurate model parameter estimates. DQMP is promising for estimating multiple high-dimensional parameters and can be generalized to global optimization for many other complex nonconvex functions and imaging of physical parameters.

Keywords: Deep Q-learning (QL); nonconvex function; multiple parameters optimization; global optima; *k*-D board move

Submitted Oct 19, 2022. Accepted for publication May 16, 2023. Published online Jun 27, 2023.
doi: 10.21037/qims-22-1147
View this article at: https://dx.doi.org/10.21037/qims-22-1147

Introduction

Model fitting is a branch of nonlinear regression that simultaneously extracts multiple model parameters by fitting experimental data to a specific model. Estimation of multiple model parameters is of great importance in many measurement and imaging applications (1-3). When the fitting function *f* is nonconvex and there are few known constraints, achieving global convergence for parameter estimation is challenging.

Current optimization methods trap suboptimal solutions in local minima because of nonconvexity of the function *f*. There is no general algorithm for solving these problems, and the theoretical guarantees regarding convergence to global optima for common algorithms are weak or nonexistent. The established field of optimization is extensive, consisting of basic methods such as Gauss-Newton and gradient descent (4,5), as well as combinations of those methods, such as Levenberg-Marquardt (6,7). Each of these methods has their strengths and weaknesses.

The simulated annealing algorithm is a classical stochastic scheme for searching global optima by minimizing the system of energy through an annealing schedule (8). Evolutionary methods that imitate biological creatures' behaviors are presented continuously (9,10). Among them, a genetic algorithm is a bio-inspired heuristic search through heredity and mutation (11,12). Particle swarm optimization is another bio-inspired stochastic approach based on the best positions experienced so far by each particle in the whole swarm (13).

In theory, current stochastic and evolutionary schemes can jump out of local extrema and increase the probability of finding global solutions. However, jumping from current local extrema may introduce other local extrema, ultimately yielding inconsistent solutions unless the solutions can be guided by any valid prior knowledge that may be available.

Deep learning (DL) methods have the capacity to learn from prior knowledge. Deep neural networks can establish maps from input to output and may be scaled to model arbitrary mappings. The adaptive and nonlinear responses of deep neural networks can be trained to model highly complex systems. With the successful application of DL in AlphaGo (14,15), the power of DL has been validated in a variety of applications (16-22).

In recent years, DL applied to regression tasks has been reportedly capable of solving multi-parameter optimization and curve fitting problems (21-24). However, learning a large number of model parameters and network weights is a complex optimization problem itself due to its network-like nature. Convergence may be difficult to achieve when applying DL to learn multiple model parameters (21,24).

In human learning, feedback from past activity is important. In a similar vein, Reinforcement Learning (RL) is a powerful, agent-based artificial intelligence (AI) algorithm in which the agents learn the optimal set of actions through their interaction with the environment. RL is able to make appropriate responses because of reinforcing events. These events can include human feedback to responses through rewards and punishments as quantified by a value function. The goal of RL is to take actions that maximize the value function at every step (14,15,24-29). In this way, past experiences can guide RL in learning from new experiences that are still similar to previous ones.

Q-learning (QL) (30) is a model-free, agent-based RL method that can adapt to an environment through utilizing prior knowledge learned from past experiences. The central idea of QL is its Q-value function. The algorithm seeks to be rewarded while also avoiding punishment for its current and next action in the form of an increasing value function. Due to a cumulative feedback mechanism (30), the agent learns to associate the optimal action for each state (31) in pursuit of increasing its value function. QL is widely used in decision-making, gambling, and random event processing problems (32-34). Combinations of DL and RL/QL have been successfully implemented to solve complex human activities, such as AlphaGo for the board game "Go" (14,15). A deep Q-Network can learn from prior human experience and predict the value function through training (14).

The aim of this paper is to develop a novel Deep Q-learning of Model Parameters (DQMP) algorithm that finds a global optimum for estimating multiple model parameters when the objective function is complex and nonconvex. Inspired by RL methods, a novel idea for parameter optimization was first formulated as a decision-making problem for selecting parameter configurations through a reward/punishment mechanism. Furthermore, to combine data with prior knowledge, a Deep Reward Network (DRN) was proposed to learn the global reward function. This process integrated both visible and hidden state feedbacks. Then, a novel DQMP scheme was proposed to maximize the Q-value function. This strategy guides the DQMP search towards the global optimum.

DQMP was validated on functions as Fourier series, exponential series expansion functions, and harmonic signals, all of which are widely used to characterize a variety of signals and experimental data. From the signals and experimental data, model parameters or coefficients that depict the physical phenomenon can be extracted and imaged.

Methods

With given experimental data and fitting models, the goal was to extract the global model parameters. No patient data or animal data were used in this paper. The ideal dataset in this work was generated by function *f* and then degraded by adding different levels of noise to generate the experimental data.

Let Y denote the experimental data. Suppose Y can be modeled by $Y = f(t;\theta)$, where $\theta = [\theta_1, \theta_2, \ldots, \theta_k]$ is the k-dimensional true parameters. Let $\hat{\theta}$ be the estimates of θ. Accordingly, the data predicted by $\hat{\theta}$ is given by $\hat{Y} = f\left(t;\hat{\theta}\right)$. Given the experimental data Y and the specific model expression $f(t;\theta)$, the goal is to estimate θ by solving the optimization problems through fitting the experimental data Y to the model *f*. When $\hat{Y}$ fits to Y closely, $\hat{\theta}$ is assumed to be close to θ. However, when θ is the k-D parameter vector and the data fitting errors are nonconvex, there may be many subsets of $\hat{\theta}$ that satisfy $\hat{Y} \sim Y$. Therefore, parameter fitting should also be included to guide global searching. A new method of the global optimization is proposed by minimizing the following objective function:

$$\hat{\theta} = \underset{\hat{\theta}}{\operatorname{argmin}} \left(\beta_1 \underbrace{\left| Y - \hat{Y} \right|}_{visible} + \beta_2 \underbrace{\left| \theta - \hat{\theta} \right|}_{hidden} \right) \quad [1]$$

$$\text{subject to} \quad Y = f(t;\theta), \hat{Y} = f\left(t;\hat{\theta}\right)$$

where β_1 and β_2 are weights that balance the contributions of the two fitting error terms describing data fittings and model parameters, respectively.

In Eq. [1], the *visible* term involves measurable data that depends on the parameters, whereas the *hidden* term directly evaluates the parameters themselves.

Inspired by QL, a novel idea of updating k-D parameters resembling a chess move in a k-D chess board was proposed. In parameter space, both visible (data fitting) and hidden (parameter fitting) states were given reward/punishment values. For a k-parameter optimization problem, the next action was subdivided into 3^k possible moves in the parameter space, comparable to a chess move in a k-D board game. Each parameter had three possible independent candidate actions, including unchanged (0), move forward (+), and move backward (–). Selection of the next candidate move in parameter space was based on the state of the current model fit, which included both the visible and hidden state feedbacks. In the fitting problem, the state is referred to $s:\left\{\hat{\theta}, f\left(t;\hat{\theta}\right)\right\}$ and includes the visible state $f\left(t;\hat{\theta}\right)$ in which the difference between current data and the desired data is measured by $\left| f(t;\theta) - f\left(t;\hat{\theta}\right) \right|$ and the hidden state $\hat{\theta}$ in which the errors between the current parameter configuration $\hat{\theta}$ and the true parameter configuration θ are measured by $\left| \theta - \hat{\theta} \right|$. In this way, we elegantly converted the parameter optimization to a decision-making problem by minimizing Eq. [1] through a set of state-action decisions in parameter space.

To help understand the new idea, *Figure 1* illustrates the global searching scheme in 3-D parameter space. The current parameter state, illustrated as the yellow dot $\hat{\theta}$, can move in one of 27 possible directions, illustrated as red dots, $\hat{\theta}_{a_1}, \ldots, \hat{\theta}_{a_{27}}$ in the next step by taking corresponding actions $a_1, \ldots, a_{27}$. For each move, a value function will be rewarded. In this way, we formulate k-parameter optimization to

Updating of parameters $\hat{\theta} = \left[\hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3\right]$

Actions a	Updating steps $\Delta\hat{\theta}_a$		
a_1	$+\Delta\hat{\theta}_1$	$+\Delta\hat{\theta}_2$	$+\Delta\hat{\theta}_3$
a_2	$+\Delta\hat{\theta}_1$	$+\Delta\hat{\theta}_2$	0
a_3	$+\Delta\hat{\theta}_1$	$+\Delta\hat{\theta}_2$	$-\Delta\hat{\theta}_3$
a_4	$+\Delta\hat{\theta}_1$	0	$+\Delta\hat{\theta}_3$
a_5	$+\Delta\hat{\theta}_1$	0	0
a_6	$+\Delta\hat{\theta}_1$	0	$-\Delta\hat{\theta}_3$
a_7	$+\Delta\hat{\theta}_1$	$-\Delta\hat{\theta}_2$	$+\Delta\hat{\theta}_3$
a_8	$+\Delta\hat{\theta}_1$	$-\Delta\hat{\theta}_2$	0
a_9	$+\Delta\hat{\theta}_1$	$-\Delta\hat{\theta}_2$	$-\Delta\hat{\theta}_3$
a_{10}	$+\Delta\hat{\theta}_1$	$+\Delta\hat{\theta}_2$	$+\Delta\hat{\theta}_3$
⋮	⋮	⋮	⋮
a_{27}	$-\Delta\hat{\theta}_1$	$-\Delta\hat{\theta}_2$	$-\Delta\hat{\theta}_3$

$$\hat{\theta} \overset{a}{\leftarrow} \hat{\theta} + \Delta\hat{\theta}_a$$

There are n =3^k possible actions at the current state in *k*-D parameter space. For each parameter, it has 3 states:
0 unchanged;
+ move forward;
– move backward.

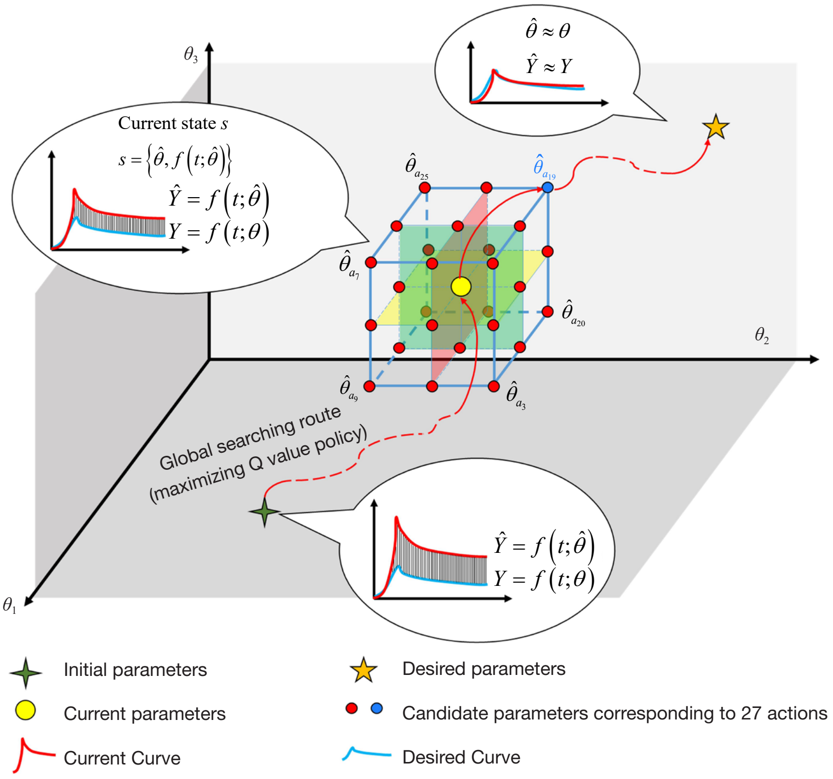

Figure 1 Schematic illustration of a global search of state actions in a 3-D parameter space. The horizontal and vertical axes are the basis of the three parameters θ_1, θ_2 and θ_3. The yellow center point denotes the current parameter configuration $\hat{\theta}$, and the 27 edge points are the candidate actions. The decision to select an action is based on maximizing the Q-value policy. The orange star point denotes the next move through action a_{19} that may maximize the Q-value function. The current curve in every step can be generated by function $\hat{Y} = f\left(t;\hat{\theta}\right)$.

be the decision-making of a *k*-D board game. Therefore, updating $\hat{\theta}$ in the parameter space resembles a chess move selection in the *k*-D board game by maximizing Q-value policy.

The Q-value is central to the QL method. QL is a powerful scheme for agents to learn to act optimally by experiencing the consequences of actions judged by a long-term discounted reward, in which actions are selected to obtain the maximum benefits Q-value.

QL consists of a set of states *S*, a set of actions *A*, and a reward function r:$S \times A \rightarrow R^+$. It uses the Q-value to affect the feedback produced by the actions of every step. The policy π maps states to actions as π: *S*→*A*. The state-action series operates as *s*→*a*→*s'*→*a'*. The Q-value is the expected discounted reward for executing action *a* at state *s* and the next step optimal action *a'* at state *s'* by episodes thereafter. The policy is maximizing Q-value by (30):

$$Q^{(t+1)}(s,a) \leftarrow \underbrace{Q^{(t)}(s,a)}_{cumulative} + \xi\left[\underbrace{r^{(t)}(s,a)}_{immediate} + \gamma \underbrace{\max_{a'} Q^{(t)}(s',a')}_{future} - Q^{(t)}(s,a)\right] \quad [2]$$

where $Q^{(t)}(s,a)$ is the cumulative reward, $r^{(t)}(s,a)$ is the immediate reward, and $Q^{(t)}(s',a')$ is the future reward.

$r(s,a)$ is the immediate reward of selecting action *a* at distinct state *s*. The reward can be any positive value for an action. The reward function should be a decreasing function of the fitting errors in the fitting problems. $Q(s',a')$ is the Q-value found by selecting the next state-action pairs (s',a'). $\xi \in [0,1]$ is the learning rate and $\gamma \in [0,1]$ is the discount factor. The recommended value of ξ is 0.6 and of *r* is 0.5.

For global optimization of (1), the optimal action was taken to increase a value function as $\hat{\theta} \rightarrow \theta$. Then, the global parameter search was formulated to be a state-action decision-making problem in the parameter space by increasing the Q-value function policy. The parameter update was formulated to be $a = \underset{a_i}{\text{argmax}}\, Q(s, a_i)$ at every step, and parameters were updated by $\hat{\theta} \overset{a}{\leftarrow} \hat{\theta} + \Delta\widehat{\theta_a}$ as indicated in the left table of *Figure 1*. $\Delta\widehat{\theta_a}$ are adaptive steps. In the experiment, we set $\Delta\widehat{\theta_a} \approx 0.01\hat{\theta}$.

The Q-value has a crucial role in QL. To guide the global parameter search, the Q-value function should integrate both the data fitting (visible) and parameter fitting (hidden) feedbacks. As indicated in Eq. [1], data fitting errors can be calculated directly, but the hidden parameter fittings are unknown and should be learned.

In order to learn the prior rewards from hidden states, a DRN was proposed to learn the reward function whose global constraints are absorbed from both visible and hidden states. In this way, a novel DQMP algorithm was proposed, where a DRN was proposed to predict global reward values comprising both the data (visible) and parameter fitting (hidden) feedbacks. DQMP iteratively updated the state through convergence such that a global solution could be found following the maximizing Q-value policy.

DQMP

The Q-value is a weighted sum of the immediate reward, cumulative reward, and future reward. The learning of the reward function *r*(*s*,*a*) that rewards both visible and hidden state feedbacks is crucial to guide the global search. Let R_d denote the data fitting reward (visible) and R_θ the parameter fitting reward (hidden). The reward function *r*(*s*,*a*) was formulated as Eq. [3], where the global reward R_g combines both the hidden reward R_θ and visible reward R_d, as expressed by Eqs. [4-1], [4-2] and [4-3].

$$\mathrm{r}(\mathrm{s},\mathrm{a}) = \beta_\mathrm{g}\mathrm{R}_\mathrm{g} + (1-\beta_\mathrm{g})\mathrm{R}_\mathrm{d} \qquad [3]$$

$$\mathrm{R}_\mathrm{g} = \beta_\theta \mathrm{R}_\theta + \beta_\mathrm{d}\mathrm{R}_\mathrm{d} \qquad [4\text{-}1]$$

$$\mathrm{R}_\theta = 1 - \frac{\left\|\hat{\theta}-\theta\right\|_2}{\sqrt{\mathrm{k}}} \qquad [4\text{-}2]$$

$$\mathrm{R}_\mathrm{d} = \begin{cases} 0, & \overline{|\Delta|} > \mathrm{e}_{\max} \\ g\left(\overline{|\Delta|}\right), & \mathrm{e}_{\min} \le \overline{|\Delta|} \le \mathrm{e}_{\max} \\ 1, & \overline{|\Delta|} < \mathrm{e}_{\min} \end{cases} \qquad [4\text{-}3]$$

$\beta_g \in [0,1]$ is the weight to balance the global reward and the data fitting reward. $\beta_\theta, \beta_d \in [0,1]$ are adjustable weights for R_θ and R_d, respectively. *k* is the number of parameters. If $\hat{\theta}$ and θ are far apart, R_θ is negative to act as punishment. $g(\cdot)$ is a decreasing function, e_{min} and e_{max} are the two thresholds such that $g(e_{min})$ =1 and $g(e_{max})$ =0, $g\left(\overline{|\Delta|}\right) = \left(\log_{10}\left(\overline{|\Delta|}\right)\right)^2 / C$, and *C* is a normalized constant to make $g(\cdot)$ within [0,1] and is recommended to be 100. Let $Y = [y(1), y(2), \ldots, y(m)]$, $\hat{Y} = [\hat{y}(1), \hat{y}(2), \ldots, \hat{y}(m)]$, then $\overline{|\Delta|} = \frac{1}{m}\sum_{i=1}^{m}\left|\hat{y}(i) - y(i)\right|$ is the sample mean value of the Mean Absolute Error (MAE) between the current data and desired data. The recommended value of β_g is 0.02, β_θ is 0.6, β_d is 0.4, e_{min} is 10^{-10}, and e_{max} is 1.

Learn the hidden feedbacks via the DRN

The DRN was designed to predict R_g, which consisted of rewards from both visible and hidden states. The schematic illustration of DRN is shown in *Figure 2*.

The structure of the DRN is shown on the left of *Figure 2*. A Long Short-Term Memory (LSTM) neural network was used to construct the DRN. LSTM is a special Recurrent Neural Network (RNN) that is appropriate for dealing with sequence data modeling. Compared to an ordinary RNN, LSTM architecture is better at dealing with long time sequence data, allows for unlimited state numbers, and avoids problems related to vanishing and exploding gradients (35). The input of the DRN was the difference between the current data and desired data $\left(\Delta(s) = \hat{Y} - Y\right)$, followed by several LSTM layers and fully connected layers. All of the hidden layers are followed by rectified linear unit (ReLU) nonlinearity. The outputs of the DRN were the global rewards $\left\{\hat{R}_{g1}, \ldots, \hat{R}_{gn}\right\}\left(n = 3^k\right)$ for each action. The depth of the DRN depends on the number of parameters.

To prevent overfitting, the dropout rate was designed such that it increased with the depth of network. Moreover, a batch norm was added before the fully connected layer to normalize the diverse parameters and was then followed by a ReLU nonlinearity. The learning rate increased with the depth of the network. The recommend dropout rate was 0.1–0.3 and the learning rate was 5×10^{-4}–1×10^{-3}.

The loss function L_{Reward} of the DRN was given by:

$$L_{Reward}\left(\boldsymbol{R_g}, \boldsymbol{\hat{R}_g}\right) = \sum_{i=1}^{n}\left|R_{gi} - \hat{R}_{gi}\right| \qquad [5]$$

The generation of the training data set is displayed on the right of *Figure 2*. First, two parameter configurations, true parameter θ and current parameter $\hat{\theta}$, were randomly

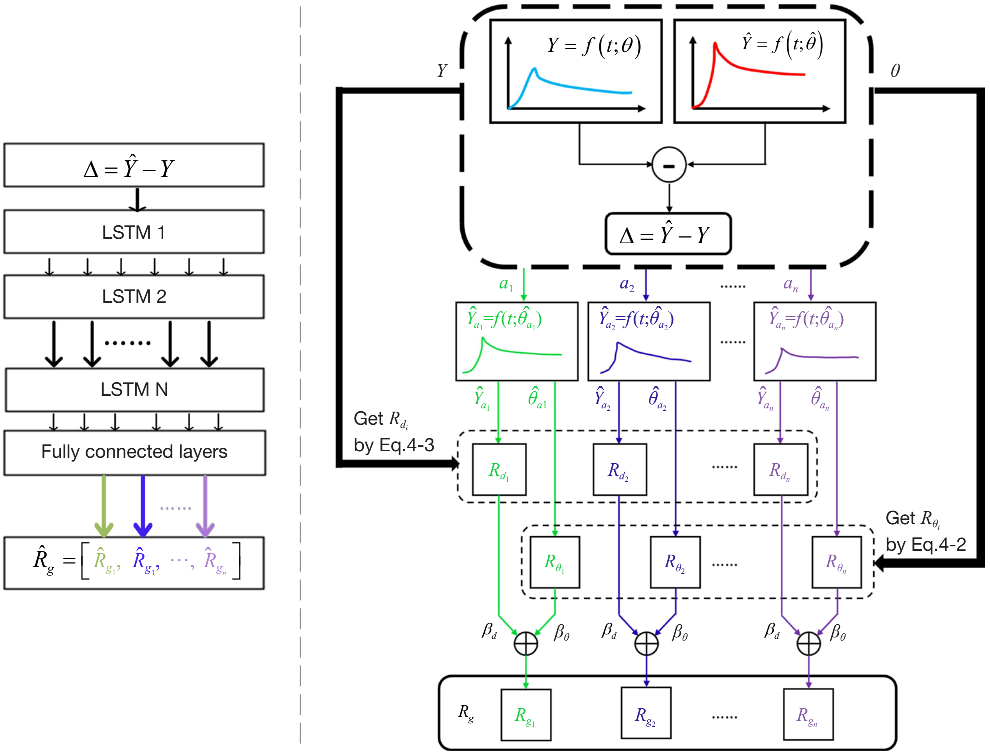

Figure 2 Schematic illustration of the DRN to learn and predict the global reward. The right figure shows the flow of generating training data and the left figure shows the structure of DRN, which has several LSTM layers followed by fully connected layers. All of the hidden layers are followed by ReLU nonlinearity. DRN, Deep Reward Network; LSTM, Long Short-Term Memory; ReLU, rectified linear unit.

generated in the parameter space. Second, the two sets of data, generated by θ and $\hat{\theta}$ according to function *f* in Eq. [1], exactly simulated the desired experimental data $Y = f(t;\theta)$ and the immediate data $\hat{Y} = f(t;\hat{\theta})$, respectively. Then the difference $\Delta(s) = \hat{Y} - Y$ was input to the DRN. Next, the current parameters $\hat{\theta}$ performed actions $a_i\,(i=1,\ldots,n)$ to achieve *n* candidate parameter configurations $\hat{\theta}_{a_i}\,(i=1,\ldots,n)$. With that, the corresponding immediate data was generated by $f(t;\hat{\theta}_{a_i})$. Then, for each candidate action a_i, using Eqs. [4-2] and [4-3], one can calculate R_d, R_θ, and R_g. The map from the input Δ (*s*) and output $R_g(s,a_i),(i=1,\ldots,n)$ was set up by the DRN as illustrated on the left of *Figure 2*. As R_g includes both the curve fitting reward R_d and parameter fitting reward R_θ, the DRN can predict the global reward. In this way, the method to reward the current fitting by doing actions $a_i\,(i=1,\ldots,n)$ was recorded in global rewards $R_g(s,a_i)$. In total, 1,000,000 pair-wise parameters θ and $\hat{\theta}$ were generated in the training set. We randomly split the datasets, with 80% for training, 10% for validation, and 10% for testing sets. After training, the DRN can predict the global reward R_g when provided with the current and desired experimental data in every immediate fitting step.

Q-value integrating rewards from both hidden and visible states

Applying Eqs. [3], [4] to [2], a novel Deep QL method was proposed which updated the Q-value as expressed by:

$$Q^{(t+1)}(s,a) \leftarrow Q^{(t)}(s,a) + \xi\left[\underbrace{\beta_g\underbrace{(\beta_\theta R_\theta + \beta_d R_d)}_{R_g} + (1-\beta_g)R_d}_{r^{(t)}(s,a)} + \gamma \max\nolimits_{a'} Q^{(t)}(s',a') - Q^{(t)}(s,a)\right] \qquad [6]$$

The idea of QL was similar to decisions in a Chess or Go game. Before one moves by choosing a candidate action *a*, one must consider the value function it brings to the current and all possible candidate next steps. In the immediate fitting environment, the current state *s* is the immediate curve denoted by $s:\hat{Y}=f\left(t;\hat{\theta}\right)$ and the next state *s′* is obtained by taking action *a* from the immediate curve $\hat{Y}$. That is, the next step curve $s':\hat{Y}'=f\left(t;\widehat{\theta'}\right)$. Whereby $\widehat{\theta'}\overset{a}{\leftarrow}\hat{\theta}+\Delta\hat{\theta}_a$, $s'\overset{a}{\leftarrow}s$ is obtained. *Q*(*s′*,*a′*) is the value function for the next state-action pairs (*s′*,*a′*).

The global reward R_g for every state-action pair can be learned and predicted by the DRN. In this way, the Q-value integrated rewards from both hidden and visible states as R_g contains both data fitting and parameter fitting rewards.

DQMP algorithm

A schematic illustration of DQMP is shown in *Figure 3*. $\hat{\theta}$ denotes the current estimates. Given the immediate data determined by $f\left(t;\hat{\theta}\right)$ and the desired data *Y*, the global reward value R_g was predicted by the DRN. The global search of parameter $\hat{\theta}$ was conducted via parameter updating through maximizing the Q-value policy. In each step, action *a* was selected by $a:a\leftarrow\underset{a'}{\operatorname{argmax}}Q\left(s,a'\right)$, and $\hat{\theta}$ was updated by $\hat{\theta}\overset{a}{\leftarrow}\hat{\theta}+\Delta\hat{\theta}_a$.

Ideally, Q(*s′*,*a′*) should be maximized instead of $r^{(t)}(s',a')$. However, global fitting is possibly an infinite state, given that (*s′*,*a′*) has been visited previously. So, Q(*s′*,*a′*) may be taken from the Q-table. However, if (*s′*,*a′*) is visited for the first time, computation of Q(*s′*,*a′*) will result in a recursive process. To improve the computation efficiency, $r^{(t)}(s',a')$ was optimized instead of Q(*s′*,*a′*). As the Q-value is inherently the cumulative reward function, the degradation is reasonable.

The pseudo-code of DQMP is provided in Algorithm 1. The algorithm iteration stops until the curve fitting error $\overline{|\Delta|}$ is small enough or the maximum number of iterations has been reached.

Algorithm 1 Algorithmic flow of Deep Q-Learning of Model Parameters (DQMP). $\hat{\theta}=\mathrm{DQMP}\left(Y,k\right)$

```
Input:
    Y - Experimental data; $Y=f\left(t;\theta\right)$; f is the mathematical modeling function.
    k - The number of model parameters to be estimated.
    where $\theta$ are true k-D parameters
Output:
    $\hat{\theta}$ – Estimated global optimal k-D parameters approaching the global optimal solution $\theta$
j =1
Initialize Q-table
Initial guess of $\theta:\hat{\theta}^{(1)}$ // $\hat{\theta}^{(1)}$ can be conducted by any fitting algorithm
while (-convergence)
    $\hat{Y}^{(j)}=f\left(t;\hat{\theta}^{(j)}\right)$
    $R_g=\mathrm{DRN}\left(\hat{Y}^{(j)}-Y\right)$ // DRN is Deep Reward Network, see section Learn the hidden feedbacks via the Deep Reward Network, $R_g$ is a vector
    for all actions a:(a=a1,…,$a_{3k}$)
        $R_{d,a}\leftarrow g\left(\overline{|\Delta|}\right)$ // $\overline{|\Delta|}=\frac{1}{m}\left|\hat{Y}_a^{(j)}-Y\right|$, $\hat{Y}_a^{(j)}$ is the estimation of Y when selecting action a
        $r\left(s,a\right)\leftarrow\beta_g R_{g,a}+\left(1-\beta_g\right)R_{d,a}$ // $R_{g,a}$ is the global rewards corresponding to action a
        call $\mathrm{DRN}\left(\hat{Y}_a^{(j)}-Y\right)$ to predict $R_{g,a'}$
        for all next actions $a'$:$\left(a'=a_1',\ldots,a_{3^k}'\right)$
            $r\left(s',a'\right)\leftarrow\beta_g R_{g,a'}+\left(1-\beta_g\right)R_{d,a'}$
        end
        $Q^{(j)}\left(s,a\right)\leftarrow Q^{(j-1)}\left(s,a\right)+\xi\left[r^{(j-1)}\left(s,a\right)+\gamma\max_{a'}Q^{(j-1)}\left(s',a'\right)-Q^{(j-1)}\left(s,a\right)\right]$
    end
    choose action $a:a\leftarrow\mathrm{argmax}_a\,Q\left(s,a\right)$
    $\hat{\theta}^{(j+1)}\leftarrow\hat{\theta}^{(j)}+\Delta\hat{\theta}_{\boldsymbol{a}}$
    $j\leftarrow j+1$
end
return $\hat{\theta}=\hat{\theta}^{(j)}$
```

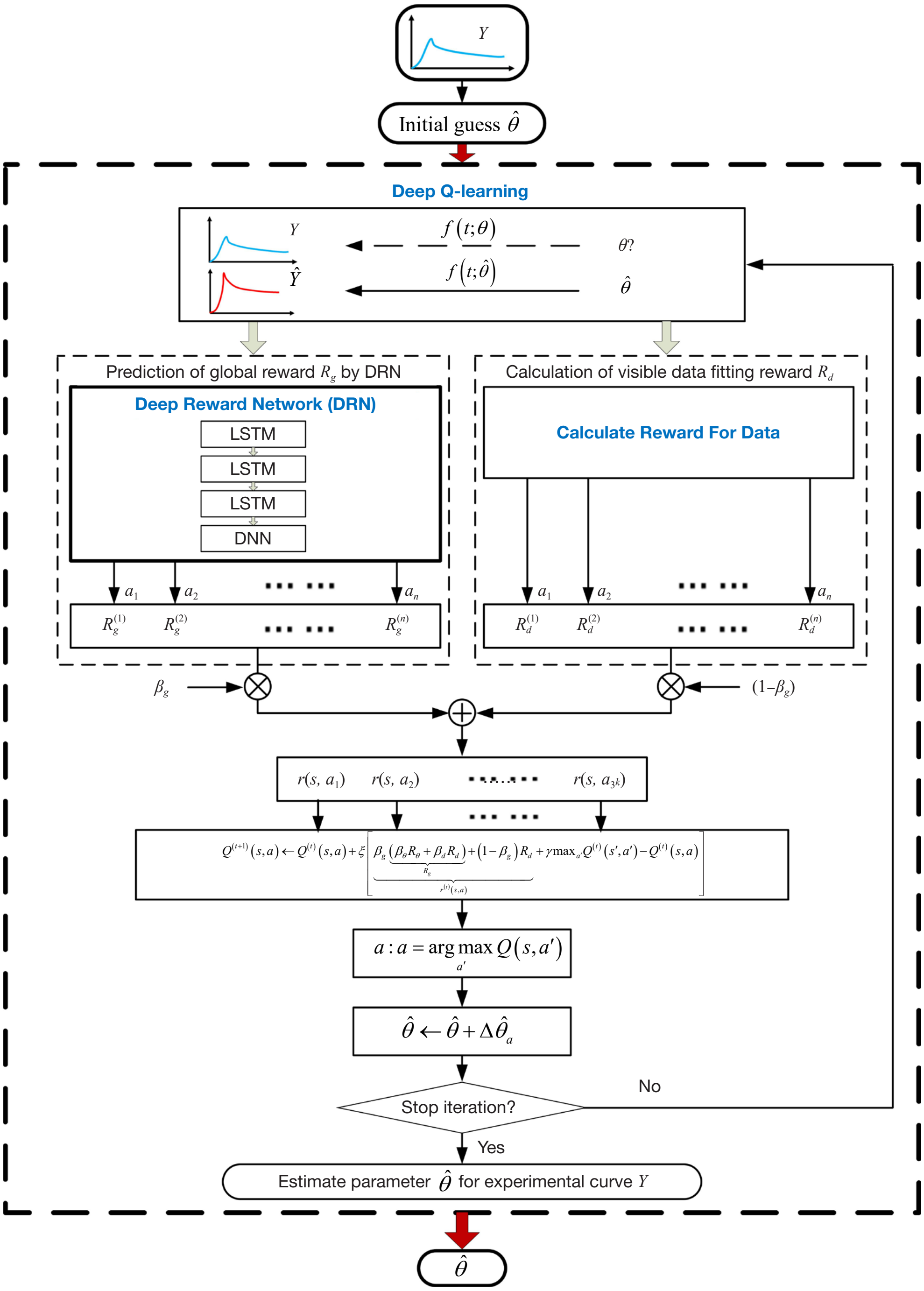
Y
Initial guess $\hat{\theta}$
Deep Q-learning
Y
$\hat{Y}$
$f(t;\theta)$
θ?
$f(t;\hat{\theta})$
$\hat{\theta}$
Prediction of global reward R_g by DRN
Deep Reward Network (DRN)
LSTM
LSTM
LSTM
DNN
a_1 a_2 a_n
$R_g^{(1)}$ $R_g^{(2)}$ $R_g^{(n)}$
Calculation of visible data fitting reward R_d
Calculate Reward For Data
a_1 a_2 a_n
$R_d^{(1)}$ $R_d^{(2)}$ $R_d^{(n)}$
β_g
$(1-\beta_g)$
$r(s, a_1)$ $r(s, a_2)$ $r(s, a_{3k})$
$Q^{(t+1)}(s,a) \leftarrow Q^{(t)}(s,a) + \xi \left[\underbrace{\beta_g \underbrace{(\beta_\theta R_\theta + \beta_d R_d)}_{R_g} + (1-\beta_g) R_d}_{r^{(t)}(s,a)} + \gamma \max_{a'} Q^{(t)}(s',a') - Q^{(t)}(s,a) \right]$
$a : a = \arg\max_{a'} Q(s,a')$
$\hat{\theta} \leftarrow \hat{\theta} + \Delta\hat{\theta}_a$
Stop iteration?
No
Yes
Estimate parameter $\hat{\theta}$ for experimental curve Y
$\hat{\theta}$

Figure 3 Schematic illustration of the DQMP algorithm, where DRN is used to predict global rewards R_g. DQMP, Deep Q-Learning of Model Parameters; DRN, Deep Reward Network.

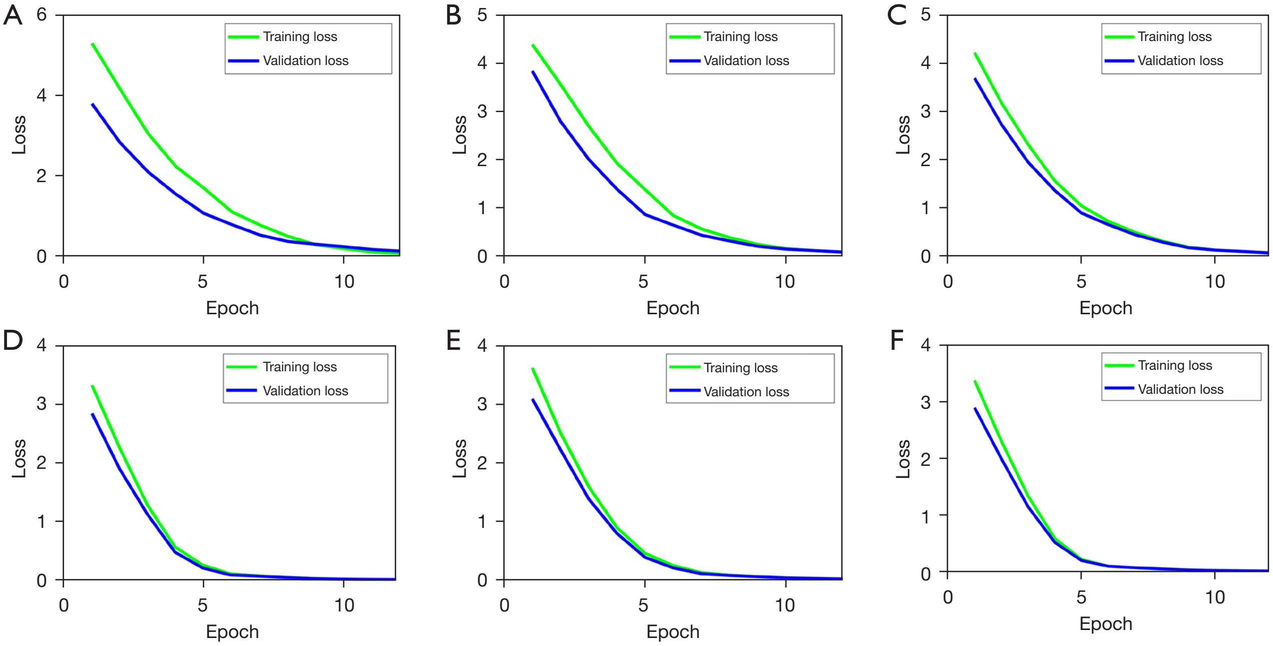

Figure 4 The convergence of the Deep Reward Network evaluation. The loss functions of the training data and validation data for f_1–f_6 are provided in (A-F). (A) Training and validation loss for Fourier function. (B,C) Training and validation loss for Exponential function. (D) Training and validation loss for Relaxation function. (E) Training and validation loss for Creep function. (F) Training and validation loss for Harmonic equation function.

Results

First, the convergence of the proposed DRN was evaluated. *Figure 4* provided the loss function of the DRN training. In this work, the DRN was designed with 5–7 LSTM layers and 4 fully connected layers, for parameter numbers of 3, 5, and 6, respectively.

It can be seen that f_1 converged slowly due to the maximum parameters to be trained, while f_4, f_5, and f_6 converged faster due to fewer training parameters. In general, the loss values of the six functions all converged after 12 epochs. The dropout rate was designed to increase with the depth of network. For the first three layers, the dropout rate was set as 0.1. For Layers 4 and 5, the dropout rate was set as 0.2, and for Layers 6 and 7, the dropout rate was set as 0.3. The learning rate was larger with the depth of the network. Specifically, the learning rates for training f_1–f_6 were set to 1×10^{-3}, 8.8×10^{-4}, 8.8×10^{-4}, 6.5×10^{-4}, 6.5×10^{-4}, and 5×10^{-4}. respectively. From here, the proposed DQMP method was used to estimate model parameters of general functions.

k-D parameter search evaluation on several general functions

Fourier series, exponential series expressions, Boltzmann integral expressions, and harmonic signals are representative forms that are widely used to characterize a variety of signals and physical behaviors of matter. From these representative forms, model parameters that depict the physical phenomena can be extracted and imaged.

The goal was to estimate the model parameters, or coefficients a_k and b_k by using the global optimizer through fitting model functions to experimental data.

The simulation data was generated with the parameter θ by six functions, f_1 to f_6. We demonstrated the DQMP using the above functions as provided in *Table 1*. The DQMP was also compared with QL and the Least Squares Method (LSM).

The representative curve fitting is provided in *Figure 5*. The blue circles denote the representative experimental data, whereas the corresponding fitting data predicted by the estimated parameters are shown by red lines. As shown in *Figure 5*, all three fitting methods can fit the data with

R^2>0.98. However, the fitting parameters were different among the three methods. DQMP can closely approach the global true parameters. This can be further confirmed by the statistical analysis on the fitting of 500 randomly generated data curves, in which the random Gaussian noise with a noise level of 1% (variance=1%, maximum value of |*f*|) was added to the data. The parameters estimated by the three methods are provided in *Table 2*. For each fitting method, the mean relative errors for all parameters from six functions, specifically from 500 randomly generated curves' data for each of the six functions, were calculated. The statistical analysis on fitting errors was conducted. As shown in *Figure 6*, parameters estimated by DQMP had the smallest relative errors. However, QL with only data fitting error constraints, i.e., no parameter constraints were introduced and R_g=0, performed worse than DQMP. Similarly, the LSM algorithm for extracting parameters showed the largest deviation to the ideal values which performed worse than the other two methods. All in all, the proposed DQMP method not only yielded a precise fit between the simulated data and the prediction curves across all functions, but also yielded precise fitting parameters whose relative errors were about 4% for the worst case, whereas the relative errors of the fitting parameters by QL and LSM were about 17% and 21%, respectively. Overall, the fitting parameters by the DQMP were the best approach to global solutions, as the parameters were the most closest to true ones.

k-D parameter search evaluation on imaging applications

The following cases provide the simulation imaging on f_4 and f_5 functions, where the parameters (a_1,a_2,a_3) are denoted by [E_0,α,τ], respectively.

The simulation image was generated with four sets of parameters [20000, 0.7, 800], [40000, 0.5, 600], [60000, 0.3, 400] and [80000, 0.1, 200] by f_4 and [2000, 0.7, 80], [4000, 0.5, 40], [6000, 0.3, 60] and [8000, 0.1, 20] by f_5 in the four 8×8 sub-region. As shown in *Figure* 7 and *Figure 8*, the corresponding four ideal curves were generated by f_4 and f_5 and were further degraded by adding random Gaussian noise (variance =10^{-6}) to simulate the 256 experimental noisy curves.

As shown in *Figure* 7 and *Figure 8*, the first line provides the ideal curve and images. The 2nd–4th lines provide the *k*-D parameter search imaging by the proposed DQMP, QL, and LSM algorithms. The representative fitting of the noisy data was shown in the first column. The estimated parameters were imaged as shown in the 2nd–4th column. From *Figure 7B* and *Figure 8B*, all 256 noisy curves were fitted with R^2≥0.97, and we can see the searched parameters were close to the ideal values and robust to Gaussian noise. The images of elastic modulus E_0 and fluidity α were almost uniform. The viscosity image of τ had slight noise fluctuation. All three imaged parameters can reflect the true parameters well. The *k*-D parameter search evaluation on imaging applications confirmed the accuracy and robustness of the proposed DQMP, indicating its potential of finding parameters close to the global solutions.

Discussion

The efficiency of the proposed DQMP algorithm was demonstrated by the curve fitting of general functions f_1–f_6 (*Figure 5*) and simulation imaging (*Figure* 7, *Figure 8*). The convergence of the algorithm was evaluated in *Table 2* and *Figure 6*, which showed that the parameters estimated by the DQMP algorithm were the closest to the global solutions for all cases when compared to other methods.

Yet, there are several issues that can be improved. First, the current DRN is simple. For many parameters, a more complex deep network may be needed to train the global reward function in a high dimensional parameter space. Second, the range of the parameters should be covered in the training process, otherwise the convergence of the DRN may be poor for the validation data. The fitting problem is usually in an engineering context, so we can more precisely cover the realistic range of fitting parameters from previous experiences when approaching a similar problem. Moreover, β_θ, β_d and β_g are weights to balance the contribution from the parameter fitting rewards, data fitting rewards, and DRN, respectively. In this work, recommendations for these parameters were provided through a combination of experience and trial-and-error. These considerations for parameter ranges may add to the versatility of the algorithm but may also bias the results.

While the proposed DQMP method was tested in general functions in this study, the proposed frameworks can be generalized to global optimization for many other complex, nonconvex functions. It can also be used in

Table 1 General functions with multiple parameters

Function	Function expression	Function description
Fourier series	$f_1(t)=\sum_{k=1}^{N}a_k\cos(2k\pi t+b_k),\quad t\in[0,2], N=3$	The Fourier series is widely used in signal processing and signal analysis. Through Fourier transform, any signal satisfying the Dirichlet conditions can be approximately expressed in multiple Fourier series form
Exponential series	$f_2(t)=a_0+\sum_{k=1}^{N}a_k\exp(-t/b_k),\quad t\in[2,50], N=2$ $f_3(t)=a_0-\sum_{k=1}^{N}a_k\exp(-t/b_k),\quad t\in[2,50], N=2$	Exponential functions are widely used to describe a time-dependent viscoelastic behavior, so as to obtain the mechanical parameters of the tested substance
Boltzmann hereditary integral operators	$f_4(t)=\int_{-\infty}^{t}G(t-u)\frac{d\varepsilon(t)}{du}du,\quad t\in[0,5]$ Where $G(t)=a_1\left[1+\frac{(t/a_3)^{-a_2}}{\Gamma(1-a_2)}\right]$ $\Gamma(x)=\int_{0}^{+\infty}u^{x-1}e^{-u}du,\quad x\in(0,+\infty)$ And loading $\varepsilon(t)$ is set as: $\varepsilon(t)=\begin{cases}2.5\times10^{-4}\cdot t, & 0\le t<2\\ 5\times10^{-4}, & 2\le t\le 5\end{cases}$ $f_5(t)=\int_{-\infty}^{t}J(t-u)\frac{d\sigma(t)}{du}du,\quad t\in[0,5]$ Where $J(t)=\frac{1}{a_1}\left[1-\mathbf{E}_{a_2,1}\left(-\left(\frac{t}{a_3}\right)^{a_2}\right)\right]$ $\mathbf{E}_{\beta_1,\beta_2}(z)=\sum_{k=0}^{\infty}\frac{z^k}{\Gamma(k\beta_1+\beta_2)},\quad \beta_1,\beta_2\in(0,+\infty)$ $\Gamma(x)=\int_{0}^{+\infty}u^{x-1}e^{-u}du,\quad x\in(0,+\infty)$ And loading $\sigma(t)$ is set as: $\sigma(t)=\begin{cases}2.5\times10^{-4}\cdot t, & 0\le t<2\\ 5\times10^{-4}, & 2\le t\le 5\end{cases}$	The Boltzmann integral is a superposition principle of which can express the physical behaviors of soft matter under different excitation
Harmonic equation	$u(t)=f_6(t)=u_c\cos(\lambda t)+u_s\sin(\lambda t),\quad t\in[0,1]$ Where $u_c=q_c\varphi_1+q_s\varphi_2$ $u_s=-q_c\varphi_2+q_s\varphi_1$ $\varphi_1=a_1+a_3\lambda^{a_2}\cos(a_2\pi/2)$ $\varphi_2=a_3\lambda^{a_2}\sin(a_2\pi/2)$ $q_c=1, q_s=1, \lambda=2\pi$ And loading $q(t)$ is set as: $q(t)=q_c\cos(\lambda t)+q_s\sin(\lambda t),\quad t\in[0,1]$	Harmonics signal are used to describe a line-elastic or viscoelastic behavior of the soft matter, when the input signal and the output signal are both harmonic signals

a_k, b_k denote the multiple parameters to be extracted when fitting experimental data to functions.

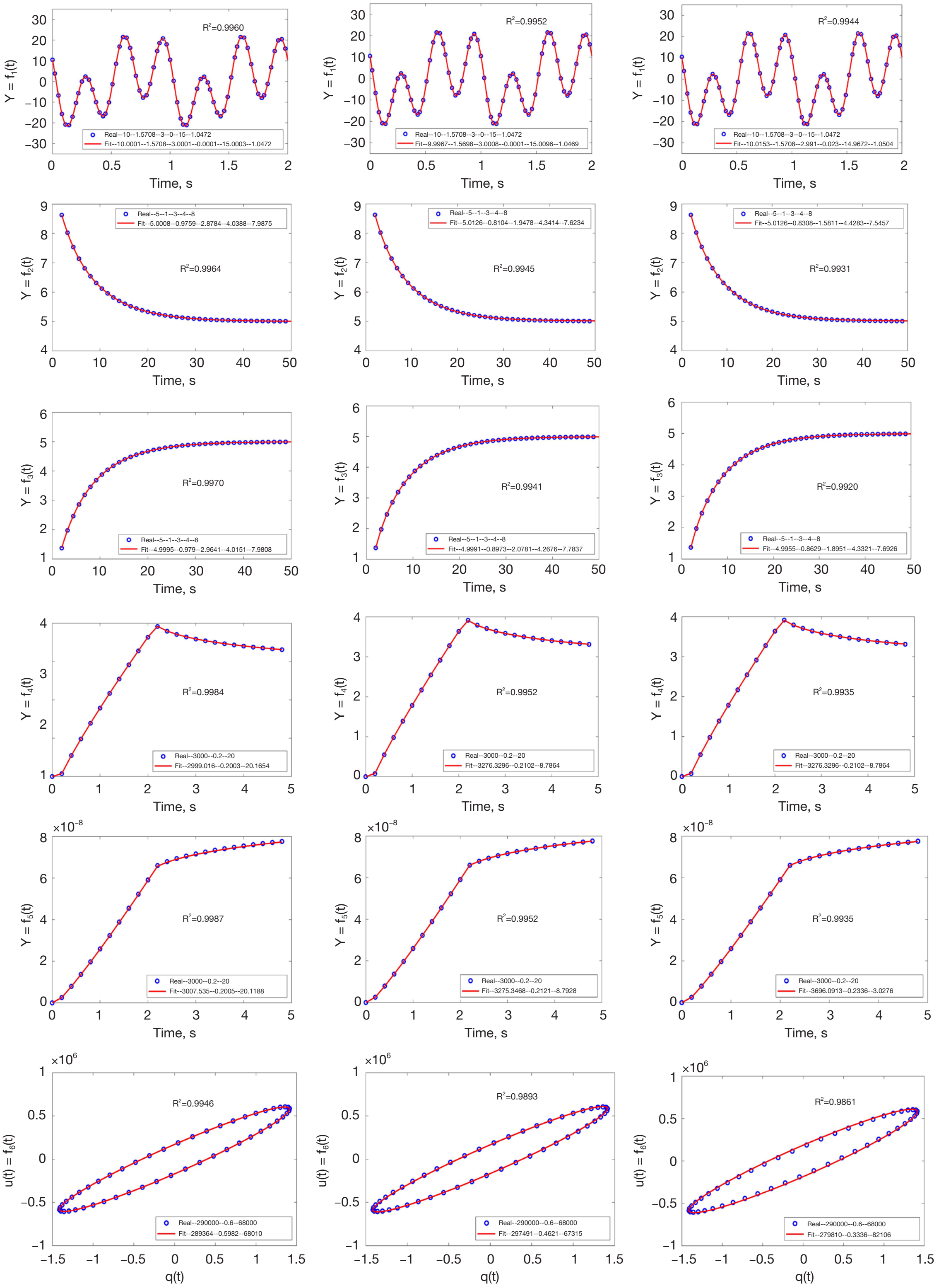

Figure 5 Comparison of fittings of the simulation data generated with parameters θ by six functions. The curves in 1st–3rd columns are the fitting results by the proposed DQMP, QL, and LSM, respectively. The simulated data are drawn with blue circles and the predicted fitting data are drawn with red lines. DQMP, Deep-Q Learning of Model Parameters; QL, Q-Learning; LSM, Least Squared Method.

Table 2 Estimated parameters by different fitting algorithms, each based on 500 randomly generated noisy curves

Estimated parameters	DQMP	QL	LSM
f_1			
a_1=10	10.0005±0.0572	10.0024±0.0273	10.0025±0.0321
b_1=1.5708	1.5708±0.0005	1.5707±0.0028	1.5707±0.0029
a_2=3	3.0003±0.005	3.0014±0.0267	3.0016±0.0268
b_2=2	2.0003±0.00010	2.0034±0.00023	2.0034 ±0.00025
a_3=15	14.9995±0.0032	14.9992±0.0262	14.9993±0.0269
b_3=1.0472	1.0472±0.0001	1.0473±0.0019	1.0473±0.0023
f_2			
a_0=5	5.0000±0.0035	5.0004±0.0095	4.9992±0.0100
a_1=1	1.0145±0.0379	1.0291±0.1319	1.0571±0.2367
b_1=3	3.0318±0.1533	3.0923±0.5176	3.1038±0.6085
a_2=4	4.0017±0.0152	3.9840±0.1464	3.9586±0.1927
b_2=8	8.0001±0.0961	8.0109±0.2234	8.0360±0.2482
f_3			
a_0=5	5.0001±0.0024	5.0003±0.0055	5.0007±0.0072
a_1=1	1.0038±0.0873	1.0241±0.1256	1.0596±0.2108
b_1=3	3.0142±0.2423	3.0441±0.3028	3.0846±0.5147
a_2=4	3.9979±0.0164	3.9824±0.0891	3.9475±0.2191
b_2=8	8.0002±0.0622	8.0155±0.1385	8.0469±0.2402
f_4			
a_1=3,000	3,000.0139±1.9903	3,000.6130±43.6399	3,004.1580±77.6099
a_2=0.2	0.2000±0.0101	0.1997±0.0151	0.1997±0.0172
a_3=20	20.0871±0.3967	20.3162±3.0011	20.5077±4.9772
f_5			
a_1=3,000	3,001.4350±2.4677	2,998.2520±44.1450	3,002.9460±80.4765
a_2=0.2	0.2001±0.0149	0.2008±0.0190	0.2007±0.0191
a_3=20	20.0910±0.2913	20.2999±3.0090	20.4573±5.1680
f_6			
a_1=290,000	299,958.04±172.66	289,975.92±864.23	289,970.78±1,470.50
a_2=0.6	0.5965±0.0189	0.5813±0.0423	0.5673±0.0804
a_3=68,000	68,104.37±233.18	68,401.24±518.50	69,414.44±1,824.72

Values are shown as mean ± standard deviation. DQMP, Deep-Q Learning of Model Parameters; QL, Q-Learning; LSM, Least Squared Method; f, function.

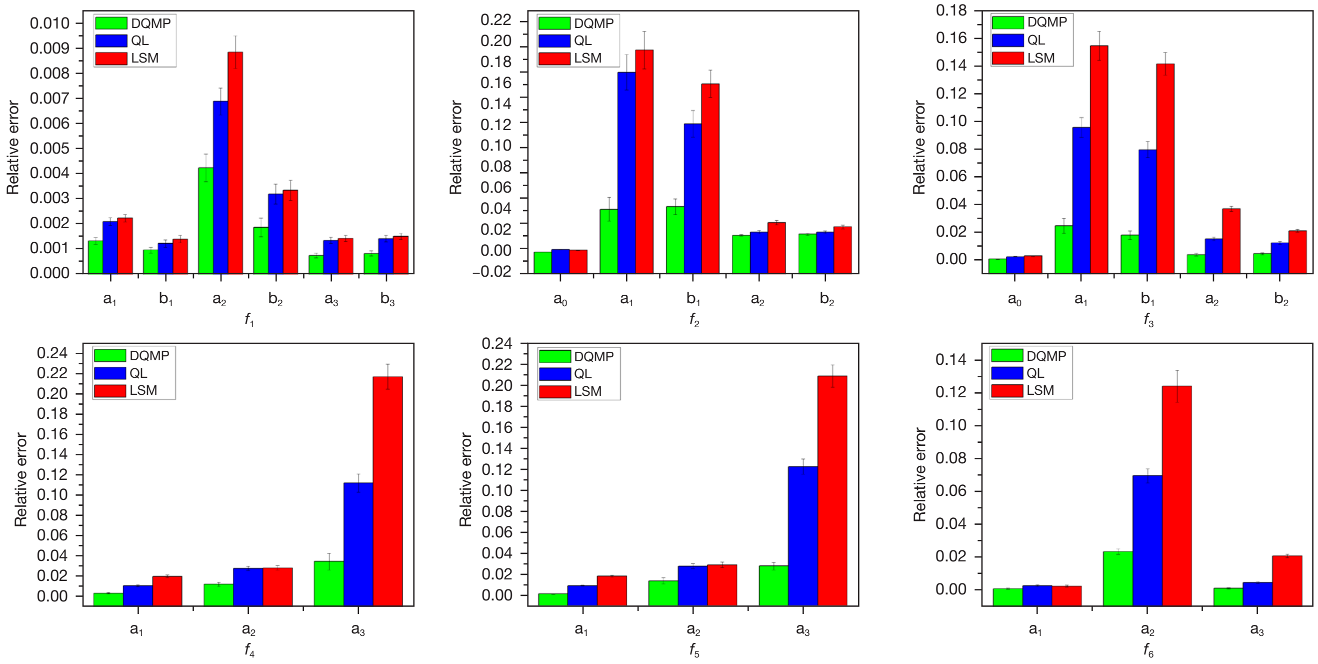

Figure 6 The plot of relative errors of model parameter fitting based on 6 functions f_1 to f_6. Data are represented as mean ± standard deviation. DQMP, Deep-Q Learning of Model Parameters; QL, Q-Learning; LSM, Least Squared Method.

physical parameter imaging, particularly in the field of radiological imaging. Due to DRN, DQMP is expected to obtain reliable estimations for multiple parameter imaging as it combines both the parameter fitting and curve fitting rewards to guide the global search. It should be noted that for some insensitive parameters such as viscosity τ, variation from values of tens to hundreds has very little influence on the curve change (36). Therefore, the imaging may be noisy. To address this issue, future investigation will be conducted by adjusting the contribution of the parameter fitting reward and curve fitting reward in the DRN training.

Conclusions

This is the first work to convert a model parameter optimization task into a state-action decision-making task in the *k*-D parameter space. We leveraged the integration of QL with DL to build a model designed to learn global reward values from both visible (data fitting) and hidden states (parameter fitting) and proposed a DQMP scheme for global parameter optimization for any complex, nonconvex function. Through DQMP, *k*-D parameter searching in each step resembled the decision-making of action selection from 3^k configurations, just like a chess move in a *k*-D board game. The proposed DQMP combined prior knowledge through DRN. An appropriate decision was made by maximizing the Q-value, which combined the current and future reward functions from both visible and hidden states, so as to iteratively update parameters toward the global solution. In summary, the novelty of the work is, as follows:

- A model parameter optimization problem was converted into a state-action decision making problem in the *k*-D parameter space, which resembled the decision-making of a *k*-D move game.
- To guide global searching, a DRN was proposed to learn the global reward from both hidden and visible states.
- A novel DQMP method integrated both current and

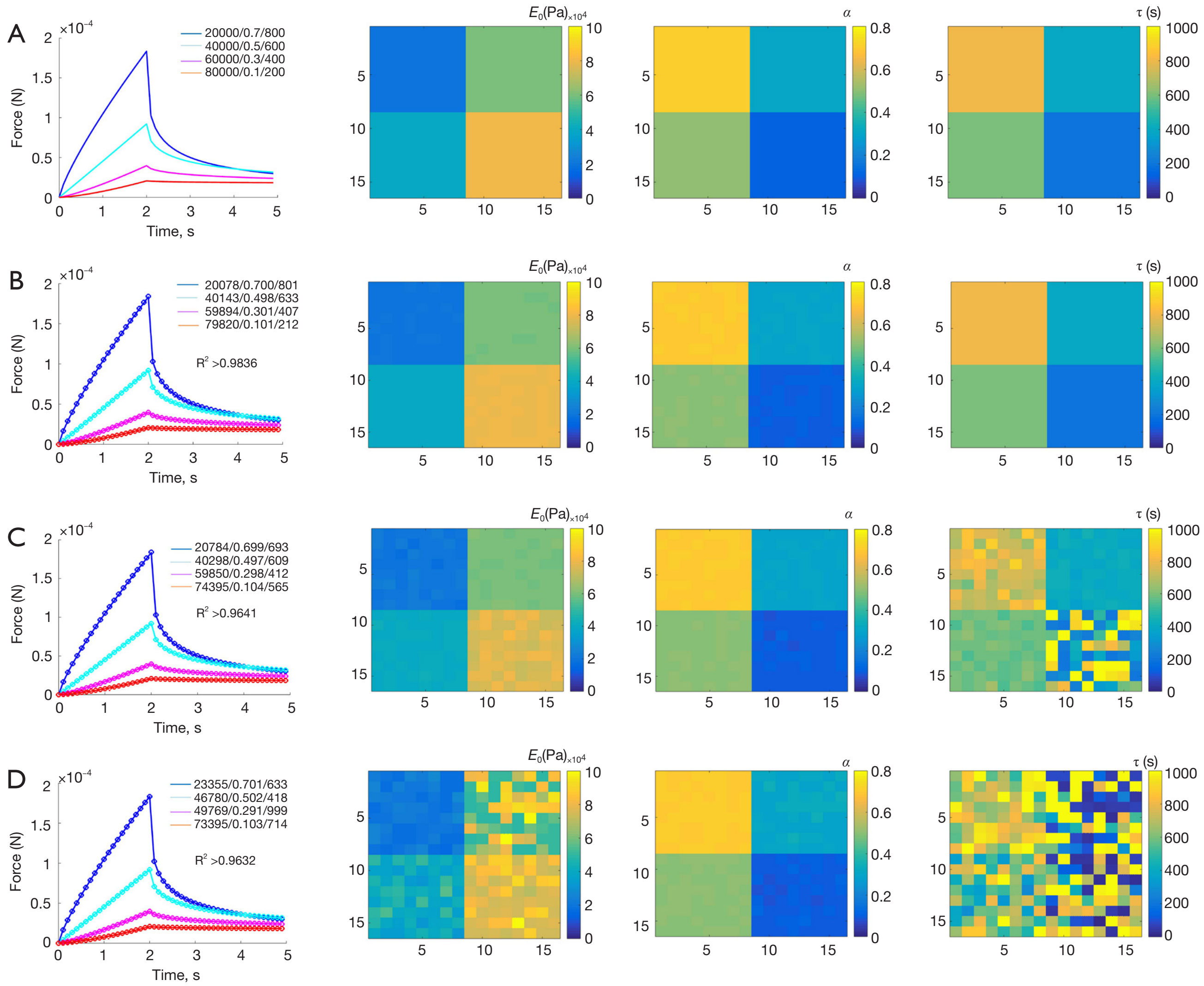

Figure 7 Representative curves generated by f_4 with parameters [20000, 0.7, 800], [40000, 0.5, 600], [60000, 0.3, 400] and [80000, 0.1, 200] for 4 regions. The fits of noisy curves by DQMP, QL, and LSM algorithms were shown from top to bottom. The corresponding viscoelastic parameters [E_0, α, τ] in the 16×16 matrices (left to right) for simulation parameters (A), and fitted parameters using (B) DQMP, (C) QL, and (D) LSM algorithms. DQMP, Deep-Q Learning of Model Parameters; QL, Q-Learning; LSM, Least Squared Method.

future global reward functions, which lead to global searching iteratively by maximizing Q-value in the parameter space.

The proposed DQMP method has demonstrated capability of finding global optimal model parameters and shows potential for the extraction or imaging of physical parameters in many applications. Overall, DQMP can accurately find global model parameters with high accuracy and consistency, both of which are crucial for the development of new fitting and imaging algorithms. This method shines a light on global optimization of multiple parameters in a variety of fitting problems.

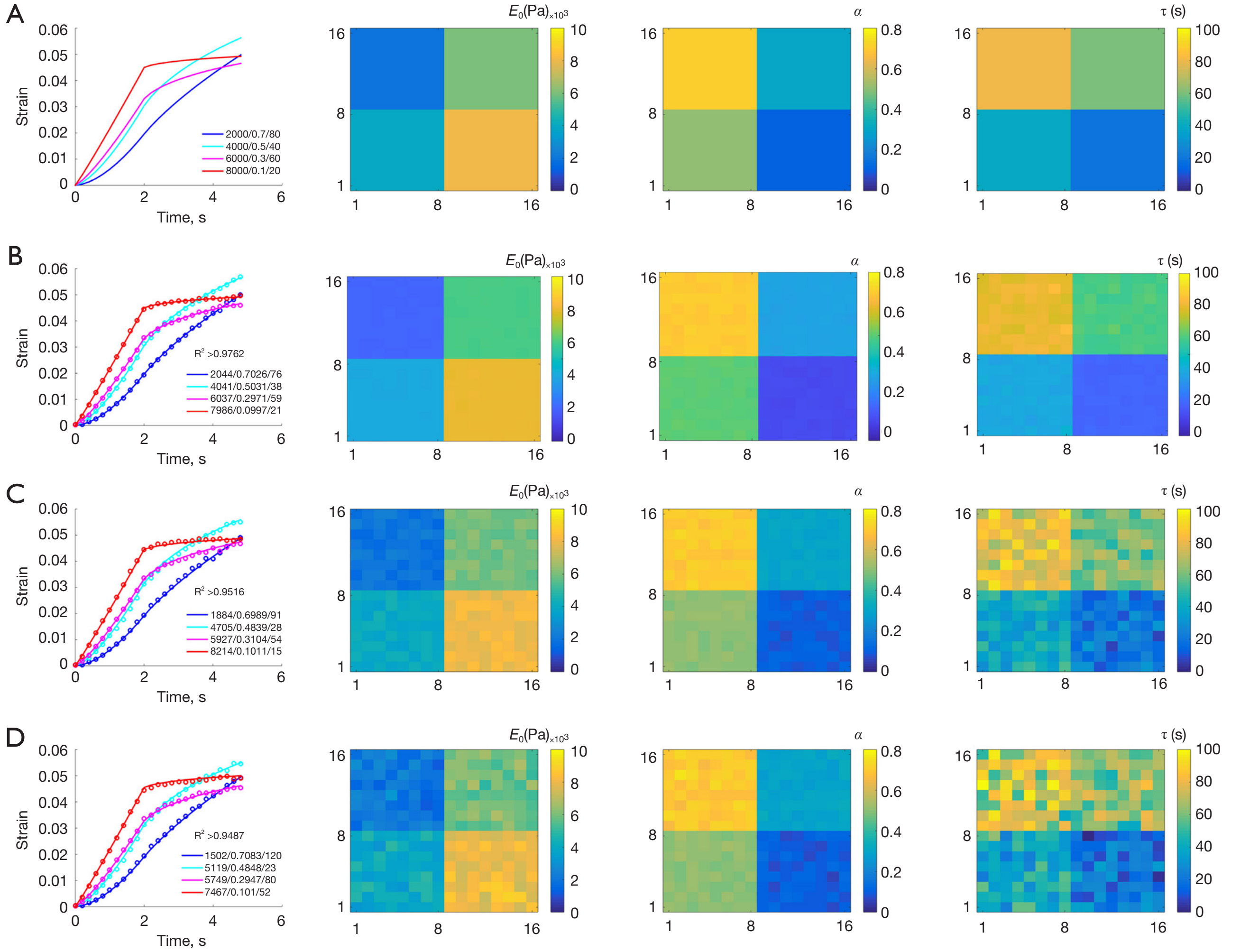

Figure 8 Representative curves generated by f_5 with parameters [2000, 0.7, 80], [4000, 0.5, 40], [6000, 0.3, 60] and [8000, 0.1, 20] for 4 regions. The fit of noisy curves by DQMP, QL, and LSM algorithms were shown from top to bottom. The corresponding viscoelastic parameters [E_0, α, τ] in the 16×16 matrices (left to right) are (A) simulation parameters image, (B) imaged parameters by DQMP, (C) by QL, and (D) by LSM algorithms. DQMP, Deep-Q Learning of Model Parameters; QL, Q-Learning; LSM, Least Squared Method.

Acknowledgments

Funding: This study was financially supported by the National Natural Science Foundation of China (Nos. 62171366 and 61871316; to Hongmei Zhang).

Footnote

Conflicts of Interest: All authors have completed the ICMJE uniform disclosure form (available at https://qims.amegroups.com/article/view/10.21037/qims-22-1147/coif). HZ reports that this study was financially supported by the National Natural Science Foundation of China (Nos. 62171366 and 61871316). The other authors have no conflicts of interest to declare.

Ethical Statement: The authors are accountable for all aspects of the work in ensuring that questions related to the accuracy or integrity of any part of the work are appropriately investigated and resolved. No patient data or animal data were used in this paper, the ethical approval and informed consent were not required.

Open Access Statement: This is an Open Access article distributed in accordance with the Creative Commons

References

1. Li S, Fang H, Shi B. Remaining useful life estimation of Lithium-ion battery based on interacting multiple model particle filter and support vector regression. Reliab Eng Syst Saf 2021;210:107542.
2. Bayoumi AS, El-Sehiemy RA, Mahmoud K, Lehtonen M, Darwish MMF. Assessment of an improved three-diode against modified two-diode patterns of MCS solar cells associated with soft parameter estimation paradigms. Appl Sci 2021;11:1055.
3. Redmond DP, Chiew YS, Major V, Chase JG. Evaluation of model-based methods in estimating respiratory mechanics in the presence of variable patient effort. Comput Methods Programs Biomed 2019;171:67-79.
4. Von Stumberg L, Wenzel P, Khan Q, Cremers D. Gn-net: The gauss-newton loss for multi-weather relocalization. IEEE Robot Automat Lett 2020;5:890-7.
5. Mele M, Magazzino C, Schneider N, Nicolai F. Revisiting the dynamic interactions between economic growth and environmental pollution in Italy: evidence from a gradient descent algorithm. Environ Sci Pollut Res Int 2021;28:52188-201.
6. Ly HB, Nguyen MH, Pham BT. Metaheuristic optimization of Levenberg–Marquardt-based artificial neural network using particle swarm optimization for prediction of foamed concrete compressive strength. Neural Comput Applic 2021;33:17331-51.
7. Jouha W, El Oualkadi A, Dherbécourt P, Joubert E, Masmoudi M. Silicon carbide power MOSFET model: An accurate parameter extraction method based on the levenberg–marquardt algorithm. IEEE Trans Power Electron 2018;33:9130-3.
8. Assad A, Deep K. A hybrid harmony search and simulated annealing algorithm for continuous optimization. Inf Sci 2018;450:246-66.
9. Hedar AR, Deabes W, Amin, HH, Almaraashi M, Fukushima M. Global sensing search for nonlinear global optimization. J Glob Optim 2022;82:753-802.
10. Chakraborty S, Saha AK, Sharma S, Chakraborty R, Debnath S. A hybrid whale optimization algorithm for global optimization. J Ambient Intell Humaniz Comput 2023;14:431-7.
11. Katoch S, Chauhan SS, Kumar V. A review on genetic algorithm: past, present, and future. Multimed Tools Appl 2021;80:8091-126.
12. Pedrozo HA, Dallagnol AM, Schvezov CE. Genetic algorithm applied to simultaneous parameter estimation in bacterial growth. J Bioinform Comput Biol 2021;19:2050045.
13. Sengupta S, Basak S, Peters RA 2nd. Particle Swarm Optimization: A survey of historical and recent developments with hybridization perspectives. Mach Learn Knowl Extr 2019;1:157-91.
14. Holcomb SD, Porter WK, Ault SV, Mao G, Wang J. Overview on DeepMind and Its AlphaGo Zero AI. Proceedings of the 2018 International Conference on Big Data and Education. 2018;67-71.
15. Silver D, Hubert T, Schrittwieser J, Antonoglou I, Lai M, Guez A, Lanctot M, Sifre L, Kumaran D, Graepel T, Lillicrap T, Simonyan K, Hassabis D. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science 2018;362:1140-4.
16. Qu B, Cao J, Qian C, Wu J, Lin J, Wang L, Ou-Yang L, Chen Y, Yan L, Hong Q, Zheng G, Qu X. Current development and prospects of deep learning in spine image analysis: a literature review. Quant Imaging Med Surg 2022;12:3454-79.
17. Panwar H, Gupta PK, Siddiqui MK, Morales-Menendez R, Singh V. Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet. Chaos Solitons Fractals 2020;138:109944.
18. Ni M, Zhao Y, Wen X, Lang N, Wang Q, Chen W, Zeng X, Yuan H. Deep learning-assisted classification of calcaneofibular ligament injuries in the ankle joint. Quant Imaging Med Surg 2023;13:80-93.
19. Gupta A, Anpalagan A, Guan L, Khwaja AS. Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues. Array 2021;10:100057.
20. Nguyen H, Kieu ML, Wen T, Cai C. Deep learning methods in transportation domain: a review. IET Intell Transp Syst 2018;12:998-1004.
21. Lu B, Ni C, Zheng Z, Liu T. A global optimization algorithm based on multi-loop neural network control. J Syst Eng Electron 2019;30:1007-24.
22. Li G, Yang Y, Qu X, Cao D, Li K. A deep learning based image enhancement approach for autonomous driving at

night. Knowledge-Based Systems 2021;213:106617.

23. Wåhlstrand Skärström V, Krona A, Lorén N, Röding M. DeepFRAP: Fast fluorescence recovery after photobleaching data analysis using deep neural networks. J Microsc 2021;282:146-61.
24. Cai L, Ren L, Wang Y, Xie W, Zhu G, Gao H. Surrogate models based on machine learning methods for parameter estimation of left ventricular myocardium. R Soc Open Sci 2021;8:201121.
25. Fujimoto S, Gu SS. A minimalist approach to offline reinforcement learning. Advances in Neural Information Processing Systems 34 (NeurIPS 2021). 2021;34:20132-45.
26. He X, Wang K, Huang H, Miyazaki T, Wang Y, Guo S. Green Resource Allocation Based on Deep Reinforcement Learning in Content-Centric IoT. IEEE Trans Emerg Top Comput 2020;8:781-96.
27. Agarwal R, Schwarzer M, Castro PS, Courville AC, Bellemare M. Deep reinforcement learning at the edge of the statistical precipice. Advances in Neural Information Processing Systems 34 (NeurIPS 2021). 2021;34:29304-20.
28. Janner M, Li Q, Levine S. Offline reinforcement learning as one big sequence modeling problem. Advances in Neural Information Processing Systems 34 (NeurIPS 2021). 2021;34:1273-86.
29. Wurman PR, Barrett S, Kawamoto K, MacGlashan J, Subramanian K, Walsh TJ, et al. Outracing champion Gran Turismo drivers with deep reinforcement learning. Nature 2022;602:223-8.
30. Watkins CJ, Dayan P. Q-Learning. Machine Learning 1992;8:279-92.
31. Vázquez-Canteli JR, Nagy Z. Reinforcement learning for demand response: A review of algorithms and modeling techniques. Applied Energy 2019;235:1072-89.
32. Arslan G, Yüksel S. Decentralized Q-Learning for Stochastic Teams and Games. IEEE Trans Autom Control 2017;62:1545-58.
33. Raffensperger PA, Bones PJ, McInnes AI, Webb RY. Rewards for pairs of Q-learning agents conducive to turn-taking in medium-access games. Adaptive Behavior 2012;20:304-18.
34. Yu KH, Chen YA, Jaimes E, Wu WC, Liao KK, Liao JC, Lu KC, Sheu WJ, Wang CC. Optimization of thermal comfort, indoor quality, and energy-saving in campus classroom through deep Q learning. Case Stud Therm Eng 2021;24:100842.
35. Fan D, Sun H, Yao J, Zhang K, Yan X, Sun Z. Well production forecasting based on ARIMA-LSTM model considering manual operations. Energy 2021;220:119708.
36. Zhang H, Lu T, Zhang Q, Zhou Y, Zhu H, Harms J, Yang X, Wan M, Insana MF. Solutions to ramp-hold dynamic oscillation indentation tests for assessing the viscoelasticity of hydrogel by Kelvin-Voigt fractional derivative modeling. Mech Mater 2020;148:103431.

Cite this article as: Zhang H, Liang S, Matkovic LA, Momin S, Wang K, Yang X, Insana MF. Deep Q-learning to globally optimize a *k*-D parameter search for medical imaging. Quant Imaging Med Surg 2023;13(8):4879-4896. doi: 10.21037/qims-22-1147