IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received February 25, 2021, accepted March 7, 2021, date of publication March 10, 2021, date of current version March 19, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3065311*

# An Improved Partial-Form MFAC Design for Discrete-Time Nonlinear Systems With Neural Networks

**YE YANG[1], CHEN CHEN[1], AND JIANGANG LU[1,2]**

[1]State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China
[2]Zhejiang Laboratory, Hangzhou 311121, China

Corresponding author: Jiangang Lu (lujg@zju.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61590925 and Grant U1609212, in part by the National Key Research and Development Plan of China under Grant 2017YFC0210403, in part by the Fundamental Research Funds for the Central Universities [Zhejiang University New Generation Industrial Control System (NGICS) Platform], and in part by the Major Scientific Project of Zhejiang Laboratory under Grant 2020MC0AE01.

**ABSTRACT** This article investigates the partial-form model-free adaptive control (MFAC) issue for a class of discrete-time nonlinear systems. An improved partial-form MFAC design named IPFMFAC-NN is proposed, where neural networks are introduced to enhance the control performance. With the excellent approximation ability of radial basis function (RBF) neural networks, the pseudo gradient (PG) values of control method can be directly approximated online using the measured input and output data of the controlled system. Besides, long short-term memory (LSTM) neural networks are used to tune the essential parameters of the control method online with system error set and gradient information set. Finally, the effectiveness and applicability are verified by SISO discrete nonlinear system simulation and three-tank system simulation, and experimental results demonstrate that the proposed method achieves the best control performance in all five indices. Especially compared with the partial-form MFAC, the proposed method reduces the *RMSE* index by 43.83% and 6.39%, respectively in two simulations, making it a promising control method for discrete-time nonlinear systems.

**INDEX TERMS** LSTM neural networks, partial-form model-free adaptive controller, RBF neural networks, three-tank system.

## I. INTRODUCTION

Since Kalman proposed the concept of state-space in the 1960s [1], [2], modern control theory has made significant developments, and many research fields and branches have been derived [3], such as linear system theory, system identification, optimal control, and robust control [4], etc. These theories have been widely and successfully used in practice, especially in aerospace, national defense, and military industry [5].

With the development of science and technology, industrial production processes are becoming more and more complex. It is arduous to establish precise mathematical models based on physical and chemical mechanisms in many practical situations [6]. However, the industrial process data can be easily measured and stored, containing a lot of information related to the controlled system's actual dynamics. Therefore, it is of great significance to utilize these measured data for modeling and control, mostly when an accurate model of the controlled system cannot be obtained [7].

Data-driven control (DDC) [8] refers to a control method in which the controller's design only utilizes the online and offline I/O data of the controlled system without relying on precise mathematical models [9]. Nowadays, many kinds of data-driven control (DDC) methods have emerged, such as proportional-integral-derivative (PID) [10], fractional-order controller (FOC) [11], model-free adaptive control (MFAC) [12], simultaneous perturbation stochastic approximation (SPSA) [13], virtual reference feedback tuning (VRFT) [14], iterative feedback tuning (IFT) [15], lazy learning (LL) [16], and so on. These DDC methods have been widely used on in practice after years of development. Besides, the study of neural networks has made remarkable academic achievements in recent years,

The associate editor coordinating the review of this manuscript and approving it for publication was Haiquan Zhao.

and researches based on the combination of data-driven methods and neural networks have also derived some theoretical results. Rădac *et al.* [17] presented a new iterative data-driven algorithm to solve the optimization problems for nonlinear processes, which uses linear controllers accounting for operational constraints and employs a quadratic penalty function approach to compensate for process nonlinearities and uncertainties; Yacoub *et al.* [18] proposed an active noise control (ANC) system based on the combination of finite-impulse response filter and functional-link neural network, its combined structure can not only compensate for the nonlinear phenomena that often appear in the real-world ANC applications, but also improve the attenuation performance; Yan *et al.* [19] developed a multivariate nonlinear controller design method for MIMO nonlinear systems via VRFT and neural networks, and restated the model reference control problem with time-domain model in the absence of transfer functions and simplify the objective function of VRFT without a linear filter; Zouari *et al.* [20] introduced two techniques named robust neural adaptive control and neural indirect adaptive control to ensure the robustness of uncertain nonlinear multivariable systems, and the study of the stability and robustness of both techniques was performed by Lyapunov theory; Esparza *et al.* [21] considered a generic virtual reference tuning (VRT) methodology, which includes open-loop and closed-loop setups. This proposed VRT discussed the computation of some required gradients via backpropagation through time and showed the achieved improvements over the standard 1-degree of freedom control loop when auxiliary sensors are used.

Among above mentioned DDC methods, MFAC and its variants are popular in industrial process control [12]. MFAC uses dynamic linearization technology to establish the equivalent dynamic linearization model of the controlled object at each sampling time, and then it estimates the pseudo partial derivative (PPD) values or pseudo gradient (PG) values of this model to approximate the local dynamic of the controlled system, so as to obtain the expected control performance [22]. Compared with ordinary DDC methods such as PID [10] or fractional-order controller [11], the equivalent dynamic linearization method MFAC has the advantages of simple structure, convenient design, etc., and can equivalently convert a discrete-time nonlinear system into a series of dynamic linearized data models based on I/O measurement data.

In the control scheme of partial-form MFAC, individual parameters are essential which affect the stability of the controlled system. However, as the controlled system status changes, these parameters should be tuned accordingly, so they are nonlinear and time-varying. Tuning parameters is time-consuming and labor-intensive, and inappropriate values can bring about poor control performance [24]. Therefore, the essential parameters online tuning work is of great practical significance [25]. So far, there are few theoretical results about the parameters tuning of MFAC. Wang *et al.* [26] proposed a controller parameter tuning method based on optimization technology; however, there is no comparative experiment to show the superiority of the algorithm; Chen and Lu [27] introduced BP neural network help MFAC complete parameters tuning work; however, authors did not compare it with other advanced parameter tuning algorithms; Gao *et al.* [28] introduced particle swarm optimization (PSO) to optimize the MFAC by searching the optimal parameters; however, this algorithm has long calculation time.

In practical application scenarios, as the amount of data grows and the controlled systems become more complex, ordinary neural networks such as BP neural networks can no longer tune parameters accurately, which are easy to fail into local optimum [29]. In previous research work [30], it has been found that the LSTM neural network can estimate the parameters to be tuned in the MFAC in real time, and its optimization effect on the MFAC is more obvious than that of the BP neural network, which proves the superiority of LSTM neural network and the necessity of introducing it to tune essential parameters online of partial-form MFAC. Moreover, for partial-form MFAC, PG's change will also become complicated when the controlled system is strongly nonlinear. Suppose only the projection algorithm of partial-form MFAC is used to estimate PG [9]; in that case, there is a risk that the estimated values differ too much from ideal values, thereby affecting control performance. Although theoretical analysis demonstrates that even if there is a deviation in PG values estimation, the system's stability can be guaranteed as long as the appropriate weighting factor is selected in the control scheme [9]. However, to improve the partial-form MFAC control performance, optimizing the PG estimation algorithm is of research significance. RBF neural networks have strong function approximation ability [23], and they can be introduced to estimate PG of partial-form MFAC. Besides, the topology of the RBF neural network is simple, which can ensure its calculation efficiency.

This article considers that PG of the dynamic linearization model in partial-form MFAC can be estimated more accurately, and essential parameters in the control scheme should be tuned in real time according to changes in the controlled system. An improved partial-form MFAC design named IPFMFAC-NN is proposed. The feature of the proposed method is that all introduced neural networks are trained online using the measured I/O data of the controlled system, and no off-line training and physical model of the controlled system are involved. Compared with ordinary control algorithms, the proposed method can reflect the nonlinear characteristics of the system and tune essential parameters online in real time, and neural networks introduced by the proposed method has strong learning ability and fast convergence speed. Therefore, it can achieve better control performance and stability in dealing with the general discrete-time nonlinear system problems

The significant contributions of this article are summarized as follows:

1) The RBF neural network is introduced to estimate the PG of the dynamic linearization data model in

partial-form MFAC, which merely depends on the controlled system's measured I/O data. The RBF neural network is a local approximation neural network with fast convergence speed and can avoid falling into local optima. It is very suitable for the online estimation of PG.

2) The LSTM neural network is introduced to perform essential parameters online tuning work. As an improved RNN, the LSTM neural network utilizes the gate mechanism to control information entering and exiting with the input gate, the forget gate, and the output gate. Each gate has a strong ability to process massive data feature information, which alleviates the gradient exploding and vanishing problems of the RNN neural network and improves the performance of parameters tuning work.
3) This article carries out qualitative and quantitative comparisons between the proposed method and other tested methods. The ablation analysis is conducted to prove the effectiveness of PG estimation and online parameters self-tuning. Besides, this article verifies the superiority and applicability of the proposed method by comparing it with advanced methods in cited references. The experimental results demonstrate that the proposed method achieves the best performance in both SISO discrete nonlinear system simulation and three-tank system simulation.

The outline of this article proceeds as follows: Section II is problem formulation. Section III introduces the proposed method's architecture and mathematical principles, including the PG estimation and essential parameters online tuning mechanisms. Section IV is the experimental part, which demonstrates the superiority and stability of the proposed method in both SISO discrete nonlinear system simulation and three-tank water system simulation. Section V concludes this article and discuss future work.

## II. PROBLEM FORMULATION

Consider a class of SISO discrete-time nonlinear systems [31]:

$$y(k+1) = f\left(y(k), \cdots, y\left(k-n_y\right), u(k), \cdots, u\left(k-n_u\right)\right) \tag{1}$$

where $y(k) \in R$, $u(k) \in R$ are defined as the output and input of the system at time $k$; $n_y$ and $n_u$ represent the unknown order of the system; $f(\cdots) : R^{n_u+n_y+2} \mapsto R$ is a nonlinear function. $\boldsymbol{U}_L(k) \in \mathbf{R}^L$ is defined as a vector composed of all control input signals in a sliding time window $[k-L+1, k]$

$$\boldsymbol{U}_L(k) = [u(k), \cdots, u(k-L+1)]^{\mathrm{T}} \tag{2}$$

where $L$ is the linearization length, and $\boldsymbol{U}_L(k) = \mathbf{0}_L$ when $k \leqslant 0$, $\mathbf{0}_L$ is a zero vector of dimension L

In order to dynamically linearize nonlinear system (1), the following assumptions are provided [31]:

*Assumption 1:* $f(\cdots)$ has continuous derivative with respect to the $(n_y+2)th$ variable to the $(n_y+L+1)th$ variable.

*Assumption 2:* System (1) satisfies the generalized Lipschitz condition: for any $k_1 \neq k_2, k_1, k_2 \geqslant 0$ and $\boldsymbol{U}_L(k_1) \neq \boldsymbol{U}_L(k_2)$, then $|y(k_1+1) - y(k_2+1)| \leqslant b\|\boldsymbol{U}_L(k_1) - \boldsymbol{U}_L(k_2)\|$ is established, where $b$ is a constant.

For the nonlinear system (1) that satisfies *assumption* 1 and *assumption* 2, there exists a time-varying parameter vector $\phi_{p,L}(k) \in \mathbf{R}^L$ called pseudo gradient (PG) that can transfer system (1) into the following partial-form dynamic linearization (PFDL) model:

$$\Delta y(k+1) = \boldsymbol{\phi}_{p,L}^{\mathrm{T}}(k)\Delta \boldsymbol{U}_L(k) \tag{3}$$

Equation (3) is an equivalent dynamic linear representation of the system (1), it is a linear time-varying model with a simple incremental form which is essentially different from the traditional models. For the design of partial-form MFAC, the following criterion function is considered [9]:

$$J(u(k)) = \left|y^*(k+1) - y(k+1)\right|^2 + \lambda|u(k) - u(k-1)|^2 \tag{4}$$

where $\lambda > 0$ is a weighting factor that limits the change in the control input, which is often applied in control system design, as it guarantees the smoothness of the control input signal, and $y^*(k+1)$ is the desired output signal. The criterion function $J$ consists of two parts, the term $|y^*(k+1) - y(k+1)|^2$ is introduced to minimize system error, and the term $\lambda|u(k) - u(k-1)|^2$ is introduced to prevent excessive changes in control input. This criterion function also takes into account the changes in the output and input of the controlled system, which is more general than the ordinary one-step forward prediction error criterion function in dealing with nonlinear process problems. Substituting Eq. (3) into criterion function (4), taking the derivative of $u(k)$ and making it equal to zero, the optimal solution can be determined. With regard to system (1), the following partial-form MFAC control scheme can be obtained:

$$\begin{aligned}\hat{\boldsymbol{\phi}}_{p,L}(k) = \hat{\boldsymbol{\phi}}_{p,L}(k-1) &+ \frac{\eta\Delta \boldsymbol{U}_L(k-1)}{\mu + \|\Delta \boldsymbol{U}_L(k-1)\|^2} \\ &\times (y(k) - y(k-1) - \hat{\boldsymbol{\phi}}_{p,L}^{\mathrm{T}}(k-1)\Delta \boldsymbol{U}_L(k-1))\end{aligned} \tag{5}$$

$$\begin{aligned}\hat{\boldsymbol{\phi}}_{p,L}(k) = \hat{\boldsymbol{\phi}}_{p,L}(1), \quad &if \left|\hat{\boldsymbol{\phi}}_{p,L}(k)\right| \leq \varepsilon \ or |\boldsymbol{U}_L(k-1) \leq \varepsilon \\ &or \ sign\left(\hat{\phi}_1(k)\right) \neq sign\left(\hat{\phi}_1(1)\right)\end{aligned} \tag{6}$$

$$\begin{aligned}u(k) = u(k-1) &+ \frac{\rho_1\hat{\phi}_1(k)\left(y^*(k+1) - y(k)\right)}{\lambda + \left|\hat{\phi}_1(k)\right|^2} \\ &- \frac{\hat{\phi}_1(k)\sum_{i=2}^{L}\rho_i\hat{\phi}_i(k)\Delta u(k-i+1)}{\lambda + \left|\hat{\phi}_1(k)\right|^2}\end{aligned} \tag{7}$$

where $\mu > 0$, $\eta \in (0, 2]$, and $\rho_i \in (0, 1]$, $i = 1, 2, \cdots, L$ is an important parameter introduced as a penalty factor for a more general and flexible control rule. $\mu$ is the weighting factor that limits the variance of the control input $u(k)$. $\hat{\boldsymbol{\phi}}_{p,L}(1)$

is the initial value of $\hat{\boldsymbol{\phi}}_{p,L}(k)$. Eq. (6) is the reset algorithm of PG.

As shown in the control scheme (5) – (7), partial-form MFAC only uses the online I/O data measured by the closed-loop controlled system for controller design and has nothing to do with the mathematical model structure as well as the system order, and it has the advantages of small calculation amount and easy implementation. In addition, partial-form MFAC can be combined with other model-based control theories and methods for modular design to achieve complementary advantages in work.

It should be emphasized that PG needs to be determined accurately to realize the partial-form MFAC. Considering that the mathematical model of the controlled system is unknown and PG is time-varying, it is difficult to obtain its exact value. The accuracy of the default PG projection algorithm in the original partial-form MFAC can be improved. Besides, [9], [26], [27] all explain that the parameters $\rho_i$ and $\lambda$ are of great importance to the design of partial-form MFAC, including theoretical analysis and simulation results showing that appropriate selection of these parameters can ensure the controlled system's stability and achieve better control performance [9]. Therefore, an improved partial-form design based on neural networks is proposed.

## III. THE PROPOSED IPFMFAC-NN METHOD

In this section, an improved partial-form MFAC based on neural networks is proposed to realize the accurate PG estimation and complete the online parameters tuning work of the control scheme in partial-form MFAC. More specifically, the proposed method introduces RBF neural networks to estimate the PG of the dynamic linearization data model in partial-form MFAC. LSTM neural networks are utilized to tune essential parameters online to achieve better control performance.

### A. PG ESTIMATION BASED ON RBFNN

In 1989, Moody and Darken [32] proposed the RBF neural network, a feedforward network with a three-layer structure of the input layer, hidden layer, and output layer. The RBF neural network has been proved to be an effective method for nonlinear estimation, with simple structure, fast convergence speed, and avoiding falling into local optimum [33]. Therefore, the RBF neural network is introduced to estimate the time-varying PG values.

The topology of the RBF neural network is shown in Fig. 1, and it is introduced to estimate the PG values. The input is the I/O information vector of the controlled system:

$$\boldsymbol{x}(k) = [y(k), \ldots, y(k-i), u(k-1), \ldots, u(k-j)] \quad (8)$$

where $i$ and $j$ are two integers. The output of the $nth$ neuron in the hidden layer is:

$$R_n(\boldsymbol{x}(k)) = \exp\left(-\frac{\|\boldsymbol{x}(k) - \boldsymbol{c}_n(k-1)\|^2}{b_n^2(k-1)}\right), \quad n = 1, 2, \cdots, m \quad (9)$$

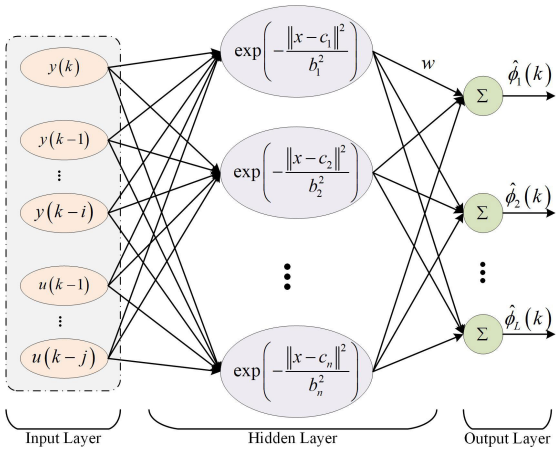



**FIGURE 1.** Structure diagram of recurrent neural network.

where $m$ is the number of hidden layer node, $\boldsymbol{c}_n$ denotes the center vector of the $nth$ hidden neuron, $\|\boldsymbol{x} - \boldsymbol{c}_n\|$ is the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{c}_n$, and $b_n$ is the radius of the $nth$ hidden neuron. The $lth$ output of RBF neural network is:

$$\hat{\phi}_l(k) = \sum_{n=1}^{m} w_{nl}(k-1) R_n(\boldsymbol{x}(k)), \quad l = 1, 2, \cdots, L \quad (10)$$

where $w_{nl}$ is the connection weights between the hidden neurons and the $lth$ output layer. Therefore, the estimated value of PG at time $k$ is:

$$\hat{\boldsymbol{\phi}}_{p,L}(k) = [\hat{\phi}_1(k), \hat{\phi}_2(k), \cdots, \hat{\phi}_L(k)] \quad (11)$$

Take the one-step ahead squared error as the performance indicator function:

$$J = \frac{1}{2} e(k+1)^2 = \frac{1}{2} \left(y^*(k+1) - y(k+1)\right)^2 \quad (12)$$

The training parameters of the RBF neural network can be obtained by the following gradient descent method [34]:

$$\begin{aligned} w_{nl}(k+1) &= w_{nl}(k) - \beta \frac{\partial J}{\partial w_{nl}(k)} + \alpha \Delta w_{nl}(k) \\ \frac{\partial J}{\partial w_{nl}(k)} &= \frac{\partial J}{\partial e(k+1)} \frac{\partial e(k+1)}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \\ &\quad \times \frac{\partial u(k)}{\partial \hat{\phi}_l(k)} \frac{\partial \hat{\phi}_l(k)}{\partial w_{nl}(k)} \end{aligned} \quad (13)$$

$$\begin{aligned} b_n(k+1) &= b(k) - \beta \frac{\partial J}{\partial b_n(k)} + \alpha \Delta b_n(k) \\ \frac{\partial J}{\partial b_n(k)} &= \frac{\partial J}{\partial e(k+1)} \frac{\partial e(k+1)}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \\ &\quad \times \frac{\partial u(k)}{\partial \hat{\phi}_l(k)} \frac{\partial \hat{\phi}_l(k)}{\partial R_n(k)} \frac{\partial R_n(k)}{\partial b_n(k)} \end{aligned} \quad (14)$$

$$\begin{aligned} c_n(k+1) &= c_n(k) - \beta \frac{\partial J}{\partial c_n(k)} + \alpha \Delta c_n(k) \\ \frac{\partial J}{\partial c_n(k)} &= \frac{\partial J}{\partial e(k+1)} \frac{\partial e(k+1)}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \\ &\quad \times \frac{\partial u(k)}{\partial \hat{\phi}_l(k)} \frac{\partial \hat{\phi}_l(k)}{\partial R_n(k)} \frac{\partial R_n(k)}{\partial c_n(k)} \end{aligned} \quad (15)$$

where $\beta$ and $\alpha$ are the learning rate and inertia coeffient, respectively. The partial derivatives of $u(k)$ with respect to $\hat{\phi}_l(k)$ are as follows:

$$\frac{\partial u(k)}{\partial \hat{\phi}_1(k)} = \rho_1(y^*(k+1) - y(k))\frac{\lambda - \hat{\phi}_1(k)^2}{(\lambda + \hat{\phi}_1(k)^2)^2} - \sum_{i=2}^{L} \rho_i \hat{\phi}_i(k)\Delta u(k-i+1)\frac{\lambda - \hat{\phi}_1(k)^2}{(\lambda + \hat{\phi}_1(k)^2)^2} \tag{16}$$

$$\frac{\partial u(k)}{\partial \hat{\phi}_l(k)} = -\frac{\hat{\phi}_1(k)\rho_l \hat{\phi}_l(k)\Delta u(k-l+1)}{\lambda + \left|\hat{\phi}_1(k)\right|^2}, \quad 2 \le l \le L \tag{17}$$

It should be noted that the RBF neural network is a local approximation neural network, and its activation function is usually a Gaussian radial basis function, and the function image is attenuated on both sides and radially symmetric. When the selected center is very close to the query point, it has a real mapping effect on the input, so it has the advantages of faster learning speed and not easy to fall into the local optimum, which is suitable for real-time PG estimation. As a comparison, in a global approximation neural network such as a BP neural network, the weight coefficients in the network all have an impact on the output, so it needs to adjust the weight of each sample learning, which leads to slow convergence and easy to fall into local extremes.

### B. PARAMETERS ONLINE TUNING BASED ON LSTM

A recurrent neural network [35] was proposed by Jordan in 1986. It takes sequence data as input, recursively in the evolution direction of the sequence, and all nodes are connected in a chain. Fig. 2 illustrates the RNN structure

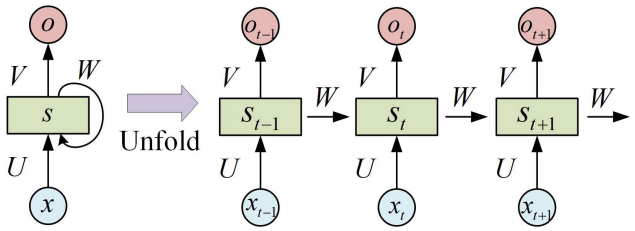



**FIGURE 2.** Structure diagram of recurrent neural network.

The forward propagation formulas for the RNN are shown as follows:

$$s_t = f(Ux_t + Ws_{t-1}) \tag{18}$$

$$o^t = g(Vs_t) \tag{19}$$

where $x_t$ represents the network input at time $t$, $s_t$ is the hidden state, $o_t$ is the network output, $U$ is the weight coefficient of the hidden layer to the output layer, $V$ is the weight coefficient of the hidden layer to the output layer, and $W$ is the hidden value of the previous moment. $f(x)$ is the tanh activation function, and $g(x)$ is the *softmax* activation function.

The training method of the RNN is the back-propagation through time (BPTT) [36].Take the weight coefficient $U$ to be updated as an example, the partial derivative formula of coefficient $U$ at time $t$ is:

$$\frac{\partial L_t}{\partial U} = \sum_{k=0}^{t} \frac{\partial L_t}{\partial o_t}\frac{\partial o_t}{\partial s_t}\left(\prod_{j=k+1}^{t}\frac{\partial s_j}{\partial s_{j-1}}\right)\frac{\partial s_k}{\partial U} \tag{20}$$

where $L_t$ is the loss function at time $t$. According to Eq. (18) and tanh activation function, partial derivative formula (20) can be converted into the following form:

$$\frac{\partial L_t}{\partial U} = \sum_{k=0}^{t} \frac{\partial L_t}{\partial o_t}\frac{\partial o_t}{\partial s_t}\left(\prod_{j=k+1}^{t}\tanh' W\right)\frac{\partial s_k}{\partial U} \tag{21}$$

The activation function tanh and its derivative curves are shown in Fig. 3. It can be seen that for most of the training process, the value of $\tanh'$ is less than 1. If the value range of the coefficient $W$ is in the interval [0,1], then term $\prod_{j=k+1}^{t}\tanh' W$ will approach zero when time $t$ is very large. Similarly, when coefficient $W$ is very large, $\prod_{j=k+1}^{t}\tanh' W$ will approach infinity. Therefore this is the problem of gradient vanishing and exploding in RNN neural networks which limits the use of RNN in actual scenarios.

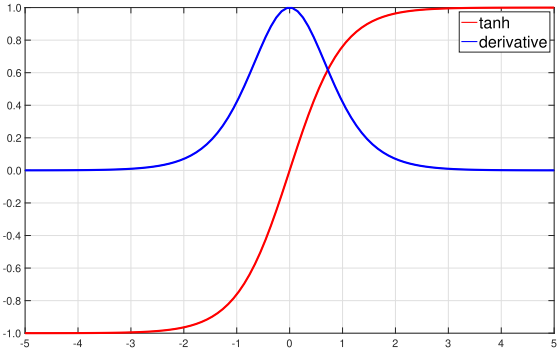



**FIGURE 3.** Tanh activation function and its derivative curve.

LSTM neural network can be regarded as an optimization of the RNN proposed by Hochreiter and Schmidhuber in 1997 [37]. Compared with an RNN, the LSTM neural network has the advantage of a gate mechanism to alleviate the gradient problems of an RNN. In the above-mentioned RNN gradient problem, the key to the gradient problem is $\frac{\partial s_t}{\partial s_{t-1}}$, and the similar term in the LSTM backpropagation can be expanded as follows:

$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c}_t \tag{22}$$

$$\frac{\partial c_t}{\partial c_{t-1}} = \frac{\partial c_t}{\partial f_t}\frac{\partial f_t}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial c_{t-1}} + \frac{\partial c_t}{\partial i_t}\frac{\partial i_t}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial c_{t-1}} + \frac{\partial c_t}{\partial \widetilde{c}_t}\frac{\partial \widetilde{c}_t}{\partial h_{t-1}}\frac{\partial h_{t-1}}{\partial c_{t-1}} + \frac{\partial c_t}{\partial c_{t-1}} \tag{23}$$

where $c_t$ is the cell state which is regarded as the part of hidden state of LSTM neural networks, $f_t$ is the forget gate,

$i_t$ is the input gate, $\widetilde{c}_t$ is the candidate cell state. Partial derivatives in Eq. (23) can be written as follows:

$$\begin{aligned}\frac{\partial c_t}{\partial c_{t-1}} &= c_{t-1}\sigma'(\cdot)w_f * o_{t-1}\tanh'(c_{t-1}) \\ &+\widetilde{c}_t\sigma'(\cdot)w_i * o_{t-1}\tanh'(c_{t-1}) \\ &+ i_t\tanh'(\cdot)w_c * o_{t-1}\tanh'(c_{t-1}) + f_t \end{aligned} \quad (24)$$

where $\sigma$ is the sigmoid activative function and $w_f$,$w_f$,$w_c$ are weight coefficients. Different from $\frac{\partial s_t}{\partial s_{t-1}}$, $\frac{\partial c_t}{\partial c_{t-1}}$ is a polynomial including $f_t \in [0, 1]$, and it can take on either values that are greater than 1 or values in the range [0,1] at any time step. Thus if the time step $t$ is very large, it is not guarenteed that $\frac{\partial c_t}{\partial c_{t-1}}$ will end up converging to zero or infinity, which can explain why LSTM neural networks can alleviate gradient problems of RNN neural networks.

Therefore, the gate mechanism in the hidden layer is suitable for processing and predicting important events with very long intervals and delays in the time series [38]. The architecture of an LSTM neural network is shown in Fig. 4, and it is used to tune essential parameters $\rho$ and $\lambda$ online mentioned in section II.

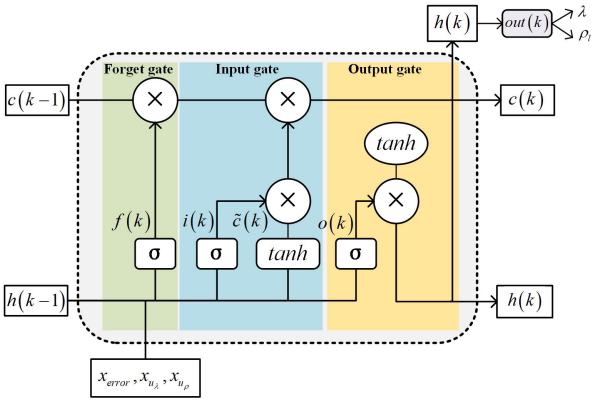



**FIGURE 4.** Architecture of the LSTM neural network.

The input consists of the system error set and the gradient information sets, which are defined as follows:

$$\begin{aligned} x_{error} &= \left[e(k), e(k)-e(k-1), \sum_{t=0}^{k} e(k)\right] \\ x_{u_\lambda} &= \left[\frac{\partial u(k-1)}{\partial \lambda}, \frac{\partial u(k-2)}{\partial \lambda}, \frac{\partial u(k-3)}{\partial \lambda}\right] \\ x_{u_\rho} &= \left[\frac{\partial u(k-1)}{\partial \rho_1}, \frac{\partial u(k-2)}{\partial \rho_1}, \frac{\partial u(k-3)}{\partial \rho_1}, \cdots\right. \\ &\quad \frac{\partial u(k-1)}{\partial \rho_l}, \frac{\partial u(k-2)}{\partial \rho_l}, \frac{\partial u(k-3)}{\partial \rho_l}, \cdots \\ &\quad \left.\frac{\partial u(k-1)}{\partial \rho_L}, \frac{\partial u(k-2)}{\partial \rho_L}, \frac{\partial u(k-3)}{\partial \rho_L}\right] \end{aligned} \quad (25)$$

where $x_{error}$ is defined as system error set, $x_{u_\lambda}$ and $x_{u_\rho}$ are defined as the gradient information sets. The overall input of the LSTM neural network is as follows:

$$X_k = \left[x_{error}, x_{u_\lambda}, x_{u_\rho}\right] \quad (26)$$

Then the LSTM neural network starts forward propagation calculation, and the outputs of all gates of the LSTM hidden layer are as follows:

$$\begin{aligned} net_{fi}(k) &= w_{fi}\left[X_k, h_{k-1}\right] + b_{fi} \\ f_i(k) &= \sigma\left(net_{fi}(k)\right) \quad (27) \\ net_{Ii}(k) &= w_{Ii}\left[X_k, h_{k-1}\right] + b_{Ii} \\ I_i(k) &= \sigma\left(net_{Ii}(k)\right) \quad (28) \\ net_{\tilde{c}i}(k) &= w_{ci}\left[X_k, h_{k-1}\right] + b_{ci} \\ \tilde{c}_i(k) &= \tanh\left(net_{\tilde{c}i}(k)\right) \quad (29) \\ c_i(k) &= c_i(k-1)\odot f_i(k) + I_i(k)\odot\tilde{c}_i(k) \quad (30) \\ net_{oi}(k) &= w_{\mathrm{oi}}\left[X_k, h_{k-1}\right] + b_{\mathrm{oi}} \\ o_i(k) &= \sigma\left(net_{oi}(k)\right) \quad (31) \\ h_i(k) &= o_i(k)\odot tanh\left(c_i(k)\right), \quad i = 1, 2, \ldots lstmnum \\ & \quad (32) \end{aligned}$$

where $f_i(k)$ is the output of forget gate, $I_i(k)$ and $\tilde{c}_i(k)$ are two part of input gate output, $o_i(k)$ is the output of output gate, $h_i(k)$ is the output of the LSTM hidden layer, $w_{fi}$, $w_{Ii}$, $w_{ci}$, $w_{oi}$ are the weight coefficients of each gate state; $b_{fi}$, $b_{Ii}$, $b_{ci}$, $b_{oi}$ are the bias coefficients of each gate state, *lstmnum* is the number of hidden layers. $\sigma$ is the sigmoid activation function and the tanh is also the activation function, their formulas are shown as follows:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad (33)$$

$$\tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \quad (34)$$

The outputs of the LSTM output layer are as follows:

$$onet_l(k) = w_{mh}h_i(k) + b_{mh} \quad (35)$$

$$out_l(k) = \sigma\left(onet_l(k)\right) \quad (36)$$

where $w_{mh}$ and $b_{mh}$ are the weight and bias coefficients. Finally, the tuned parameters of partial-form MFAC are determined as follows:

$$\begin{aligned} \lambda &= out_{l1}(k) \\ \rho_l &= out_{l(l+1)}(k), \quad l = 1, 2, \cdots, L \end{aligned} \quad (37)$$

Combining the systematic error $e(k)$, the control input $u(k)$ can be calculated. Take Eq. (12) as the performance indicator function.The weight and bias coefficients to be learned in LSTM are updated using a chain-based backpropagation algorithm (BPTT). For the sake of brevity, only the updated formulas of the weight coefficients are presented as follows:

$$\begin{aligned} w_{fi}(k+1) &= w_{fi}(k) - \eta\frac{\partial J}{\partial w_{fi}} \\ \frac{\partial J}{\partial w_{fi}} &= \frac{\partial J}{\partial y(k+1)}\frac{\partial y(k+1)}{\partial u(k)}\frac{\partial u(k)}{\partial out_l(k)}\frac{\partial out_l(k)}{\partial onet_l(k)} \\ &\quad \times\frac{\partial onet_l(k)}{\partial h_i(k)}\frac{\partial h_i(k)}{\partial c_i(k)}\frac{\partial c_i(k)}{\partial f_i(k)}\frac{\partial f_i(k)}{\partial net_{fi}(k)} \\ &\quad \times\frac{\partial net_{fi}(k)}{\partial w_{fi}} \end{aligned} \quad (38)$$

$$
\begin{aligned}
w_{Ii}(k+1) &= w_{Ii}(k) - \eta \frac{\partial J}{\partial w_{Ii}} \\
\frac{\partial J}{\partial w_{Ii}} &= \frac{\partial J}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \frac{\partial u(k)}{\partial out_l(k)} \frac{\partial out_l(k)}{\partial onet_l(k)} \\
&\quad \times \frac{\partial onet_l(k)}{\partial h_i(k)} \frac{\partial h_i(k)}{\partial c_i(k)} \frac{\partial c_i(k)}{\partial I_i(k)} \frac{\partial I_i(k)}{\partial net_{Ii}(k)} \\
&\quad \times \frac{\partial net_{Ii}(k)}{\partial w_{Ii}}
\end{aligned} \tag{39}
$$

$$
\begin{aligned}
w_{ci}(k+1) &= w_{ci}(k) - \eta \frac{\partial J}{\partial w_{ci}} \\
\frac{\partial J}{\partial w_{ci}} &= \frac{\partial J}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \frac{\partial u(k)}{\partial out_l(k)} \frac{\partial out_l(k)}{\partial onet_l(k)} \\
&\quad \times \frac{\partial onet_l(k)}{\partial h_i(k)} \frac{\partial h_i(k)}{\partial c_i(k)} \frac{\partial c_i(k)}{\partial \tilde{c}_i(k)} \frac{\partial \tilde{c}_i(k)}{\partial net_{ci}(k)} \\
&\quad \times \frac{\partial net_{\tilde{c}i}(k)}{\partial w_{ci}}
\end{aligned} \tag{40}
$$

$$
\begin{aligned}
w_{oi}(k+1) &= w_{oi}(k) - \eta \frac{\partial J}{\partial w_{oi}} \\
\frac{\partial J}{\partial w_{oi}} &= \frac{\partial J}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \frac{\partial u(k)}{\partial out_l(k)} \frac{\partial out_l(k)}{\partial onet_l(k)} \\
&\quad \times \frac{\partial onet_l(k)}{\partial h_i(k)} \frac{\partial h_i(k)}{\partial o_i(k)} \frac{\partial o_i(k)}{\partial net_{oi}(k)} \frac{\partial net_{oi}(k)}{\partial w_{oi}}
\end{aligned} \tag{41}
$$

$$
\begin{aligned}
w_{mh}(k+1) &= w_{mh}(k) - \eta \frac{\partial J}{\partial w_{mh}} \\
\frac{\partial J}{\partial w_{mh}} &= \frac{\partial J}{\partial y(k+1)} \frac{\partial y(k+1)}{\partial u(k)} \frac{\partial u(k)}{\partial out_l(k)} \frac{\partial out_l(k)}{\partial onet_l(k)} \\
&\quad \times \frac{\partial onet_l(k)}{\partial w_{mh}}
\end{aligned} \tag{42}
$$

where $\eta$ is the learning rate parameter. The updated principle of the bias coefficients is the same as the update of the weight coefficients. The most important component of the above weight update is $\partial u(k)/\partial out_l(k)$, which is the partial derivative $u(k)$ of with respect to $\lambda$ and $\rho_l$, and the formulas are as follows:

$$
\begin{aligned}
\frac{\partial u(k)}{\partial \lambda} &= -\frac{\rho_1 \hat{\phi}_1(k)\,(y^*(k+1) - y(k))}{\left(\lambda_s + \left|\hat{\phi}_1(k)\right|^2\right)^2} \\
&\quad + \frac{\hat{\phi}_1(k) \sum_{l=2}^{L} \rho_l \hat{\phi}_l(k) \Delta u(k-l+1)}{\left(\lambda + \left|\hat{\phi}_1(k)\right|^2\right)^2}
\end{aligned} \tag{43}
$$

$$
\frac{\partial u_l(k)}{\partial \rho_l} = \begin{cases} \dfrac{\hat{\phi}_1(k)\,(y^*(k+1) - y(k))}{\lambda + \left|\hat{\phi}_1(k)\right|^2}, & l = 1 \\ -\dfrac{\hat{\phi}_1(k)\hat{\phi}_l(k)\Delta u(k-l+1)}{\lambda + \left|\hat{\phi}_1(k)\right|^2}, & 2 \le l \le L \end{cases} \tag{44}
$$

### C. CONTROL SCHEME

The general framework of the proposed method is shown in Fig. 5, and the $z^{-1}$ denotes the backward time shift. An improved partial-form MFAC based on neural networks for discrete-time nonlinear systems is proposed. To be specific, this proposed method takes the I/O information vector of the controlled system as the input of RBF neural networks, and the RBF neural networks estimate the PG values online. Then the system error set and the gradient information set are taken together as the input of LSTM neural networks, and essential parameters $\lambda$ and $\rho$ are tuned online by LSTM neural networks. These introduced neural networks all take the one-step-ahead squared error as the performance indicator function, quickly training and updating the weight and bias coefficients. The PG estimation value and tuned parameters $\lambda$ and $\rho$ are passed into controller, then the input signal $u(k)$ and the output $y(k+1)$ can be calculated.

In summary, the control scheme of the proposed method is constructed as follows:

1) PG online estimation by RBF neural networks:

$$
\hat{\phi}_l(k) = \sum_{n=1}^{m} w_{rbf}(k-1) R_n(\boldsymbol{x_{rbf}}(k)) \tag{45}
$$

$$
\hat{\boldsymbol{\phi}}_{p,L}(k) = [\hat{\phi}_1(k), \hat{\phi}_2(k), \cdots, \hat{\phi}_L(k)] \tag{46}
$$

2) Essential parameters online tuning by LSTM neural networks:

$$
out_l(k) = \sigma\left(w_{lstm} h(\boldsymbol{x_{lstm}}(k)) + b_{lstm}\right) \tag{47}
$$

$$
\begin{aligned}
\lambda &= out_{l1}(k) \\
\rho_l &= out_{l(l+1)}(k)
\end{aligned} \tag{48}
$$

3) Control method:

$$
\begin{aligned}
u(k) &= u(k-1) + \frac{\rho_1 \hat{\phi}_1(k)\,(y^*(k+1) - y(k))}{\lambda + \left|\hat{\phi}_1(k)\right|^2} \\
&\quad - \frac{\hat{\phi}_1(k) \sum_{i=2}^{L} \rho_i \hat{\phi}_i(k) \Delta u(k-i+1)}{\lambda + \left|\hat{\phi}_1(k)\right|^2}
\end{aligned} \tag{49}
$$

$$
\Delta \boldsymbol{U}_L(k) = [\Delta u(k), \cdots, \Delta u(k-L+1)]^{\mathrm{T}} \tag{50}
$$

$$
y(k+1) = y(k) + \boldsymbol{\phi}_{p,L}^{\mathrm{T}}(k) \Delta \boldsymbol{U}_L(k) \tag{51}
$$

4) Weight coefficient training of neural networks:

$$
w_{nl}(k+1) = w_{rbf}(k) - \beta \frac{\partial J}{\partial w_{rbf}(k)} + \alpha \Delta w_{rbf}(k) \tag{52}
$$

$$
w_{lstm}(k+1) = w_{lstm}(k) - \eta \frac{\partial J}{\partial w_{lstm}} \tag{53}
$$

where $R_n(\cdots)$ represents the output of RBFNN's hidden layer, $h(\cdots)$ represents the output of LSTM's hidden layer, $\boldsymbol{x_{rbf}}(k)$ and $\boldsymbol{x_{lstm}}(k)$ represent the inputs of RBF neural networks and LSTM neural networks respectively. Eq. (52) and (53) show the weight coefficients training of output layers. The weight coefficients of the other layers are trained similarly, which is omitted here for brevity.

## IV. SIMULATION AND EXPERIMENTAL RESULTS

In this section, two simulations, single input single output (SISO) discrete nonlinear system simulation, and three-tank system simulation are given to demonstrate the

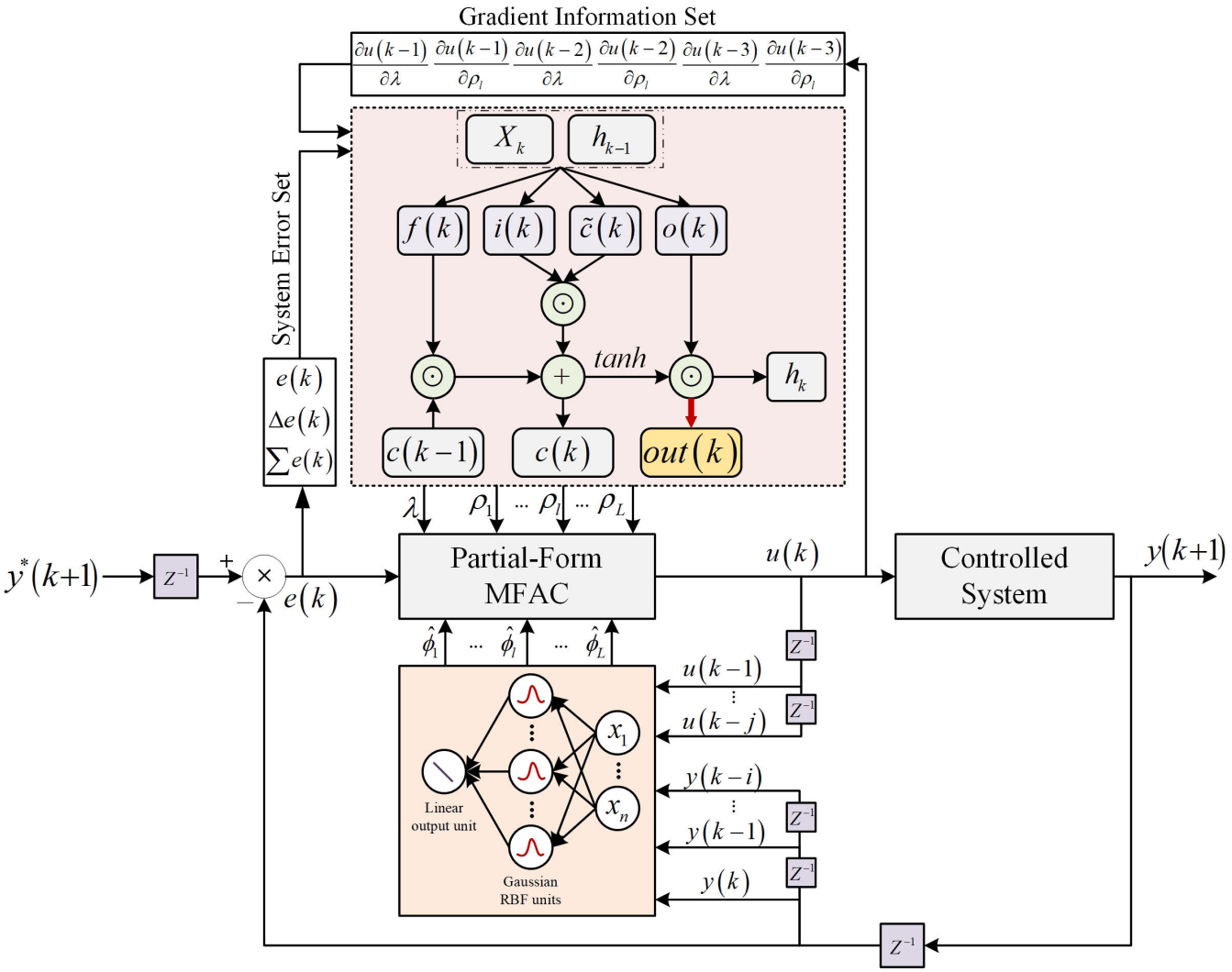



**FIGURE 5.** Structure diagram of the proposed algorithm.

effectiveness and applicability of the proposed method. The comparison methods selected in these two simulations are PID [15], partial-form MFAC [8], BP-based partial-form MFAC [27], and PSO-based partial-form MFAC [28], which are named by the following abbreviations for simplicity: PID, PFMFAC, PFMFAC-BP, and PFMFAC-PSO. Besides, to evaluate the effectiveness of PG estimation and essential parameters online tuning, a temporary tested method named PFMFAC-PG is introduced for subsequent ablation analysis. PFMFAC-PG can be considered as IPFMFAC-NN without the parameter tuning module based on LSTM.

### A. SISO DISCRETE NONLINEAR SYSTEM SIMULATION

A classical SISO discrete nonlinear system can be described as [9]:

$$\begin{aligned} y(k+1) = {} & \frac{2.5y(k)y(k-1)}{1+y^2(k)+y^2(k-1)} + 1.2u(k) \\ & + 0.09u(k)u(k-1) + 1.6u(k-2) \\ & + 0.7\sin(0.5(y(k)+y(k-1))) \\ & \times \cos(0.5(y(k)+y(k-1))) \end{aligned} \tag{54}$$

The desired value of the system output is as follows:

$$y^*(k+1) = 5\sin(k\pi/50) + 2\cos(k\pi/20) \tag{55}$$

The proposed method initialization work is shown in the table 1. The control input linearization length $L$ is 3, which means that the number of PG estimation at each time step is 3, and the number of parameters to be tuned at each time step is 4 ($\lambda$, $\rho_1$, $\rho_2$ and $\rho_3$). In neural networks, the number of hidden layers is approximately twice the number of input layers, which can fully learn nonlinear features. The initial values of the partial-form MFAC parameters in the table 1 are consistent with the initial values set in the cited reference [9].

All tracking curves of tested methods are shown in Fig. 6 and Fig. 7, which illustrate the control performance of each tested methods. It should be noted that the tracking curve of PFMFAC in Fig. 6 is almost consistent with the corresponding experimental tracking curve in the cited reference [9] so that it can be used as the benchmark for the comparison of the proposed algorithm.

**TABLE 1. Initialization in SISO discrete nonlinear system simulation.**

| Initialization object | Value |
|---|---|
| $y(k), k = 1, 2$ | 0 |
| $u(k), k = 1, 2, 3$ | 0 |
| $\mu$ | 1 |
| $\varepsilon$ | $10^{-5}$ |
| $L$ | 3 |
| $\hat{\phi}_{p,L}(k), k = 1$ | [1 0 0] |
| $\lambda$ | 0.5 |
| $\rho_l, l = 1, 2, \cdots, L$ | 0.5 |
| $RBF$ layers number | 5-10-3 |
| $LSTM$ layers number | 15-29-4 |
| $\beta$ | 0.5 |
| $\alpha$ | 0.02 |
| $\eta$ | 0.05 |

Fig. 6 demonstrates the tracking curves of PFMFAC, PFMFAC-PG, and the proposed method. As shown in Fig. 6, The tracking performance of these three methods is improved gradually. In the first 20s, the tracking curves of PFMFAC and PFMFAC-PG have apparent fluctuations, and the fluctuation of the PFMFAC-PG tracking curve decreases rapidly after 20s. In comparison, the PFMFAC tracking curve does not show a steady trend until after the 40s, reflecting that PG values estimated by RBF neural networks are closer to the ideal value. Compared with PFMFAC-PG, the proposed IPFMFAC-NN performs better in tracking the target curve, not only has the smaller fluctuation in the first 20s but also can track the target curve smoothly in the follow-up, proving that the online tuning of essential parameters $\lambda$ and $\rho$ can further optimize the control performance.

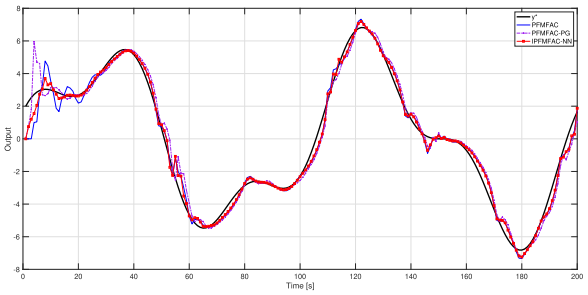



**FIGURE 6. Tracking curves of PFMFAC, PFMFAC-PG, IPFMFAC-NN.**

Fig. 7 compares the proposed algorithm with methods in other cited references. Similar to Fig. 6, the proposed method achieves the best tracking performance among all tested methods, which can quickly track the target curve while tracking curves of other methods show fluctuations of different degrees in the first 20s. After the 20s, except for the PID method, all tested methods can track the target curve well, of which the proposed method has the best tracking performance, proving the superiority of RBF neural networks and LSTM neural networks in optimizing the tracking performance. Take the period between the 20s and 35s as an example, there is a nonnegligible gap between the tracking curve of other algorithms and the target curve, and the tracking curve

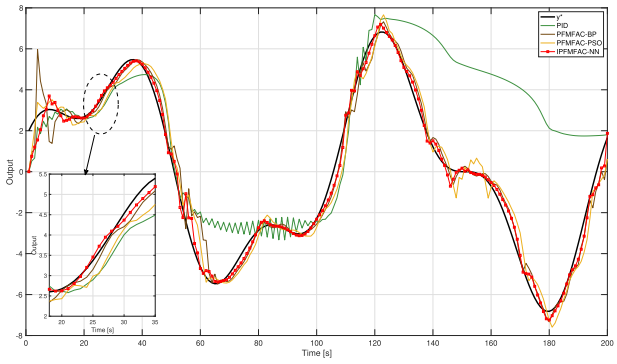



**FIGURE 7. Tracking curves of cited methods and IPFMFAC-NN.**

of the proposed method is closer to the target curve, which can be found from the detailed view in Fig. 7.

Fig. 8 demonstrates the controller input curves of each algorithm. Each method control input curve is similar to the corresponding tracking curve trend in Fig. 6 and Fig. 7. It can be found that the control input curve of the proposed method has smaller fluctuations than other curves, indicating that the proposed method's control input is more stable

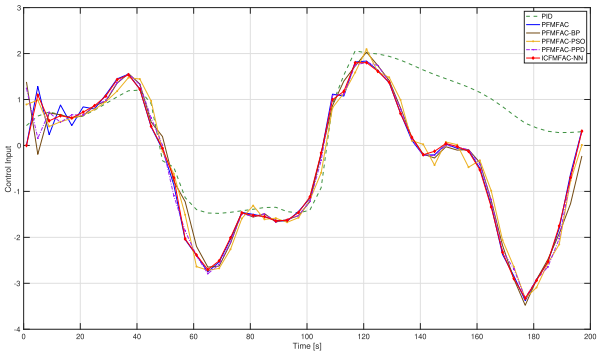



**FIGURE 8. Control input of all tested methods.**

Fig. 9 demonstrates the parameter online tuning results of $\lambda$ and $\rho$. As described in these two subfigures, the proposed algorithm can tune these essential parameters sensitively, and the difference between parameters tuning results and the corresponding default values is relatively small, which guarantees the stability of parameters online tuning work. Therefore, combined with Fig. 6, the proposed method can tune essential parameters online sensitively to obtain better tracking performance than ordinary partial-form MFAC, reflecting the effectiveness of parameters online tuning. Moreover, it is worth noting that value curves of $\lambda$ and $\rho$ have a certain degree of similarity which can be explained with the theoretical analysis. Eq. (7) presents the control scheme which can be converted to the following form:

$$\Delta u(k) = \frac{\rho_1 \hat{\phi}_1(k)\left(y^*(k+1) - y(k)\right)}{\lambda + \left|\hat{\phi}_1(k)\right|^2} - \frac{\hat{\phi}_1(k)\sum_{i=2}^{L} \rho_i \hat{\phi}_i(k)\Delta u(k-i+1)}{\lambda + \left|\hat{\phi}_1(k)\right|^2}$$

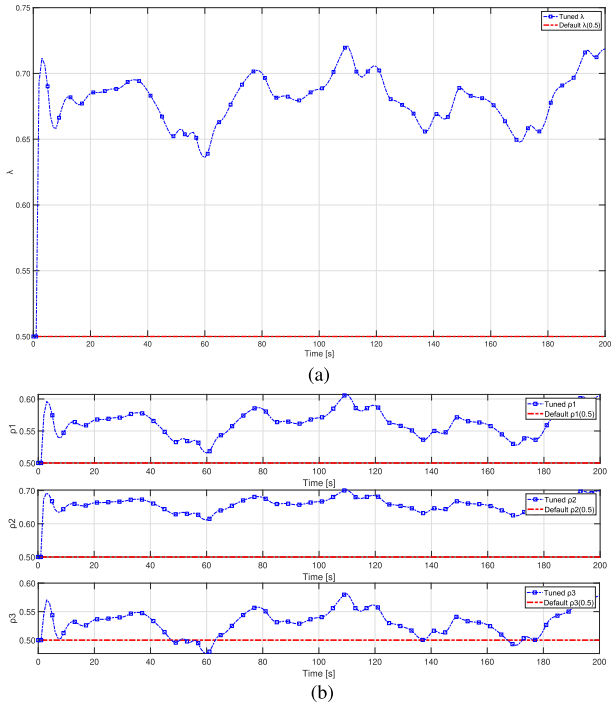



**FIGURE 9.** Parameter tuning results of λ(a) and $\rho$(b).

$$= \frac{\rho_1}{\lambda + \left|\hat{\phi}_1(k)\right|^2}\hat{\phi}_1(k)\left(y^*(k+1) - y(k)\right) - \frac{\rho_2}{\lambda + \left|\hat{\phi}_1(k)\right|^2}\hat{\phi}_1(k)\hat{\phi}_2(k)\Delta u(k-1) - \frac{\rho_3}{\lambda + \left|\hat{\phi}_1(k)\right|^2}\hat{\phi}_1(k)\hat{\phi}_3(k)\Delta u(k-2) \vdots - \frac{\rho_L}{\lambda + \left|\hat{\phi}_1(k)\right|^2}\hat{\phi}_1(k)\hat{\phi}_L(k)\Delta u(k-L+1) \quad (56)$$

where λ and $\rho$ are introduced to ensure the smoothness of input $u(k)$, and the fluctuation of the PG value is relatively small, which means λ and $\rho$ play a role in preventing the term $\rho_l\phi_1(k)/\left(\lambda + |\phi_1(k)|^2\right)$ from changing too much. Therefore, the value curves of λ and $\rho$ have similar trends

Fig. 10, Fig. 11, and Fig. 12 demonstrate the PG estimated value curves of partial-form MFAC and the proposed method. As described in these three figures, the proposed method's PG estimation curves tend to be stable after a short period of fluctuation. In contrast, the dynamics of PFMFAC's PG values are so complicated that the default projection estimation algorithm cannot track its true values well, which results in its tracking performance is not as good as the IPFMFAC-NN in Fig. 6.

All the figures shown above can prove that the proposed algorithm can accurately estimate PG values of partial-form MFAC and tune essential parameters λ and $\rho$ online sensitively. Therefore, it can achieve the best tracking

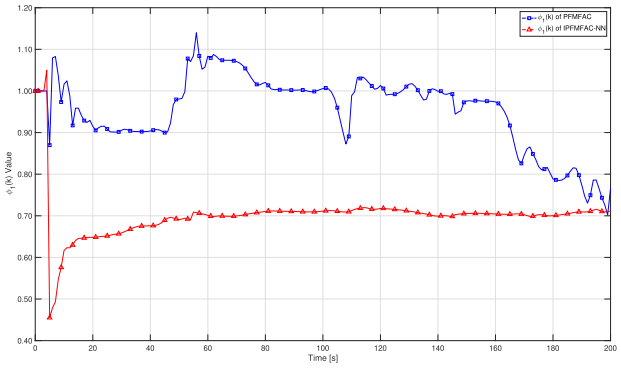



**FIGURE 10.** $\hat{\phi}_1$ estimated value curve.

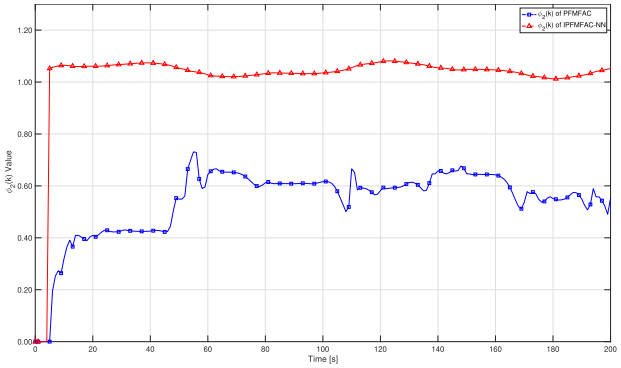



**FIGURE 11.** $\hat{\phi}_2$ estimated value curve.

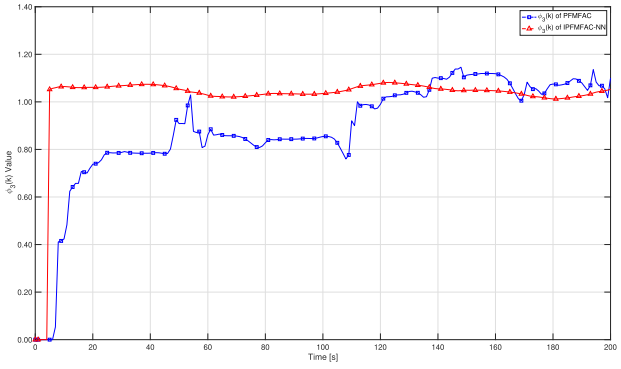



**FIGURE 12.** $\hat{\phi}_3$ estimated value curve.

performance in the SISO discrete nonlinear system simulation, proving the superiority of RBF neural networks and LSTM neural networks.

### B. THREE-TANK SYSTEM SIMULATION

The three-tank system [39] is a typical nonlinear and time-delayed object, which belongs to the controlled object in the liquid level control system. The output $Y$ (cm) of the system is the liquid level of the third tank, and the control input $U$ is the flow opening (%) into the tank. The working principle of the three-tank system is: when the liquid level rises to a certain high and low pressure, the outflow is increased to be equal to the inflow, to re-establish a balanced relationship, the liquid level finally stabilized at a certain height, the schematic diagram of the three-tank system is

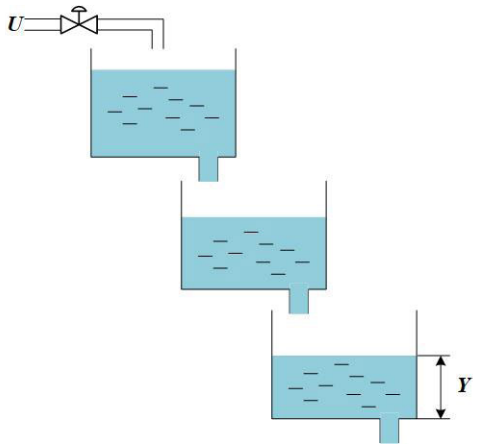



**FIGURE 13.** Schematic diagram of the three-tank system.

shown in Fig. 13. The transfer function of the liquid level $Y$ and control input $U$ in the simple three-tank system is calculated as follows:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{Ke^{-\tau s}}{(T_1 s + 1)(T_2 s + 1)(T_3 s + 1)} \tag{57}$$

where $K$ is system gain, $\tau$ is delay factor, $T1$, $T2$, $T3$ are time constants. In this simulation, the following parameters are selected:

$$[K \quad \tau \quad T_1 \quad T_2 \quad T_3] = \begin{cases} [4.5 \quad 24 \quad 8 \quad 8 \quad 8] & k < 400 \\ [5 \quad 24 \quad 8 \quad 8 \quad 8] & 400 \le k < 800 \\ [5 \quad 40 \quad 6 \quad 6 \quad 6] & 800 \le k < 1000 \end{cases} \tag{58}$$

Based on the transfer function and above mentioned parameters, the following three-tank system can be calculated:

$$y(k+1) = \begin{cases} 2.6475\, y(k) - 2.3364\, y(k-1) + 0.6873\, y(k-2) \\ +0.001334\, u(k-24) + 0.00486\, u(k-25) \\ +0.001106\, u(k-26), k < 400 \\ 2.6475\, y(k) - 2.3364\, y(k-1) + 0.6873\, y(k-2) \\ +0.001482\, u(k-24) + 0.0054\, u(k-25) \\ +0.001229\, u(k-26), 400 \le k < 800 \\ 2.5394\, y(k) - 2.1496\, y(k-1) + 0.6065\, y(k-2) \\ +0.003406\, u(k-40) + 0.01203\, u(k-41) \\ +0.0026\, u(k-42), 800 \le k < 1000 \end{cases} \tag{59}$$

The desired value of the system output is as follows:

$$y^*(k) = 10 \tag{60}$$

The proposed method initialization work is shown in the table2. The control input linearization length $L$ is 3, which means that the number of RBF neural networks output layers is 3, and the number of LSTM neural networks output layers is 4 ($\lambda$, $\rho_1$, $\rho_2$ and $\rho_3$). To prove the superiority of the proposed method, the initial values of the partial-form MFAC

**TABLE 2.** Initialization in three-tank system simulation.

| Initialization object | Value |
|---|---|
| $y(k), k = 1, 2$ | 0 |
| $u(k), k = 1, 2, 3$ | 0 |
| $\mu$ | 1 |
| $\varepsilon$ | $10^{-5}$ |
| $L$ | 3 |
| $\hat{\phi}_{p,L}(k), k = 1$ | [1 0 0] |
| $\lambda$ | 15 |
| $\rho_l, l = 1, 2, \cdots, L$ | 0.05 |
| $RBF$ layers number | 7-15-3 |
| $LSTM$ layers number | 15-29-4 |
| $\beta$ | 0.5 |
| $\alpha$ | 0.02 |
| $\eta$ | 0.05 |

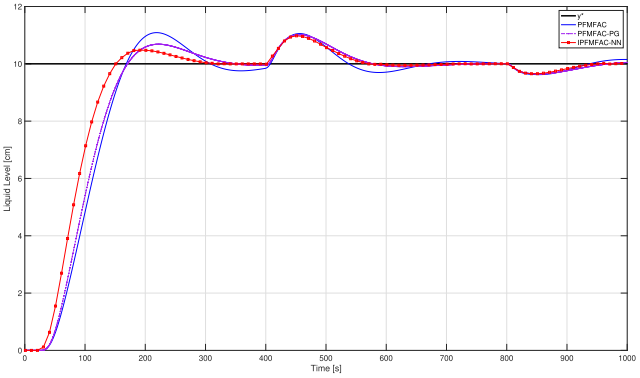



**FIGURE 14.** Tracking curves of PFMFAC, PFMFAC-PG, IPFMFAC-NN.

parameters in the table 2 are consistent with the initial values set in the cited reference [39].

Similar to the SISO discrete nonlinear system simulation, it also should be noted that the tracking curve of PFMFAC in Fig. 14 is almost consistent with the corresponding experimental curves in the cited reference [39], which can be regarded as the benchmark to reflect the superiority of the proposed method.

All tracking curves of tested methods are shown in Fig.14 and Fig.15, which are used to illustrate each tested method's control performance. Fig. 14 demonstrates the tracking curves of PFMFAC, PFMFAC-PG, and the proposed method. As described in Fig. 14, in the first 400s, the proposed method has the shortest rise time and tends to be stable quickly, while the other two tested methods have longer rise time and larger overshoots. Similarly, between 400s and 800s, the tracking curves of the proposed method and PFMFAC-PG tend to be stable at 600s after a small fluctuation. In comparison, the PFMFAC method does not reach a steady-state until 650s later, reflecting the effectiveness of the PG estimation method for improving the tracking performance. In the last 200s, the tracking performances of all tested methods are very close. From the overall point of view, the proposed method's tracking performance is the best among the three, showing the necessity of PG value estimation and parameters online tuning in the partial-form MFAC.

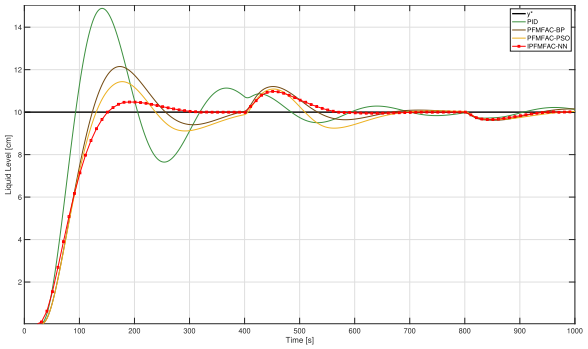



**FIGURE 15.** Tracking curves of cited methods and IPFMFAC-NN.

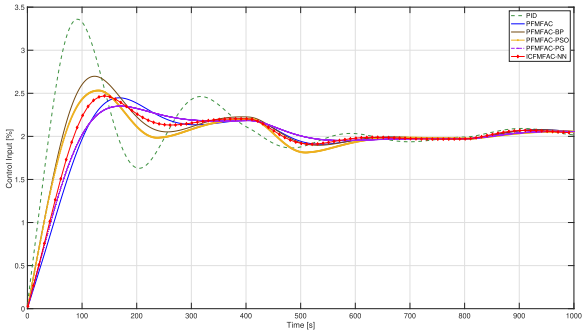



**FIGURE 16.** Control input of all tested methods.

In Fig. 15, the proposed method is compared with other methods in cited references. From the overall tracking curves of all tested methods, the proposed method achieves the best tracking performance, tracking the target value quickly, and has the best stability. Although the other three tested methods' rise time is shorter, their curves fluctuate considerably and fail to reach a stable state in the first 400s, showing that these algorithms cannot guarantee control stability while shortening the response speed. The algorithm proposed in this article can reach a stable state fastest and has the smallest overshoot, which reflects the superiority of RBF neural networks and LSTM neural networks for algorithm optimization

Fig. 16 demonstrates the controller input curves of each algorithm. Each method control input curve is similar to the corresponding tracking curve trend in Fig. 14 and Fig. 15.

Fig. 17 demonstrates the parameter online tuning results of parameter $\lambda$ and $\rho$. As described in these two subfigures, these tuned parameter values are relatively close to the corresponding default values, which guarantees the accuracy of parameters online tuning work. Therefore, compared with ordinary partial-form MFAC, the proposed method can tune parameters online sensitively to obtain better tracking performance, reflecting the effectiveness of LSTM neural networks in parameters tuning. Besides, the changes of these two kind of tuned parameters curves have a certain degree of similarity, the reason for this phenomenon is similar to that in the SISO discrete nonlinear system simulation.

Fig. 18, Fig. 19, and Fig.20 demonstrate the PG estimated value curves of partial-form MFAC and the proposed method. As shown in these three figures, PG values of PFMFAC

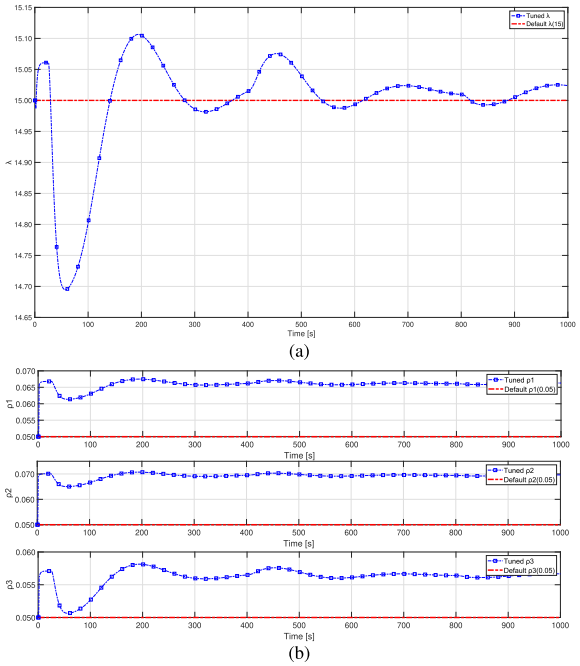



**FIGURE 17.** Parameter tuning results of $\lambda$(a) and $\rho$(b).

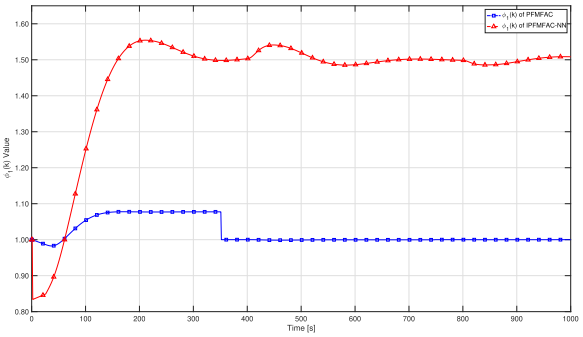



**FIGURE 18.** $\hat{\phi}_1$ estimated value curve.

are equal to the initial values $\hat{\boldsymbol{\phi}}_{p,L}(1) = [1 \quad 0 \quad 0]$ after 350s, indicating that the PG parameter estimation algorithm in the partial-form MFAC triggers the reset mechanism after 350s. Therefore, in this three-tank system simulation with a large time-lag, the PG value calculated by the PFMAFC projection algorithm is constant after 350s and cannot reflect the controlled system's dynamic characteristics. In contrast, the algorithm proposed in this article can estimate the PG value very well, and estimated values can tend to be stable within a specific range after the initial stage of rising, proving RBF neural networks' effectiveness.

All the figures shown above can prove that the PG projection algorithm in partial-form MFAC will fail to deal with control process with a large time-lag, and the proposed algorithm can accurately estimate PG of partial-form MFAC and tune the parameters $\lambda$ and $\rho$ online correctly and sensitively. Therefore, it can achieve the best tracking performance in the three-tank water system simulation, proving the effectiveness of the introduced RBF neural networks and LSTM neural networks.

**TABLE 3.** Evaluation results of SISO discrete nonlinear system simulation.

| | $RMSE$ | $IAE$ | $IAVU$ | $MO$ | $ICR(0.1)$ |
|---|---|---|---|---|---|
| PID [15] | 3.812 | 526.086 | 69.755 | 9.602 | 0.960 |
| PFMFAC [8] | 0.771 | 89.523 | 75.022 | 2.844 | 0.830 |
| PFMFAC-BP [27] | 0.703 | 97.012 | 52.389 | 3.250 | 0.825 |
| PFMFAC-PSO [28] | 0.751 | 119.099 | 46.332 | 1.821 | 0.915 |
| PFMFAC-PG | 0.651 | 89.357 | 42.488 | 3.233 | 0.815 |
| **IPFMFAC-NN** | **0.433** | **59.348** | **34.803** | **1.344** | **0.660** |

**TABLE 4.** Evaluation results of three-tank system simulation.

| | $RMSE$ | $IAE$ | $IAVU$ | $MO$ | $ICR(0.1)$ |
|---|---|---|---|---|---|
| PID [15] | 2.899 | 1398.005 | 29.815 | 4.885 | 0.869 |
| PFMFAC [8] | 2.754 | 1280.840 | 16.455 | 1.088 | 0.750 |
| PFMFAC-BP [27] | 2.689 | 1204.468 | 19.717 | 2.147 | 0.747 |
| PFMFAC-PSO [28] | 2.674 | 1199.050 | 19.168 | 1.435 | 0.772 |
| PFMFAC-PG | 2.652 | 1186.492 | 14.469 | 1.016 | 0.594 |
| **IPFMFAC-NN** | **2.578** | **991.334** | **14.312** | **0.978** | **0.532** |

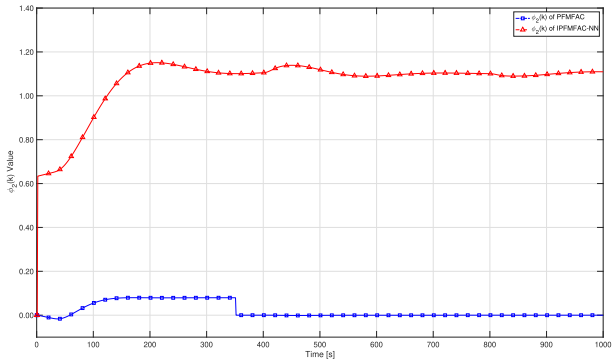



**FIGURE 19.** $\hat{\phi}_2$ estimated value curve.

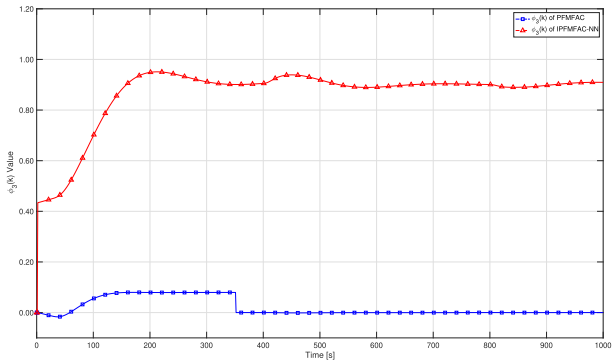



**FIGURE 20.** $\hat{\phi}_3$ estimated value curve.

### C. SIMULATION RESULTS AND ANALYSIS

To further analyze the control performance of the proposed method, five individual performance indices are introduced for quantitative comparison. The first two indices are used to evaluate each method's tracking accuracy, which are the root mean square error (*RMSE*) and the integral absolute error (*IAE*). The integral absolute variation of the control signal (*IAVU*) is used to evaluate the stability of the control input. This article proposes two other indices to evaluate the algorithm. The first is the maximum overshoot (*MO*), which is presented to evaluate the tracking instability, and the imprecise control ratio (*ICR*) is used to calculate the time proportion of imprecise control. These five indices are expressed in (61) – (65) as follows:

$$RMSE = \sqrt{\frac{1}{N}\sum_{k=1}^{N} e(k)^2} \tag{61}$$

$$IAE = \int_0^t |e_i(t)|\, dt \tag{62}$$

$$IAVU = \int_0^t \left|\frac{du(t)}{dt}\right| dt \tag{63}$$

$$MO = \max\left(\left(y(1) - y^*(1)\right), \cdots, \left(y(N) - y^*(N)\right)\right) \tag{64}$$

$$ICR(\xi) = \frac{1}{N}\sum\nolimits_{k=1}^{N} IC(k, \xi)$$
$$IC(k, \xi) = \begin{cases} 0 & when \left|y(k) - y^*(k)\right| < \xi \\ 1 & when \left|y(k) - y^*(k)\right| \geq \xi \end{cases} \tag{65}$$

The performance evaluation results of each algorithm in the experiment are presented in Tables 3 and 4. It is easy to find that the proposed method performs best in all indices in two simulations. In comparison with the original PFMFAC, both the proposed algorithm and the PFMFAC-PG exhibit significant improvement in various indices, demonstrating that accurate estimation of PG and essential parameters online tuning of partial-form MFAC is necessary and meaningful in the field of control theory.

#### 1) ANALYSIS OF SISO DISCRETE NONLINEAR SYSTEM SIMULATION

For results of SISO discrete nonlinear system simulation as listed in Table 3, PID has the worst performance on various indices and can be reflected in Fig. 7. PFMFAC-BP and PFMFAC-PSO introduce different algorithms to tune the parameters of partial-form MFAC. However, these two algorithms only have better performance in the two indices of *RMSE* and *IAVU*, and Fig. 7 shows that these two algorithms solve the problem of the fluctuation of the tracking curve in the first 40s, but the tracking performance is not as good as

PFMFAC afterward, indicating that the effect of parameter tuning is not stable enough.

PFMFAC-PG, which introduces RBF neural networks to estimate PG values, achieves better results than PFMFAC under all indices except *MO* in the first simulation, mainly because PFMFAC-PG has a more obvious fluctuation in the first 10s, but the tracking performance is better than PFMFAC afterwards. Therefore, it can be proved that the PG estimation by RBF neural networks is effective. In addition, in comparison with PFMFAC-BP, PFMFAC-PG achieves better results in all five indices and it is reduced by 7.39% in the *MFAC* index, which reflects the advantages of the local approximation neural network.

Furthermore, compared with PFMFAC, the proposed method is reduced by 43.83%, 33.71%, 53.61%, 52.74% and 20.48% in all indices, indicating that the proposed method has a significant improvement in control performance and control input stability. Besides, compared with PFMFAC-PG, the proposed method is reduced by 33.48% and 18.09% in the two main indices of *RMSE* and *IAVU*, proving the effectiveness of parameter tuning and the superiority of LSTM neural networks.

#### 2) ANALYSIS OF THREE-TANK SYSTEM SIMULATION

In the three-tank system simulation as listed in Table 4, all algorithms except PID have relatively small gaps in various indices, which can be reflected from these methods' tracking curves in Fig. 14 and Fig. 15. Similarly, tracking curves of PFMFAC-BP and PFMFAC-PSO have faster response speed but greater fluctuations than PFMFAC's curve, so they only have better performance in the two indices of *RMSE* and *IAVU*, indicating the necessity of introducing different indices to measure the effectiveness of the method.

In addition, PFMFAC-PG achieves better results than PFMFAC under all indices to verify the effectiveness of PG estimation by RBF neural networks, it is reduced by 3.70%, 7.36%, 12.07%, 6.61% and 20.80% in all indices. Similar to the results of the SISO discrete nonlinear system simulation, PFMFAC-PG achieves better results than PFMFAC-BP in all five indices, the most obvious of which is that it is reduced by 26.6% compared to PFMFAC-BP under the *IAVU* index, showing the advantages of RBF neural networks.

Finally, compared with PFMFAC, the proposed method is reduced by 6.39%, 23.91%, 13.02%, 10.11% and 29.07% in all indices, and it also performs better in all indices than PFMFAC-PG and reduces the two main indices of *RMSE* and *IAVU* by 2.68% and 0.95%. The above analysis proves the effectiveness of parameters tuning work and PG estimation, and also reflects the superiority of the proposed algorithm.

Generally speaking, the methods in the cited references only achieve better results than partial-form MFAC in some indices, indicating that the optimization effect is not good enough, and it also proves the necessity of multiple indices to measure the performance of the method. In addition, through ablation analysis, the optimization effects of the RBF neural network and the LSTM neural network are respectively proved. Therefore, the proposed method has the significant improvement in all indices, which shows the rationality of the algorithm design.

## V. CONCLUSION

This article proposes an improved partial-form MFAC named IPFMFAC-NN for a class of discrete-time nonlinear systems. The importance and originality of this article are that the proposed method introduces RBF neural networks to estimate PG values of partial-form MFAC and uses LSTM neural networks to tune essential parameters online sensitively, which significantly improves the control performance. The correctness of the proposed method has been verified by SISO discrete nonlinear system simulation and three-tank system simulation, and five evaluation indices are introduced. The experimental results demonstrate that the proposed algorithm achieves the best control performance among all tested methods. The proposed method thus has advantages in stability and accuracy. This joint optimization method is not available in previous theoretical results.

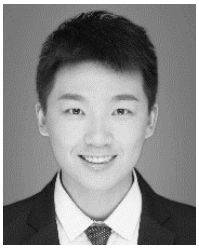
The proposed method in this article still has room for improvement. In Fig. 6 and Fig. 14, the tracking errors of IPFMFAC-NN do not converge to zero. In fact, the system output at the next moment may also be related to the system output in the sliding window. Therefore, in the future research, the partial-form dynamic linearization can be expanded to take into account the control input and output of the system in the sliding time window, which can better capture the complex dynamic characteristics of the controlled system.

More importantly, the proposed method is a pure DDC method and only validates its advantages in SISO discrete nonlinear system simulation and classical practical industrial scene like the three-tank system. However, it has not been evaluated in the actual industrial scene with some factors such as measurement noise and control saturation [40]. Therefore, in the future work, noise reduction techniques like wavelet denoising [41] will be introduced into the algorithm to deal with some disturbance factors in the actual production process, which is different from the experiments conducted in this article such as the chemical production process, oil refining production process, etc. Studying these factors in the actual industrial scene is an important research area from an engineering perspective.

## REFERENCES

[1] R. E. Kalman, "On the general theory of control systems," in *Proc. 1st Int. Conf. Autom. Control*, Moscow, Russia, 1960, pp. 481–492.
[2] R. Kalman and J. Bertram, "General approach to control theory based on the methods of lyapunov," *IRE Trans. Autom. Control*, vol. 4, no. 3, p. 20, Dec. 1959.
[3] B. Gough, B. Wilson, and D. Matovich, "Advanced model-based control for continuous process industries," in *Proc. IEEE Ind. Appl. Dyn. Model. Control Appl. Ind. Workshop*, Apr. 1998, pp. 33–39.
[4] H. Guo, J. Xu, and Y.-H. Chen, "Robust control of fault-tolerant permanent-magnet synchronous motor for aerospace application with guaranteed fault switch process," *IEEE Trans. Ind. Electron.*, vol. 62, no. 12, pp. 7309–7321, Dec. 2015.
[5] M. A. S. Aboelela, M. F. Ahmed, and H. T. Dorrah, "Design of aerospace control systems using fractional PID controller," *J. Adv. Res.*, vol. 3, no. 3, pp. 225–232, Jul. 2012.

[6] G. Franceschini and S. Macchietto, "Model-based design of experiments for parameter precision: State of the art," *Chem. Eng. Sci.*, vol. 63, no. 19, pp. 4846–4872, Oct. 2008.
[7] S. J. Qin, "Survey on data-driven industrial process monitoring and diagnosis," *Annu. Rev. Control*, vol. 36, no. 2, pp. 220–234, Dec. 2012.
[8] Z.-S. Hou and J.-X. Xu, "On data-driven control theory: The state of the art and perspective," *Acta Automatica Sinica*, vol. 35, no. 6, pp. 650–667, Oct. 2009.
[9] Z. Hou and S. Jin, *Model Free Adaptive Control: Theory and Applications*. Berlin, Germany: Springer, 2013.
[10] K. Heong Ang, G. Chong, and Y. Li, "PID control system analysis, design, and technology," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 4, pp. 559–576, Jul. 2005.
[11] I. Podlubny, "Fractional-order systems and fractional-order controllers," *Inst. Exp. Phys., Slovak Acad. Sci., Kosice*, vol. 12, no. 3, pp. 1–18, 1994.
[12] J. Shangtai, H. Zhongsheng, and W. Weihong, "Model-free adaptive control used in permanent magnet linear motor," in *Proc. Chin. Control Conf.*, Jul. 2006, pp. 748–751.
[13] J. C. Spall, "Simultaneous perturbation stochastic approximation," in *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley, 2003, pp. 176–207. [Online]. Available: https://onlinelibrary.wiley.com/action/showCitFormats?doi=10.1002%2F0471722138.ch7
[14] M. C. Campi, A. Lecchini, and S. M. Savaresi, "Virtual reference feedback tuning: A direct method for the design of feedback controllers," *Automatica*, vol. 38, no. 8, pp. 1337–1346, Aug. 2002.
[15] S. Abedini and H. Zarabadipour, "Tuning of an optimal PID controller with iterative feedback tuning method for DC motor," in *Proc. 2nd Int. Conf. Control, Instrum. Autom.*, Dec. 2011, pp. 611–615.
[16] G. Bontempi, M. Birattari, and H. Bersini, "Lazy learning for local modelling and control design," *Int. J. Control*, vol. 72, nos. 7–8, pp. 643–658, Jan. 1999.
[17] M.-B. Rădac, R.-E. Precup, E. M. Petriu, and S. Preitl, "Iterative data-driven tuning of controllers for nonlinear systems with constraints," *IEEE Trans. Ind. Electron.*, vol. 61, no. 11, pp. 6360–6368, Nov. 2014.
[18] R. R. Yacoub, R. T. Bambang, A. Harsoyo, and J. Sarwono, "Dsp implementation of combined fir-functional link neural network for active noise control," *Int. J. Artif. Intell.*, vol. 12, no. 1, pp. 36–47, 2014.
[19] P. Yan, D. Liu, D. Wang, and H. Ma, "Data-driven controller design for general mimo nonlinear systems via virtual reference feedback tuning and neural networks," *Neurocomputing*, vol. 171, pp. 815–825, 2016.
[20] F. Zouari, K. B. Saad, and M. Benrejeb, "Robust neuronal adaptive control for a class of uncertain nonlinear complex dynamical multivariable systems," *Int. Rev. Model. Simul.*, vol. 5, no. 5, pp. 2075–2103, 2012.
[21] A. Esparza, A. Sala, and P. Albertos, "Neural networks in virtual reference tuning," *Eng. Appl. Artif. Intell.*, vol. 24, no. 6, pp. 983–995, Sep. 2011.
[22] J. T. Jose, "Pseudo gradient guided particle swarm optimization for economic load dispatch of wind thermal system," in *Proc. 2nd Int. Conf. Current Trends Eng. Technol. (ICCTET)*, Jul. 2014, pp. 92–96.
[23] A. G. Bors, "Introduction of the radial basis function (RBF) networks," in *Proc. Online Symp. Electron. Eng.*, 1996, pp. 1–7.
[24] R. T. Gunasekara and S. Filizadeh, "Internal model design for power electronic controllers," in *Proc. 15th IET Int. Conf. AC DC Power Transmiss. (ACDC)*, 2019, pp. 1–6.
[25] W. Wang, Z. Hou, H. Huo, and S. Jin, "Data-driven based controller design and its parameters tuning method," *J. Syst. Sci. Math. Sci.*, vol. 30, no. 6, pp. 792–805, 2010.
[26] J. Wang, C. Ji, L. Cao, and Q. Jin, "Model free adaptive control and parameter tuning based on second order universal model," *J. Central South Univ.*, vol. 43, no. 5, pp. 1795–1802, 2012.
[27] C. Chen and J. Lu, "Design of self-tuning SISO partial-form model-free adaptive controller for vapor-compression refrigeration system," *IEEE Access*, vol. 7, pp. 125771–125782, 2019.
[28] S.-Z. Gao, X.-F. Wu, L.-L. Luan, J.-S. Wang, and G.-C. Wang, "PSO optimal control of model-free adaptive control for PVC polymerization process," *Int. J. Autom. Comput.*, vol. 15, no. 4, pp. 482–491, Aug. 2018.
[29] L. Yijun, T. Jiali, J. Hongfen, Z. Guangping, C. Dan, and Y. Zhimin, "GA-BP neural networks for environmental quality assessment," in *Proc. Int. Conf. Netw. Digit. Soc.*, vol. 2, 2010, pp. 126–129.
[30] Y. Yang, C. Chen, and J. Lu, "Parameter self-tuning of SISO compact-form model-free adaptive controller based on long short-term memory neural network," *IEEE Access*, vol. 8, pp. 151926–151937, 2020.
[31] R. Cao, L. Bai, and Z. Hou, "Study on model-free learning adaptive control in permanent magnet linear motor," in *Proc. Chin. Control Decis. Conf.*, Jul. 2008, pp. 2946–2949.
[32] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Comput.*, vol. 1, no. 2, pp. 281–294, Jun. 1989.
[33] R. Mahanty and P. D. Gupta, "Application of rbf neural network to fault classification and location in transmission lines," *IEE Proc.-Gener., Transmiss. Distrib.*, vol. 151, no. 2, pp. 201–212, 2004.
[34] S.-I. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, nos. 4–5, pp. 185–196, Jun. 1993.
[35] M. Gherrity, "A learning algorithm for analog, fully recurrent neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, 1989, pp. 643–644.
[36] P. Campolucci, A. Uncini, and F. Piazza, "Causal back propagation through time for locally recurrent neural networks," in *Proc. IEEE Int. Symp. Circuits Syst. Circuits Syst. Connecting World. (ISCAS)*, vol. 3, May 1996, pp. 531–534.
[37] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[38] F. A. Gers, J. Schmidhuber, and F. Cummins, *Learning to Forget: Continual Prediction With LSTM*. Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 1999. [Online]. Available: https://dl.acm.org/doi/book/10.5555/870468
[39] F. Tahir, N. Iqbal, and G. Mustafa, "Control of a nonlinear coupled three tank system using feedback linearization," in *Proc. 3rd Int. Conf. Electr. Eng.*, Apr. 2009, pp. 1–6.
[40] M. Bodson, J. S. Jensen, and S. C. Douglas, "Active noise control for periodic disturbances," *IEEE Trans. Control Syst. Technol.*, vol. 9, no. 1, pp. 200–205, Jan. 2001.
[41] S. Sardy, P. Tseng, and A. Bruce, "Robust wavelet denoising," *IEEE Trans. Signal Process.*, vol. 49, no. 6, pp. 1146–1152, Jun. 2001.

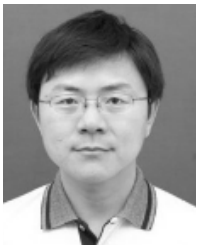


**YE YANG** received the B.S. degree from the Bell Honors School, Nanjing University of Posts and Telecommunications, Nanjing, China, in 2018. He is currently pursuing the Ph.D. degree with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Zhejiang, China.

His current research interests include artificial intelligence in industries, intelligent modeling, intelligent control, and industrial big data.

**CHEN CHEN** received the B.S. degree from the College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2018. She is currently pursuing the Ph.D. degree with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Zhejiang, China.

Her current research interests include data-driven control, intelligent control, and their industrial applications.



**JIANGANG LU** received the B.S. and Ph.D. degrees from Zhejiang University, China, in 1989 and 1995, respectively.

He is currently a Full Professor with the Institute of Industrial Process Control and the College of Control Science and Engineering, Zhejiang University. Up to now, he has published one book and more than 120 papers in journals and conferences. His research interests include artificial intelligence in industries, intelligent sense, intelligent modeling, intelligent control, intelligent optimization, and industrial big data. He is a member of the Technical Committee on Process Control in Chinese Association of Automation, and the standing Director of the Zhejiang Association of Automation. He has received the First Prize of China's National Science and Technology Progress Award in 2013, the Geneva International Invention Gold Medal Award in 2013, and the Second Prizes of Zhejiang Province Science and Technology Progress Award in 1999 and 2007.