Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 8935423, 13 pages
https://doi.org/10.1155/2020/8935423

Research Article

Position Control of Magnetic Levitation Ball Based on an Improved Adagrad Algorithm and Deep Neural Network Feedforward Compensation Control

Zhouxiang Wei, Zhiwen Huang, and Jianmin Zhu

College of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

Correspondence should be addressed to Jianmin Zhu; jmzhu_usst@163.com

Received 14 August 2020; Accepted 12 December 2020; Published 21 December 2020

Academic Editor: Xianming Zhang

To control the position of the magnetic levitation ball more accurately, this paper proposes a deep neural network feedforward compensation controller based on an improved Adagrad optimization algorithm. The control structure of the controller consists of a deep neural network identifier, a deep neural network feedforward compensator, and a PID controller. First, the dynamic inverse model of the magnetic levitation ball is established by the deep neural network identifier which is trained online based on the improved Adagrad algorithm, and the trained network parameters are dynamically copied to the deep neural network feedforward compensator. Then, the position control of the magnetic levitation ball system is realized by the output of the feedforward compensator and the PID controller. Simulations and experiments illustrate that the accuracy of the deep network feedforward compensation control based on an improved Adagrad algorithm is higher, and its control system shows good dynamic and static performance and robustness to some extent.

1. Introduction

Magnetic levitation system is widely used in magnetic bearing, magnetic suspension vibration isolator, magnetic suspension train, and many other fields because it is contactless, frictionless, and noiseless, etc. A magnetic levitation ball system is a magnetic levitation system with a single degree of freedom [1]. It is a typical control system to study the phenomenon of magnetic levitation and verify various controllers.

Recently, the controllers of the magnetic levitation ball have received a great deal of attention from a number of researchers, and they have done relevant research and obtained rich research results, such as PID control [2, 3], GPI control [4, 5], sliding mode control [6, 7], and robust control [8–10]. The design of the above controllers requires a dynamic model of a magnetic levitation ball system. However, the precise dynamic model of the magnetic levitation ball system is difficult to obtain, so the model-free control method is more suitable for it.

Feedforward control is an effective method for tracking problems, which predicts the external interference in advance and generates appropriate control law according to the prediction and input signals. Feedforward control can eliminate disturbance in time, so it can make the control system have higher control accuracy. It is widely used in various domains of the industry. In [11], the trajectory tracking problem of the cable-driven robot is solved by using feedforward compensation. In [12, 13], feedforward control has achieved excellent results in machine tool control. In [14], the hot rolling mill is well controlled by the feedforward controller. But feedforward control is difficult to design because the precise mathematical model of the controlled plant is usually difficult to obtain.

The neural network can approximate continuous functions closely, so there has been much research on the use of the neural network for model-free control. By training the neural network online or offline, the performance of the control system can be improved, and finally, a satisfactory

control effect can be achieved [15–20]. Nowadays, neural network control is promisingly used in magnetic levitation ball system. In [21], electromagnetic parameters are approximated by the neural network. On this basis, a neural network controller is proposed. In [22, 23], a magnetic levitation ball model is approximated by RBF neural networks. The model parameters are identified offline by applying the structured nonlinear parameter optimization method. Based on the model, a predictive controller is designed. In [24], a new neural network controller for a magnetic levitation ball system is proposed. The proposed controller comprises a neural network controller and a robust controller. In [25], a dynamic model of control system error and control law is established online by BP neural network. Based on the model, a neural network controller is designed with a PID controller. The controllers identify relevant parameters of the control system by the neural network, and the controller is designed by identified parameters. Although these controllers have achieved good results, the control accuracy can be further improved.

The control accuracy of the above methods is affected by the accuracy of the neural network model. Therefore, improving the modeling accuracy of the neural network is an effective way to improve the control accuracy of the neural network controller. Generally, the complex neural network can improve the modeling accuracy of the controlled plant. In order to fully train the complex neural network, multiple iterations are needed. However, the computation time generated by multiple iterations is difficult to meet the requirements of real-time control. In order to solve the above problems, this paper proposed a deep neural network feedforward compensation controller based on an improved Adagrad algorithm. First, a dynamic inverse model of the controlled plant is established online by the deep neural network, which is regarded as a model identifier. Meanwhile, the parameters of the identifier are dynamically copied to the feedforward compensator which has the same structure as the identifier. Thus, the proposed controller in this paper is designed by the deep neural network identifier, feedforward compensator, and PID controller. Then, an improved Adagrad optimization algorithm with better convergence accuracy and faster convergence rate is proposed to solve the problem of slow training of the neural network and network parameter delay caused by online training of the neural network controller. The improved Adagrad optimization algorithm was used to train the identifier. Finally, simulation and experiment are conducted to verify the effectiveness of the proposed controller. The main innovations of this article are summarized as follows:

(1) An improved Adagrad optimization algorithm is proposed to solve the problems that may occur in the online training of the neural network controller, such as delay, slow training, and control accuracy

(2) A deep neural network feedforward compensation controller is proposed by combining the PID controller with a deep neural network. The effect of the proposed controller is proved by simulation and experiment

The rest of the paper is organized as follows. Section 2 introduces the magnetic levitation ball control system. Section 3 introduces the feedforward compensation control method of a deep neural network based on an improved Adagrad optimization algorithm. Section 4 verifies the performance of the improved Adagrad optimization algorithm with benchmark function and conducts simulation and experimental research on the control method proposed in this paper. Section 5 is the conclusion.

2. Magnetic Levitation Ball Position Control System

2.1. Principle of Magnetic Levitation Ball. The working principle of the magnetic levitation ball control system studied in this paper is shown in Figure 1. In the real-time control, the light transmission area of the slit of the photoelectric position sensor and the illuminance of the silicon photocell change accordingly when the position of the steel ball changes in the vertical direction. The photoelectric position sensor transforms the displacement signal of the ball into a voltage signal that changes in proportion to the level of illumination. Then, the voltage signal is input to a computer after signal conditioning and A/D conversion. The controller calculates the control law after comparing the input computer signal with the command position of the maglev ball. After D/A conversion and power amplification, the control law controls the current i in the electromagnet winding, so that the electromagnet generates the corresponding electromagnetic attraction F, and then controls the position of the steel ball.

2.2. Modeling of Magnetic Levitation Ball. In order to build the mathematical model of the magnetic levitation ball, the following assumptions must be made:

(1) Magnetic flux leak, edge effect, and reluctance between the ball and the electromagnet are ignored

(2) The ball is a homogeneous sphere, and the magnetic force is concentrated towards the center of it

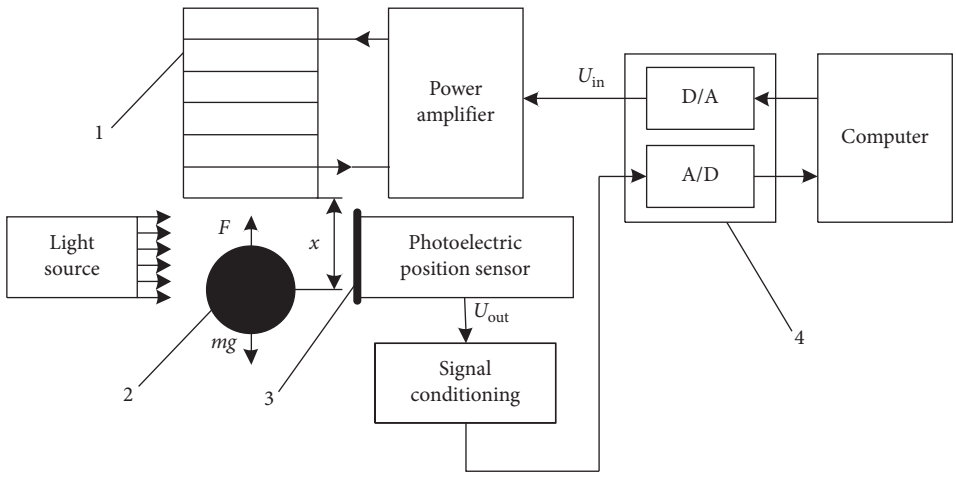
(3) There is a linear relationship between the output current and the input voltage of the power amplifier, and there is no delay

Based on the above assumptions, the mathematical model of the magnetic levitation ball system can be established through theoretical derivation.

The dynamic equation of the ball is derived from Newton's second law as follows:

$$m\frac{\mathrm{d}^2x(t)}{\mathrm{d}t^2} = F(i,x) + mg, \tag{1}$$

where m is the mass of the ball, g is the acceleration of gravity, x is the ball position, $F(i,x)$ is the electromagnetic force on the ball, and i is the instantaneous current of the electromagnet winding.



1 Electromagnet winding;
2 Magnetic levitation ball;
3 Light shield;
4 Data acquisition and control card

Figure 1: Principle of magnetic levitation ball position control system.

According to Kirchhoff laws and Biot-Savart Law of the magnetic circuit, the electromagnetic force on a ball can be deduced as follows:

$$F(i,x) = K\left(\frac{i}{x}\right)^2, \tag{2}$$

where K is the constant coefficient related to the magnetic flux of the electromagnet winding.

When the ball is in a state of balance, the formula is obtained according to the mechanical balance principle as follows:

$$mg + F(i_0, x_0) = 0, \tag{3}$$

where x_o and i_o are the air gap and the current in the coil when the magnetic levitation ball is in equilibrium.

Combined with equations (1)–(3), the formula is as follows:

$$\frac{x(s)}{i(s)} = \frac{-1}{(i_0/2g)s^2 - (i_0/x_0)}. \tag{4}$$

Taking the input voltage of the power amplifier (U_{in}) as input and the output voltage of the photoelectric position sensor (U_{out}) as output, the transfer function of the magnetic levitation ball control system can be given as follows:

$$G(s) = \frac{U_{out}(s)}{U_{in}(s)} = \frac{K_s x(s)}{K_a i(s)} = \frac{-(K_s/K_a)}{(i_o/2g)s^2 - (i_o/x_o)}, \tag{5}$$

where K_s is the gain of the photoelectric position sensor and K_a is the gain of the voltage-current power amplifier.

For the magnetic levitation ball position control experimental platform shown in Figure 1, the relevant parameters in equation (5) are shown in Table 1.

Substitute the physical parameters of the magnetic levitation ball system into equation (5). The system transfer function of the magnetic levitation ball position control system is calculated as follows:

$$G(s) = \frac{77.8421}{0.0311s^2 - 30.5250}. \tag{6}$$

Table 1: Physical parameters of magnetic levitation ball system.

Parameter (unit)	Numerical value
m(g)	22
i_0 (A)	0.6105
x_0 (mm)	20
K (Nm^2/A^2)	0.00023
K_s	−458.7156
K_a	5.8929

3. Deep Neural Network Feedforward Compensation Control

3.1. Control Principle. The feedforward compensation control of a deep neural network is designed for a magnetic levitation ball, and the structure is described in Figure 2. The main design ideas of the controller shown above are as follows:

First, the inverse model of the controlled plant is established online by the deep neural network identifier (DNNI), and then the trained parameters are dynamically copied to the deep neural network compensator (DNNC). The data of the training network is the control law and output signal produced in each control period. Second, DNNC and the PID controller are combined, and it can produce control law together. The inverse model of the controlled plant cannot be established accurately by DNNC in the early stage of control, so the PID controller is introduced to ensure the control accuracy in the early stage of control.

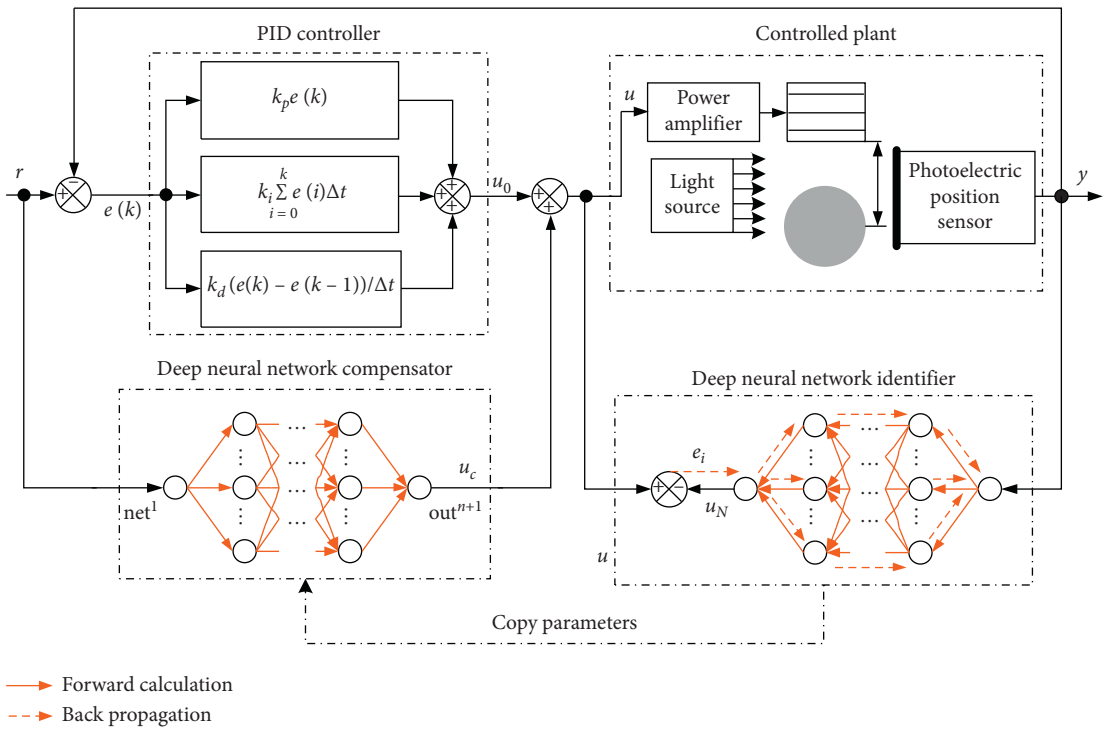


Figure 2: Deep neural network feedforward compensation control structure.

The actual control law of the controller proposed in this paper can be designed as follows:

$$u = u_0 + u_c, \tag{7}$$

where u_0 is the output of the PID controller and u_c is the output of DNNC.

The output of the PID controller is calculated as follows:

$$u_o = k_p e(k) + k_i \sum_{i=0}^{k} e(i)\Delta t + k_d \frac{(e(k) - e(k-1))}{\Delta t}, \tag{8}$$

where k_p, k_i, and k_d are hyperparameters.

Magnetic levitation ball control system is a complex nonlinear control system [26]. It is not accurate to establish the inverse model of the controlled plant only by using the neural network with a single hidden layer. The modeling accuracy of the deep neural network is higher because it has a stronger nonlinear mapping ability than that of a single hidden layer neural network [27, 28]. Therefore, the identifier and compensator established by a deep neural network can enhance the control accuracy.

3.2. Control Algorithm. In Figure 2, DNNI and DNNC have the same network structure, both of which are multi-hidden-layer neural networks. The DNNI is trained online once per control period based on real-time position y and control law u of the control system. Then, the trained parameters of the DNNI are dynamically copied to the DNNC. Based on the reference signal r, the compensation control law u_c of the current control period is determined by the forward calculation of DNNC. The control law u is calculated by u_c and PID controller output u_o. The deep neural network used in DNNI and DNNC includes an input layer, multiple hidden layers, and an output layer. The forward calculation and training of the network are as follows.

3.2.1. Forward Calculation. The whole deep neural network has $n+1$ layers, and there is n a weight matrix $w^1 \sim w^n$ in total. The input of the layer k is defined as net^k and the output as out^k. Then, the input calculation process and output calculation process of each layer are as follows:

Input layer: by directly inputting the reference signal into the neural network and pass it directly to the next layer, as follows:

$$\text{out}^1 = \text{net}^1 = y. \tag{9}$$

Hidden layer: the value of the input hidden layer becomes output after being processed by the activation function. In this paper, the Leaky ReLU function is used as the activation function. The input and output of layer k are as follows:

$$\begin{aligned} \text{net}^k &= \text{out}^{k-1} \cdot w^{k-1} + b^{k-1}, \\ \text{out}^k &= \delta\left(\text{net}^k\right), \end{aligned} \tag{10}$$

where w is the weight matrix between two layers, b is the bias, and δ is the Leaky ReLU function.

Output layer: by implementing linear operations. The output of the deep neural network is as follows:

$$\begin{aligned} \text{net}^{n+1} &= \text{out}^n \cdot w^n + b^n, \\ u_N &= \text{out}^{n+1} = \text{net}^{n+1}. \end{aligned} \tag{11}$$

The calculation process of u_c is consistent with that of u_N.

3.2.2. Training of Deep Neural Network. In Section 3, we improved the optimization algorithm. The network is trained using the improved Adagrad algorithm in the process of backpropagation.

The loss function is as follows:

$$L = \frac{1}{2}(u - u_N)^2, \tag{12}$$

where u is the reference signal, and u_N is the output of DNNI. The training objective of the deep neural network is to minimize L. The training process of parameters is as follows:

w^n error gradient:

$$g_w^n = \frac{\partial L}{\partial w^n} = (u - u_N) \cdot \text{out}^n. \tag{13}$$

w^{n-1} error gradient:

$$g_w^{n-1} = \frac{\partial L}{\partial w^{n-1}} = \frac{\partial \text{net}^n}{\partial w^{n-1}} \cdot \frac{\partial L}{\partial \text{net}^n} = \text{out}^{n-1} \cdot \xi^{n-1}, \tag{14}$$

$$\xi^{n-1} = \frac{\partial L}{\partial \text{net}^n} = \delta'(\text{net}^n) \cdot (w^n)^T \cdot (u - u_N), \tag{15}$$

where δ' is the derivative of the activation function, w^T is the transposition of the weight matrix, and ξ is the intermediate variable. The other symbols are the same as before.

w^{n-2} error gradient:

$$g_w^{n-2} = \frac{\partial L}{\partial w^{n-2}} = \frac{\partial \text{net}^{n-1}}{\partial w^{n-2}} \cdot \frac{\partial L}{\partial \text{net}^{n-1}} = \text{out}^{n-2} \cdot \xi^{n-2}, \tag{16}$$

$$\xi^{n-2} = \frac{\partial L}{\partial \text{net}^{n-1}} = \delta'\left(\text{net}^{n-1}\right) \cdot \left(w^{n-1}\right)^T \cdot \xi^{n-1}. \tag{17}$$

The above symbols are the same as before.

The calculation method of $w^{n-3} \sim w^1$ error gradient is the same as that of w^{n-2}.

b^n error gradient:

$$g_b^n = \frac{\partial L}{\partial b^n} = (u - u_N). \tag{18}$$

The above symbols are the same as before. The error gradient of $b^{n-1} \sim b^1$ is $\xi^{n-1} \sim \xi^1$.

The gradient of all network parameters can be obtained by the above formula. Then, the gradient is substituted into the appropriate optimization algorithm to update the network. The dynamic inverse model of the controlled plant is established by an online training network, and the network is trained only once per control period. Therefore, it is necessary to select the optimization algorithm with better convergence accuracy and faster convergence rate; otherwise, the network is not fully trained, which will affect the control accuracy.

Using the Adagrad algorithm instead of the gradient descent method to train the neural network can improve the accuracy of network modeling with fewer iterations, which can meet the accuracy requirements of parameter identification in neural network control. This is because the Adagrad algorithm has better convergence accuracy and convergence rate [29, 30]. However, Adagrad is still to be improved for instance. The convergence accuracy and convergence rate are far from satisfying and the learning rate is too small in the later stage of training.

In order to fully train the network with fewer iterations to make the established model have higher accuracy, this paper improves Adagrad. The formula of improved Adagrad is as follows:

$$s_t = \alpha s_{t-1} + (g_t)^2, \tag{19}$$

$$w_t = w_{t-1} - \frac{\eta}{\sqrt{s_t + \varepsilon}} g_t + \gamma(g_t - g_{t-1}), \tag{20}$$

where w_{t-1} is the parameter before the update, w_t is the parameter after the update, η is the learning rate, g_t is the gradient value, s_t is the sum of the squares of the gradient used in this update, s_{t-1} is the sum of the squares of the gradient used in the previous update, the initial value of s is 0, and ε is a smoothing term that avoids division by zero (the value is usually $le - 8$). α and γ are hyperparameters, and the value of α is 0.95, which guarantees that the learning rate will not be too low in the later stage of training.

The gradient obtained in equations (13)–(18) is substituted into equations (7) and (8) to update the network parameters. In [31], it reveals the similarity between the calculation of the parameter update amount in the process of neural network training and calculation of control law in the process of control by the PID controller. Based on this, we add a gradient differentiation term to Adagrad to build an improved Adagrad algorithm. The function of gradient differentiation is to predict the change trend of neural network parameters in the future and it is similar to the error differentiation in the PID controller. It improves the convergence accuracy and convergence rate of Adagrad by providing predictive compensation amount in the process of weight updating.

4. Results and Discussion

In this section, the effectiveness of the improved Adagrad optimization algorithm is verified by the benchmark function, and then the effectiveness of the controller proposed is verified by simulation and experiment. Simulation and experiment will compare the control effect of the four controllers. These four controllers are PID controllers and three controllers proposed in this paper. The three controllers are an Adagrad-based neural network controller (ANNC), an improved Adagrad-based neural network controller (IANNC), and an improved Adagrad-based deep neural network controller (IADNNC), respectively. The three controllers have the same control structure as shown in Figure 2, but the implementation of model identifier and

feedforward compensator is different in these controllers. Their implementation is similar to DNNI and DNNC in Figure 2, but not identical. They are implemented by different optimization algorithms and different numbers of hidden layers, as shown in Table 2.

Table 2: Differences between three controllers.

Controller	Number of hidden layers	Optimization algorithm
ANNC	1	Adagrad
IANNC	1	Improved Adagrad
IADNNC	More than 1	Improved Adagrad

4.1. Benchmark Function Verification. The performance of the Adagrad optimization algorithm and the improved Adagrad optimization algorithm is compared by five benchmark functions [32]. The benchmark functions are shown in Table 3.

The convergence accuracy of the optimization algorithm is evaluated by the result of iteratively solving the benchmark function. The function iterates from the same initial value. The maximum number of iterations is selected as 5000. The learning rates of both optimization algorithms are 0.1. The closer the result of either algorithm is to the theoretical minimum value of the reference function result, the better the convergence accuracy of the algorithm is. The experimental results are shown in Table 4.

The results in Table 4 show that the improved Adagrad algorithm has better convergence accuracy, and its solution results are closer to the theoretical minimum.

The convergence rate of the optimization algorithm is evaluated by the number of iterations needed to solve the benchmark function to reach the standard. The standard result of each benchmark function is the one with the larger result in Table 4. The learning rates of both optimization algorithms are 0.1. The fewer the number of iterations required, the faster the convergence rate of the algorithm. The experimental results are shown in Table 5.

The results in Table 5 show that the improved Adagrad algorithm has a faster convergence rate. The standard can be reached faster by solving the benchmark function iteratively with the improved Adagrad algorithm.

The results of benchmark function verification show that the improved Adagrad optimization algorithm has better convergence accuracy and faster convergence rate. So, the proposed controller will use the improved Adagrad optimization algorithm to train the network.

4.2. Simulation Results. The following simulation tests the influence of the improved Adagrad optimization algorithm on the control effect. In the simulation, the control period is 3 ms. The parameters of the PID controller are $k_p = 8$, $k_i = 6$, and $k_d = 0.2$. Parameters of the PID controller are the optimal parameters obtained by many experiments. The weight matrix and bias initial value of the neural network are random numbers within (−1, 1). There are five neurons in each hidden layer. The learning rate is 0.015, and the hyperparameter γ of the optimization algorithms in IANNC is 0.98. The activation function is Leaky ReLU. The sinusoidal signal is as in equation (18) which is in radians and one period of the trapezoidal signal is as in equation (19). Time is in seconds.

$$r(t) = \sin(0.4t),$$

$$r(t) = \begin{cases} 0.2t, & (0 < t \le 5), \\ 1, & (5 < t \le 10), \\ 1 - 0.2t, & (10 < t \le 15), \\ 0, & (15 < t \le 20). \end{cases} \tag{21}$$

The simulation results of the magnetic levitation ball follow the sinusoidal reference signal, as shown in Figures 3 and 4, and the trapezoidal reference signal, as shown in Figures 5 and 6. The results show that the IANNC provides better control in simulations.

Figures 7 and 8 show the training process of the weight matrix between the hidden layer and the output layer when using IANNC to control the magnetic levitation ball system to track sinusoidal and trapezoidal signals. Each line represents a weight parameter. It can be known from the figure that the weights in the controller will change because the network will be trained once per control period. The training data is the control law and output signal that are generated in each control period. Using the Adagrad algorithm to train the network will produce some problems. The first is the delay problem. This is because we use the data generated in the t_{n-1} control period to train the network and then use the trained network to predict the control law of the t_n control period. In the prediction process, it will be affected by various time-varying factors, so the predicted result will have certain errors. Therefore, we add a gradient differential term to Adagrad. According to the change trend of the gradient, we can predict the future parameter value, so the way of calculation will help to reduce the delay error. The second is inadequate network training. Because we train the network only once per control period, it is difficult to train the network fully in one iteration using the Adagrad algorithm. The convergence rate can be improved by increasing the differential of the gradient, which has been proved in Table 5. In order to ensure real-time performance, we can only train the network once per control period in actual control. Therefore, the improved Adagrad optimization algorithm is used to speed up the network training, so that the network model established under one training has higher accuracy, which improves the control accuracy.

The next simulation tests the influence of the number of hidden layers on the control effect. Through this simulation, the number of hidden layers in the controller is selected. PID controller parameters, neural network parameters, and sinusoidal signal are the same as those in the previous simulation. In the simulation, the feedforward compensation control effect of the neural network with 1, 2, 3, 4, 5, 6, 7, and 8 hidden layers is tested. The control reference signal is a sinusoidal signal. For each kind of neural network, the

Table 3: Benchmark functions used in experiments.

No.	Range	Function	Formulation
1	[−100, 100]	Bohachevsky2	$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1)\cos(4\pi x_2) + 0.3$
2	[−4.5, 4.5]	Beale	$f(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$
3	[−100, 100]	Bohachevsky1	$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$
4	[−5, 5]	Six-hump camel back	$f(x) = 4x_1^2 - 2.1x_1^4 + 1/3x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$
5	[−100, 100]	Bohachevsky3	$f(x) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1 + 4\pi x_2) + 0.3$

Table 4: Iterative results of two optimization algorithms.

No.	Min	Adagrad iteration results	Improved Adagrad iteration results
1	0	1.0648	0.2806
2	0	0.3442	0.2813
3	0	3.5184	0.4130
4	−1.03163	2.1352	−0.2115
5	0	3.6049	0.0001

Table 5: Iteration times of the two algorithms under the same iteration result.

No.	Standard	Adagrad iterations	Improve Adagrad iterations
1	1.0648	273	10
2	0.3442	2997	655
3	3.5184	96	8
4	2.1352	500	4
5	3.6049	364	6

average absolute value integral of the error over ten times simulation is reported for comparison.

The simulation results are shown in Figure 9.

According to the analysis of the simulation results in Figure 9, it can be known that increasing the number of hidden layers can improve the control accuracy within limited. For the magnetic levitation ball control system, it is suitable that the neural network in the controller has 5 hidden layers. When the hidden layer is increased to five layers, increasing the number of hidden layers will not only not improve the control accuracy but also increase the calculation time.

The last simulation further tests the influence of the number of hidden layers on the control effect. PID controller parameters, neural network parameters, and reference signals are the same as those in the previous simulation. In the simulation, we test the influence of IANNC and IADNNC (with five hidden layers) on the control accuracy.

The simulation results of the magnetic levitation ball follow the sinusoidal reference signal, as shown in Figure 10, and a trapezoidal reference signal, as shown in Figure 11. The results of the two controllers in the comparison figure show that the IADNNC provides the best control in simulations. We can know that the network structure of IADNNC is more complex than that of IANNC, so more data is needed to fully train the network used in IADNNC. In the initial phase, the network of IADNNC is not fully trained by enough data, resulting in a less accurately established network model and a larger error of tracking.

4.3. Experimental Results. Figure 12 shows the experimental platform of magnetic levitation ball position control. In this paper, the data acquisition control card PCI-1711 supported by MATLAB/Simulink is used to realize the real-time collection of magnetic levitation ball position and the output of control law. Based on the MATLAB/RTW software platform, the hardware in the loop experiment verification of the control method proposed in this paper is carried out.

The following experiment tests the influence of the improved Adagrad optimization algorithm on the control effect. In the experiment, the control period is 3 ms. The parameters of the PID controller are $k_p = 1.5$, $k_i = 0.3$, and $k_d = 15$. Parameters of the PID controller are the optimal parameters obtained by many experiments. The weight matrix and bias initial value of the neural network are random numbers within (−1, 1). There are five neurons in each hidden layer. The learning rate is 0.015, and the hyperparameter γ of the optimization algorithms in IANNC is 0.98. The activation function is Leaky ReLU. The sinusoidal signal is as in equation (18) which is in radians and one period of the trapezoidal signal is as in equation (19). Time is in seconds.

$$r(t) = 2.18\sin(0.5t + 2.4328) + 10.9,$$

$$r(t) = \begin{cases} 8.72 + 1.1637t, & (0 < t \le 3.75), \\ 13.08, & (3.75 < t \le 7.5), \\ 13.08 - 1.1637t, & (7.5 < t \le 11.25), \\ 8.72, & (11.25 < t \le 15). \end{cases} \quad (22)$$

The experimental results of the magnetic levitation ball follow a sinusoidal reference signal, as shown in Figures 13 and 14, and a trapezoidal reference signal, as shown in Figures 15 and 16. Table 6 shows the error range when three controllers are used to control the magnetic levitation ball.

Based on the above experimental results, it can be concluded that IANNC has a better control effect under the same PID controller parameters. IANNC can make the control system perform with better steady-state accuracy and dynamism.

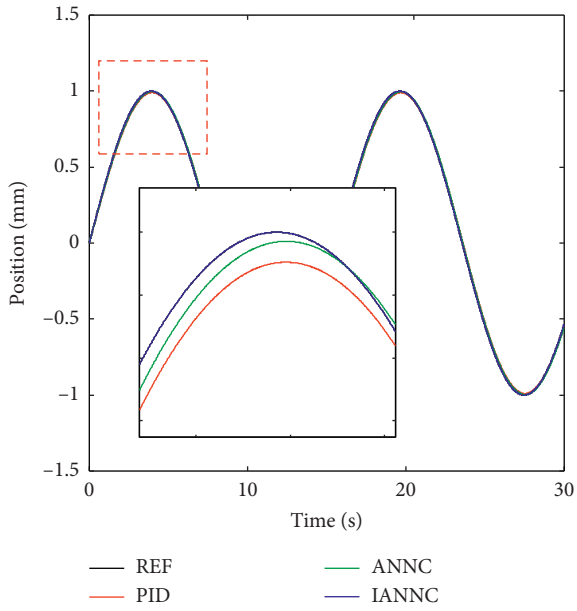


Figure 3: Sinusoidal signal position.

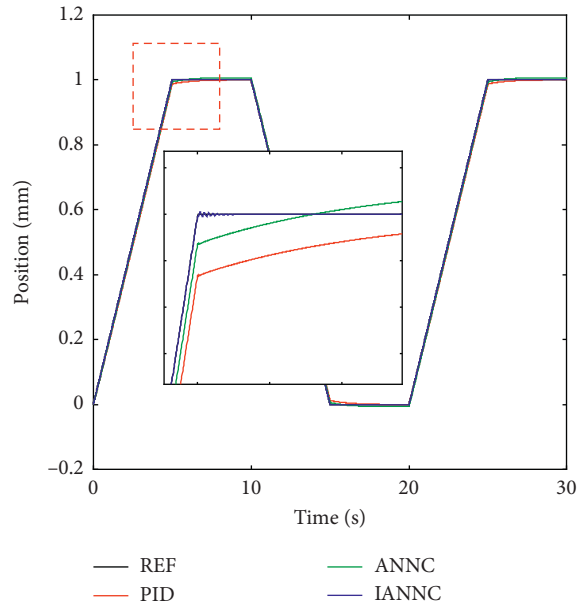


Figure 5: Trapezoidal signal position.

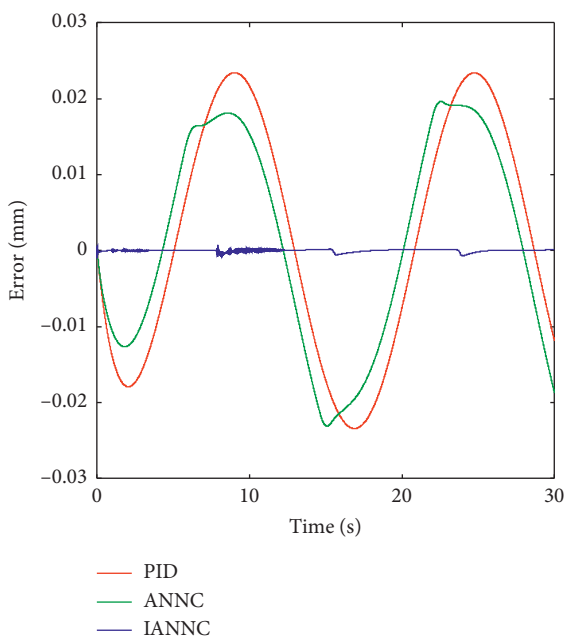


Figure 4: Sinusoidal signal error.

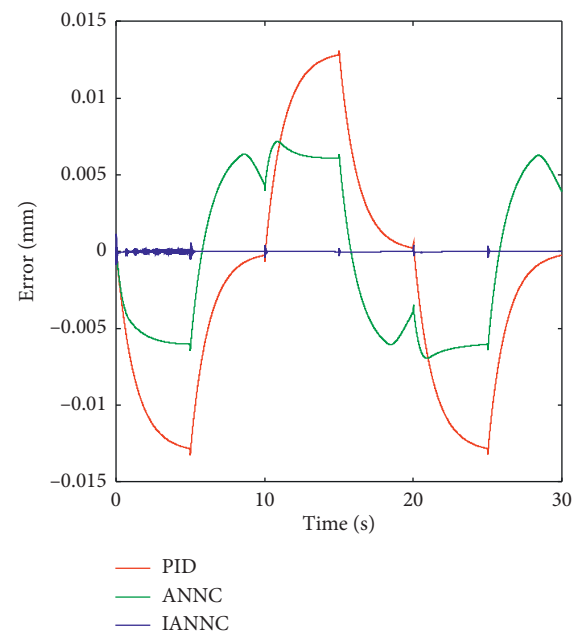


Figure 6: Trapezoidal signal error.

The next experiment tests the influence of the number of hidden layers on the control effect. PID controller parameters, neural network parameters, and reference signals are the same as those in the previous experiment. In the experiment, using the neural network with too many hidden layers will lead to a longer calculation time of the control quantity than that of the control period. Therefore, we test the influence of IANNC and IADNNC (with two hidden layers) on the control accuracy, only to verify whether increasing the number of hidden layers can improve the control accuracy and the change trend of control accuracy is shown in Figure 9.

The experimental results of the magnetic levitation ball follow a sinusoidal reference signal, as shown in Figure 17, and a trapezoidal reference signal, as shown in Figure 18. Table 7 shows the error range when two controllers are used to control the magnetic levitation ball.

Based on the above experimental analysis results, it can be concluded that increasing the number of hidden layers of neural network in the controller can improve the control accuracy. In Figure 17, the trend of error change caused by increasing the number of hidden layers is similar to that in Figure 9.

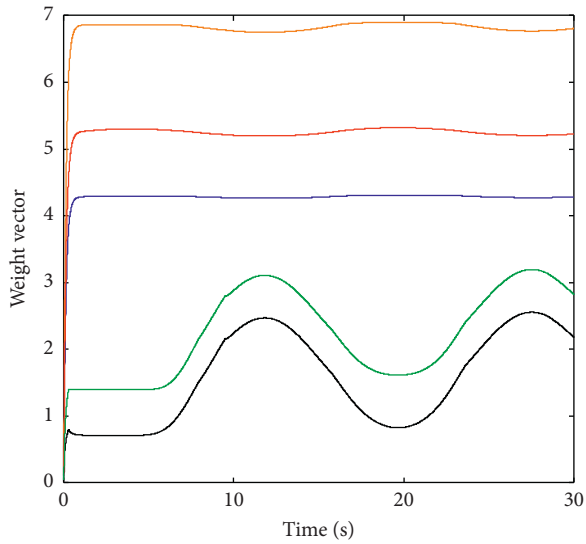


Figure 7: Sinusoidal signal weight vector.

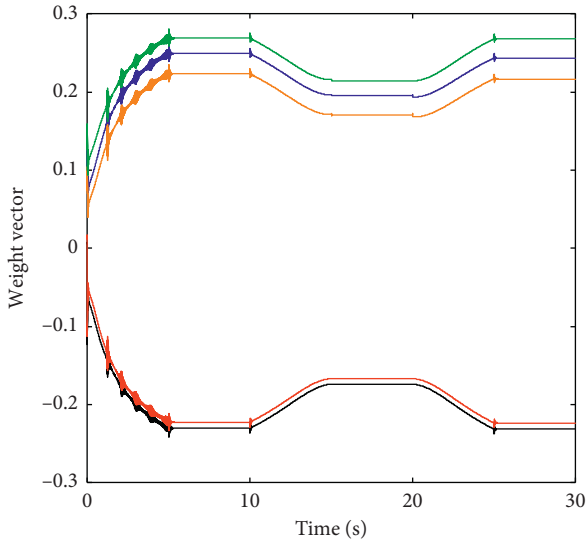


Figure 8: Trapezoidal signal weight vector.

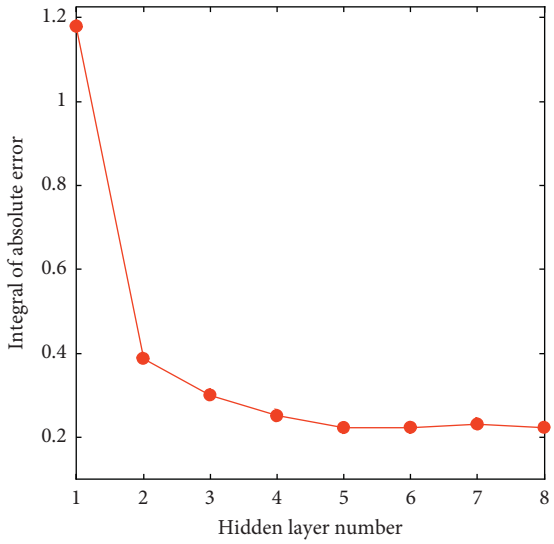


Figure 9: Effect of hidden layer number on control accuracy.

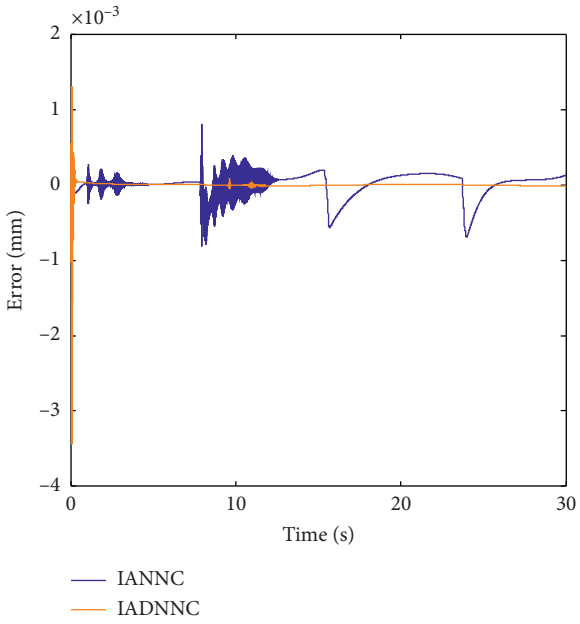


Figure 10: Sinusoidal signal error.

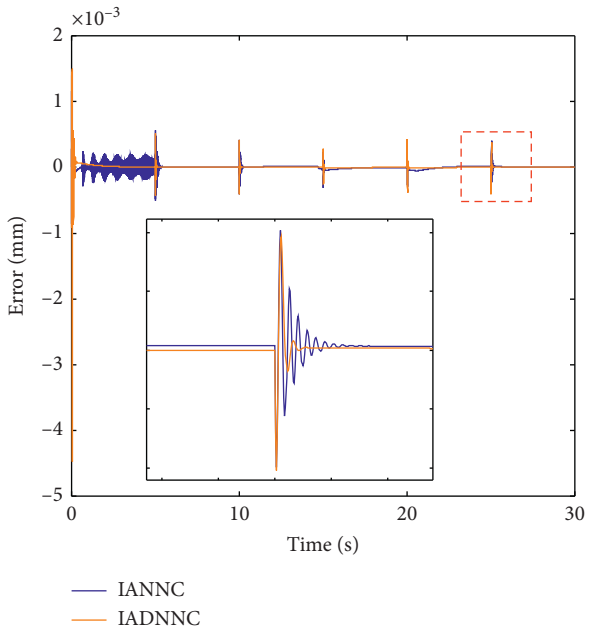


Figure 11: Trapezoidal signal error.

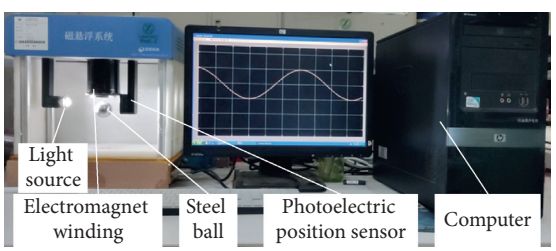


Figure 12: Magnetic levitation ball position control experiment platform.

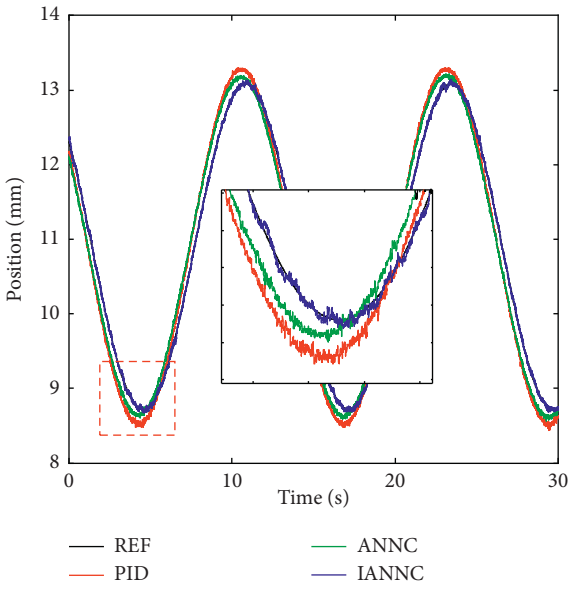


Figure 13: Sinusoidal signal position.

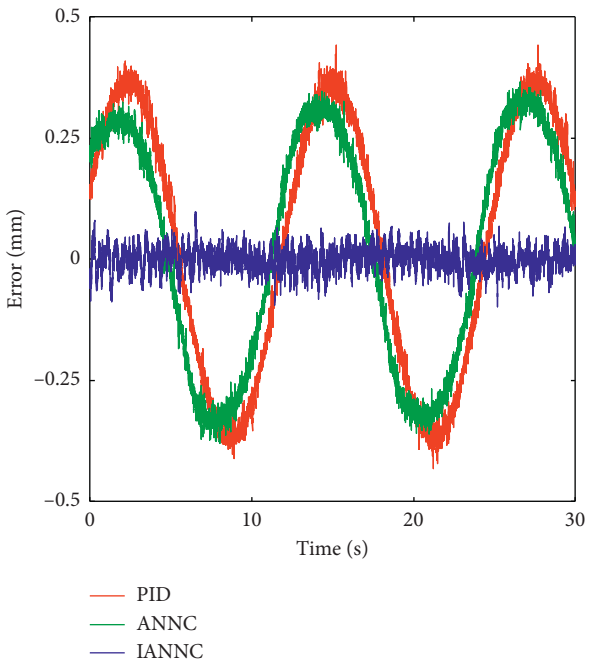


Figure 14: Sinusoidal signal error.

The root-mean-squared error (RMSE) shown in Table 8 proves that the IADNNC produces an obviously smaller RMSE than the other controllers.

In order to test the anti-interference ability of the controller IANNC and IADNNC, the interference is exerted by touching the steel ball with fingers at a random moment in the sinusoidal signal tracking experiment. The anti-interference experiment results of IANNC and IADNNC sinusoidal signal tracking control are shown in Figures 19 and 20. They prove that the control system can

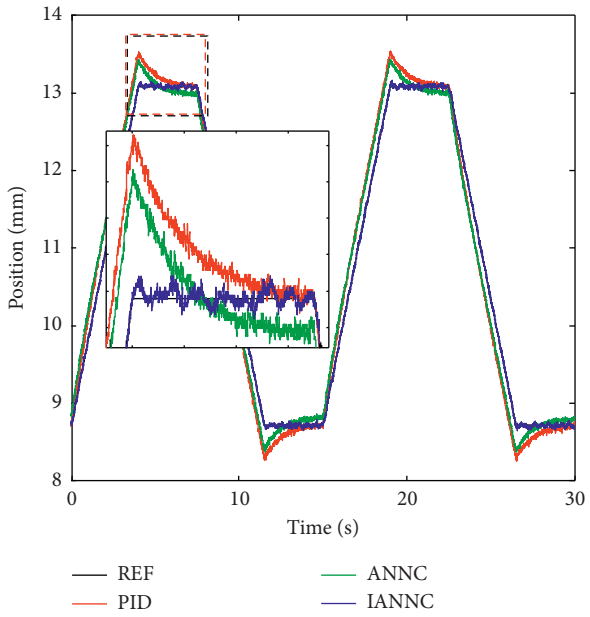


Figure 15: Trapezoidal signal position.

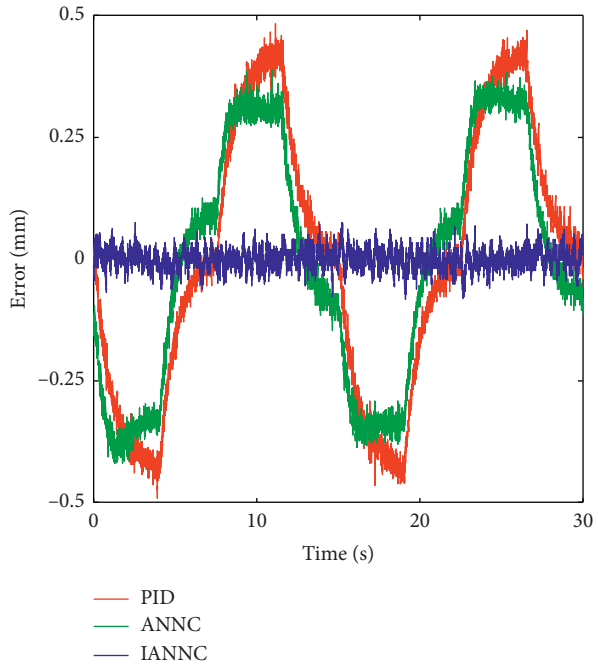


Figure 16: Trapezoidal signal error.

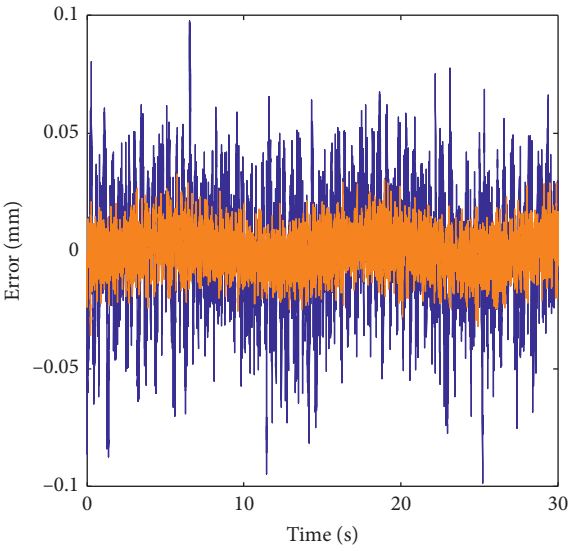


Figure 17: Sinusoidal signal error.

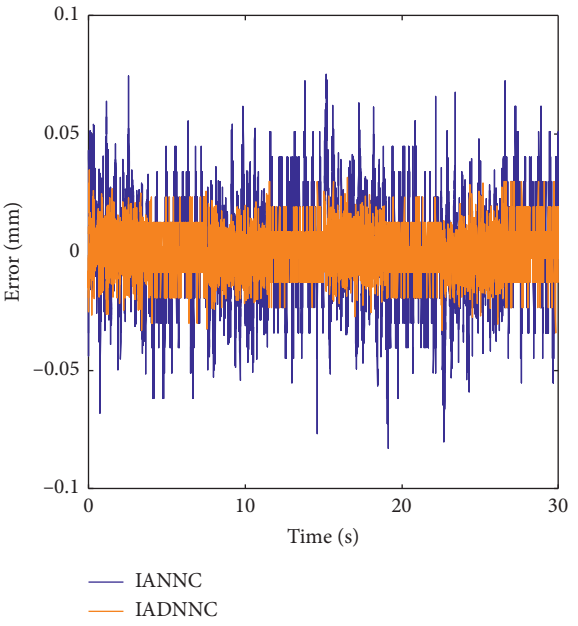


Figure 18: Trapezoidal signal error.

Table 6: Comparison of the experimental results for error range.

Controller	Sinusoidal signal (mm)	Trapezium signal (mm)
PID	[−0.45, 0.45]	[−0.50, 0.50]
ANNC	[−0.38, 0.38]	[−0.39, 0.43]
IANNC	[−0.10, 0.10]	[−0.09, 0.09]

Table 7: Comparison of the experimental results for error range.

Controller	Sinusoidal signal (mm)	Trapezium signal (mm)
IANNC	[−0.10, 0.10]	[−0.09, 0.09]
IADNNC	[−0.035,0.035]	[−0.035, 0.035]

Table 8: Comparison of the experimental results for RMSE.

Controller	Sinusoidal signal (mm)	Trapezium signal (mm)
PID	0.2671	0.2731
ANNC	0.2415	0.2436
IANNC	0.0237	0.0206
IADNNC	0.0084	0.0092

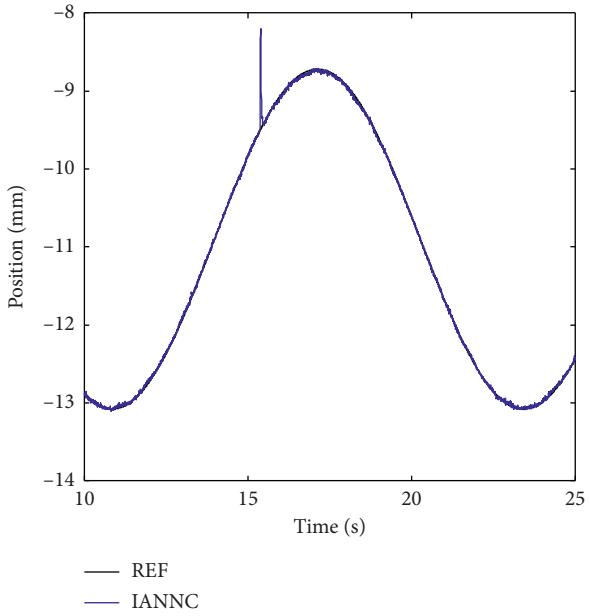


Figure 19: IANNC anti-interference results.

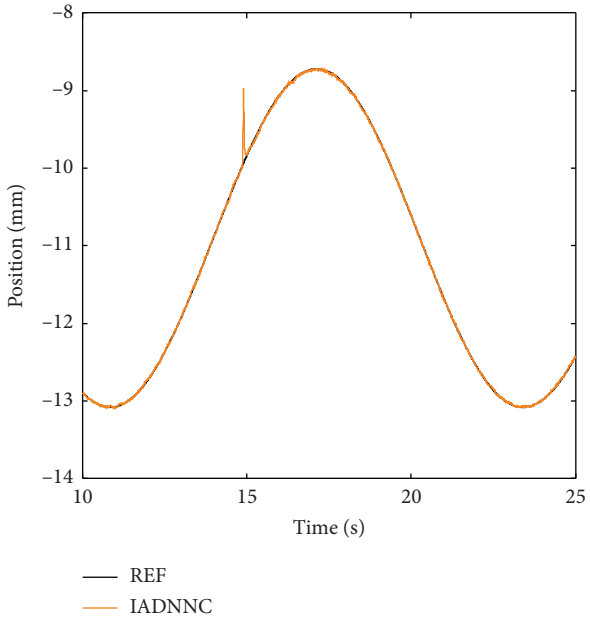


Figure 20: IADNNC anti-interference results.

quickly return to a stable state after being disturbed by the external environment, and the control system has certain robustness.

5. Conclusion

In this paper, a deep neural network feedforward compensation controller based on the improved Adagrad optimization algorithm is proposed. The improved Adagrad optimization algorithm and deep neural network are used to solve the problems that may occur in the application of the neural network in the control field, such as delay, slow training of the neural network, model accuracy, and control accuracy. By improving the accuracy of the dynamic inverse model of the controlled plant in the neural network controller, the control accuracy can be further improved without changing the controller structure. In the proposed controller, the dynamic inverse model of the controlled plant is established by a deep neural network that is trained by an improved Adagrad optimization algorithm. Then, a feedforward compensation controller is obtained by mixing this model with the PID controller. The simulation and experimental results demonstrate that the proposed IADNNC has a good control effect. The proposed controller can also be applied to other nonlinear systems, particularly time-varying systems and systems with uncertainty which are difficult to be obtained by mathematical methods. The application of other deep neural network models in the control field will be a topic for future study.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest in this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Nos. 51775323 and 51375289).

References

[1] W. Barie and J. Chiasson, "Linear and nonlinear state-space controllers for magnetic levitation," *International Journal of Systems Science*, vol. 27, no. 1, pp. 1153–1163, 1996.

[2] C.-M. Lin, M.-H. Lin, and C.-W. Chen, "SoPC-based adaptive PID control system design for magnetic levitation system," *IEEE Systems Journal*, vol. 5, no. 2, pp. 278–287, 2011.

[3] A. Mercado-Uribe and J. A. Moreno, "Homogeneous integral controllers for a magnetic suspension system," *Control Engineering Practice*, vol. 97, Article ID 104325, 2020.

[4] R. Morales, V. Feliu, and H. Sira-Ramirez, "Nonlinear control for magnetic levitation systems based on fast online algebraic identification of the input gain," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 4, pp. 757–771, 2011.

[5] R. Morales, H. Sira-Ramírez, and V. Feliu, "Adaptive control based on fast online algebraic identification and GPI control for magnetic levitation systems with time-varying input gain," *International Journal of Control*, vol. 87, no. 8, pp. 3077–3097, 2014.

[6] S. Pandey, V. Dourla, P. Dwivedi, and A. Junghare, "Introduction and realization of four fractional-order sliding mode controllers for nonlinear open-loop unstable system: a magnetic levitation study case," *Nonlinear Dynamics*, vol. 98, no. 1, pp. 601–621, 2019.

[7] J. D. J. Rubio, "Hybrid controller with observer for the estimation and rejection of disturbances," *ISA Transactions*, vol. 65, pp. 445–455, 2016.

[8] B. Bidikli and A. Bayrak, "A self-tuning robust full-state feedback control design for the magnetic levitation system," *Control Engineering Practice*, vol. 78, pp. 175–185, 2018.

[9] J. D. J. Rubio, "Robust feedback linearization for nonlinear processes control," *ISA Transactions*, vol. 74, pp. 155–164, 2018.

[10] Z.-J. Yang and M. Tateishi, "Adaptive robust nonlinear control of a magnetic levitation system," *Automatica*, vol. 37, no. 7, pp. 1125–1131, 2001.

[11] Y. Y. Wang, L. F. Liu, J. W. Chen et al., "Practical robust control of cable-driven robots with feedforward compensation," *Advance in Engineering Software*, vol. 145, Article ID 102801, 2020.

[12] D. Huo, K. Cheng, and F. Wardle, "A holistic integrated dynamic design and modelling approach applied to the development of ultraprecision micro-milling machines," *International Journal of Machine Tools and Manufacture*, vol. 50, no. 4, pp. 335–343, 2010.

[13] D. Wang, J. Wu, L. Wang, Y. Liu, and G. Yu, "A method for designing control parameters of a 3-DOF parallel tool head," *Mechatronics*, vol. 41, pp. 102–113, 2017.

[14] K. Prinz, A. Steinboeck, and A. Kugi, "Optimization-based feedforward control of the strip thickness profile in hot strip rolling," *Journal of Process Control*, vol. 64, pp. 100–111, 2018.

[15] C. Chen, S. Zhu, and Y. Wei, "Closed-loop control of nonlinear neural networks: the estimate of control time and energy cost," *Neural Networks*, vol. 117, pp. 145–151, 2019.

[16] L. H. Fu and D. Wang, "Fusion control of flexible logic control and neural network," *Mathematical Problems in Engineering*, vol. 2014, Article ID 913549, 17 pages, 2014.

[17] X. Ji, X. He, C. Lv, Y. Liu, and J. Wu, "Adaptive-neural-network-based robust lateral motion control for autonomous vehicle at driving limits," *Control Engineering Practice*, vol. 76, pp. 41–53, 2018.

[18] G. H. Nguyen, J.-H. Shin, and W.-H. Kim, "Autotuning controller for motion control system based on intelligent neural network and relay feedback approach," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 3, pp. 1138–1148, 2015.

[19] M. AlDhaifallah, S. Alsabbah, and I. Mujtaba, "A predictive neural network-based cascade control for PH reactors," *Mathematical Problems in Engineering*, vol. 2016, Article ID 5638632, 7 pages, 2016.

[20] D. Xu, J. Liu, X.-G. Yan, and W. Yan, "A novel adaptive neural network constrained control for a multi-area interconnected power system with hybrid energy storage," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 8, pp. 6625–6634, 2018.

[21] J. D. J. Rubio, L. X. Zhang, E. Lughofer et al., "Modeling and control with neural networks for a magnetic levitation system," *Neurocomputing*, vol. 227, pp. 113–121, 2017.

[22] Y. Qin, H. Peng, W. Ruan, J. Wu, and J. Gao, "A modeling and control approach to magnetic levitation system based on state-dependent ARX model," *Journal of Process Control*, vol. 24, no. 1, pp. 93–112, 2014.

[23] Y. Qin, H. Peng, F. Zhou, X. Zeng, and J. Wu, "Nonlinear modeling and control approach to magnetic levitation ball system using functional weight RBF network-based state-dependent ARX model," *Journal of the Franklin Institute*, vol. 352, no. 10, pp. 4309–4338, 2015.

[24] C.-M. Lin, Y.-L. Liu, and H.-Y. Li, "Sopc-Based function-link cerebellar model articulation control system design for magnetic ball levitation systems," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 4265–4273, 2014.

[25] J. M. Zhu, Z. Q. Shen, X. R. Li, and B. C. Qi, "Magnetic levitation ball position control based on neural network feedback compensation control," *Chinese Journal of Scientific Instrument*, vol. 35, no. 5, pp. 976–986, 2014.

[26] M. T. Gençoğlu and P. Agarwal, "Use of quantum differential equations in sonic processes," *Applied Mathematics and Nonlinear Sciences*, p. 8, 2020.

[27] S. N. Deepa and I. Baranilingesan, "Optimized deep learning neural network predictive controller for continuous stirred tank reactor," *Computers & Electrical Engineering*, vol. 71, pp. 782–797, 2018.

[28] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

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

[30] Z. Huang, J. Zhu, J. Lei, X. Li, and F. Tian, "Tool wear predicting based on multi-domain feature fusion by deep convolutional neural network in milling operations," *Journal of Intelligent Manufacturing*, vol. 31, no. 4, pp. 953–966, 2019.

[31] W. P. An, H. Q. Wang, Q. Y. Sun et al., "A PID controller approach for stochastic optimization of deep networks," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8522–8531, Salt Lake City, UT, USA, June 2018.

[32] B. Dogan and T. Olmez, "A new metaheuristic for numerical function optimization: vortex search algorithm," *Information Sciences*, vol. 293, pp. 125–145, 2015.