electronics

Article

Active Disturbance Rejection Control of Bearingless Permanent Magnet Synchronous Motor Based on Genetic Algorithm and Neural Network Parameters Dynamic Adjustment Method

Xin Wang and Huangqiu Zhu *

School of Electrical and Information Engineering, Jiangsu University, Zhenjiang 212013, China; 2111907009@stmail.ujs.edu.cn
* Correspondence: zhuhuangqiu@ujs.edu.cn

Abstract: In order to solve the problem of poor control performance, caused by fixed parameters of the active disturbance rejection control (ADRC) in bearingless permanent magnet synchronous motors (BPMSM), a dynamic parameters adjustment method of ADRC, based on a genetic algorithm and back-propagation neural network (GA-BPNN), is proposed. Firstly, the ADRC control models of motor-side and suspension-side are established, according to the motor speed formula and suspension force formula. Secondly, the BPNN algorithm is used to dynamically adjust the parameters of the ADRC, and the operation processes of BPNN are deduced, according to the chain rule. Thirdly, in order to avoid the problem of getting out of control, caused by the convergence failure of BPNN, a GA based on floating point coding is used to optimize the initial value of BPNN. Finally, these methods are integrated to form a BPMSM control system, based on the GA-BPNN-ADRC, and the effectiveness is verified on an experimental platform. The experimental results, show that the proposed method reduces the failure probability of the system from 35.61% to 0%, and the anti-interference ability and dynamic performance of the speed and displacement of the control system are significantly improved.

Keywords: bearingless permanent magnet synchronous motor (BPMSM); genetic algorithm (GA); neural network (NN); active disturbance rejection control (ADRC)

check for updates

Citation: Wang, X.; Zhu, H. Active Disturbance Rejection Control of Bearingless Permanent Magnet Synchronous Motor Based on Genetic Algorithm and Neural Network Parameters Dynamic Adjustment Method. *Electronics* **2023**, *12*, 1455. https://doi.org/10.3390/electronics12061455

Academic Editor: Teresa Orlowska-Kowalska

Received: 16 February 2023
Revised: 15 March 2023
Accepted: 17 March 2023
Published: 19 March 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

A bearingless permanent magnet synchronous motor (BPMSM) combines magnetic bearing technology with a permanent magnet synchronous motor, and controls the currents of stator windings, to realize the suspension and rotation of the rotor. Since the transmission shaft is not in contact with the stator, the BPMSM has the characteristics of no mechanical friction and wear, a high critical speed, being maintenance free, a long life, and producing no pollution [1–3]. Since the BPMSM superimposes a torque magnetic field and suspension magnetic field, the whole system is a high-order, multivariable, and strong coupling nonlinear system. Therefore, the classical proportional integral differential (PID) control does not meet the high-performance control requirements of the BPMSM [4], which is mainly caused by the following factors: 1. The PID controller constantly adjusts the control signal according to the amplitude of the error, to achieve closed-loop control. However, when a disturbance is suddenly applied, the PID controller cannot respond quickly, therefore, its response to the disturbance has a certain lag. At the same time, in order to better suppress the disturbance, a large control force will be used at the initial stage, which may cause the system to oscillate. 2. The integral link can effectively eliminate the static error, but it will degrade the stability of the system. At the same time, the PID controller has a poor ability to restrain change disturbance. Therefore, it is particularly important to replace the classical PID algorithm with a modern control algorithm.

In order to make bearingless motors have high-performance control effects, many control methods have been applied, such as sliding mode control [5,6] , internal model control [7], neural network control [8], active disturbance rejection control (ADRC) [4,9–12], etc. Owing to the sliding mode control, internal model control, and neural network control methods being heavily dependent on the motor mathematical model, it is difficult to replace the PID control. The ADRC control is independent of the actual model, and can use the extended state observer to compensate for the total disturbance in real-time, so as to achieve a high-performance control effect. However, the control performance of the ADRC is affected by the fixed parameters. The solution is, to adjust these parameters dynamically. It is possible to achieve dynamic parameter adjustment using fuzzy control, neural network, and intelligent algorithms. Nonetheless, intelligent algorithms require a lot of computing, while fuzzy control relies on actual experience. Neural network methods are a suitable approach, with the benefits of quick convergence and appropriate computing.

In previous studies, some scholars have used the ADRC to achieve high-performance control of bearingless motors. In [4], the cascaded extended state observers (ESOs) were used to realize the ADRC control of speed and displacement, which improved the dynamic performance and control accuracy of the whole control system. However, since two ESOs were used in the control system, more parameters needed to be adjusted to increase the parameter complexity. In [9], the corresponding ESO and state error feedback (SEF) were designed based on the derived displacement and rotational speed integral series standard mathematical model of the single-winding flux-switching bearingless motor, as the basis of ADRC. At the same time, the adjustment process of the parameters was analyzed, by combining the parameters with the bandwidth. In [12], the fuzzy neural network inverse system and the ADRC method were used, to realize the high-performance decoupling control of the BPMSM. Although the performance of the control system was better than that of the PID method, the entire algorithm required too much computation. At the same time, it was difficult for the inverse system to make adjustments to the actual working conditions, so the dynamic performance was poor. These papers applied the ADRC to bearingless motors to achieve high-performance control, however, there is a little research on dynamic parameter adjustment.

A single constant parameter cannot affect the performance of the ADRC control method, despite its strong control performance. Silva et al. (2021) regarded the error and the incremental change in the ADRC parameters as the input and output signals of the fuzzy control algorithm, respectively, to realize the dynamic adjustment of parameters [10]. However, fuzzy control heavily depends on expert knowledge, and insufficient fuzzy control experience will negatively impact the performance of the entire control system. In [11], the particle swarm genetic algorithm was used in a bearingless induction motor, to optimize the parameters of the ADRC, which greatly improved the performance of the control system. The real-time performance of this approach, however, was challenging to satisfy, because it was created with 10 populations and 100 iterations. Similarly, other intelligent optimization algorithms can also optimize the ADRC parameters. Although the optimized parameters can improve the performance of the control system, it is difficult to satisfy the real-time performance of the system [13,14]. In [15], a back propagation neural network (BPNN) was used to update the parameters of ESO in the PMSM. Compared with intelligent algorithms with more population and iterations, this method had less computation, so it could meet the real-time requirements. However, the initial values randomly generated in the BPNN had a great impact on the stability of the system. If the initial values generated were not appropriate, the control system would not be stable. At the same time, the learning rate of the BPNN had a great influence on the stability and convergence rate of the system. Although the aforementioned papers could enhance the performance of the control system, by dynamically adjusting the ADRC parameters, there were still drawbacks. Therefore, it is necessary to continue researching the dynamic adjustment of ADRC parameters.

In order to improve the control effect of the BPMSM system, this paper proposes an ADRC method based on the GA-BPNN. According to the previous analysis, the initial

values of the BPNN have a great impact on the performance of the system. Therefore, the GA is firstly used to optimize the initial value of the BPNN. Since this process is completed offline, it has no impact on the real-time performance of the control system. Moreover, additional momentum and an adaptive learning rate are used, to improve the BPNN, in order to accelerate the training speed and prevent local minimums. Then, the results of the BPNN dynamic training are transferred to ADRC, to realize the dynamic adjustment of the parameters. In the whole process, the operation of the BPNN is completed in a separate timer interrupt, so it will not interfere with the real-time control effect of BPMSM. The proposed method has the following advantages: 1. It can effectively reduce the probability of failure when BPNN dynamically adjusts parameters. 2. A small amount of calculations can meet the requirements of real-time control. 3. The dynamic response and steady-state performance of the motor speed are improved, and the anti-interference ability of the displacement loop is also improved.

The rest of this paper is structured as follows. In Section 2, the mathematical model of BPMSM and the algorithm structure of ADRC are derived. In Section 3, the processes of dynamic parameter adjustment of the BPNN algorithm are analyzed and deduced in detail. The GA algorithm for dynamically adjusting the initial value of the BPNN is investigated in Section 4. In Section 5, the whole control process is analyzed in detail and the feasibility of this method is proved by experiments. The content of this paper is summarized in Section 6.

2. The BPMSM Mathematical Model and ADRC Algorithm

2.1. The BPMSM Mathematical Model

The torque windings and the suspension force windings of the BPMSM are placed in the stator slots, the schematic diagram of the BPMSM is shown in Figure 1. When the pole pairs of the torque windings, P_M, and the suspension force windings, P_B, differ by 1, and the current frequency connected to the two sets of windings is equal, i.e., $P_M = P_B \pm 1$, $\omega_M = \omega_B$, BPMSM will be able to stabilize suspension. At this time, the corresponding expressions of the suspension force in the control system are

$$\begin{cases} \Sigma F_x = F_{ux} + F_{dx} - F_x = m\ddot{x} \\ \Sigma F_y = F_{uy} + F_{dy} - F_y = m\ddot{y} \end{cases} \quad (1)$$

where F_{ux} and F_{uy} are the unbalanced magnetic pull in the x- and y-axis static coordinate system, respectively, F_{dx} and F_{dy} are the additional disturbing forces in the x- and y-axis direction, respectively, and F_x and F_y are, respectively, the controllable suspension force in the x- and y-axis direction.

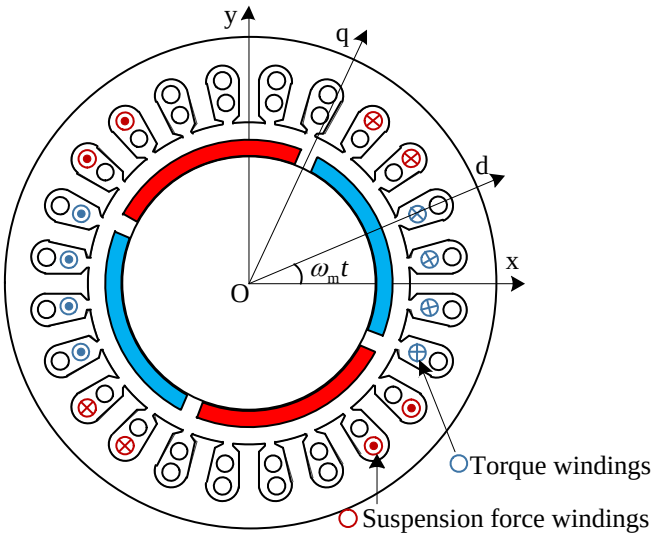


Figure 1. Schematic diagram of the BPMSM.

The expressions of the controllable suspension force corresponding to the x- and y-axis directions are [1]

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = (K_M + K_L) \begin{bmatrix} \lambda_{Md} & \lambda_{Mq} \\ -\lambda_{Mq} & \lambda_{Md} \end{bmatrix} \begin{bmatrix} i_{Bd} \\ i_{Bq} \end{bmatrix} \tag{2}$$

where K_M is the Maxwell force constant and K_L is the Lorentz force constant. λ_{Md} and λ_{Mq} are the flux linkages of the d- and q-axis at motor-side, respectively. i_{Bd} and i_{Bq} are the currents of the d- and q-axis at suspension-side, respectively.

The expressions of electromagnetic torque and mechanical motion equation are

$$T_e = 1.5 P_M \lambda_f i_{Mq} + M'd\left(I_f i_{Bq} + i_{Mq} i_{Bd}\right) + M'q\left(I_f i_{Bd} - i_{Mq} i_{Bq}\right) \tag{3}$$

$$J\frac{d\omega_m}{dt} = T_e - T_L - B\omega_m \tag{4}$$

where P_M is the number of motor pole pairs, λ_f is the permanent magnet flux linkage, J is the moment of inertia, ω_m is the mechanical angular speed of the motor, T_L is the load torque, and B is the viscous damping. I_f is the equivalent excitation current amplitude of the permanent magnet. i_{Md} and i_{Mq} are the currents of the d- and q-axis at motor-side, respectively. M' is the mutual inductance coefficient. d and q are the displacements of the d- and q-axis, respectively.

2.2. Establishment of the ADRC Control Algorithm

The high-performance control requirements of BPMSM cannot be met by a standard PID control, since there are several coupling considerations involved in its operation. The PID controller constantly adjusts the control signal according to the size of error during operation, therefore, its response to interference has a certain lag. At the same time, although the integral link can effectively eliminate the static error, it will reduce the stability of the system. The ADRC achieves a high-performance control effect by actively observing the total disturbance of the system and actively suppressing the disturbance. Therefore, ADRCs are used to replace the PID controllers of the BPMSM speed loop and displacement loop [16,17]. At the same time, in order to ensure the rapidity of the current loop, PI controllers are still used for the current loop at motor-side and suspension-side.

2.2.1. ADRC Design of Motor-Side Speed Loop

According to the previous expressions of the electromagnetic torque and mechanical motion equations, the following expression can be constructed.

$$\begin{aligned} \frac{d\omega_M}{dt} &= \frac{1.5 P_M^2 \lambda_f}{J} i_{Mq} + \frac{1}{J} \begin{bmatrix} M'd\left(I_f i_{Bq} + i_{Mq} i_{Bd}\right) + \ldots \\ M'q\left(I_f i_{Bd} - i_{Mq} i_{Bq}\right) - \ldots \\ P_M T_L - B\omega_M \end{bmatrix} \\ &= bu + f_w \end{aligned} \tag{5}$$

where b is the motor parameter gain, $b = \frac{1.5 P_M^2 \lambda_f}{J}$, and f_w is the total disturbance.

According to the above formula, the form of (5) conforms to the normal form of the ADRC, so the ADRC can be used to control the speed. In the standard ADRC, the tracking differentiator (TD) has the ability to soften the input signal. The use of TD depends on the actual control requirements. The ADRC of the motor-side speed loop, shown in Figure 2, can be constructed.

The state equation in linear ESO (LESO) can be written as

$$\begin{cases} e_0 = z_{M21} - \omega_M \\ \dot{z}_{M21} = z_{M22} - \beta_1 e_0 + b_s u_M \\ \dot{z}_{M22} = -\beta_2 e_0 \end{cases} \tag{6}$$

where z_{M21} is the speed observation value, z_{M22} is the disturbance observation value, and β_1 and β_2 are the observer gains.

The state equation in linear SEF (LSEF) can be written as

$$\begin{cases} e_1 = \omega_M^* - z_{M21} \\ u_s = \beta_3 e_1 \end{cases} \tag{7}$$

where ω_M^* is the given value of motor speed, β_3 is the gain of LSEF, and u_s is the LSEF output value.

The compensation output of the disturbance is

$$u_M = u_s - \frac{z_{M22}}{b_s} \tag{8}$$

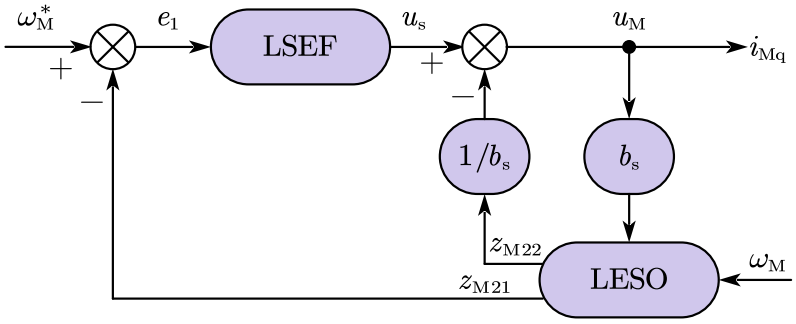


Figure 2. Block diagram of speed loop ADRC.

2.2.2. ADRC Design of Suspension-Side Displacement Loop

The ADRC structure of the suspension-side displacement loop is similar to that of the motor-side speed loop. According to (1), the following form of the active disturbance rejection normal form can be constructed.

$$\begin{cases} \ddot{x} = -\frac{1}{m}F_x + \frac{1}{m}(F_{ux} + F_{dx}) \\ \quad = \frac{1}{m}F_x + \frac{1}{m}(F_{ux} + F_{dx} - 2F_x) \\ \quad = b_1 u + f_{x1} \\ \ddot{y} = -\frac{1}{m}F_y + \frac{1}{m}\left(F_{uy} + F_{dy}\right) \\ \quad = \frac{1}{m}F_y + \frac{1}{m}\left(F_{uy} + F_{dy} - 2F_y\right) \\ \quad = b_1 u + f_{y1} \end{cases} \tag{9}$$

where f_{x1} and f_{y1} are the disturbance values.

According to (9), the ADRC of the suspension-side displacement loop, shown in Figure 3, can be constructed, according to the corresponding normal form.

Taking the ADRC in the x-axis direction as an example, the state equation inside the LESO can be written as

$$\begin{cases} e_2 = z_{B31} - x \\ \dot{z}_{B31} = z_{B32} - \beta_{p1} e_2 \\ \dot{z}_{B32} = z_{B33} - \beta_{p2} e_2 + b_p u_p \\ \dot{z}_{B33} = -\beta_{p3} e_2 \end{cases} \tag{10}$$

where z_{B31} is the displacement observation value, z_{B32} is the observed value of displacement differential, z_{B33} is the disturbance value, and β_{p1}, β_{p2}, and β_{p3} are the gains of the observer.

The state equation in LSEF can be written as

$$\begin{cases} e_3 = x - z_{\mathrm{B31}} \\ u_0 = \beta_{\mathrm{p4}} e_3 - \beta_{\mathrm{p5}} z_{\mathrm{B32}} \end{cases} \tag{11}$$

where β_{p4} and β_{p5} are the gains of LSEF.

The compensation output of the disturbance is

$$u_{\mathrm{p}} = u_0 - \frac{z_{\mathrm{B33}}}{b_{\mathrm{p}}} \tag{12}$$

The ADRC methods of motor-side speed loop and suspension-side displacement loop both contain observer gains and motor correlation coefficients. Fixed gains and coefficients find it difficult to cope with different motor operating conditions. Therefore, in order to ensure that BPMSM is always functioning at its peak performance, it is necessary to dynamically adjust these gains and coefficients, in accordance with the motor's operating conditions.

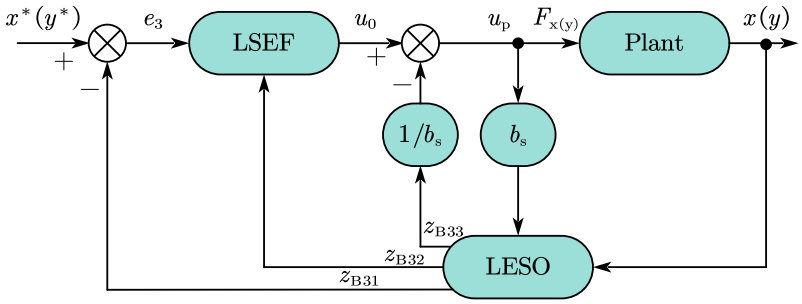


Figure 3. Block diagram of displacement loop ADRC.

3. The BPNN Algorithm and Analysis

Since many gains and coefficients in the ADRC have a crucial impact on the operation of the BPMSM, dynamic adjustment of these gains and coefficients can effectively improve the operation quality of the motor. There are many methods of dynamic adjustment, which have been mentioned in Section 1. However, most of these methods have too much computation, so it is difficult to meet the real-time requirements in some cases. The BPNN algorithm is a multilayer, feedforward neural network, trained according to the error back propagation algorithm, which includes input layer, hidden layer, and output layer. The BPNN will not have a great impact on the global training results when its local or partial neurons are damaged. In addition, the BPNN can greatly reduce the computation time, because of its computational advantages compared with an evolutionary algorithm.

The task of adjusting the ADRC parameters based on the BPNN mainly includes two parts, and the specific algorithm block diagram is shown in Figure 4. It is mainly composed of the BPNN for adjusting parameters and the traditional ADRC structure.

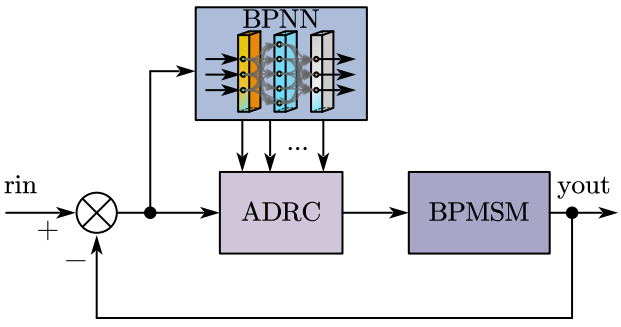


Figure 4. Block diagram of BPNN-ADRC.

In this paper, a three-layer BPNN is used. The input layer selects the important three variables in the neural network algorithm: error, e, actual output component, y, and constant item, 1. The number of hidden layer nodes is determined to be five, based on the comprehensive control effect and calculation amount. The output of the output layer corresponds to the gains of the LESO of the speed loop ADRC controller β_1, β_2, correlation coefficient b_s, and the gain in LSEF, β_3. The output layer of BPNN in suspension control corresponds to the gains of the LESO of displacement loop ADRC controller β_{p1}, β_{p2}, β_{p3}, correlation coefficient b_p, and the gains in LSEF β_{p4} and β_{p5}. The basic structure of the designed neural network is shown in Figure 5.

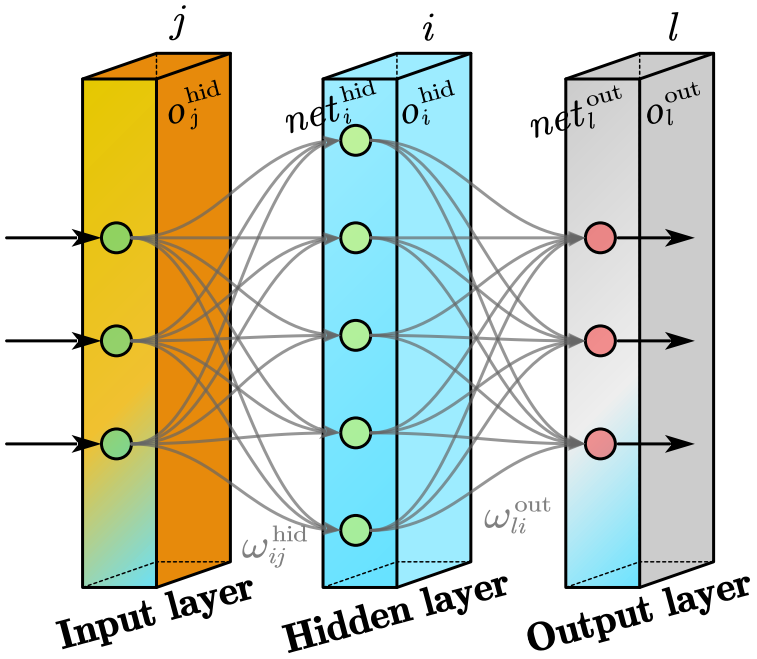


Figure 5. Basic structure of BPNN.

Take the structure in Figure 5 as an example to analyze the network. The input and output parts corresponding to the hidden layer are

$$\begin{cases} net_i^{\text{hid}}(k) = \sum_{j=1}^{3} w_{ij}^{\text{hid}} o_j^{\text{in}} \\ o_i^{\text{hid}}(k) = f\left(net_i^{\text{hid}}(k)\right) \end{cases}, i = 1, 2, \ldots, 5 \tag{13}$$

where w_{ij}^{hid} is the weight coefficient of the hidden layer.

The input and output parts of the output layer are

$$\begin{cases} net_l^{out}(k) = \sum_{i=1}^{5} w_{li}^{\text{out}} o_i^{\text{im}}(k) \\ o_l^{\text{out}}(k) = g\left(net_l^{out}(k)\right) \end{cases}, l = 1, 2, 3 \tag{14}$$

The parameters that need to be adjusted by ADRC are represented by each element of the output node. (13) and (14) employ the following functions, respectively, as the activation functions $f(\cdot)$ and $g(\cdot)$.

$$\begin{cases} f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \\ g(x) = \frac{e^x}{e^x + e^{-x}} \end{cases} \tag{15}$$

The performance evaluation function is defined as the square of the difference between the input and output. The square error is recognized for its good mathematical properties. It is continuous and differentiable, which is conducive to optimization [18,19]. The specific expression is

$$E(k) = \frac{1}{2}[\text{rin}(k) - \text{yout}(k)]^2 \tag{16}$$

The formula can be updated continuously by using the gradient descent method, that is, search and adjust according to the negative gradient direction of $E(k)$, to the weight coefficients. However, the convergence speed of the traditional BPNN gradient descent algorithm is slow, therefore, the momentum term needs to be added in the weight iteration process, to accelerate the gradient descent process.

The updating process of the parameters of the accelerated gradient descent method is

$$\Delta w_{li}^{\text{out}}(k) = -\eta \left[(1-\mu) \frac{\partial E(k)}{\partial w_{li}^{\text{out}}(k)} + \mu \frac{\partial E(k-1)}{\partial w_{li}^{\text{out}}(k-1)} \right] \tag{17}$$

where η is the learning rate, μ is the momentum factor.

The weight calculation process of the output layer is

$$\begin{aligned} h(k) &= \frac{\partial E(k)}{\partial w_{li}^{\text{out}}(k)} \\ &= \frac{\partial E(k)}{\partial y(k)} \cdot \frac{\partial y(k)}{\partial o_l^{\text{out}}(k)} \cdot \frac{\partial o_l^{\text{out}}(k)}{\partial net_l^{\text{out}}(k)} \cdot \frac{\partial net_l^{\text{out}}(k)}{\partial w_{li}^{\text{out}}(k)} \end{aligned} \tag{18}$$

The first item on the right side of (18) can be simplified as

$$\frac{\partial E(k)}{\partial y(k)} = -e(k) \tag{19}$$

Since the second term in (18) is unknowable and hard to calculate, the following formulation can be used in its place.

$$\frac{\partial y(k)}{\partial o_l^{\text{out}}(k)} \to \text{sgn}\left(\frac{\partial y(k)}{\partial \Delta o_l^{\text{out}}(k)} \right) \tag{20}$$

The third and fourth terms on the right side of (18) can be expressed as

$$\frac{\partial o_l^{\text{out}}(k)}{\partial net_l^{\text{out}}(k)} = g'\left(net_l^{\text{out}}(k)\right) \tag{21}$$

$$\frac{\partial net_l^{\text{out}}(k)}{\partial w_{li}^{\text{out}}(k)} = o_i^{\text{hid}}(k) \tag{22}$$

where $g'(\cdot) = g(\cdot)(1 - g(\cdot))$.

The following formula is derived by substituting (18)–(22) into (17).

$$\Delta w_{li}^{\text{out}}(k) = -\eta \left[\begin{array}{c} (1-\mu)\delta_l^{\text{out}}(k) o_i^{\text{hid}}(k) + \dots \\ \mu \delta_l^{\text{out}}(k-1) o_i^{\text{hid}}(k-1) \end{array} \right] \tag{23}$$

where $\delta_l^{\text{out}}(k) = -e(k) \cdot \text{sgn}\left(\frac{\partial y(k)}{\partial \Delta o_l^{\text{out}}(k)} \right) \cdot g'\left(net_l^{\text{out}}(k)\right)$.

Similarly, the weight update formula of the input layer is

$$\Delta w_{ij}^{\text{hid}}(k) = -\eta \left[\begin{array}{c} (1-\mu)\delta_i^{\text{hid}}(k) o_j^{\text{hid}}(k) + \dots \\ \mu \delta_i^{\text{hid}}(k-1) o_j^{\text{hid}}(k-1) \end{array} \right] \tag{24}$$

where $\delta_i^{\text{hid}}(k) = \delta_l^{\text{out}}(k) \cdot w_{li}^{\text{out}}(k) \cdot f'\left(net_l^{\text{out}}\right)$, $f'(\cdot) = (1 - f(\cdot))/2$.

The adaptive learning rate is utilized to enhance the optimization performance of the BPNN, by accelerating training and preventing the network from reaching the local minimum. The following form can be used to iteratively update the specific learning rate.

$$\eta(k) = 2^{\mathrm{sgn}(h(k)\cdot h(k-1))}\eta(k-1) \tag{25}$$

4. The Improved Genetic Algorithm

4.1. Basic Theory of Genetic Algorithms

The GA processes multiple individuals in the population at the same time, achieves the balance between global exploration and local development by setting selection pressure, evaluates multiple solutions in the search space, and reduces the risk of falling into the local optimal solution. At the initial stage of the algorithm operation, setting small selection pressure can make the algorithm have better global exploration ability and carry out a wide area search. At the later stage of the algorithm operation, setting a large selection pressure can make the algorithm perform a relatively fine local search. Compared with other evolutionary algorithms, the GA has strong robustness and can find the global optimal solution, which is suitable for determining the optimal initial value of the BPNN. Although this method can also be used to directly update the parameters of ADRC, it cannot meet the real-time requirements of the system, because it takes more time to calculate the GA algorithm online. The initial value of BPNN obtained by the GA is obtained offline, so it will not affect the real-time performance of the control system.

The flow chart of the GA is shown in Figure 6. First, the population of the GA is initialized and the fitness value after initialization is calculated. Floating point coding means that each gene value of an individual is represented by a floating point number within a certain range, and the coding length of an individual is equal to the number of digits of its decision variable. The floating point coding method is utilized to code and calculate the initial values, due to the huge number of optimized initial values and the requirement for high accuracy and operational efficiency. Then, the iterative process is carried out. In the iteration process, operations such as selection, crossover and mutation are required. Among them, the roulette method is selected in the selection process, which determines the probability of being selected, by the size of the fitness function. The crossover operation selects the arithmetic crossover method to linearly combine the two individuals and generate new individuals. The mutation operation selects the boundary mutation method, which has good performance when the optimal solution approaches the boundary.

The expressions of the arithmetic crossover method are

$$\begin{cases} X_A^{t+1} = \alpha X_B^t + (1-\alpha)X_A^t \\ X_B^{t+1} = \alpha X_A^t + (1-\alpha)X_B^t \end{cases} \tag{26}$$

where α is a random number and X_A^t and X_B^t are two individuals.

The expression of the boundary mutation method is

$$X^{t+1} = X^t + \lambda\left(bound - X^t\right) \tag{27}$$

where *bound* represents the boundary value, $\lambda = \mathrm{rand} \cdot \left(1 - \frac{pop}{Maxgen}\right)^2$, pop represents the population number, and *Maxgen* represents the maximum number of iterations.

After selection, crossover, mutation, and other operations, new individuals need to be substituted into the control model to solve, and the corresponding performance standards need to be obtained in the model. Here, the integral of time absolute error (ITAE) is selected, which can reflect both the size of error and the speed of error convergence, that is, the control accuracy and convergence speed are considered. The calculation method of performance criteria is

$$obj = \omega_1 \int_0^\infty t|e_{\mathrm{n}}(t)|\mathrm{d}t + \omega_2\sigma_n + \omega_3 \int_0^\infty t|e_{\mathrm{x}}(t)|\mathrm{d}t + \omega_4\sigma_{\mathrm{x}} \tag{28}$$

where ω_1, ω_2, ω_3, and ω_4 are weights. $e_n(t)$ and $e_x(t)$ are the errors. σ_n and σ_x are the speed drop and displacement drop after the disturbances. This paper takes $\omega_1 = 1$, $\omega_2 = 0.01$, $\omega_3 = 0.01$, and $\omega_4 = 0.1$.

When the GA algorithm reaches the maximum number of iterations, the optimization process of the initial parameters can be completed. The final output requires the optimal initial value of BPNN.

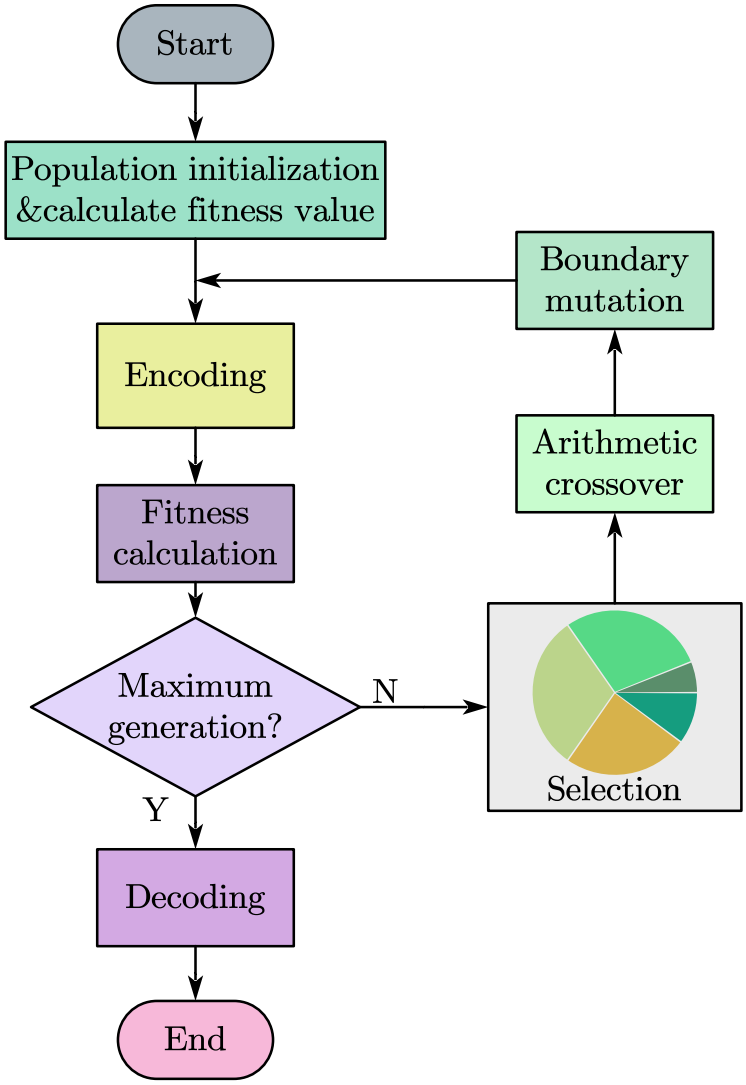


Figure 6. The flow chart of the genetic algorithm.

4.2. Performance Comparison of Different Genetic Algorithms

In order to verify that the GA method used has better optimization capability than the commonly used GA algorithm (binary encoding and fundamental bit mutation), the performances of the two methods are tested. The tested functions are the Matyas function, the Booth function, and the Beale function. The test results are shown in Figure 7. To ensure fairness, the number of species used in the test is consistent with the crossover probability and mutation probability. That is, the population number is set as 10, the crossing probability is 0.9, and the mutation probability is 0.1. The maximum number of iterations is set differently according to their performance differences. It can be seen from Figure 7, that the proposed method can obtain the extreme values of the three test functions more quickly. In addition, the use of binary encoding has a negative impact on the optimization of BPNN initial values. If there are too many BPNN initial variables, the number of codes will be far greater than the real number coding method. In addition, because of the increase in the amount of coding,

more storage space will be consumed during code execution, which is not conducive to the storage of other data.

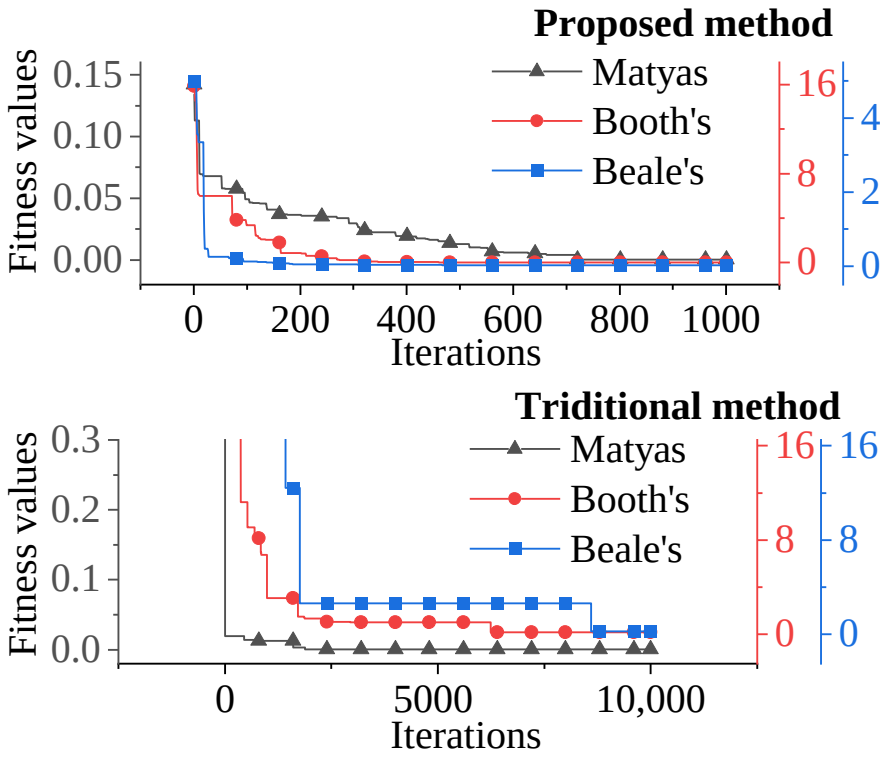


Figure 7. Results of the genetic algorithm performance test.

5. Experimental Analysis

5.1. Establishment and Analysis of the GA-BPNN Parameter Update Algorithm

In this paper, the GA is used to optimize the initial value of BPNN, to avoid the situation that the system becomes unstable. At the same time, the BPNN is used to continuously update the ADRC parameters, to improve the performance of the control system. The general block diagram of the BPMSM control system based on the GA-BPNN-ADRC, is shown in Figure 8.

The whole control is divided into suspension vector control and motor vector control. Between them, the proposed GA-BPNN algorithm is used to update the parameters of the displacement loop ADRC in suspension vector control, and the speed loop ADRC in motor vector control. Since the current loop in suspension vector control and motor vector control can be equivalent with a first-order link, a simple and efficient PI controller can be used. The GA algorithm is used to optimize the initial value of the BPNN offline. In this algorithm, 10 populations and a maximum of 30 iterations are set. Through calculation and analysis, the total number of additions, subtractions, multiplications, and divisions used by a single GA algorithm is 6390, and the total time consumption is 319.5 μs. According to the previous BPNN network structure, the number of additions and multiplications used by the BPNN algorithm in the whole control system is 459, the number of divisions is 26, the number of exponential operations is 44, and the total time consumption is 26.25 μs. Among them, the exponential operation uses the lookup table method, to reduce the calculation time. Based on this data, it can be found that the total number of computations of BPNN is far less than that of the GA algorithm. As a result, the BPNN algorithm better meets the performance requirements when real-time requirements are high. The GA-BPNN in the speed loop takes the speed error and actual speed as the input of the module. After real-time adjustment, it outputs three gains and correlation coefficients in (6)–(8). The GA-BPNN in the displacement loop takes displacement error and actual displacement as the input of the

module. After real-time adjustment, it outputs five gains and one correlation coefficient in (10)–(12).

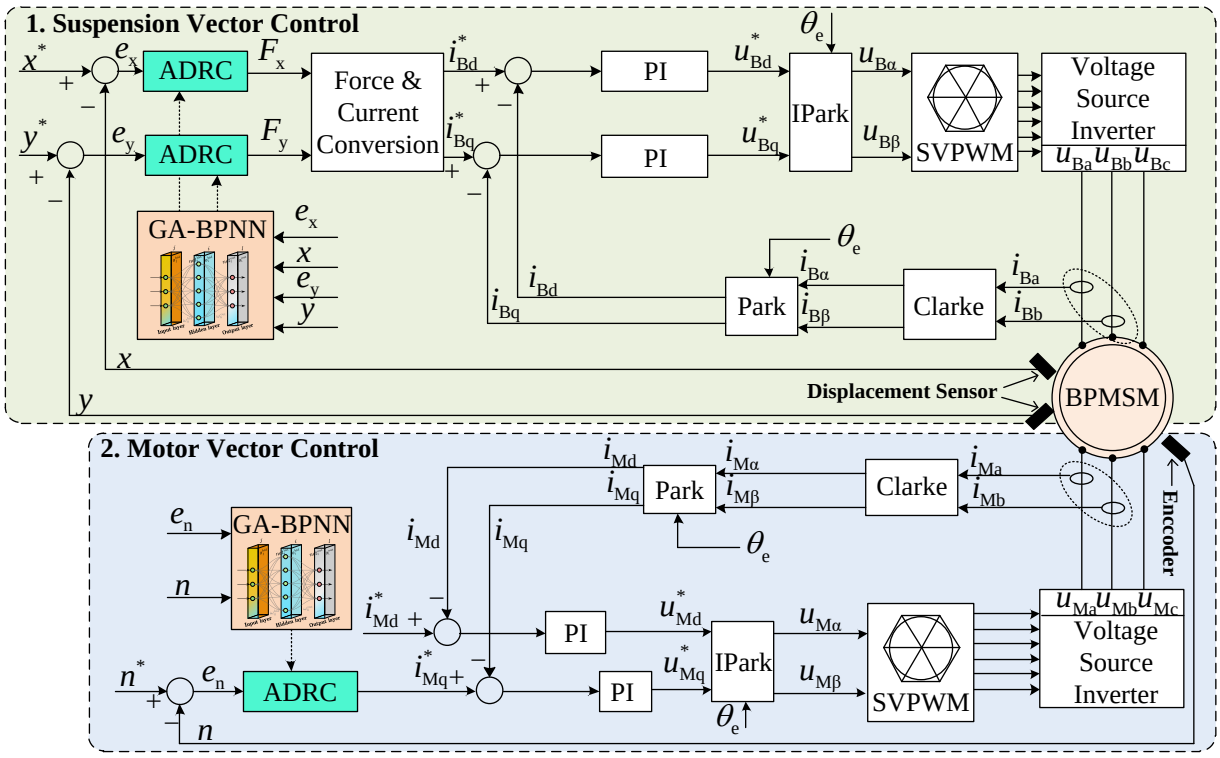


Figure 8. The general block diagram of the BPMSM control system.

5.2. Analysis of Experimental Results

In order to verify the effectiveness of the proposed GA-BPNN-ADRC algorithm applied to the BPMSM, the corresponding experiments are carried out on the experimental platform. The experimental prototype platform and specific parameters are shown in Figure 9 and Table 1. Since the calculation amount of the proposed algorithm is much higher than that of the vector control algorithm without BPNN and GA, TMS320F28377D is used to implement the proposed algorithm. The controller has a main frequency of 200 MHz, which can realize the control algorithm faster. The interrupt frequency in the control unit is 10 kHz, and Mitsubishi Electric's intelligent power module PS21965 is used, which has a collector emitter voltage of 600 V and the ability to output 20 A rated current. In Figure 9, the upper power drive circuit board supplies power to the torque windings of the BPMSM, and the lower power drive circuit board supplies power to the suspension force windings of the BPMSM. Two AC voltage regulators supply the torque windings and suspension force windings, and the auxiliary power supplies power for the power drive circuits and eddy current sensors. The interface circuit boards modulate the motor speed signals, and the eddy current displacement sensor signals into voltage signals of 0–3 V, then sends them to the digital signal processor (DSP) for data processing, and completes the closed-loop control in the DSP. Finally, the torque and suspension force are controlled by the power drive circuit boards. The initial parameters of the speed loop ADRC, in Section 2.2.1, are set as $\beta_1 = 2.5 \times 10^5$, $\beta_1 = 2 \times 10^3$, $\beta_3 = 40$, and $b_s = 166$. The initial parameters of the displacement loop ADRC, in Section 2.2.2, are set as $\beta_{p1} = 8 \times 10^3$, $\beta_{p2} = 1.2 \times 10^3$, $\beta_{p3} = 60$, $\beta_{p4} = 15$, $\beta_{p5} = 8 \times 10^{-3}$, and $b_p = 6.25$. The PI parameters of the current loop at the motor-side are set to $K_p = 0.23$ and $K_i = 1.7 \times 10^{-2}$. The PI parameters of the current loop at the suspension-side are set to $K_p = 0.5$ and $K_i = 4.5 \times 10^{-3}$.

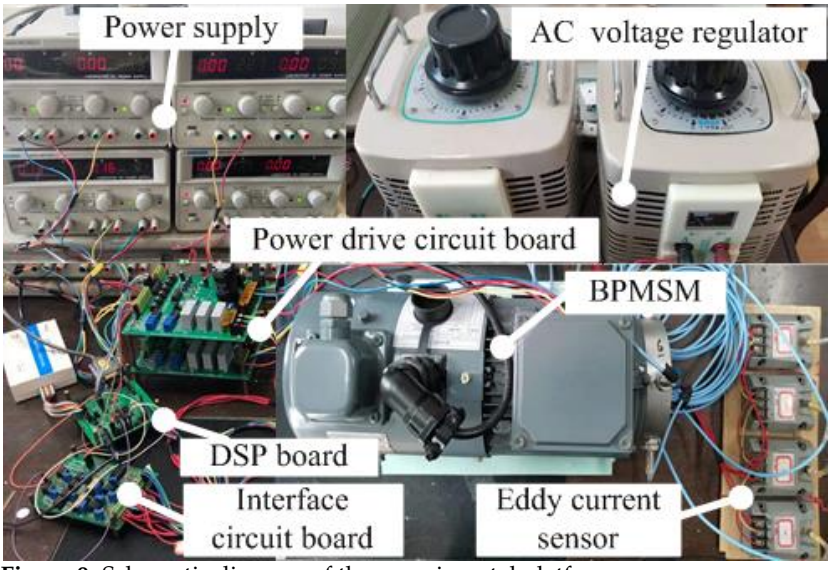


Figure 9. Schematic diagram of the experimental platform.

Table 1. The parameters of the prototype [1].

Symbol	Value	Symbol	Value
U_N (V)	220	n_N (r/min)	3,000
P_N (kW)	1.1	P_M/P_B	1/2
λ_f (Wb)	0.166	M'	6.67
R_M (Ω)	2.32	$R_B(\Omega)$	1.85
L_M (mH)	13.42	L_B (mH)	2.34

In order to verify that the randomly generated initial values, when the BPNN optimizes the ADRC parameters, have adverse effects on the entire system, the failure probability test of the control system that does not use the GA algorithm to optimize the initial values, was carried out, in the form of a simulation model. The failure probability refers to the proportion of the number of times that the system lost control in the process of using BPNN, to the total number of times it ran. The smaller the proportion, the more stable the system is. The model was repeated for 10,000 times, the number of runaway times was accumulated in the counter, and the failure probability was finally calculated. The failure probability was recorded at the 100th, 500th, 1000th, and 10,000th times. The specific data are shown in Table 2.

Table 2. Failure probability of systems with random initial values.

Number of Cycles	Failure Probability (Initial Value)	
	Random	GA Optimization
100	40%	0
500	34.40%	0
1000	35.30%	0
10,000	35.61%	0

The GA was used to optimize the initial value of the BPNN. The population number, maximum iteration number, crossover probability, and mutation probability were set to 10, 30, 0.9, and 0.1, respectively. The fitness value of the system is shown in Figure 10. It can be seen from Figure 10 that in the process of optimization, the fitness value of the target decreased, which means that the proposed GA algorithm was effective in finding the optimal initial value.

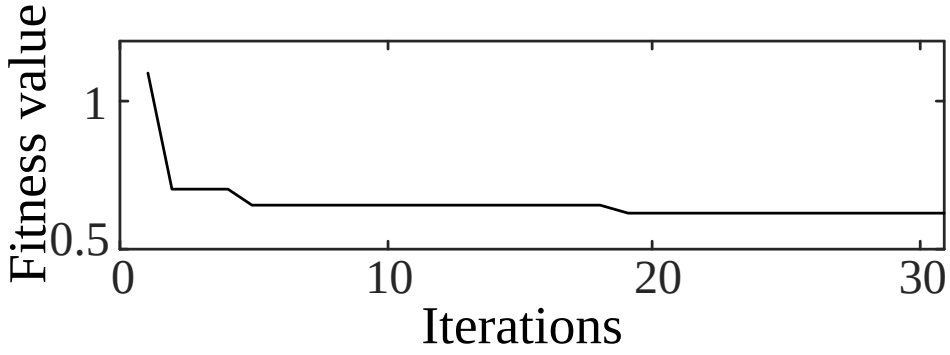


Figure 10. The fitness value of the system.

After the optimizations of initial values were completed, the optimal initial values could be substituted into the BPNN, to start real-time operation control. Since the BPNN does not require parameters to be updated in every control cycle, the execution frequency of the BPNN can be appropriately reduced. Here, the number of BPNN executions was set to one tenth of the interrupt frequency. The process of BPNN adjusting the ADRC parameters in real-time is shown in Figure 11.

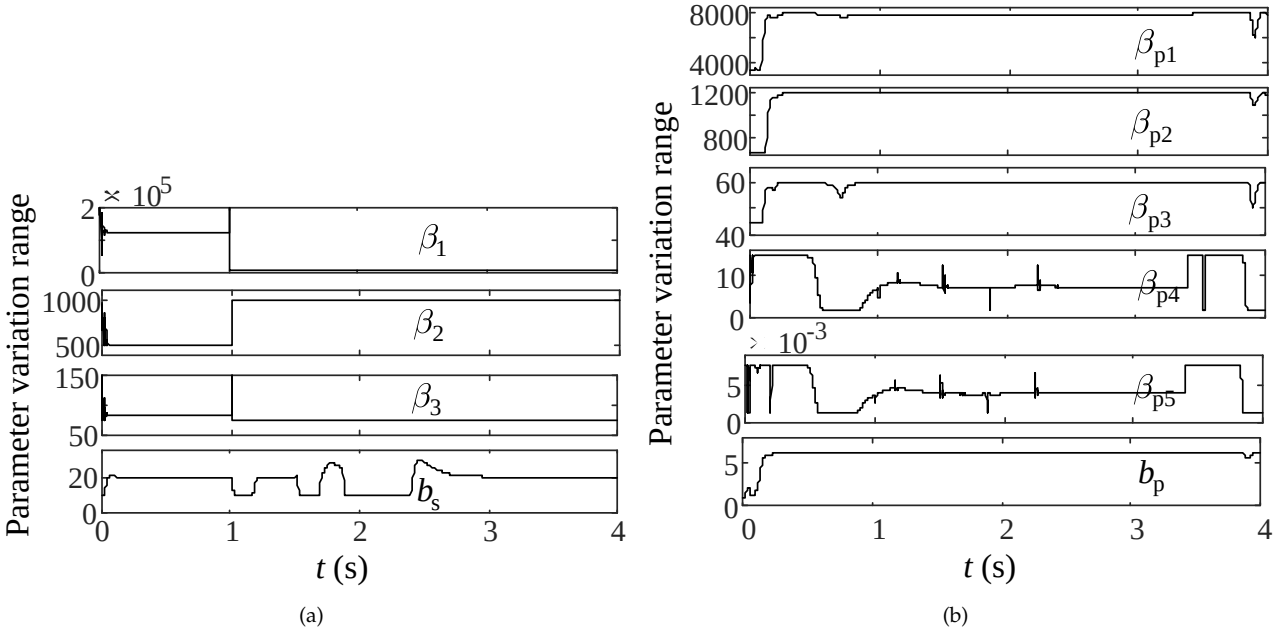


Figure 11. ADRC parameter adjustment process. (**a**) Motor-side, (**b**) suspension-side.

After the parameters were adjusted, the performance of the whole control system was favorable. Wherein, Figure 12 shows the speed waveforms of the different methods, and Figure 13 shows the displacement waveforms of the different methods. It can be seen from Figure 12, that after the GA optimization of the initial value, the speed waveform at low speed and high speed has a faster recovery speed and less jitter. As can be seen from Figure 13, after the GA optimizes the initial value, the anti-interference ability of displacement becomes stronger and the amplitude of displacement jitter decreases.

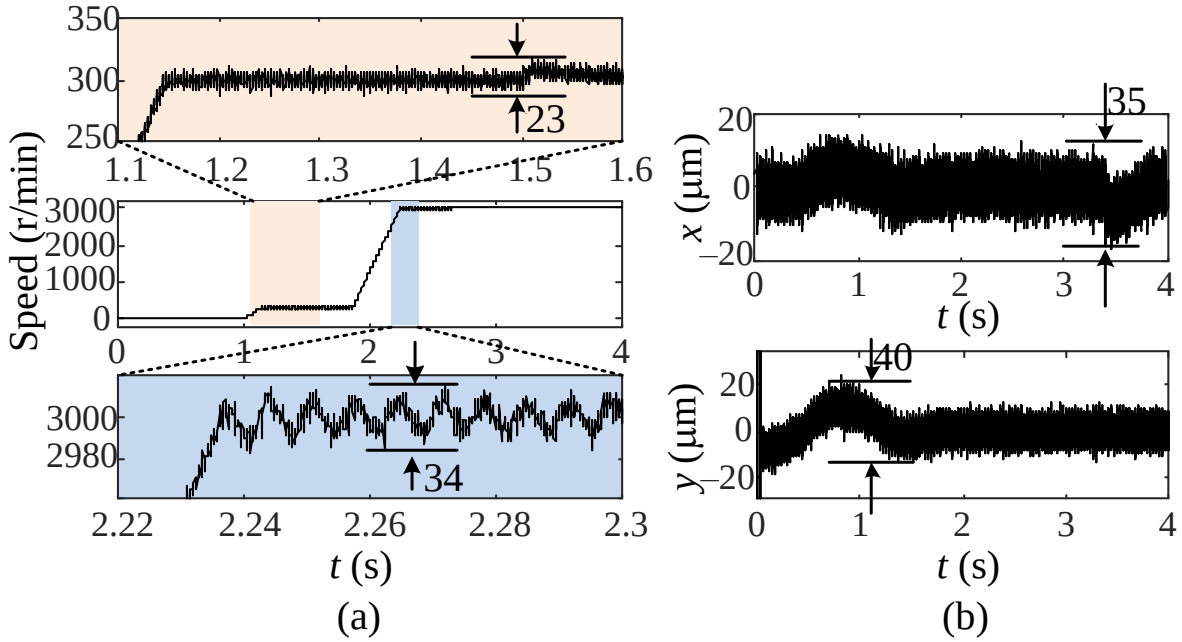


Figure 12. Speed waveforms in different methods. (**a**) Initial value not optimized, (**b**) initial value optimized by GA.

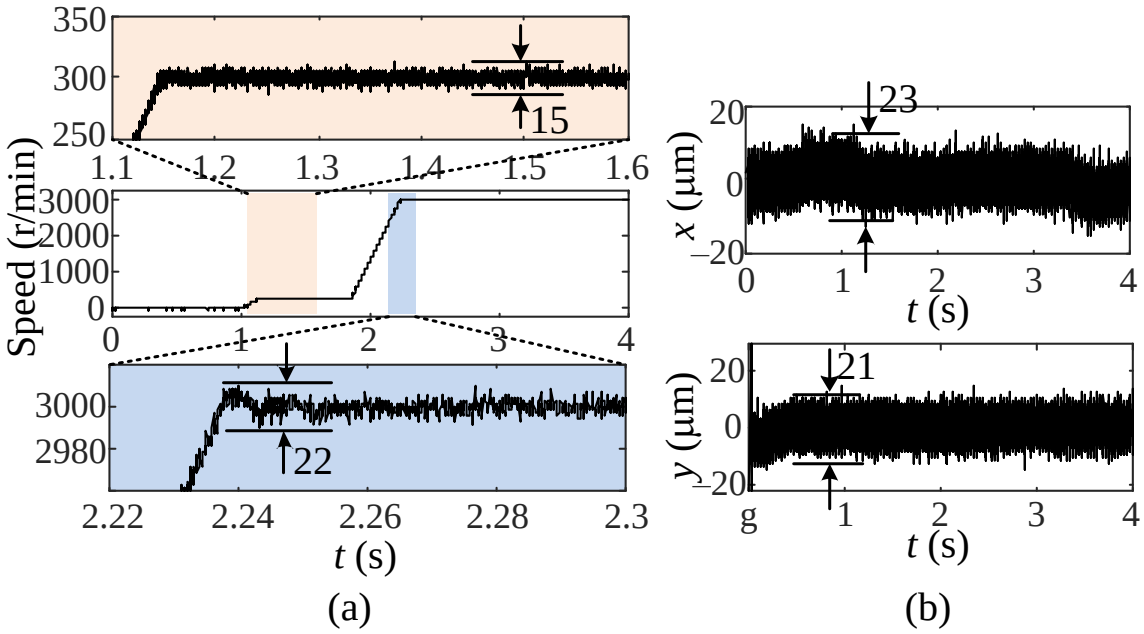


Figure 13. Displacement waveforms in different methods. (**a**) Initial value not optimized, (**b**) initial value optimized by GA.

6. Conclusions

In this paper, the GA is used to optimize the initial value of BPNN, and the optimized BPNN is applied to the real-time adjustment of ADRC parameters, so that the control effect is significantly improved. Not only the ADRC control models of motor-side and suspension-side are derived, but also the chain updating process of BPNN is derived. The genetic algorithm, based on floating point coding, is used to optimize the initial value of BPNN. The results show that the main contributions of the proposed method are as follows

(1) The floating point coded GA is used to optimize the initial value of BPNN, which effectively reduces the failure probability of the system, on the premise of reducing the amount of calculations.
(2) The BPNN algorithm, with adaptive learning rate and additional momentum factor, is used to dynamically adjust the ADRC parameters. Compared with the evolutionary algorithm, the algorithm has fewer computations and can meet the real-time control requirements of the control system.
(3) The proposed method dramatically enhances the control effect, resulting in a faster dynamic response and better steady-state performance for the motor speed waveform, and at the same time, the anti-interference ability of the displacement loop has been greatly improved.

Although this method improves the performance of the control system, in the process of BPNN initial value optimization, only single objective GA is used to achieve the initial value optimization, according to the control requirements. Multi-objective optimization techniques, including NSGA-II and MOPSO, can also be utilized to complete the optimization process, depending on various control needs.

Author Contributions: Conceptualization, X.W. and H.Z.; methodology, X.W.; validation, X.W. and H.Z.; formal analysis, X.W.; investigation, X.W.; resources, X.W.; data curation, X.W.; writing—original draft preparation, X.W.; writing—review and editing, X.W.; visualization, X.W.; supervision, H.Z.; project administration, H.Z.; funding acquisition, H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the Postgraduate Research and Practice Innovation Program of Jiangsu Province, grant number KYCX22_3655.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Nomenclature

α	Random number
B	Viscous damping
b	Motor parameter gain
β_1, β_2	The observer gains
β_3	Gain of LSEF
β_{p1}, β_{p2}, β_{p3}	Gains of observer
β_{p4}, β_{p5}	Gains of LSEF
bound	Boundary value
d, q	Displacements of d- and q-axis, respectively
η	Learning rate
F_x, F_y	Controllable suspension forces in x- and y-axis static coordinate system, respectively
F_{ux}, F_{uy}	Unbalanced magnetic pull in x- and y-axis static coordinate system, respectively
F_{dx}, F_{dy}	Additional disturbing forces in x- and y-axis static coordinate system, respectively
f_w	Total disturbance
f_{x1}, f_{y1}	Disturbance values
i_{Md}, i_{Mq}	Currents of d- and q-axis at motor-side, respectively
i_{Bd}, i_{Bq}	Currents of d- and q-axis at suspension-side, respectively
I_f	Equivalent excitation current amplitude of the permanent magnet
J	Moment of inertia
K_M, K_L	Maxwell force constant and Lorentz force constant, respectively
λ_{Md}, λ_{Mq}	Flux linkages of d- and q-axis at motor-side, respectively
λ_f	Permanent magnet flux linkage
M'	Mutual inductance coefficient
μ	Momentum factor
Maxgen	Maximum number of iterations

ω_{m}	Mechanical angular speed of the motor
ω_{M}^{*}	Given value of electric angular speed of the motor
P_{M}	Number of motor pole pairs
pop	Population number
T_{L}	Load torque
u_{s}	LSEF output value
w_{ij}^{hid}	Weight coefficient of the hidden layer
X_A^t, X_B^t	Two individuals
z_{M21}	The speed observation value
z_{M22}	The disturbance observation value
z_{B31}	The displacement observation value
z_{B32}	The observed value of displacement differential
z_{B33}	The disturbance value
ADRC	Active disturbance rejection control
BPMSM	Bearingless permanent magnet synchronous motor
ESO	Extended state observer
GA	Genetic algorithm
ITAE	Integral of time absolute error
NN	Neural network
PID	Proportional integral differential
SEF	State error feedback

References

1. Wang, X.; Zhu, H. Vibration Compensation Control of BPMSM With Dead-Time Effect Based on Adaptive Neural Network Band-Pass Filter. *IEEE Trans. Power Electron.* **2022**, *37*, 7145–7155. [CrossRef]
2. Valente, G.; Formentini, A.; Papini, L.; Gerada, C.; Zanchetta, P. Performance Improvement of Bearingless Multisector PMSM With Optimal Robust Position Control. *IEEE Trans. Power Electron.* **2019**, *34*, 3575–3585. [CrossRef]
3. Zhu, T.; Cao, X.; Yu, Q.; Deng, Z.; Hao, Z. Direct Torque Control With Phase Commutation Optimization for Single-Winding Bearingless Switched Reluctance Motor. *IEEE Trans. Power Electron.* **2022**, *37*, 13238–13249. [CrossRef]
4. Yang, Z.; Jia, J.; Sun, X.; Xu, T. An Enhanced Linear Adrc Strategy for a Bearingless Induction Motor. *IEEE Trans. Transp. Electrif.* **2022**, *8*, 1255–1266. [CrossRef]
5. Xie, K.; Yuan, Y.; Sun, Y.; Yang, F.; Li, G.; Huang, Y. Super-spiral Sliding Mode Controller Design for Single-Winding Bearingless Switched Reluctance Motor. *IET Electr. Power Appl.* **2022**, *16*, 1492–1501. [CrossRef]
6. Yang, Z.; Ding, Q.; Sun, X.; Zhu, H.; Lu, C. Fractional-Order Sliding Mode Control for a Bearingless Induction Motor Based on Improved Load Torque Observer. *J. Franklin Inst.* **2021**, *358*, 3701–3725. [CrossRef]
7. Zhu, H.; Shen, L. Decoupling Control of Outer Rotor Coreless Bearingless Permanent Magnet Synchronous Generator Based on Online Least Squares Support Vector Machine Inverse System and Internal Model Controllers. *Prog. Electromagn. Res. C* **2023**, *128*, 1–15. [CrossRef]
8. Jiang, J.; Zhu, H.; Wang, X. Decoupling Control Of Outer Rotor Coreless Bearingless Permanent Magnet Synchronous Generator Based On Fuzzy Neural Network Inverse System. *IEEE Trans. Transp. Electrif.* 2023, *to be published*. [CrossRef]
9. Chen, Y.; Zhou, Y. Active Disturbance Rejection and Ripple Suppression Control Strategy with Model Compensation of Single-Winding Bearingless Flux-Switching Permanent Magnet Motor. *IEEE Trans. Ind. Electron.* **2022**, *69*, 7708–7719. [CrossRef]
10. da Silva, W.; Salazar, A.O.; Vieira, P.V.F.; Jácome, M.C.; Villarreal, E.R.L. Radial Position Control of a Bearingless Machine with Active Disturbance Rejection Control Fuzzy an Approach. In Proceedings of the ICA-ACCA, Online, 22–26 March 2021; pp. 1–5.
11. Yang, Z.; Lu, C.; Sun, X.; Ji, J.; Ding, Q. Study on Active Disturbance Rejection Control of a Bearingless Induction Motor Based on an Improved Particle Swarm Optimization–Genetic Algorithm. *IEEE Trans. Transp. Electrif.* **2021**, *7*, 694–705. [CrossRef]
12. Zhu, H.; Gu, Z. Active Disturbance Rejection Control of 5-Degree-of-Freedom Bearingless Permanent Magnet Synchronous Motor Based on Fuzzy Neural Network Inverse System. *ISA Trans.* **2020**, *101*, 295–308. [CrossRef] [PubMed]
13. Zhou, X.; Gao, H.; Zhao, B.; Zhao, L. A GA-based Parameters Tuning Method for an ADRC Controller of ISP for Aerial Remote Sensing Applications. *ISA Trans.* **2018**, *81*, 318–328. [CrossRef]
14. Wang, Y.; Fang, S.; Hu, J. Active Disturbance Rejection Control Based on Deep Reinforcement Learning of Pmsm for More Electric Aircraft. *IEEE Trans. Power Electron.* **2023**, *38*, 406–416. [CrossRef]
15. Li, Z.; Yang, K.; Zhang, Y.; Liu, A.; Yang, F. Improved Active Disturbance Rejection Control of Permanent-Magnet Synchronous Motor Based on BP Neural Network. In Proceedings of the ICEMS, Hamamatsu, Japan, 24–27 November 2020; pp. 2137–2141.
16. Lin, P.; Wu, Z.; Liu, K.; Sun, X. A Class of Linear-Nonlinear Switching Active Disturbance Rejection Speed and Current Controllers for PMSM. *IEEE Trans. Power Electron.* **2021**, *36*, 14366–14382. [CrossRef]
17. Qu, L.; Qiao, W.; Qu, L. Active-Disturbance-Rejection-Based Sliding-Mode Current Control for Permanent-Magnet Synchronous Motors. *IEEE Trans. Power Electron.* **2021**, *36*, 751–760. [CrossRef]

18. Botchkarev, A. A New Typology Design of Performance Metrics to Measure Errors in Machine Learning Regression Algorithms. *Interdiscip. J. Inf. Knowl. Manag.* **2019**, *14*, 45–76. [CrossRef] [PubMed]
19. Naser, M.; Alavi, A. Error Metrics and Performance Fitness Indicators for Artificial Intelligence and Machine Learning in Engineering and Sciences. In *Architecture Structures and Construction*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 1–19.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.