Particle Swarm Optimization Algorithm Based on Natural Selection and Simulated Annealing for PID Controller Parameters

Minlan Jiang(✉), Ying Wu, Lan Jiang, and Fei Li

College of Mathematics, Physics and Information Engineering,
Zhejiang Normal University, Jinhua 321004, China
xx99@zjnu.cn

Abstract. The values of a PID controller's parameters determine the controller's effect. The particle swarm optimization (PSO) algorithm is often used to optimize the controller's parameters. However, PSO has some inherent defects, such as premature convergence and easily turning into a local optimization. In this paper, an improved particle swarm optimization algorithm based on a natural selection strategy and a simulated annealing mechanism is proposed to optimize the PID controller's parameters. In the improved PSO algorithm, the natural selection strategy is used to accelerate the rate of convergence, and the simulated annealing mechanism is employed to ensure the accuracy of the search and increase its ability to avoid local optima. The improved algorithm not only guarantees the convergence speed but also has a better ability to jump out of the local optimum trap. To verify the performance of the improved algorithm, four types of algorithms are selected to optimize the PID controller parameters of the Second-order Time-delayed System and the Permanent Magnet Synchronous Motor (PMSM) Servo System. They are the PSO algorithm, the optimization algorithm proposed in this paper (NAPSO), the seeker optimization algorithm (SOA), and the genetic algorithm (GA). The results show that the improved algorithm has a better optimal solution.

Keywords: PID · PSO · PMSM · GA

1 Introduction

In industrial control, proportion integration differentiation (PID) controllers have been widely used. The parameter optimization of PID controller has great influence on PID control system. In recent years, Zhou et al. [1] used particle swarm optimization to find an optimal set of PID control parameters in the target space, and an air-conditioning temperature control system was designed as an example. Wei et al. [2] use a GA algorithm to optimize the PID controller parameters, and they designed a PID controller parameter optimization system that consists of a microcontroller module. The host computer processes the real-time voltage and uses ITAE to evaluate the effectiveness of the optimized PID controller parameters. Zhong et al. [3] proposed a multi-agent simulated annealing algorithm based on a particle swarm optimization algorithm to address continuous function optimization problems. Lin et al. [4] used the multi-agent

H. Song et al. (Eds.): SIMUtools 2019, LNICST 295, pp. 363–373, 2019.
https://doi.org/10.1007/978-3-030-32216-8_35

simulated annealing algorithm to predict protein structure. However, there is no single improved PSO algorithm that is based on the methods of natural selection and simulated annealing. This paper adds two methods into the basic PSO algorithm simultaneously; then, a new PSO algorithm based on natural selection and simulated annealing (NAPSO) is produced. The simulation results show that using the improved algorithm to optimize the PID controller parameters can achieve better control performance.

This paper is organized as follows: Sect. 2 gives a brief description of the PSO algorithm and optimization algorithm. Section 3 introduces the PID Control and discusses the process of NAPSO-PID optimization. Section 4 reports the Second-order Time-delayed System, PMSM servo System and simulation results. Finally, Sect. 5 gives the conclusions of this paper.

2 PSO Algorithm and NAPSO Optimization Algorithm

2.1 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) was proposed in 1995 by Dr. Eberhart and Dr. Kennedy and was derived from research on bird flocks' preying behavior. In PSO, every single solution is a "bird" in the search space. We call it a "particle". All of the particles have fitness values that are evaluated by the fitness function to be optimized, and each particle has a velocity to determine the direction and distance of its flight.

At the beginning of the algorithm, the particle swarm is initialized to a set of random values in the solution space. Then, the particles are "flown" through the problem space by following the current optimum particles. In each iteration, every particle is updated by tracking the two best positions: the first position is the best position that it has achieved so far, and this position is called the personal best position (P_{best}). The other position that is tracked by the particle swarm optimizer is the best position that is obtained so far by all particles in the population. This position is the global best and is called the global best position (g_{best}).

The update equation of the velocity and position is shown by the following expression:

$$\begin{aligned} v_{i.d}(t+1) &= \omega v_{i.d}(t) \\ &+ c_1 r_1 [p_{best} - x_{i.d}(t))] + c_2 r_2 [g_{best} - x_{i.d}(t))] \end{aligned} \tag{1}$$

$$x_{i.d}(t+1) = x_{i.d}(t) + v_{i.d}(t+1) \tag{2}$$

In the D dimension space, t is the iteration number, $v_{i,d}(t)$ is the velocity of particle i at iteration t, $\chi_{i.d}(t)$ is the position of particle i at iteration t, and ω is the inertia weight to be used to control the impact of the previous history of velocities. Here, c_1 is the cognition learning factor, c_2 is the social learning factor, γ_1 and γ_2 are random numbers that are uniformly distributed in [0, 1], P_{best} is the particle best value for the individual variable of particle i, and g_{best} is the global best position variable of the particle swarm.

2.2 NAPSO Algorithm

When a particle's speed, position ℓ and fitness value have been updated, the particle moves to a random position ℓ_1' in its neighborhood and computes its new fitness value. The general procedure of NAPSO is illustrated in the flow chart in Fig. 1.

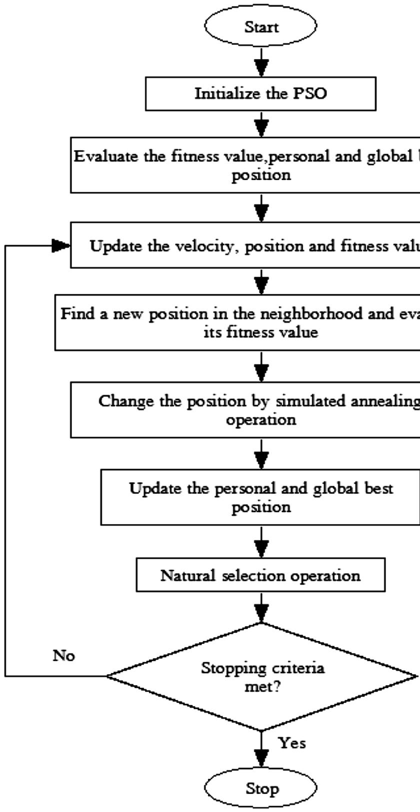

Fig. 1. Flow chart of the NAPSO algorithm

$$\ell_1' = \ell + [v_{max} - v_{min}] * \gamma_1 \quad (3)$$

Where γ_1 is the normally distribution random numbers of D-dimension that are distributed in [0, 1].determines whether to accept the new position and then updates the particle's P_{best} and g_{best}. The simulated annealing operation can significantly increase the ability of the algorithm to jump out of the local optimum trap. The system uses the simulated annealing algorithm to determine whether to stay in the new position and thereby improve the rapid convergence effect of the birds. In the improved algorithm, at the end of each iteration, all particles have been ranked by their fitness values, from best to worst, using the better half to replace the other half. In this way, the stronger adaptability particles are saved.

However, both operations have their own disadvantages. The simulated annealing operation will slow the speed of convergence, thus increasing the convergence time. The natural selection operation will reduce the diversity of the samples. These two operations can compensate for each other. The simulated annealing operation can increase the sample diversity, and the natural selection operation can speed up the convergence. The two operations complement each other in the improved algorithm to both ensure the convergence speed of the algorithm and guarantee that the ability of the algorithm to jump out of the local optimal trap will be strengthened.

3 PID Controller Parameter Optimization by NAPSO

3.1 The PID Control System Based on NAPSO

The proportional-integral-derivative (PID) controller is widely used in industrial control systems. The “textbook” version of the PID algorithm is described by [5]:

$$u(t) = k_p e(t) + k_i \int_0^t e(t)dt + k_d \frac{de(t)}{dt} \tag{4}$$

Where k_p is the proportional gain, k_i is the integral gain, and k_d is the derivative gain.

The core of the PID controller parameters optimization is to use an algorithm to optimize the PID controller's parameters: k_p, k_i and k_d. When using the NAPSO algorithm to optimize the PID controller's three parameters, Block diagram of the NAPSO-PID control system is shown in Fig. 2.

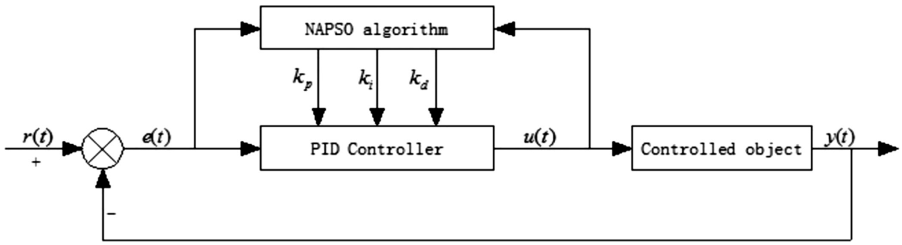

Fig. 2. Block diagram of the NAPSO-PID control system

In each iteration, the NAPSO algorithm will optimize the values of the three parameters and deliver the results to the PID controller. The PID controller calculates the control volume $u(t)$ according to formula (4) and delivers it to the controlled object, which gradually reduces the deviation [6].

3.2 PID Parameter Optimization Based on NAPSO

The essence of the PID controller parameters optimization is a parameter optimization problem based on an objective function. The objective function is the fitness function. In this paper, the fitness function is defined as an ITAE (Integral Time Absolute Error) index, which is often used to reflect the system's quality in a control system. ITAE represents the integral of the time multiplied by the absolute value of the error, which can be expressed as follows:

$$J = \int_0^{\infty} t|e(t)|dt \tag{5}$$

The NAPSO algorithm is applied to optimize the PID controller parameters as follows:

Step 1: Initialize ω, c_1, c_2, and calculate the fitness value of each particle. Define the initial p_{best} and initial g_{best}. Set T to be the simulated temperature; the initial T is 5000 °C, and the lower limit of T is 1 °C.
Step 2: Update the position l and velocity of each particle.
Step 3: Evaluate the fitness value f', randomly find a new position l_1' in the neighborhood of the particle, and calculate the new fitness value (f_1') of the new position.
Step 4: Compare the fitness value with the new fitness, and evaluate the difference, $\Delta f = f_1' - f'$. If $\Delta f < 0$ and $f_1' < g_{best}$, then replace the original position with the new position.
Step 5: If $f_1' > g_{best}$, then keep the original position. If $\Delta f > 0$ and $f_1' < g_{best}$, then generate a random number $(rand(1))$ as a probability. But not $p = \exp((-1) * (f_1' - f')/T) > r4$, r_4 is a random number that is uniformly distributed in [0, 1]. Then accept the new position. According to the position of the particle, update the personal best position and global best position.
Step 6: When all of the updates of the particles are finished, then rank all of the particles according to the fitness value. Replace the information of half of the particles (position and velocity) with the information of the other half (the better) particles, and update the temperature $T = T * 0.9$.
Step 7: Stop searching when the maximum iteration limit or the fitness value limit is reached, and then output the three variables of the particles and their corresponding fitness value; otherwise, return to step 2.

To verify the validity of the algorithm, two different systems (Second-order Time-delayed System and PMSM Servo System) have been considered to illustrate the effectiveness of the proposed method. At the same time, the particle swarm optimization (PSO), seeker optimization algorithm (SOA) [7, 8] and genetic algorithm (GA) were also applied to show the performance of the proposed algorithm.

4 Experiments

4.1 PMSM Servo System PID Controller Parameters Optimization and Simulation Results

As an important part of Computer numerical control(CNC) machine tools, the servo system directly determines the quality of the machining performance. Currently, the permanent magnet synchronous motor (PMSM) servo system is a high performance servo system that is commonly used in CNC machine tools [9]. The state equation of the PMSM servo system in d-q coordinates is expressed as follows [10]:

$$\begin{bmatrix} \dot{i}_d \\ \dot{i}_q \\ \dot{\omega}_r \end{bmatrix} = \begin{bmatrix} -R/L & p_n\omega_r & 0 \\ -p_n\omega_r & -R/L & -p_n\varphi_f/L \\ 0 & 3p_n\varphi_f/2J & 0 \end{bmatrix} \begin{bmatrix} i_d \\ i_q \\ \omega_r \end{bmatrix} + \begin{bmatrix} u_d/L \\ u_q/L \\ -T_L/J \end{bmatrix} \tag{6}$$

To simplify the controller design of the PMSM servo system, $i_d = 0$ is often used in vector control. The PMSM servo system is shown in Fig. 3.

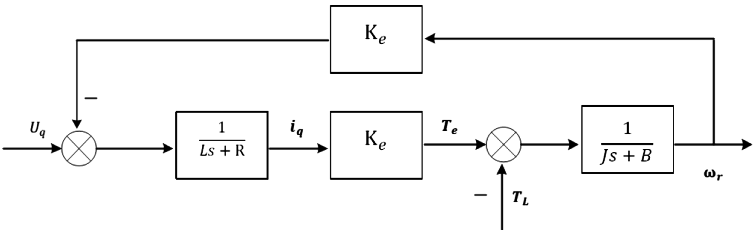

Fig. 3. Block diagram of the PMSM servo system

Here, $K_c = \frac{3}{2}p_n\varphi_f$ is the torque coefficient, $K_e = p_n\varphi_f$ is the back EMF constant, and B is the viscosity damping coefficient.

According to Eq. (6) and Fig. 3, the CNC machine feeding servo system in the field orientation (controlled under) can be expressed as follows:

$$G_2(S) = \frac{K_c}{LJS^2 + (RJ + LB)S + BR + K_cK_e} \tag{7}$$

The parameters of the PMSM servo system are designed as follows:

$$L = 8.5e - 3(\mathrm{H}), R = 2.875(\Omega),$$

$$J = 0.8e - 3(\mathrm{km} \cdot \mathrm{m}^2), B = 0.02(\mathrm{N} \cdot \mathrm{m}/(\mathrm{rad/s})), p_n = 4, \varphi_f = 0.175(\mathrm{Wb}),$$

Equation (7) can be rewritten as follows:

$$G_2(S) = \frac{1.05}{6.8 \cdot 10^{-6} S^2 + 2.47 \cdot 10^{-3} S + 0.7925} \tag{8}$$

In the experiment, the system expressed by Eq. (8) is used as the second controlled object. The system PID controller parameters are optimized by NAPSO, PSO, SOA and GA. The comparisons of the step responses based on the four algorithms are displayed in Fig. 4.

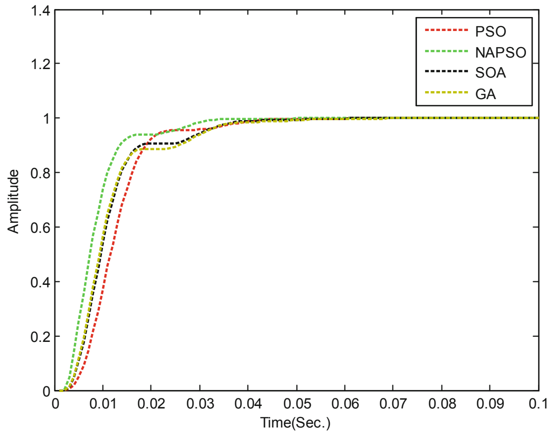

Fig. 4. Step responses of the PMSM servo system

Figure 4 shows that the NAPSO optimization step response curve is clearly superior and the convergence speed of the NAPSO is faster than the others. At 0.035 s, the curve of the NAPSO algorithm is closest to the curve of the idea output. The simulation results of the PMSM feed system are presented in Table 1.

Table 1. Simulation results of the PMSM servo system

Tuning method	k_p	k_i	k_d	Settling time (sec)	Overshoot (%)	ITAE value
PSO	0.1094	61.6899	0	0.03741	0	12.3795
NAPSO	0.2058	92.2902	3.7148e − 004	0.02914	0	8.3796
GA	0.1363	68.0213	0	0.03615	0	11.8784
SOA	0.1116	68.3148	0	0.03562	0	11.3453

As Table 1 shows, the settling time is approximately 0.029 s for NAPSO compared with approximately 0.037, 0.36 and 0.035 for PSO, GA and SOA, respectively. In addition, NAPSO has the best ITAE value of the four algorithms, acquiring an ITAE value of 8.3796, and the ITAE values of PSO, GA and SOA are 12.3795, 11.8784 and

11.3453, respectively. It is clear that NAPSO has the best ITAE value, settling time and overshoot among the four algorithms in the second experiment.

4.2 Second-Order Time-Delayed System PID Controller Parameters Optimization and Simulation Results

In modern industrial process control, many systems can similarly be seen as a first- or second-order typical system. In this paper, select the Second-order Time-delayed System as the first controlled object. The mathematical expression of the first controlled object is the following:

$$G_1(s) = \frac{0.05}{s^2 + 0.2s + 0.05} e^{-3s} \tag{9}$$

Next, with the particle swarm optimization (PSO), seeker optimization algorithm (SOA) and genetic algorithm (GA) to optimize the PID controller's parameters. For the first experiment, to make a fair comparison, the maximum generation, population size, minimum fitness value, range of gains, dimension of search space and initial positions are identical for all of the algorithms. The maximum number of generations is 100, the minimum fitness value is 0.1, the size of the population is 100, and the dimension of the search space is 3. The parameters for the PSO were set as follows: the acceleration constants c_1 and c_2 are 2, the dimension is 3, and the inertia weight $w = 0.9$. The parameters of the SOA are designed as follows: the minimum membership degree $U_{\min} = 0.0111$, the maximum membership degree $U_{\max} = 0.95$, the maximum weight $w_{\max} = 0.9$, and the minimum weight $w_{\min} = 0.1$. In the GA algorithm, the crossover probability is 0.9, and the mutation probability updates in a self-adaptive manner.

The optimize results of the four types of algorithm are shown in Fig. 5, and the step responses are presented in Fig. 6

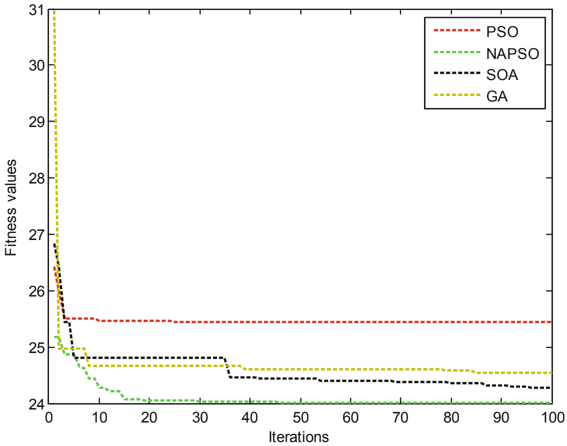

Fig. 5. Response curve of the second-order time-delayed system

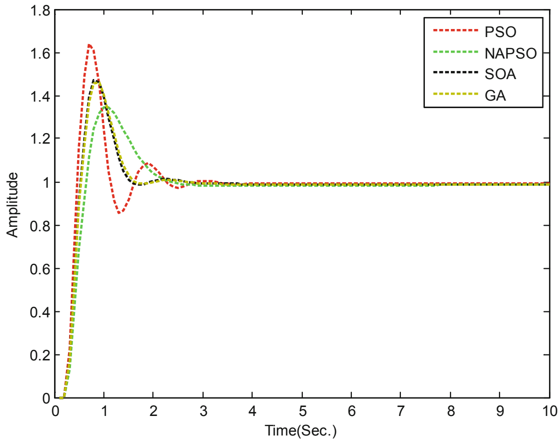

Fig. 6. ITAE curves of the second-order time-delayed system

In Fig. 5, at the 15th iteration, the ITAE value of the NAPSO algorithm is approximately 24.08, and the PSO, GA, and SOA are approximately 25.4, 24.6, and 24.8, respectively. It can clearly be seen that the NAPSO algorithm has the best optimized result among the four types of algorithms, and when the four types of algorithms obtain the same fitness value, the NAPSO algorithm requires a much smaller number of iterations than the other algorithms.

In Fig. 6, compared with PSO, the NAPSO has a smaller overshoot and shorter oscillation cycle, and the system can achieve stability more quickly. The accurate simulation results of the Second-order Time-delayed System are given in Table 2.

Table 2. Simulation results of the Second-order Time-delayed System

Tuning method	k_p	k_i	k_d	Settling time (sec)	Overshoot (%)	ITAE value
PSO	100	0	80.8380	2.6408	63.79	25.3166
NAPSO	55.8028	0.6136	47.2314	2.1245	36.72	24.0218
GA	75.9074	0.6447	60.7940	1.5371	48.80	24.7880
SOA	77.3687	68.3148	62.7102	1.4745	47.42	24.3978

In Table 2, the ITAE value is approximately 24.0218 for NAPSO compared with approximately 25.3166, 24.7880 and 24.3978 for the PSO, GA and SOA, respectively. Furthermore, the system overshoot is 36.72% in the case of NAPSO. Compared with NAPSO, the overshoots of the PSO, GA and SOA are 63.79%, 48.80% and 47.42%, respectively. Overall, the NAPSO has the smaller ITAE value and overshoot among the four algorithms in the first experiment.

From the two experiments, it can be seen that when the initial positions and range of gains are identical for all of the algorithms, the different algorithms produce different values, and the NAPSO algorithm has the better ITAE values in the two experiments. This finding indicates that the NAPSO algorithm has a better capability for a global search than the other two algorithms. It is also clearly obvious that in the NAPSO algorithm, the rapid convergence has improved because the update of the velocity and positions of the particles no longer depend too much on the current best particle.

5 Conclusions

The optimization of the PID controller parameters is a hotspot in modem manufacturing technology. In this study, an improved PSO algorithm (NAPSO) based on simulated annealing and natural selection is proposed and used in PID controller parameters optimization. The simulation experiments show that the proposed algorithm performs well in the Second-order Time-delayed System and the PMSM Servo System. The results of the NAPSO were compared with PSO, SOA and GA, with the result that the NAPSO has higher accuracy and faster convergence than the other three algorithms. The proposed algorithm can provide a new method for addressing PID controller parameter optimization and has definite value for applications in modem manufacturing technology. However, the NAPSO has the disadvantage of strong randomicity. In the future, we plan to study which is the more effective method for improving the prediction results.

Acknowledgment. This work was supported in part by the National Natural Science Foundation of China (Nos. 51305407, 61571104).

References

1. Zhou, Y., Nie, J., Han, N., Chen, C., Yue, Z.: Study on PID parameters tuning based on particle swarm optimization. In: Advanced Materials Research, vol. 823, pp. 432–438 (2013)
2. Wei, C., Liu, Z., Li, P., Chen, J., Zhang, S.: The PID parameters optimization method based on GA for NC power supply. In: Huang, B., Yao, Y. (eds.) Proceedings of the 5th International Conference on Electrical Engineering and Automatic Control. LNEE, vol. 367, pp. 97–104. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-48768-6_12
3. Zhong, Y., Ning, J., Zhang, H.: Multi-agent simulated annealing algorithm based on particle swarm optimisation algorithm. Int. J. Comput. Appl. Technol. **43**(4), 335–342 (2012)
4. Lin, J., Ning, J., Du, Q., Zhong, Y.: Multi-agent simulated annealing algorithm based on particle swarm optimization algorithm for protein structure prediction. J. Bionanoscience Sci. **7**(1), 84–91 (2013)
5. Laskawski, M., Wcislik, M.: Sampling rate impact on the tuning of PID controller parameters. Int. J. Electron. Telecommun. **62**(1), 43–48 (2016)
6. Modares, H., Alfi, A., Naghibi, S., Sistani, M.B.N.: Parameter estimation of bilinear systems based on an adaptive particle swarm optimization. Eng. Appl. Artif. Intell. **23**(7), 1105–1111 (2010)
7. Zhu, Y., Dai, C., Chen, W.: Seeker optimization algorithm for several practical applications. Int. J. Comput. Intell. Syst. **7**(2), 353–359 (2014)
8. Chen, C., Li, J., Luo, J., Xie, S., Li, H.: Seeker optimization algorithm for optimal control of manipulator. Ind. Robot **43**(6), 677–686 (2016)

9. Premkumara, K., Manikandan, B.V.: Speed control of brushless DC motor using bat algorithm optimized adaptive neuro-fuzzy inference system. Appl. Soft Comput. **32**, 403–419 (2015)
10. Ouledali, O., Meroufel, A., Wira, P., Bentouba, S.: Direct torque fuzzy control of PMSM based on SVM. Energy Proc. **74**, 1314–1322 (2015)