International Journal of Automation and Computing
15(1), February 2018, 103-114
DOI: 10.1007/s11633-016-0990-6

Hybrid Particle Swarm Optimization with Differential Evolution for Numerical and Engineering Optimization

Guo-Han Lin[1,2] Jing Zhang[1] Zhao-Hua Liu[3]

[1]College of Electrical and Information Engineering, Hunan University, Changsha 410082, China
[2]College of Electrical and Information, Hunan Institute of Engineering, Xiangtan 411101, China
[3]School of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

Abstract: In this paper, a hybrid particle swarm optimization (PSO) algorithm with differential evolution (DE) is proposed for numerical benchmark problems and optimization of active disturbance rejection controller (ADRC) parameters. A chaotic map with greater Lyapunov exponent is introduced into PSO for balancing the exploration and exploitation abilities of the proposed algorithm. A DE operator is used to help PSO jump out of stagnation. Twelve benchmark function tests from CEC2005 and eight real world optimization problems from CEC2011 are used to evaluate the performance of the proposed algorithm. The results show that statistically, the proposed hybrid algorithm has performed consistently well compared to other hybrid variants. Moreover, the simulation results on ADRC parameter optimization show that the optimized ADRC has better robustness and adaptability for nonlinear discrete-time systems with time delays.

Keywords: Particle swarm optimization (PSO), active disturbance rejection control (ADRC), differential evolution algorithm, chaotic map, parameter tuning.

1 Introduction

Particle swarm optimization (PSO) was proposed by Kennedy and Eberhart in 1995, which was inspired by the bird flocking social behavior[1, 2]. Due to its implementation simplicity, few parameters and fast convergence, PSO has been successfully applied in many areas[3−5]. While the PSO algorithm can converge quickly in early stages, it is prone to diversity loss during iterations, and may get trapped in a local optimum. Therefore, how to overcome the local optimum drawback is still an important issue for PSO applications. Hybridization is one of the most efficient strategies to improve the performance of optimization algorithms[6]. Researchers have conducted many related studies and proposed various hybrid algorithms based on PSO to deal with the problems of early loss of diversity, premature convergence and slow convergence rate. Differential evolution (DE) is a population based stochastic search algorithm, and was developed by Storn and Price[7]. Many hybrid versions of DE and PSO have been presented in the past decade. In [8], an algorithm combining DE algorithm and PSO, called DEPSO-R, was proposed for economic dispatch problems. In that method, a particle′s position was updated only if its offspring particle has better fitness, this strategy makes the algorithm has less computational complex than some existing hybrid algorithms. Another hybrid PSO algorithm coupled by a differential operator with the velocity update scheme is proposed in [9], and the hybrid PSO (PSO-DV) algorithm was reported with robust performance on seven global optimization problems compared with DE, PSO and other PSO variants. Another hybrid algorithm named DEPSO-KL was presented by Kim and Lee[10]; in this algorithm, each individual particle updates its current position according to a predefined probability. Through adaptive selection of control parameters, DEPSO-KL can find the optimized solution with small numerical oscillations.

In this paper, a novel hybrid version of PSO and differential evolution (DE), called HCPSODE, is proposed. A new nonlinear strategy for decreasing inertia weights, along with a chaotic map with greater Lyapunov exponent, is introduced to balance the exploration and exploitation abilities of the proposed algorithm. A DE operator is used to help particles jump out of stagnation when the diversity of the particles decreases rapidly at later stages of iterations. The performance of the proposed HCPSODE is evaluated by solving twelve of the CEC2005 contest functions and eight CEC2011 real-world optimization problems, and studying its application to parameter optimization of ADRC in controlling nonlinear discrete-time systems with time delays.

Research Article
Manuscript received April 23, 2014; accepted May 11, 2015; published online June 20, 2016
Recommended by Associate Editor Dong-Ling Xu
This work was supported by National Natural Science Foundation of China (Nos. 61174140 and 61203016), Ph. D. Programs Foundation of Ministry of Education of China (No. 20110161110035) and China Postdoctoral Science Foundation Funded Project (No. 2013M540628).

2 Hybrid PSO algorithm

2.1 Basic PSO

In the basic PSO algorithm, a swarm is generated randomly in the search space. Each particle in the swarm represents a candidate solution, and is treated as a point flying in the solution space. Each particle remembers its best position denoted by p_{best}. The globally best position is denoted by g_{best}, which represents the best position among all particles. The movement of each particle is determined by its own previous best position and the globally best position. The velocity and position of particle i are altered by the following recursive equations:

$$V_{id}(t+1) = \omega V_{id}(t) + c_1 \times \text{rand}_1()(p_{\text{best}}(t) - X_{id}(t)) + c_2 \times \text{rand}_2()(g_{\text{best}}(t) - X_{id}(t)) \tag{1}$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \tag{2}$$

where $X_i = (x_{i1}, x_{i2}, \cdots, x_{id})$ and $V_i = (v_{i1}, v_{i2}, \cdots, v_{id})$ are current position and current velocity of the i-th particle respectively, ω is the inertia weight which balances the local and global search during the optimization process, c_1 and c_2 are cognitive and social acceleration factors respectively, $\text{rand}_1()$ and $\text{rand}_2()$ are uniformly distributed random numbers in the range between 0 and 1.

2.2 Differential evolution

Differential evolution (DE) has gained much attention due to its effectiveness and simplicity. As a parallel direct-search meta-heuristic algorithm, DE is characterized by memorizing of individual optimal value and sharing mutual information. Its basic idea is the use of differential variation and crossover recombination between individuals to generate trial vectors. Based on a greedy selection strategy, an individual is generated from the target vector and trial vectors, and is entered to the next generation. The procedures for implementing a DE algorithm can be summarized as the following steps[11]:

Step 1. Initialize D-dimensional population vectors with NP individuals, and each individual can be described as follows:

$$X_i^g = (x_{i1}^g, x_{i2}^g, \cdots, x_{iD}^g), \quad (i = 1, 2, \cdots, NP) \tag{3}$$

where x_{ij}^g represent the j-th dimension of the i-th individual in the g_{th} generation.

Step 2. Randomly select one individual from current population as the target vector, and select two other different individuals to produce the differential vector. A mutation vector can be generated through the following strategy:

$$v_{i,j}^g = x_{r1,j}^g + F \times (x_{r2,j}^g - x_{r3,j}^g) \tag{4}$$

where $r_1, r_2, r_3 \in (1, 2, \cdots, NP)$ are integers randomly generated and mutually different, and are also different from i; $F \in (0, 2)$ is a scaling factor, and controls the amplification of the differential vector.

Step 3. After generation of the mutation vector, for each target vector $x_{i,j}^g$, a crossover is produced between the target vector and the mutation vector to generate a trial vector $u_{i,j}^g$.

$$u_{i,j}^g = \begin{cases} v_{i,j}^g, & \text{rand}(0,1) \le CR \\ x_{i,j}^g, & \text{otherwise} \end{cases} \tag{5}$$

where u denotes the trial vector, v is the mutation vector, $CR \in [0,1]$ is called the crossover factor, and rand(0,1) is a uniform random number generator.

Step 4. Using a greedy selection strategy, the selection operation is performed to choose the better one from the target vector and the trial vector to enter the next generation.

$$x_i^{g+1} = \begin{cases} u_i^g & \text{when} \quad f(u_i^g) < f(x_i^g) \\ x_i^g & \text{when} \quad f(u_i^g) \ge f(x_i^g) \end{cases} \tag{6}$$

where x_i^{g+1} is the new individual in the next generation, and $f()$ is the fitness function. When the fitness function value of the trial vector u_i^g is smaller than that of the target vector x_i^g, the next generation will be replaced by u_i^g.

Step 5. Repeat until the termination condition is met, or it reaches to the maximum number of iterations.

2.3 Hybrid PSO and DE with a chaotic map

For complex optimization problems, the basic PSO algorithm can easily converge to local optimum, which leads to slow convergence speed and premature convergence. The main reason for premature convergence is that the diversity of particles decreases rapidly during iterations. Chaos theory has been applied into many fields including physics, engineering and biology[12]. The main feature of chaotic systems is their sensitivity to initial conditions, even a minute change in initial conditions can later lead to considerably different behaviors. Several chaotic time-series sequences, such as the logistic map, have been applied in optimization. Optimization algorithms based on chaos theory are stochastic search methodologies that differ from any of existing evolution algorithms[13]. In the basic PSO algorithm, parameters such as the inertia weight ω, the cognitive and social acceleration factors c_1, c_2, are key factors that determine the PSO convergence performance[14, 15]. In our proposed algorithm, a new nonlinear strategy is adopted for decreasing the inertia weight:

$$\omega_i = \omega_{\text{end}} + (\omega_{\text{start}} - \omega_{\text{end}})(1-q)$$
$$\begin{cases} q = \dfrac{t}{t_{\max}}, & \text{if } \dfrac{g_{\text{best}}}{p_{\text{best}}} < \dfrac{t_i}{t_{\max}} \\ q = \dfrac{g_{\text{best}}}{p_{\text{best}}}, & \text{if } \dfrac{g_{\text{best}}}{p_{\text{best}}} \ge \dfrac{t_i}{t_{\max}} \end{cases} \tag{7}$$

where $t_{\max}$ is the maximum number of iterations, t_i is the current number of iterations, ω_{start} and ω_{end} are the maximum and minimum of the inertia weight. The parameters $\text{rand}_1()$ and $\text{rand}_2()$ affect the convergence performance of PSO algorithm, in our proposed algorithm, an iterative chaotic map with infinite collapses (ICMIC) is used because it has greater Lyapunov exponent and is more sensitive to the initial value. Using of chaotic sequences in PSO can improve the global convergence and help the algorithm escape from local minima[16]. The ICMIC equation is given by

$$x_{n+1} = \sin\left(\frac{a}{x_n}\right), \quad n = 0, 1, 2, \cdots, \quad x_0 \neq 0, \quad a > 0. \quad (8)$$

The ICMIC map used in this work is illustrated in Fig.1.

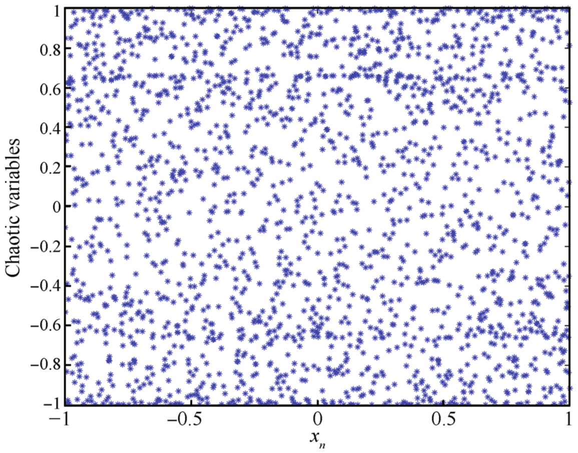


Fig. 1 ICMIC chaotic map

The velocity and position of particles in the proposed algorithm are updated by the following equations:

$$V_{id}(t+1) = \omega V_{id}(t) + c_1 \times r_{c1}(t)(p_{\text{best}}(t) - X_{id}(t)) + c_2 \times r_{c2}(t)(g_{\text{best}}(t) - X_{id}(t)) \quad (9)$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (10)$$

where $r_{c1}(t)$ and $r_{c2}(t)$ are calculated from ICMIC map.

During the iteration process of basic PSO, once a particle finds a better solution, all the other particles will be attracted and gathered to it quickly. When dealing with problems with many local minima, the swarm will be stagnated due to the lack of momentum and the algorithm stops evolving.

To avoid the stagnation often encountered by PSO, DE algorithm is incorporated into the PSO. When the swarm settles into stagnation state, DE is used to provide the necessary momentum for particles to roam across the search space and escape from the local optimum. Criteria for detecting stagnation includes maximum swarm radius, cluster analysis and objective function fitness. Differing from aforementioned criteria, in our study, the median velocity of vector norm of the particles (denoted by v_m) is defined as the state when v_m approaches a pre-specified stagnation threshold λ. λ is a positive scalar value to be specified by the user. An empirical study shows that the proposed method is not too sensitive to the stagnation threshold. As the stagnation threshold determines when the DE operator is to be merged into PSO, if it is set to reasonable conservative value, the hybrid chaotic particle swarm optimization with differential evolution (HCPSODE) method yields improved results.

The flowchart of HCPSODE algorithm is shown in Fig. 2.

3 Numerical experiments

A group of benchmark test problems have been selected to evaluate the performance of the proposed HCPSODE algorithm. The benchmark test problems consist of twelve ($F_6 - F_{17}$) multimodal test functions proposed in the CEC2005 special session on real-parameter optimization and 10 problems related to bound constrained optimization proposed in the CEC2011 competition on real world optimisation problems.

Among the CEC2005 benchmark suite, $F_6 - F_{12}$ are basic multimodal functions, $F_{13} - F_{14}$ are expanded multimodal functions and $F_{15} - F_{17}$ are hybrid compositions of functions with a large number of local minima.

More details about these test problems can be found in [17−18].

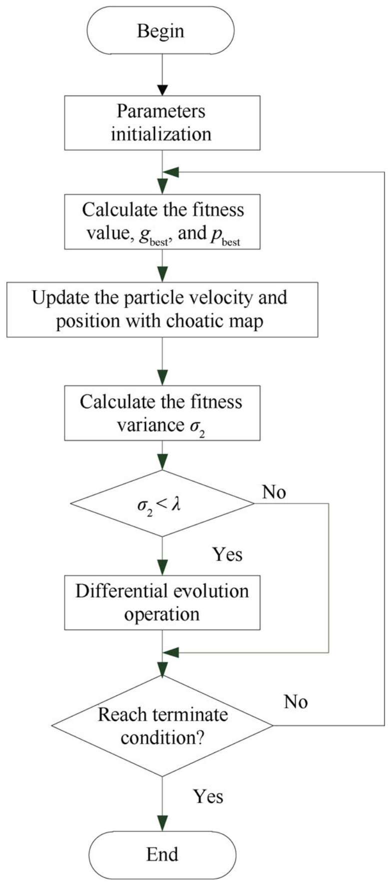


Fig. 2 The optimization process of HCPSODE

3.1 Parameter settings and performance metrics

As for the CEC2005 real-parameter optimization problems, simulation was conducted on HCPSODE algorithm and compared with PSO-DV, CDEPSO[19], DEPSO-EPV[20] and DEPSO[21], the parameters set up for the involved algorithms are as follows: for the HCPSODE, the inertia weight ω_{start} and ω_{end} in (7) are set to 0.9 and 0.4 respectively, the parameter a in (8) is set at 5.56,

$x_0 = 0.9$, and the acceleration coefficients are set to be $c_1 = c_2 = 1.494\,45$, the pre-specified stagnation threshold λ is set to 0.05. The crossover factor $CR = 0.9$ and the scale factor $F = 0.8$. The population size NP is set two times of decision variables, D. For the DEPSO, $\omega = 0.4$, $CR = 0.1$, $c_1 = c_2 = 2$, the population size is five times of the number of decision variables. For CDEPSO, acceleration coefficients c_1 and c_2 are set at 1.496 2, the inertia factor is 0.729 8, the population size is taken six times of the number of decision variables, both scaling factor F and crossover constant CR are set at 0.5, and for PSO-DV, $F = 0.8$ and $CR = 0.9$ are same as [9], The population size was five times of decision variables. For DEPSO-EPV, $F = 0.5$ and $CR = 0.9$, population size: $NP = D$, PSO topology: ring with neighborhood radius $n_r = 2$, $c_1 = c_2 = 2.05$.

The number of decision variables, D, was set to 10, 30 and 50 for all the test functions. To reduce the random discrepancy and make a fair comparison between different algorithms, 25 independent runs of all algorithms were executed, and the stopping condition of each run is based on the maximum number of function evaluations (FEs) that was set to $D \times 10\,000$.

The experiments were carried out on a PC with AMD Athlon (tm) II X2 250 processor, 3.00 GHz and 3.25 GB memory, and windows XP3 operating system.

In the experimental study, the solution error value, defined as $f(x) - f(x^*)$ was recorded, the mean and standard deviation of the solution error value were used to evaluate the performance of the hybrid PSO and DE variants, where x is the global optimum of the benchmark function and x^* is the best solution found by the algorithm after $D \times 10\,000$ function evaluations.

To get statistically sound conclusions, the Wilcoxon rand sum test was used to test whether the difference between different algorithms results was statistically significant. The test was conducted at 0.05 significance level. The Wilcoxon test results (h) is summarized to indicate the number of functions in which HCPSODE performs significantly better than (denoted by $+$), almost the same as (denoted by $\approx$), and significantly worse than (denoted by$-$) the other four involved algorithm, respectively.

As for the CEC 2011 real-world optimization problems, experiments were conducted between HCPSODE and the top three methods, including GA-MPC[22], DE-ACr[23] and SAMODE[24], which were proposed in the CEC 2011 competition. A different stagnation detection strategy was used to test the CEC2011 instances. During the implementation, if there is no improvement on the g_{best} after several iterations, that means the swarm is trapped into local optima and stagnation is detected.

3.2 Experimental results and discussions

Tables 1 shows the results of the mean, standard deviation of the function error and the Wilcoxon rank sum test achieved by the five algorithms for all the selected CEC2005 test problems. The best results among those obtained by all algorithms are marked in bold. The Wilcoxon test results is to denote that HCPSODE performs significantly better than (+), almost the same as ($\approx$), and significantly worse ($-$) than its peer algorithm respectively. Figs. 3–14 show the convergence curve of the average function error for the benchmark functions with 30 dimensions.

From Table 1 as well as the convergence curve figures, it is clear that the HCPSODE is ranked first among the five

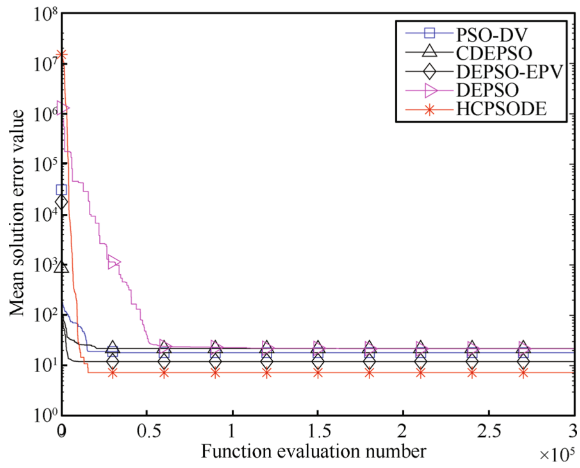


Fig. 3 The convergence curve of F_6

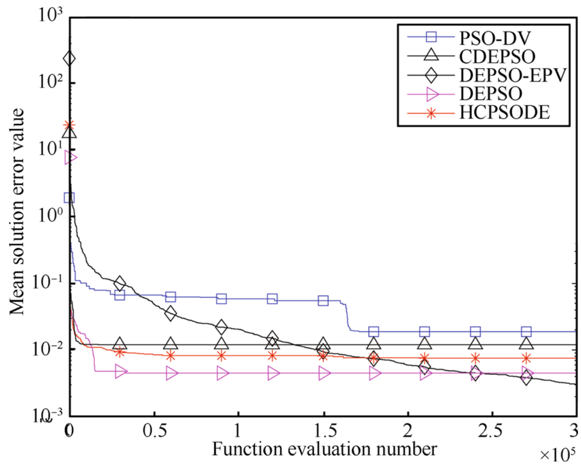


Fig. 4 The convergence curve of F_7

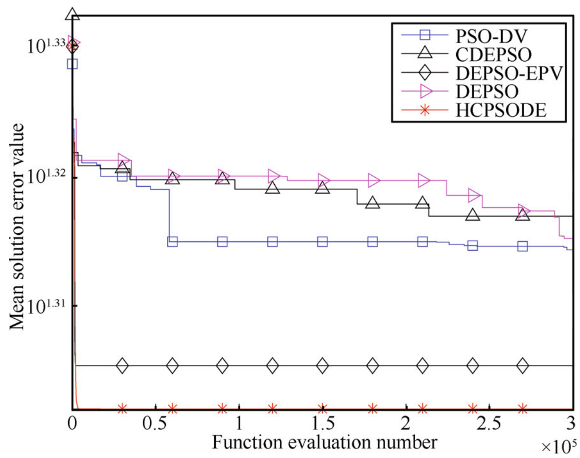


Fig. 5 The convergence curve of F_8

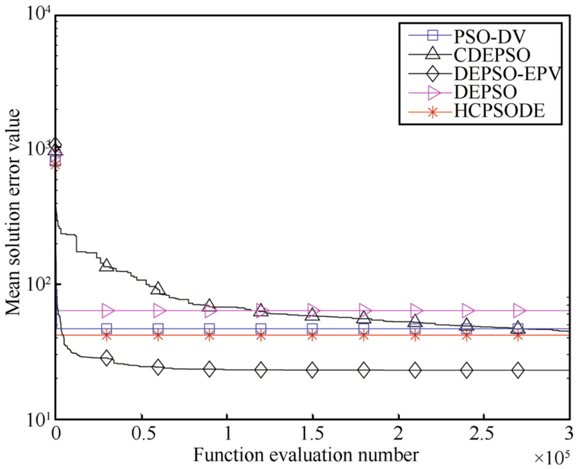


Fig. 6 The convergence curve of F_9

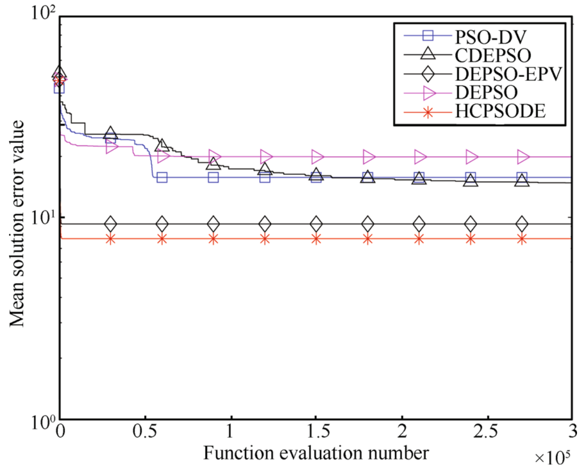


Fig. 7 The convergence curve of F_{10}

Table 1 Search results comparisons among different algorithms

F_i	D	PSO-DV mean±standard deviation	CDEPSO mean±standard deviation	DEPSO-EPV mean±standard deviation	DEPSO mean±standard deviation	HCPSODE mean±standard deviation
	10	1.05E+01±5.42E−01+	1.95E+01±4.22E−01+	1.05E+01±4.69E−01+	3.04E+01±2.21E+01+	**6.69E+00±3.78E−01**
F_6	30	1.80E+01±6.32E−01+	2.18E+01±5.21E−01+	1.20E+01±5.10E−01+	3.28E+01±2.56E+01+	**7.27E+00±4.10E−01**
	50	2.12E+01±4.88E−01+	3.14E+01±5.40E−01+	1.98E+01±4.76E−01+	5.21E+01±3.58E+01+	**9.08E+00±4.26E−01**
	10	7.58E−11±7.21E−12+	3.30E−12±1.01E−12+	5.45E−10±8.12E−09+	**3.13E−14±8.99E−13**−	6.25E−13±2.89E−12
F_7	30	4.23E−02±4.69E−02+	1.25E−02±5.41E−02+	8.69E−02±1.39E+00+	**5.20E−03±1.86E−02**−	8.96E−03±6.12E−02
	50	1.50E+06±2.47E+06+	9.01E+07±3.29E+08+	8.36E+06±1.13E+06+	**5.35E+05±4.27E+05**−	9.54E+05±7.01E+05
	10	2.03E+01±7.83E−02≈	2.03E+01±8.08E−02≈	2.01E+01± 1.05E−01≈	2.03E−01±6.31E−02≈	**2.01E+01±1.21E−01**
F_8	30	2.08E+01±5.88E−02≈	2.09E+00±4.70E−02≈	2.01E+01±1.39 E−01≈	2.09E+01±5.77E−02≈	**2.01E+01± 1.32E−01**
	50	2.11E+01±3.29E−02≈	2.03E+01±7.75E−02≈	2.00E+01±7.92 E−02≈	2.11E+01±3.46E−02≈	**2.00E+01±6.17E−02**
	10	6.60E+00±2.66E+00≈	9.92E+00±2.50E+00≈	**5.01E+00±1.70E+00**−	6.92E+00±3.23 E+00≈	7.52E+00±3.23E+00
F_9	30	7.75E+01±8.06E+01≈	8.02E+02±1.55E+01+	**4.76E+01±8.54E+00**−	8.55E+01±8.19E+01≈	6.19E+01±6.15 E+01
	50	**1.25E+02±1.91E+01**−	2.56E+03±2.89E+02≈	5.51E+02±1.86E+001≈	7.78E+02±1.35E+02≈	7.54E+02±1.97E+02
	10	4.91E+00±1.20E+00≈	4.97E+00±4.81E−01≈	6.51E−01±9.36E−01≈	4.73E+00±6.63 E−01≈	**5.27E−01±7.21E−01**
F_{10}	30	2.02E+01±4.92E+00≈	4.51E+01±1.54E+00≈	1.12E+01±5.97E+00≈	6.54E+01±1.74E+00≈	**1.02E+01±3.35E+00**
	50	5.45E+01±2.03E+00≈	5.07E+01±2.54E+00≈	2.88E+01±5.25E+00≈	5.35E+01±1.45E+00≈	**2.84E+01±2.83E+00**
	10	3.32E+01±1.41E+02+	1.10E+02±7.56E+01+	5.64E+03±9.95E+03+	**4.78E+00±7.67E+00**−	2.30E+01±3.81E+01
F_{11}	30	2.41E+03±3.42E+03+	1.55E+04±6.34E+03+	9.68E+03±1.51E+04+	2.07E+04±3.05E+03+	**6.53E+02±4.27E+03**
	50	4.65E+04±1.53E+04+	6.66E+04±1.71E+04+	6.74E+04±1.09E+04+	8.57E+04±1.04E+04+	**1.72E+04±1.06E+04**
	10	4.49E−01±7.12E−02≈	2.77E−01±1.00E−01≈	3.06 E−01±4.97 E−02≈	5.97E−01±9.70E−02≈	**2.86E−01±4.71E−02**
F_{12}	30	2.06E+00±2.01E−01+	1.53E+00±3.26E−01+	1.46E+00±1.25E−01+	3.64E+00±6.07E−01+	**7.15E−01±6.37E−02**
	50	3.96E+00±3.74E−01+	3.09E+00±5.85E−01≈	3.15E+00±1.32E−01≈	1.22E+01±7.28E−01+	**2.97E+00±2.10E−01**
	10	2.48E+00±4.97E−01+	2.55E+00±6.58E−01+	3.21E+00±2.54E−01+	2.00E+00±8.25E−01+	**1.05E+01±5.40E−02**
F_{13}	30	1.12E+01±5.78E−01+	1.08E+01±3.58E−01+	1.20E+01±2.12E−01+	1.38E+01±5.41E−01+	**5.98E+00±5.47E−02**
	50	3.54E+01±5.23E−01+	3.87E+01±2.89E−01+	3.23E+01±3.69E−01≈	3.89E+01±7.32E−01+	**3.21E+01±3.67E−02**
	10	7.11E+00±2.58E−01+	6.57E+00±2.54E−01+	4.52E+00±3.42E−01+	5.09E+00±2.99E−01+	**2.10E+00±2.51E−01**
F_{14}	30	3.20E+01±2.14E−01+	2.56E+01±6.24E−01+	1.18E+01±5.02E−01+	1.65E+01±8.95E−01+	**9.98E+00±4.23E−01**
	50	6.24E+01±5.38E−01+	5.24E+01±9.10E−01+	4.51E+01±2.35E−01+	4.27E+01±1.20E+00+	**2.32E+01±4.23E−01**
	10	2.10E+02±2.00E+01+	2.98E+02±2.02E+01+	2.58E+02±4.00E+01+	3.68E+02±3.14E+01+	**2.02E+02±1.10E+01**
F_{15}	30	2.82E+02±2.14E+01+	3.39E+02±2.24E+01+	3.01E+02±4.02E+01+	4.12E+02±3.95E+01+	**2.63E+02±1.23E+01**
	50	3.25E+02±2.41E+01+	3.87E+02±2.14E+01+	3.67E+02±4.11E+01+	4.58E+02±3.06E+01+	**2.88E+02±1.29E+01**
	10	2.85E+02±7.01E+01+	2.21E+02±3.02E+01+	1.28E+02±2.13E+01≈	1.52E+02±2.44E+01+	**1.24E+02±1.54E+01**
F_{16}	30	3.20E+02±8.14E+01+	2.41E+02±3.24E+01+	1.65E+02±2.02E+01≈	1.79E+02±2.95E+01+	**1.64E+02±1.23E+01**
	50	4.02E+02±6.33E+01+	3.01E+02±3.51E+01+	2.12E+02±2.23E+01≈	2.24E+02±2.20E+01+	**2.11E+02±1.63E+01**
	10	1.89E+02±3.02E+01+	2.11E+02±5.89E+01+	**9.21E+01±2.47E+01**−	8.96E+02±6.87E+01+	1.65E+02±4.54E+01
F_{17}	30	2.20E+02±3.14E+01+	2.56E+02±6.24E+01+	**1.02E+02±2.36E+01**−	1.01E+03±8.95E+01+	1.80E+02±4.23E+01
	50	2.88E+02±3.45E+01+	2.98E+02±6.54E+01+	**1.85E+02±2.46E+01**−	1.77E+03±8.12E+01+	1.96E+02±4.86E+01
	+	26	26	18	22	
h	−	1	0	5	4	
	≈	9	10	13	10	

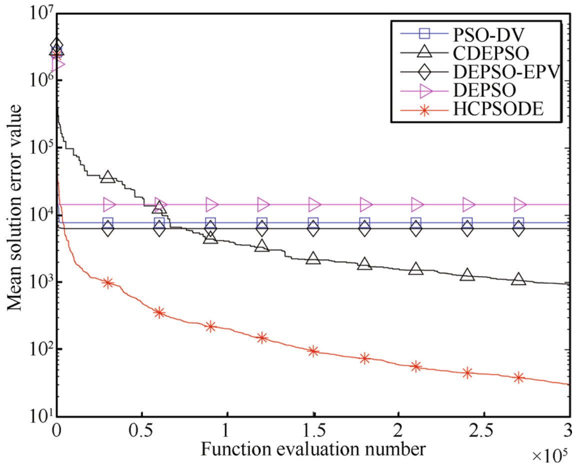


Fig. 8 The convergence curve of F_{11}

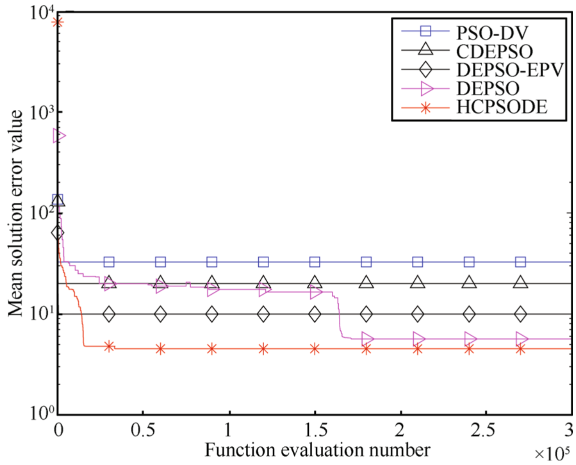


Fig. 11 The convergence curve of F_{14}

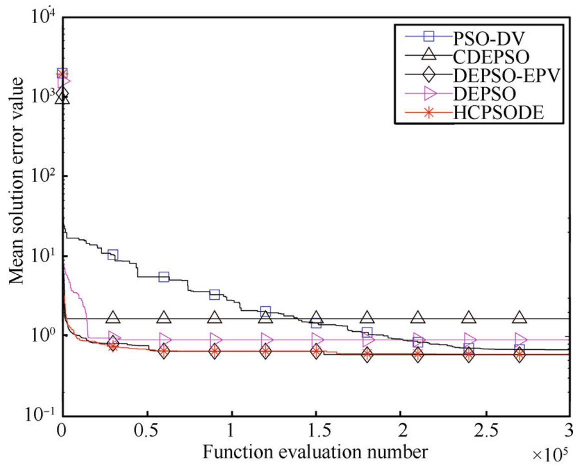


Fig. 9 The convergence curve of F_{12}

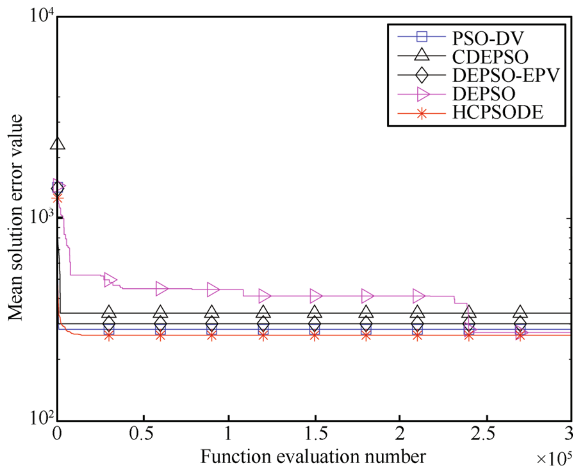


Fig. 12 The convergence curve of F_{15}

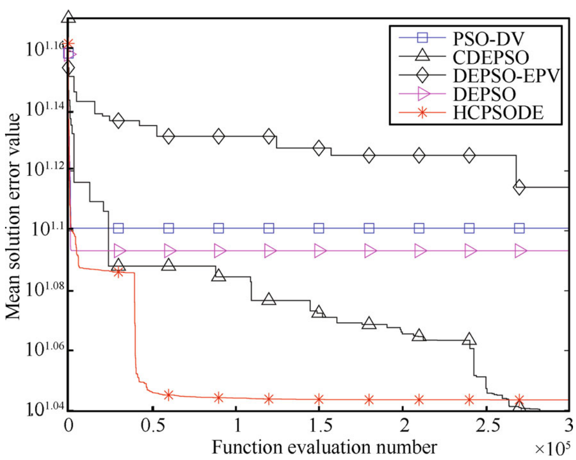


Fig. 10 The convergence curve of F_{13}

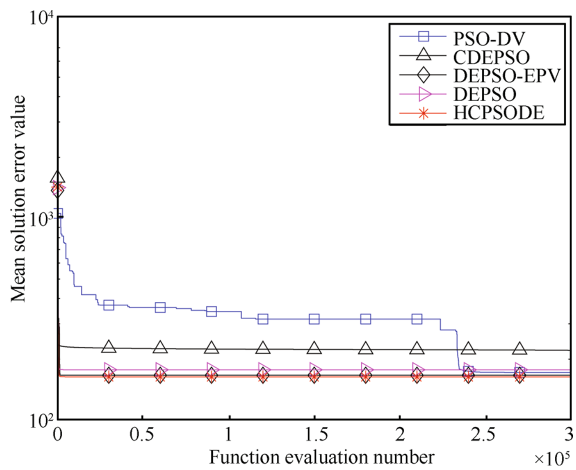


Fig. 13 The convergence curve of F_{16}

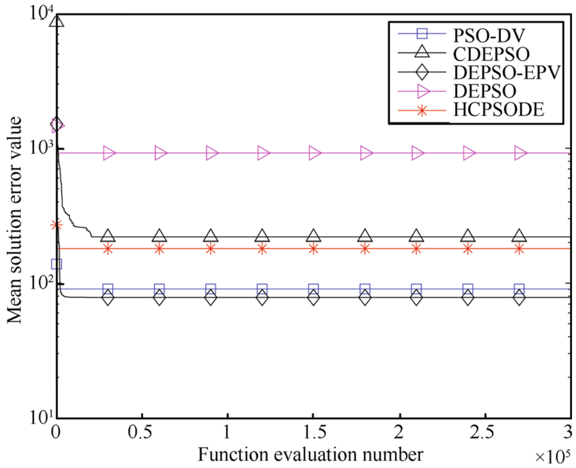


Fig. 14 The convergence curve of F_{17}

methods on solving eight functions ($F_6, F_8, F_{10}, F_{12}, F_{13}, F_{14}, F_{15}, F_{16}$), DEPSO-EPV outperforms the HCPSODE on functions F_9 and F_{17}, and DEPSO performs better than others algorithms on F_7. Among the five algorithms, CDEPSO cannot be significantly better than HCPSODE on any test function. The outstanding performance of HCPSODE is due to the chaos map with greater Lyapunov index used in HCPSODE which can help particle jump out of local optimum, when the swarm is on stagnation, hence DE operator can enhance the diversity of the swarm. The reason for DEPSO-EPV′s excellent performance on some function is due to its DE operator action on both cognitive and social experience, which makes the algorithm have better balance on exploitation and exploration.

HCPSODE is significantly better than PSO-DV, CDEPSO, DEPSO-EPV and DEPSO on 8, 8, 6, and 7 test functions, respectively. This may be because HCPSODE can improve the global search ability by detecting the stagnation, and balance exploration and exploitation ability by adjustment of inertia weight adaptively.

As to Wilcoxon rank sum test, the HCPSODE significantly outperforms its peers with 26, 26, 18 and 22 out of 12 test instances on 10, 30 and 50 dimensions respectively.

Table 2 shows the best, worst, mean and standard deviation (Std.Dev) values in the CEC2011 test instances for the HCPSODE, GA-MPC, DE-ACr, SAMODE. Except for HCPSODE, all the values are obtained from the corresponding literature.

From Table 2, we can see that the HCPSODE obtained the best values in 8 of the 10 problems ($T01$, $T02$, $0T3$, $T04$, $T05$, $T07$, $T10$ and $T13$), while the GA-MPC, DE-ACr, SAMODE obtained the best values in 6 ($T01$, $T02$, $T03$, $T04$, $T07$, and $T12$), 1 ($T06$), 5 ($T01$, $T02$, $T03$, $T04$, and $T07$), and 5 problems ($T01$, $T02$, $T03$, $T04$, and $T06$), respectively.

3.3 Diversity analysis

Swarm diversity can be used to monitor the degree of convergence or divergence and is closely linked to the exploration-exploitation tradeoff. The diversity measure used in this research is the average distance around the swarm center which is defined as

$$\operatorname{div}(S) = \frac{1}{|S|}\sum_{i=1}^{S}\sqrt{\sum_{j=1}^{D}(X_{ij}-\overline{X}_j)^2} \tag{11}$$

where S denotes the swarm, $|S|$ is the population size, D is the dimensionality of the optimization problem, X_{ij} is the value of the j-th dimension of the i-th particle, and $\overline{X}_j$ is the average value for dimension j over all particles. Fig.15 illustrates the swarm diversity of the basic PSO and HCPSODE algorithm on solving F_6 with 30 dimensions.

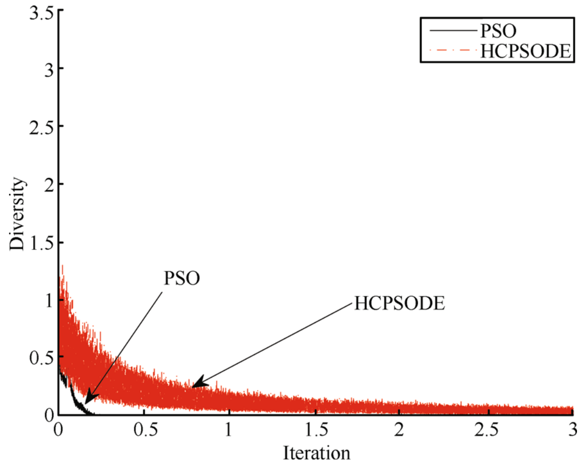


Fig. 15 Diversity curve F_6

It is clearly shown that the diversity of basic PSO decreases dramatically, and the diversity of HCPSODE decreases gradually which indicates that the HCPSODE can maintain the diversity effectively and keep good balance between exploration and exploitation.

4 Application of HCPSODE to ADRC parameter optimization

In this section, the proposed HCPSODE algorithm is applied to parameter optimization of ADRC.

PID controller has been widely used in industrial control systems due to the simple structure and implementation simplicity. As an error-based feedback controller, the control law is produced by linear combination of the error between the set point and plant as well as its differentiation and integration.

$$u(t) = K_p e(t) + K_i\int_0^t e(\tau)\,\mathrm{d}\tau + K_d\frac{\mathrm{d}}{\mathrm{d}t}e(t). \tag{12}$$

However, when the parameters of the control system change in larger range or the control system is nonlinear, or when the reference input signal is not differentiable or is non-smooth, it is difficult to obtain ideal differential signal, and the performance of PID control will degrade greatly, to solve this problem exists in PID controller, a new nonlinear controller named ADRC was proposed by Han and

Wang[25, 26]. ADRC is a nonlinear controller, which is generally used for controlling a class of nonlinear uncertain systems and systems with large time-delay. ADRC combines modern control theory with signal processing techniques, and inherits the essence of PID controller. ADRC does not depend on the model of the control and does not need to measure the perturbation of the system. It is easy to implement decoupling control which has shown broad application prospects[27−29]. However, there are several key parameters in ADRC which need to be tuned before using it, and the tuning process is heavily depends on the experience and is time-consuming and tedious. The tuning of ADRC parameter has become a hot topic. The typical second-order ADRC is schematically shown in Fig. 16, where $w(t)$ is the unknown disturbance. Note that ADRC is composed of three parts: the tracking differentiator (TD), the expansion of the state observer (ESO), and the nonlinear state error feedback (NLSEF).

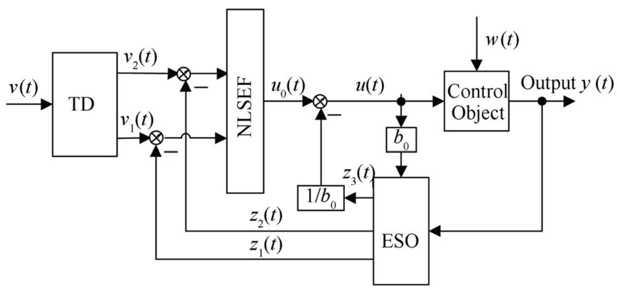


Fig. 16 The structure of ADRC

Table 2 Performance values achieved by HCPSODE and others three algorithms within 1.5×10^5 FEs

Function		GA-MPC	DE-ACr	SAMODE	HCPSODE
$T01$	Best	**0.000 000E+00**	7.209 3E−15	**0.000 000E+00**	**0.000 000E+00**
	Worst	0.000 000E+00	1.175 7E+01	1.094 227 7E+01	0.000 000E+00
	Mean	**0.000 000E+00**	8.7697E−01	1.212 025 6E+00	**0.000 000E+00**
	Std.Dev	0.000 000E+00	3.043 9E+00	3.376 217 1E+00	0.000 000E+00
$T02$	Best	**−2.842 253E+01**	**−2.842 3E+01**	**−2.842 253E+01**	**−2.842 253E+01**
	Worst	−2.711 301E+01	−2.644 37E+01	−2.610 048E+01	−2.761 562E+01
	Mean	−2.770 069E+01	−2.773 1E+01	−2.706 978E+01	**−2.795 874E+01**
	Std.Dev	4.673 052E−01	4.903 5E−01	6.624 811E−01	3.872 351E−01
$T03$	Best	**1.151 489E−05**	**1.151 5E−05**	**1.151 489E−05**	**1.151 489E−05**
	Worst	1.151 489E−05	1.151 5E−05	1.151 489E−05E	1.151 489E−05
	Mean	**1.151 489E−05**	**1.151 5E−05**	**1.151 489E−05**	**1.151 489E−05**
	Std.Dev	0.000 000E+00	0.00E+00	6.1087E−15	0.000 000E+00
$T04$	Best	**1.377 076 2E+01**	1.377 2E+01	**1.377 076 2E+01**	**1.377 076 2E+01**
	Worst	1.432 911 3E+01	2.1002E+01	1.432 911 3E+01	1.586 324E+01
	Mean	**1.381 543 0E+01**	1.733 9E+01	1.394 046 0E+01	1.412 474E+01
	Std.Dev	1.546 004 5E−01	2.976 1E+00	2.502 223 1E−01	3.685 412E−01
$T05$	Best	−3.684 537E+01	−3.684 5E+01	−3.684 393E+01	**−3.702 594E+01**
	Worst	−3.410 760E+01	−3.148 4E+01	−3.049 253E+01	−3.353 489E+01
	Mean	−3.503 883E+01	−3.472 0E+01	−3.359 474E+01	**−3.638 954E+01**
	Std.Dev	8.329 248E−01	1.446 9E+00	1.575 135E+00	1.102 657E+00
$T06$	Best	−2.906 612E+01	**−3.684 5E+01**	−2.916 612E+01	−2.918 123E+01
	Worst	−2.125 851E+01	−3.416 5E+01	−2.300 593E+01	−2.651 254E+01
	Mean	−2.748 811E+01	**−3.503 3E+01**	−2.763 470E+01	−2.824 871E+01
	Std.Dev	1.782 137E+00	1.028 7E+00	1.923 527E+00	4.894 321E−01
$T07$	Best	**5.000 000E−01**	6.659 1E−01	**5.000 000E−01**	**5.000 000E−01**
	Worst	9.334 272E−01	1.036 1E+00	9.943 334E−01	9.852 141E−01
	Mean	7.484 090E−01	8.847 7E−01	8.166 238E−01	**7.248 974E−01**
	Std.Dev	1.249 139E−01	1.057 1E−01	1.193 672E−01	1.354 785E−01
$T10$	Best	−2.184 253 9E+01	−2.160 1E+01	−2.182 166 5E+01	**−2.185 212 6E+01**
	Worst	−2.147 568 4E+01	−1.094 0E+01	−2.141 583 7E+01	−2.156 287 4E+01
	Mean	−2.170 224 9E+01	−1.675 6E+01	−2.165 890 6E+01	**−2.181 336 8E+01**
	Std.Dev	1.163 465 9E−01	4.043 7E+00	1.129 576 9E−01	1.084 596 2E−01
$T12$	Best	**7.095 559 5E+00**	1.181 4E+01	6.943 215 0E+00	6.960 568 2E+01
	Worst	1.692 489 3E+01	1.798 1E+01	1.561 880 0E+01	1.665 874 1E+01
	Mean	1.281 816 5E+01	1.536 0E+01	**1.106 747 1E+01**	1.1857452E+01
	Std.Dev	3.241 342 8E+00	1.213 3E+00	2.652 277 9E+00	3.054 521 0E+00
$T13$	Best	8.398 688E+00	8.962 4E+00	8.610 634E+00	**8.662 569E+00**
	Worst	1.081 018E+01	2.105 2E+01	1.662 200E+01	9.678 521E+00
	Mean	9.359 342E+00	1.490 9E+01	1.099 524E+01	**8.785 412E+00**
	Std.Dev	9.454 327E−01	2.763 4E+00	2.388 975E+00	5.478 521E−01

4.1 TD

TD is a dynamic component. From its input signal $v(t)$, two output signals can be obtained, $v_1(t)$ and $v_2(t)$ are the tracking signal and the differentiated signal of the input. The discrete expression of output signals are as follows:

$$v_1(k+1) = v_1(k) + Tv_2(k) \tag{13}$$

$$v_2(k+1) = v_2(k) + Tfhan(v_1(k), v_2(k), r, h_0) \tag{14}$$

where T is the sample time, r is the parameter to determine the tracking speed, h_0 is filter factor, $fhan$ is a nonlinear function and can be defined as

$$fhan = \begin{cases} -\dfrac{ra}{d}, & |a| \le d \\ -r\mathrm{sgn}(a), & a > d \end{cases} \tag{15}$$

where

$$\begin{aligned} &d = rh_0, d_0 = dh_0, y = v_1 - v_0 + h_0 v_2 \\ &a_0 = \sqrt{d^2 + 8r|y|} \\ &a = \begin{cases} v_2 + \dfrac{y}{h_0}, & |y| \le d_0 \\ v_2 + \mathrm{sgn}(y)\dfrac{(a_0 - d)}{2}, & |y| > d_0. \end{cases} \end{aligned} \tag{16}$$

The expression sgn() represents the signum function.

4.2 ESO

ESO is the core part integrated with the ADRC controller. It adopts a nonlinear structure to estimate the state of the system, the model uncertainty and the external disturbance.

$$\begin{aligned} &e(k+1) = z_1(k) - y(k+1) \\ &z_1(k+1) = z_2(k) + T[z_2(k) - \beta_1 e(k)] \\ &z_2(k+1) = z_2(k) + T[z_3(k) - \beta_2 fal(e, 0.5, \delta) + b_0 u(k)] \\ &z_3(k+1) = z_3(k) - T\beta_3 fal(e, 0.25, \delta). \end{aligned} \tag{17}$$

In (17), parameters, β_1, β_2 and β_3, are need to be tuned. The nonlinear function $fal()$ can be defined as

$$fal(e, \alpha, \delta) = \begin{cases} \dfrac{e}{\delta^{1-\alpha}}, & |e| \le \delta \\ |e|^{\alpha}\mathrm{sgn}(e), & |e| \ge \delta \end{cases} \tag{18}$$

where $0 < \alpha < 1$, $\delta < 0$.

4.3 NLSEF

NLSEF converts the linear combination of traditional PID to the nonlinear combination, and obtains a nonlinear PID controller to improve control performance. The formula of the calculation can be regarded as a nonlinear PD controller.

$$\begin{aligned} &e_1(k+1) = v_1(k+1) - z_1(k+1) \\ &e_2(k+1) = v_2(k+1) - z_2(k+1) \\ &u_0(k+1) = \lambda_1 fal(e_1(k+1), \alpha_1, \delta) + \\ &\qquad \lambda_2 fal(e_2(k+1), \alpha_2, \delta) \\ &u_1(k+1) = u_0(k+1) - \frac{z_3(k+1)}{b_0} \end{aligned} \tag{19}$$

where λ_1, λ_2, α_1, α_2 and b_0 are adjustable parameters. ADRC does not rely on the accurate mathematical model of the control system, and can achieve high control performance only need the information of the input, system output and the controller output.

As discussed above, there are many parameters in ADRC need to be tuned, including r, h_0 in TD, β_1, β_2, β_3 in ESO, λ_1, λ_2, α_1, α_2 and b_0 in NLSEF. The tuning of ADRC parameters has some rules to follow. The parameters of TD can be tuned alone because it is independent of ESO and NLSEF. In ESO, β_1, β_2 are the estimations of the object state variables, β_3 is the estimation of the total system disturbances which are compensated by NLSEF automatically. Power parameter $0 \le \alpha_i \le 1, (i = 1, 2)$ is usually set as $\alpha_1 = 1$, $\alpha_2 = 0.5$. The estimation ability of ESO is determined by β_1, β_2, β_3. λ_1, λ_2 and b_0 are the key parameters to determine the controller performance. The proposed algorithm is employed to optimize the parameters of ADRC.

4.4 Fitness function and penalty strategy

The fitness function can affect the quality of optimal design schemes and the optimization process. The functional integral of instantaneous error, such as integral of error (IE), integral of squared error (ISE), integral time square error (ITSE), integral of absolute value of error (IAE), and integral time absolute error (ITAE) are generally used as objective function to evaluate the control system performance. Aforementioned performance criteria have their own advantages and disadvantages. Optimized control parameters can yield an excellent response that will minimize the performance criteria including the overshoot, rise time, settling time, and steady-state error. To make the system output and the actuator movement more stable, a new fitness function combining ITSE with overshoot and steady state error is defined as follows:

$$J = \int_0^{\infty} w_1 |e(t)|^2 t\mathrm{d}t + w_2 u^2(t) + w_3(M_p + e_{ss}) \tag{20}$$

where $e(t)$ is the instantaneous error between reference input and system output, w_1, w_2, w_3 are the adjustable weight factors, $u(t)$ is output of controller, M_p and e_{ss} is the overshoot and the steady state error respectively.

The structure of the HCPSODE based ADRC controller is illustrated in Fig. 17.

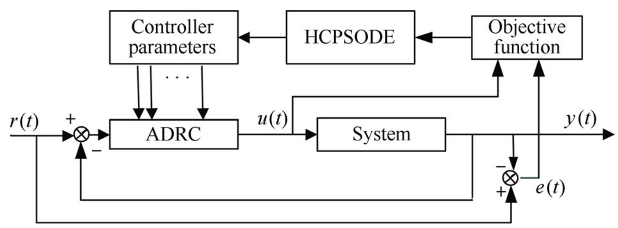


Fig. 17 The structure of HCPSODE-ADRC

4.5 Simulation study

Two examples are used in the simulation to demonstrate the effectiveness of the optimized ADRC. For comparison purpose, the HCPSODE, JADE[30], CMA-ES[31] and APSO[32] are applied to these examples.

Example 1. Given a system with the following difference equation:

$$y\,(k+1)=\sin[y(k)]+u(k)(5+\cos[y(k)u(k)]). \tag{21}$$

Example 2. A second-order system with time delay is described in the following difference equation:

$$\begin{aligned} y\,(k+2) = {} & 0.2\sin(0.5(y(k)+y(k-1)))+\\ & 0.2\sin(0.5(y(k)+y(k-1))+2u\,(k)+\\ & u\,(k-1)+\frac{4u(k)+u(k-1)}{1+0.2\cos(0.2(2y(k)+y(k-1)))}. \end{aligned} \tag{22}$$

The reference signal to be tracked by system (21) and (22) is step change.

The tracking performance of ADRC for (21) and (22) optimized by the four methods is presented in Figs. 18 and 19, respectively. For the first example, it is clearly shown that the tracking performance of ADRC optimized by HCPSODE is obviously better than APSO and CMA-ES. It took about 20, 22, 45, 47 generations for HCPSODE, JADE, APSO and CMA-ES to reach steady state respectively, and among the four optimized ADRC, there is only a slight overshoot existing in APSO. As for the second example, there exist slight overshoot and small oscillation in APSO optimized ADRC, and the HCPSODE optimized ADRC took least generations to reach steady state followed by CMA-ES, JADE, and APSO.

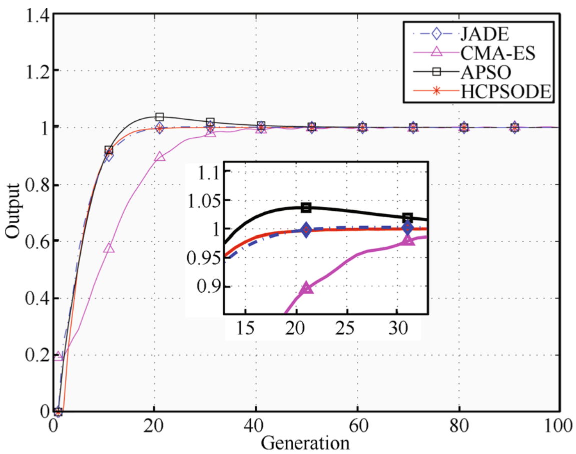


Fig. 18 Tracking performance of Example 1

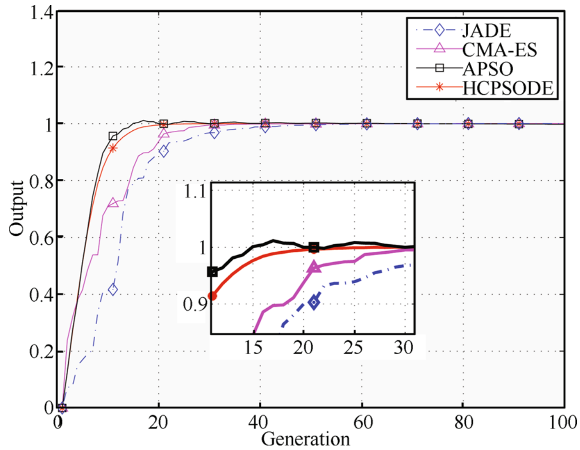


Fig. 19 Tracking performance of Example 2

5 Conclusions and future work

In this paper, a hybrid algorithm, incorporating PSO, DE and chaotic map, to solve numerical function and ADRC controller optimization is proposed. A novel nonlinear strategy for decreasing inertia weights is adopted to balance the abilities of exploration and exploitation of the proposed algorithm. To maintain the diversity of particles in the late evolution period and avoid the prematurity, a DE operator is used to help particles jump out of stagnation. The proposed algorithm utilizes a chaotic map to improve global convergence and escape from local optimum. The results obtained from twelve benchmark functions demonstrate the superiority of the proposed algorithm to five others evolution algorithms in terms of solution accuracy and convergence speed. In addition, the proposed algorithm is applied to solve the parameter optimization problem of ADRC. The evaluation results indicate its potential effectiveness to control complex discrete-time nonlinear systems with time delays.

In the field of optimization, hybridization is a direction worthy of further study as it is one of the most efficient strategies to improve the performance of many optimizers. Our future work will focus on several issues: the first is understanding how different hybrid methods improve the performance of DE and PSO. The second is expanding our method to solve more real-world problems; last but not the least, we will apply the proposed optimized ADRC controller to replace PID controller to drive permanent magnet synchronous motor (PMSM).

References

[1] R. C. Eberhart, J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, IEEE, Nagoya, Japan, pp. 39–43, 1995.

[2] J. Kennedy, R. C. Eberhart. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Network*, IEEE, Perth, Australia, pp. 1942–1948, 1995.

[3] D. M. Wonohadidjojo, G. Kothapalli, M. Y. Hassan. Position control of electro-hydraulic actuator system using fuzzy logic controller optimized by particle swarm optimization. *International Journal of Automation and Computing*, vol. 10, no. 3, pp. 181–193, 2013.

[4] K. Ishaque, Z. Salam, M. Amjad, S. Mekhilef. An improved particle swarm optimization (PSO) - Based MPPT for PV with reduced steady-state oscillation. *IEEE Transactions on Power Electronics*, vol. 27, no. 8, pp. 3627–3638, 2012.

[5] N. Talbi, K. Belarbi. Fuzzy Takagi Sugeno system optimization using hybrid particle swarm optimization and Tabu search learning algorithm. *International Journal of Tomography and Simulation*, vol. 22, no. 1, pp. 4–16, 2013.

[6] B. Xin, J. Chen, J. Zhang, H. Fang, Z. H. Peng. Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: A review and taxonomy. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 42, no. 5, pp. 744–767, 2012.

[7] R. Storn, K. Price. Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[8] V. Ramesh, T. Jayabarathi, S. Asthana, S. Mital, S. Basu. Combined hybrid differential particle swarm optimization approach for economic dispatch problems. *Electric Power Components and Systems*, vol. 38, no. 5, pp. 545–557, 2010.

[9] S. Das, A. Konar, U. K. Chakraborty. Improving particle swarm optimization with differentially perturbed velocity. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, ACM, New York, USA, pp. 177–184, 2005.

[10] P. Kim, J. Lee. An integrated method of particle swarm optimization and differential evolution. *Journal of Mechanical Science and Technology*, vol. 23, no. 2, pp. 426–434, 2009.

[11] H. M. Elragal, M. A. Mangoud, M. T. Alsharaa. Hybrid differential evolution and enhanced particle swarm optimisation technique for design of reconfigurable phased antenna arrays. *IET Microwaves, Antennas and Propagation*, vol. 5, no. 11, pp. 1280–1287, 2011.

[12] D. Levy. Chaos theory and strategy: Theory, application, and managerial implications. *Strategic Management Journal*, vol. 15, no. S2, pp. 167–178, 1994.

[13] L. dos Santos Coelho, B. M. Herrera. Fuzzy identification based on a chaotic particle swarm optimization approach applied to a nonlinear yo-yo motion system. *IEEE Transactions on Industrial Electronics*, vol. 54, no. 6, pp. 3234–3245, 2007.

[14] M. Clerc, J. Kennedy. The particle swarm: Explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.

[15] I. C. Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters*, vol. 85, no. 6, pp. 317–325, 2003.

[16] C. He, D. He, L. G. Jiang, H. W. Zhu, G. R. Hu. A chaotic map with infinite collapses. In *Proceedings of TENCON 2000*, IEEE, Kuala Lumpur, Malaysia, pp. 95–99, 2000.

[17] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, S. Tiwari. Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-parameter Optimization, Technical Report, Nanyang Technological University, Singapore, 2005.

[18] S. Das, P. N. Suganthan. Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems, Technical Report, Jadavpur University, Nanyang Technological University, Kolkata, 2010.

[19] Y. Tan, G. Z. Tan, S. G. Deng. Hybrid particle swarm optimization with differential evolution and chaotic local search to solve reliability-redundancy allocation problems. *Journal of Central South University*, vol. 20, no. 6, pp. 1572–1581, 2013.

[20] M. G. Epitropakis, V. P. Plagianakos, M. N. Vrahatis. Evolving cognitive and social experience in particle swarm optimization through differential evolution. In *Proceedings of IEEE Congress on Evolutionary Computation*, IEEE, Barcelona, Spain, pp. 1–8, 2010.

[21] W. J. Zhang, X. F. Xie. DEPSO: Hybrid particle swarm with differential evolution operator. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, IEEE, Washington DC, USA, pp. 3816–3821, 2003.

[22] S. M. Elsayed, R. A. Sarker, D. L. Essam. GA with a new multi-parent crossover for solving IEEE-CEC 2011 competition problems. In *Proceedings of 2011 IEEE Congress on Evolutionary Computation*, IEEE, USA, pp. 1034–1040, 2011.

[23] G. Reynoso-Meza, J. Sanchis, X. Blasco, J. M. Herrero. Hybrid DE algorithm with adaptive crossover operator for solving real-world numerical optimization problems. In *Proceedings of 2011 IEEE Congress on Evolutionary Computation*, IEEE, New Orleans, USA, pp. 1551–1556, 2011.

[24] S. M. Elsayed, R. A. Sarker, D. L. Essam. Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems. In *Proceedings of 2011 IEEE Congress on Evolutionary Computation*, IEEE, New Orleans, USA, pp. 1041–1048, 2011.

[25] J. Q. Han. Auto-disturbances-rejection controller and its applications. *Control and Decision*, vol. 13, no. 1, pp. 19–23, 1998. (in Chinese)

[26] J. Q. Han, W. Wang. Nonlinear tracking-differentiator. *Journal of Systems Science and Mathematical Sciences*, vol. 14, no. 2, pp. 177–183, 1994. (in Chinese)

[27] K. Erenturk. Fractional-order $PI^{\lambda}D^{\mu}$ and active disturbance rejection control of nonlinear two-mass drive system. *IEEE Transactions on Industrial Electronics*, vol. 60, no. 9, pp. 3806–3813, 2013.

[28] M. Pizzocaro, D. Calonico, C. Calosso, C. Clivati, G. A. Costanzo, F. Levi, A. Mura. Active disturbance rejection control of temperature for ultrastable optical cavities. *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 60, no. 2, pp. 273–280, 2013.

[29] J. Q. Pu, R. Y. Yuan, X. M. Tan, J. Q. Yi. An integrated approach to hypersonic entry attitude control. *International Journal of Automation and Computing*, vol. 11, no. 1, pp. 39–50, 2014.

[30] J. Q. Zhang, A. C. Sanderson. JADE: Adaptive differential evolution with optional external archive. *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[31] N. Hansen, A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[32] Z. H. Zhan, J. Zhang, Y. Li, H. S. H. Chung. Adaptive particle swarm optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, 2009.

Guo-Han Lin received the B. Sc. degree in automation from Hunan University, Changsha, China in 1996, the M. Sc. degree in control theory and control engineering from Hunan University, China in 2005, then, he joined the School of Hunan Institute of Engineering. He is currently Ph. D. degree candidate in control science and engineering, Hunan University, China.

His research interests include evolutionary computation techniques, electric machine drives, power electronics, and intelligent control theory.

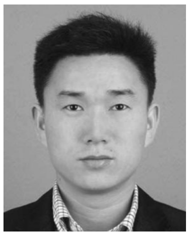
E-mail: lgh@hnu.edu.cn (Corresponding author)

ORCID iD: 0000-0001-5014-3374

Jing Zhang received his B. Sc., M. Sc. and Ph. D. degrees in control theory and control engineering from Hunan University, China in 1982, 1984 and 1997 respectively. He is currently with the College of Electrical and Information Engineering, Hunan University, China. He has published more than 100 papers in journals and conferences. He was a recipient of Second Prize for Chinese National Science and Technology Progress, China.

His research interests include optimal control, fuzzy control, and intelligent control of rotary kiln.

E-mail: zhangj@hnu.edu.cn



Zhao-Hua Liu received his M. Sc. degree in computer science and technology, and the Ph. D. degree in control science and engineering from Hunan University, China, in 2010 and 2012, respectively. Currently, he is with the School of Information and Electrical Engineering, Hunan University of Science and Technology, China.

His research interests include evolutionary computation techniques, intelligent control, computer control technology, parallel computing, and parameter identification of the permanent-magnet synchronous motor.

E-mail: zhaohualiu2009@hotmail.com