2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Optimal Approximation of Stable Linear Systems with a Novel and Efficient Optimization Algorithm

Qiaoyong Jiang, Lei Wang*, Xinhong Hei, Rong Fei, Dongdong Yang,
Feng Zou, Hongye Li, Zijian Cao, YanYan Lin
Department of Computer Science and Engineering, Xi'an University of Technology
Corresponding author*: Tel.: ++86 29 8231 2087, fax: ++86 29 8231 2216
email: jdcotor@stu.xaut.edu.cn, leiwang@xaut.edu.cn

***Abstract*—Optimal approximation of linear system models is an important task in the controller design and simulation for complex dynamic systems. In this paper, we put forward a novel nature-based meta-heuristic method, called artificial raindrop algorithm, which is inspired from the phenomenon of natural rainfall, and apply it for optimal approximation of a stable linear system. It mimics the changing process of a raindrop, including the generation of raindrop, the descent of raindrop, the collision of raindrop, the flowing of raindrop and the updating of raindrop. Five corresponding operators are designed in the algorithm. Numerical experiment is carried on the optimal approximation of a typical stable linear system in two fixed search intervals. The result demonstrates better performance of the proposed algorithm comparing with that of other five state-of-the-art optimization algorithms.**

I. INTRODUCTION

Approximation is an operation that identifies the largest value from multiple input signals. Such an operation has many applications in a variety of fields including associative memories [1], cooperative models of binocular stereo [2], Fukushima's neocognitron for feature extraction [3], and so on. As a mathematical tool, it has been widely used in the field of control engineering. For the simulation and controller design of complex dynamic systems, optimal approximation of linear systems is one of the most important tasks [1]. In the past few decades, various methods have been proposed for the model approximation problem under certain approximation error criteria, such as gradient-based search methods [1]. However, these methods often obtain a local rather than a global optimum solution. In order to get the approximation model of given linear system more efficient and effective, some intelligent algorithms have been introduced for this issue in the recent years. Representatively, the genetic algorithm (GA) [2], differential evolution algorithm (DE) [3], artificial immune algorithm (AIA) [4]–[5], have been employed in the optimal approximation of linear systems, respectively. The problem of model approximation is also addressed and solved with a memetic computing approach called 3SOME employed in the optimal approximation of linear systems [6], which is the latest progress of the problem.

According to the No Free Lunch theorem, however, there is no explicit approach to be optimized for all optimization problems. Hence, developing for new meta-heuristic methods is always an open problem. In this paper, a new optimization algorithm inspired from the phenomenon of natural rainfall, especially its primary stage, named artificial raindrop algorithm (ARA) is proposed for optimal approximation of linear system models. The algorithm is based on particle system which is a technique for modeling a class of fuzzy objects such as the smoke, water, cloud and so on [7], and the particle goes through five stages, including the produce of raindrop, the descent of raindrop, the collision of raindrop, the flowing of raindrop and the updating of raindrop. Five corresponding operators are designed to closely simulate the raindrop process of change.

The rest of this paper is organized as follows: in Section II, we first introduce the model approximation problem of linear systems. Then Section III presents the background and principles of the proposed algorithm. Section IV presents the simulation results. Section V discusses the similarities and difference between ARA and PSO, and shows how and why the proposed algorithm works well. Finally, the summary of this paper will be made in Section VI.

II. PROBLEM FORMULATION

Usually, a practical linear system can be represented a transfer function of the form in Eq.(1) [1].

$$G(s) = \frac{b_0 s^m + b_1 s^{m-1} + \cdots + b_{m-1}s + b_m}{s^n + a_1 s^{n-1} + \cdots + a_{n-1}s + a_n}. \tag{1}$$

For a high-order linear system, *m* and *n* are usually much larger. In order to simplify the system, we usually employ the second-order system with delay of the form in Eq.(2) to approximate the high-order system.

$$H(s) = \frac{k(s+b)}{s^2 + a_1 s + a_2} e^{-\tau s}. \tag{2}$$

The goal to find an optimal approximate mode *H(s)* such that the frequency-domain L2-error performance in Eq.(3) is minimized, where the frequency points, $\omega_i, i = 0, 1, 2, \cdots, N$, and the integer *N* are taken a priori.

$$J = \sum_{i=0}^{N} |G(j\omega_i) - H(j\omega_i)|. \tag{3}$$

In this case, the original system *G(s)* is asymptotically stable, the constraint, $H(0) = G(0)$, is place to make sure that the steady-state responses of the original system and approximate model are the same for the unit-step input [4]–[5].

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

III. ARTIFICIAL RAINDROP ALGORITHM

A. The idea of proposed algorithm

In this section, the idea of artificial raindrop calculation model, inspired from the phenomenon of natural rainfall, will be introduced. It is similar to other meta-heuristic that the proposed algorithm begins with an initial population called vapor, i.e. the population is composed of vapors. Then, an individual fitness value is considered as the individual altitude. In other words, the small raindrops will flow from high altitude to low altitude direction under the action of gravity. Finally, most of small raindrops will stay in the location of the lowest elevation, i.e. the location of the optimal solution. The correspondence of ARA and the changing process of a raindrop can be summarized in Table I.

TABLE I. THE CORRESPONDENCE OF ARA AND THE CHANGING PROCESS OF A RAINDROP.

ARA	The changing process of a raindrop
Search space	Natural environment
Particle	Vapor
Population size	The number of vapor
Fitness function	Altitude
Global optimal solution	Location of the lowest altitude
Raindrop formation operator	Raindrop formation process
Raindrop descent operator	Raindrop descent process
Raindrop collision operator	Raindrop collision process
Raindrop flowing operator	Raindrop flowing process
Raindrop updating operator	Raindrop updating process

1) Raindrop generation operator: As we all know, the raindrop is generated by absorbing ambient water vapor in nature. For convenience, it may be assumed that the position of the raindrop is the geometric center of ambient water vapor. From a statistical standpoint, the geometric center is a very important digital characteristic and represents the changing trend of vapor population on some level. The similar idea has been used in the estimation of distribution algorithm based on Gaussian model sampling [8]. That is why we choose the geometric center as the position of raindrop.

At each generation *t*, the raindrop generation operator φ_R^G on vapor population

$\mathbf{Pop}(t) = \{\mathbf{Vapor}_1(t), \mathbf{Vapor}_2(t), \cdots, \mathbf{Vapor}_N(t)\}$
is carried out as follows.
Define

$$\begin{aligned}\mathbf{Raindrop}(t) &= \varphi_R^G(\mathbf{Pop}(t)) \\ &= \{\mathbf{Vapor}_1(t), \mathbf{Vapor}_2(t), \cdots, \mathbf{Vapor}_N(t)\} \\ &= \{(\frac{1}{N}\sum_{i=1}^{N}\mathbf{Vapor}_{i1}(t), \cdots, \frac{1}{N}\sum_{i=1}^{N}\mathbf{Vapor}_{iN}(t)\}.\end{aligned} \quad (4)$$

where *N* is the number of population size.

2) Raindrop descent operator: According to our observation, when the effect of wind is ignored, the raindrop straight drops from the cloud to the ground by the action of gravity. From a mathematical point of view, it means that the coordinate of raindrop changes just one component. This is equivalent to a one-dimensional mutation operator or disturbance in evolutionary algorithm.

Therefore, the raindrop descent operator φ_R^D on raindrop is implemented as follows.

Define

$$\mathbf{New\_Raindrop}(t) = \varphi_R^D(\mathbf{Pop}(t)). \quad (5)$$

i.e.

$$\begin{aligned}\mathbf{New\_Raindrop}_{r_1}(t) = \mathbf{Raindrop}_{r_2}(t) + \phi \cdot (\mathbf{Raindrop}_{r_3}(t) \\ - \mathbf{Raindrop}_{r_4}(t)).\end{aligned} \quad (6)$$

where $r_1, r_2, r_3, r_4 \in \{1, 2, \cdots, D\}$ are randomly chosen indexes, *j* is the corresponding index of decision variable in **New_Raindrop**, and ϕ is a random number in the range (-1, 1).

3) Raindrop collision operator: Owing to the speed and quality, the raindrop will be split into a number of small raindrops when contacts the ground and flying in all directions. It can be assumed that the number of small raindrops is equal to the population size in order to keep the population scale stability. Define

$$\mathbf{Small\_Raindrop}(t) = \varphi_R^C(\mathbf{New\_Raindrop}(t) \cup \mathbf{Pop}(t)). \quad (7)$$

i.e.

$$\begin{aligned}\mathbf{Small\_Raindrop}_{ij}(t) = \mathbf{New\_Raindrop}_j(t) + sign(\alpha_j - 0.5)\cdot \\ \log(\beta_j) \cdot (\mathbf{New\_Raindrop}_j(t) - \mathbf{Vapor}_{kj}(t))).\end{aligned} \quad (8)$$

where i ($i = 1, 2, \cdots, N$) and j ($j = 1, 2, \cdots, D$) are the index of *i*th small raindrop and the corresponding dimension, respectively. $k \in \{1, 2, \cdots, N\}$ is randomly chosen index, α_j and β_j are two uniformly distributed random numbers in the range (0, 1) and *sign*($\cdot$) stands for sign function.

4) Raindrop flowing operator: Due to the action of gravity, the small raindrops will flow from high altitude to low altitude direction. Lastly most of small raindrops will stay in the locations with a relatively low elevation. The locations provide additional information about the promising progress direction. Denote **RP** as a raindrop pool which includes chronicles optimal solutions. In short, raindrop pool is an external archive. The raindrop flowing operator φ_R^F on small raindrops is generated as follows.

$$\mathbf{New\_Small\_Raindrop}(t) = \varphi_R^F(\mathbf{Small\_Raindrop}(t)). \quad (9)$$

i.e.

$$\begin{aligned}\mathbf{New\_Small\_Raindrop}_i(t) = \mathbf{Small\_Raindrop}_i(t) + \mathbf{d}(t, \lambda) \\ (i = 1, 2, \cdots, N).\end{aligned} \quad (10)$$

where

$$\begin{aligned}\mathbf{d}(t, \lambda) = \tau_1 \cdot rand_1 \cdot sign(F(\mathbf{RP}_{k_1}) - F(\mathbf{Small\_Raindrop}_i(t))) \\ \cdot (\lambda \cdot \boldsymbol{RP}_{k1} - \mathbf{Small\_Raindrop}_i(t)) + \tau_2 \cdot rand_2 \cdot sign(F(\mathbf{RP}_{k_2}) \\ - F(\mathbf{Small\_Raindrop}_i(t))) \cdot (\lambda \cdot \mathbf{RP}_{k2} - \mathbf{Small\_Raindrop}_i(t)).\end{aligned} \quad (11)$$

where τ_1 and τ_2 are two step parameters of **Small_Raindrop** flowing, $rand_1$ and $rand_2$ are two uniformly distributed random numbers in the range (0, 1), $F(\cdot)$ is the fitness function, *sign*($\cdot$) stands for sign function, λ is a damping coefficient, $\mathbf{RP}_{k1}$ and $\mathbf{RP}_{k2}$ are any two of candidate solutions in raindrop pool **RP**, and can be chosen by the tournament selection procedure. However, each raindrop could not have been in the flowing in a real environment. They will stay in the locations with a relatively lower elevation or evaporate after several flowing. It is necessary to introduce a parameter *Max_Flow_Number* to control the maximum number of flowing.

5) Raindrop updating operator: Evaporation is one of the most important links in the water cycle. In the land water cycle system, the water vapor produced by the evaporation is mainly from surface water. The water vapor will be into the atmosphere and further form the raindrops. In order to improve the computational performance and convergence rate for global optimization problem, the raindrop updating operator φ_R^U is executed as follows.

$$\mathbf{Pop}(t+1) = \varphi_R^U(\mathbf{Pop}(t) \cup \mathbf{Small\_Raindrop}(t)). \quad (12)$$

That is to say, we use a sort method to select the *N* best individuals as the next population from the above two populations. In our work, the ranking method is achieved by a bubble-sort procedure.

B. Algorithm procedures

As explained above, the implementation steps of ARA are summarized as follows:

Step 1. Initialization: Choose the algorithm parameters *N*, *D*, λ, τ_1, τ_2, *Max_Flow_Number*; Randomly Generate initial population **Pop**(0); Set $t = 0$.

Step 2. Evaluation: Calculate the objective function values of all vapors in **Pop**(*t*); Find the best solution **gbest**(0); **RP**(0) = **gbest**(0);

Step 3. Raindrop Generation: Get **Raindrop**(*t*) by applying raindrop generation operator φ_R^G to **Pop**(*t*);

Step 4. Raindrop Descent: Get **New_Raindrop**(*t*) by applying raindrop descent operator φ_R^D to **Raindrop**(*t*);

Step 5. Raindrop Collision: Get **Small_Raindrop**(*t*) by applying raindrop collision operator φ_R^C to **New_Raindrop**(*t*);

Step 6. Raindrop Flowing: Get **New_Small_Raindrop**(*t*) by applying raindrop flowing operator φ_R^F to **Small_Raindrop**(*t*);

Step 7. Raindrop Updating: Get **Pop**(*t*+1) by applying raindrop updating operator φ_R^U to **Pop**(*t*)∪**Small_Raindrop**(*t*) and update Raindrop pool **RP**(*t*+1);

Step 8. Termination Test: If termination condition is satisfied, export the individual with the smallest objective function value in **Pop**(*t*+1), stop the algorithm; otherwise, $t = t + 1$, go to Step 3.

IV. APPLICATION OF ARA TO OPTIMAL APPROXIMATION OF A STABLE LINEAR SYSTEM

A. Problem description

In this part, ARA is used for approximating a stable linear system. The system is taken from [9], and the transfer function is given by equation. Then the proposed ARA will be compared with that of PSO [10], DE [11], GSO [12], ABC [13] and CS [14]. The transfer function of the system is given as follows:

$$G(s) = \frac{k_d k_{r_1}(\tau_{od}s + 1)e^{-\theta_d s}}{(\tau_r s + 1)(\tau_1 s + 1)(\tau_2 s + 1) - k_{r2}k_d(\tau_{od}s + 1)e^{-\theta_d s}}. \quad (13)$$

where $k_{r1} = 0.258$, $k_{r2} = 0.281$, $k_d = 1.4494$, $\theta_d = 1.4494$, $\tau_r = 0.3684$, $\tau_1 = 1.9624$, and $\tau_2 = 0.43256$. It is desired to find the second-order model

$$H_2(s) = \frac{k_{2,p}(s + \tau_{2,z})}{a_{2,0} + a_{2,1}s + s^2} \cdot e^{-\tau_{2,d}s}. \quad (14)$$

Therefore, the parameters to be determined are $a_{2,1}$, $k_{2,p}$, $\tau_{2,z}$ and $\tau_{2,d}$. Owing to the fact that the original system is stable, each parameter lies in the interval $[0, +\infty)$.We use the ARA to search for the optimal parameters of $H_2(s)$ in two fixed search intervals, including $[0, 10]^4$, and $[0, 50]^4$.

B. Experimental platform and algorithms parameter settings

For all experiments, 50 independent runs are carried out on the same machine with a Celeron 3.40 GHz CPU, 4GB memory, and windows 7 operating system with Matlab 7.9, and conducted with 40000 function evaluations (FES) as the termination criterion. For all algorithms, the population size *N* is set to 50. The other specific parameters of algorithms are given as follows.

- **PSO Settings:** There are other three control parameters denoted ω, c_1 and c_2. As suggested in [15], a linearly decreasing inertia ω from 0.9 to 0.4 is adopted over the course of the search. c_1 and c_2 are two coefficients of PSO and set to be 1.49445.

- **DE Settings:** There are other two control parameters denoted *F* and *CR* in DE. *F* is a mutation step which is a real constant and affects the differential variation between two solutions. *CR* is a crossover rate which controls the change of the diversity of the population. As suggested in [16], *F* is set to be 0.5 and *CR* is 0.9, respectively.

- **GSO Settings:** There are other five important control parameters denoted θ_0, *a*, θ_{max}, α_{max}, and l_{max}. We adopt the same parameters setting with the original publication [12]. The θ_0 is the initial head angle of each individual and is set to $(\pi/4, \pi/4, \cdots, \pi/4)$. The constant *a* is given by $round(\sqrt{D})$, where *D* is the dimension of the problem. The maximum pursuit angle θ_{max} is set to π/a^2. The maximum turning angle α_{max} is set to $\theta_{max}/2$. The maximum pursuit l_{max} is calculated from the formula $l_{max} = ||\mathbf{U} - \mathbf{L}||$. Where U_i and L_i are the upper bounds and lower for the *i*th dimension.

- **ABC Settings:** There are other five control parameters used in ABC: the number of the food sources N_1, the number of employed bees N_2, the number of onlooker bees N_3, the umber of scour bees N_4 and the value of *limit*. As suggested in the original publication [13], $N_1 = N$, $N_2 = N_3 = N/2$, $N_4 = 1$, *limit* $= 100$.

- **CS Settings:** There are other three control parameters denoted P_a, α and β. P_a is a probability that a host can discover an alien egg and is set to 0.25. α is the step size which is related to the scales of the problem of interest. The β is a parameter which is related to Lévy flights. In the experiment, α is set to be 0.01 and β is 1.5, respectively. The above three control parameters settings are suggested in the original publication [14].

- **ARA Settings:** There are other five control parameters in ARA. The optimal setting of the ARA parameters is very difficult to obtain and will be needed to further study in the future. However, some advice is given to set the ARA parameters as follows. The value of parameter λ can be either 1 or 2, which is again a heuristic step and decided randomly with equal probability. The two step parameters are $\tau_1 = 2$ and $\tau_2 = 2$. The maximum number of flowing *Max_Flow_Number* is set to be 3. The size of **RP** is *N*.

C. Comparisons on the experimental results

The optimal approximate models and the corresponding performance indices obtained by ARA are compared with other algorithms in Table II and Table III. The convergence curves of six algorithms are shown in Fig. 1 and Fig. 2.

TABLE II. COMPARISONS OF ARA, PSO, DE, GSO, ABC AND CS IN OPTIMAL APPROXIMATION OF THE STABLE LINEAR SYSTEM ON SEARCH SPACE $[0, 10]^4$.

Algorithm	Approximate Model
PSO	$H_2(s) = \frac{0.0199772(s+6.9217567)}{s^2+1.2345129s+0.2191751} \cdot e^{-0.4489693s}$ $J = 4.13e-05$
DE	$H_2(s) = \frac{0.0134725(s+9.9834769)}{s^2+1.2524285s+0.2131921} \cdot e^{-0.4087373s}$ $J = 5.14e-05$
GSO	$H_2(s) = \frac{0.1615397(s+6.5991488)}{s^2+9.0476640s+1.6896933} \cdot e^{-1.0440751s}$ $J = 1.12e-02$
ABC	$H_2(s) = \frac{0.1441361(s+5.8249825)}{s^2+6.5715485s+1.3307853} \cdot e^{-0.9790431s}$ $J = 1.00e-03$
CS	$H_2(s) = \frac{0.0217212(s+6.5989635)}{s^2+1.2350808s+0.2271961} \cdot e^{-0.4611103s}$ $J = 6.41e-05$
ARA	$H_2(s) = \frac{0.013448(s+10.0000000)}{s^2+1.2523644s+0.2131672} \cdot e^{-0.4085342s}$ $J = \mathbf{3.63e-05}$

TABLE III. COMPARISONS OF ARA, PSO, DE, GSO, ABC AND CS IN OPTIMAL APPROXIMATION OF THE STABLE LINEAR SYSTEM ON SEARCH SPACE $[0, 50]^4$.

Algorithm	Approximate Model
PSO	$H_2(s) = \frac{0.0668230(s+40.2589210)}{s^2+25.3339184s+4.2641112} \cdot e^{-0.9360568s}$ $J = 8.53e-03$
DE	$H_2(s) = \frac{0.0065845(s+23.0111473)}{s^2+1.2630373s+0.2401598} \cdot e^{-0.3637589s}$ $J = 7.92e-06$
GSO	$H_2(s) = \frac{0.4835010(s+14.9822682)}{s^2+46.9811375s+11.4819488} \cdot e^{-1.0716813s}$ $J = 1.18e-02$
ABC	$H_2(s) = \frac{0.0266913(s+25.1777486)}{s^2+39.7536285s+8.9290529} \cdot e^{-1.0247974s}$ $J = 1.04e-02$
CS	$H_2(s) = \frac{0.0266913(s+27.5172856)}{s^2+5.3300849s+1.1641676} \cdot e^{-0.7678470s}$ $J = 5.43e-03$
ARA	$H_2(s) = \frac{0.0046815(s+34.7929249)}{s^2+1.2651678s+0.2581766} \cdot e^{-0.3509602s}$ $J = \mathbf{6.76e-06}$

From the Table II and Table III, when the search space is set to $[0, 10]^4$, and $[0, 50]^4$, the performance indices of models obtained by ARA has slightly better than that of other five algorithms. The results also suggest that a larger search space does not evidently influence the performance of ARA in approximating of the stable linear system.

V. Discussion

It is also very interesting to compare ARA with PSO. Like PSO, ARA is also introduced to deal with continuous function optimization problems. However, it is not very difficult to find that there are some major difference between ARA and PSO. Firstly, PSO is inspired from the models of coordinated animal motion and originally developed for simulating the animal

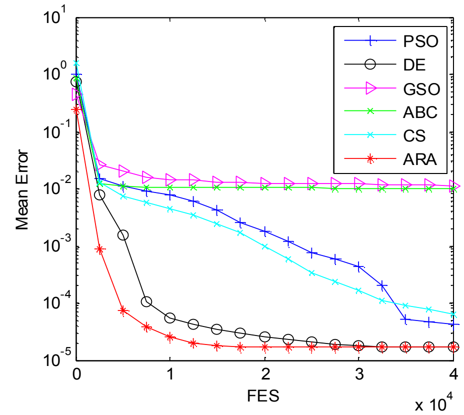

Fig. 1. Mean error of J that six algorithms search for the optimal parameters of $H_2(s)$ in the fixed space $[0, 10]^4$.

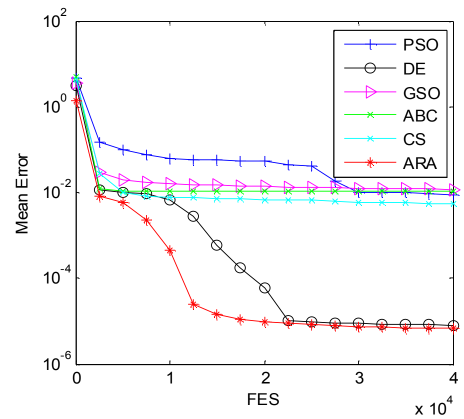

Fig. 2. Mean error of J that six algorithms search for the optimal parameters of $H_2(s)$ in the fixed space $[0, 50]^4$.

swarm behavior, mainly fish schooling and bird flocking, while ARA is proposed based on the particle system and the observation of natural rainfall. Secondly, each particle in the basic PSO contains a velocity item. Nevertheless, the concept does not appear in ARA. Thirdly, ARA uses a new learning strategy whereby all particles' historical best information to guide each particle to aim to move to a better position, which may ensure the diversity of population and avoid premature convergence.

Although the ARA and evolutionary algorithms draw inspiration from completely different disciplines, ARA still shares some similarities with the other evolutionary algorithms. For instance, all of them make use of the notion of fitness to guide search toward better solutions. Like most of evolutionary algorithms or swarm intelligence algorithms, ARA is also a population-based algorithm.

The above comparisons between ARA and other nature based heuristic algorithms may offer a possible explanation that why ARA could obtain better results on some optimization problems and it is possible for ARA to deal with more complex problems better.

VI. Conclusion

In this paper, a novel meta-heuristic algorithm-ARA is proposed for optimal approximation of a stable linear system. Numerical simulation results and comparisons show that the proposed algorithm is also effective and efficient as with other algorithms. Moreover, they further show that the propose algorithm may be a potential approach for unstable linear systems, as well as other numerical optimization problems in control and other related areas. In short, ARA, like other intelligent algorithms, may be used as a general-purpose optimization method for various practical optimization problems.

Acknowledgment

The authors express their sincere thanks to the reviewers for their significant contributions to the improvement of the final paper. This research is partly supported by the National Natural Science Foundation of China under Project Code (61272283, 61100173, 61100009, 61305083, 61304082), Shaanxi Province Natural Science Foundation (2012JQ8052), Shaanxi Province education department natural science research item (11JK0918).

References

[1] L. Fortuna, G. Nunnari, A. Gallo, *Model Order Reduction Techniques with Applications in Electrical Engineering.* Spring-Verlag, London, 1992.

[2] W. C. Zhong, J. Liu, M. Z. Xue, L. C. Jiao, "A multiagent genetic algorithm for global numerical optimization," *IEEE Trans. System, Man, and Cybern. B*, vol. 34, no. 2, pp. 1128-1141, Oct. 2004.

[3] S. L. Cheng, C. Y. Hwang, "Optimal approximation of linear systems by a differential evolution algorithm," *IEEE Transactions on Circuits and Systems-I*, vol. 43, no. 6, pp. 800-805, Nov. 1996.

[4] M. G. Gong, H. F. Du, L. C. Jiao, "Optimal approximation of linear systems by artificial immune response," *Science in China: Series F Information Sciences*, vol. 49, no. 1, pp. 63-79, Jan. 2006.

[5] M. G. Gong, L. C. Jiao, L. N. Zhang, "Baldwinian learning in clonal selection algorithm for optimization," *Information Sciences*, vol. 180, no. 8, pp. 1218-1236, Apr. 2010.

[6] G. Iacca, F. Neri, E. Mininno, Y. S. Ong and M. H. Lim, "Ockham's Razor in Memetic Computing: Three Stage Optimal Memetic Exploration," *Information Sciences*, vol. 188, no. 1, pp. 17-43, Apr. 2012.

[7] W. T. Reeves, "Particle Systems-a technique for modeling a class of fuzzy objects," *Computer Graphics*, vol. 17, no. 3, pp. 359-376, Jul. 1983.

[8] P. Larrañaga, J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation Norwell.* MA, Kluwer, 2001.

[9] S. L. Cheng, C. Y. Hwang, "Optimal approximation of linear systems by a differential evolution algorithm," *IEEE Trans. System, Man, and Cybern. A*, vol. 31, no. 6, pp. 698-707, Nov. 2001.

[10] J. Kennedy, R. Eberhart, "Particle swarm optimization," *Proceedings of IEEE International Conference on Neural Network*, pp. 1942-1948, 1995.

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

[12] S. He, Q. H. Wu, J. R. Saunders,"Group Search Optimizer: An Optimization Algorithm Inspired by Animal Searching Behavior," *IEEE Transaction on Evolutionary Computation*, vol. 13, no. 5, pp. 973-990, Oct. 2009.

[13] D. Karaboga, B. Basturk,"A powerful and efficient algorithm for numerical function. optimization: Artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459-471, Nov. 2007.

[14] X. S. Yang, S. Deb, *Cuckoo search via Lévy flights, Proceedings of World Congress on Nature & Biologically Inspired Computing (NaBIC 2009, India).* IEEE Publications, USA, 2009.

[15] J. Liang, A. K. Qin, P. N. Suganthan, S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281-295, Jun. 2006.

[16] J. Brest, S. Greiner, B. Bošković, M. Mernik, "Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646-657, Dec. 2006.