International Journal of Innovative Computing, Information and Control
Volume **9**, Number **6**, June **2013**

ICIC International ©2013 ISSN 1349-4198
pp. **2665–2688**

HYBRID TAGUCHI-CHAOS OF ARTIFICIAL BEE COLONY ALGORITHM FOR GLOBAL NUMERICAL OPTIMIZATION

Jia-Ping Tien and Tzuu-Hseng S. Li*

aiRobots Laboratory
Department of Electrical Engineering
National Cheng Kung University
No. 1, University Road, Tainan 701, Taiwan
{ n2894112; n2895109 }@mail.ncku.edu.tw; *Corresponding author: thsli@mail.ncku.edu.tw

Received March 2012; revised August 2012

Abstract. *In this paper, a new evolutionary learning algorithm is proposed by hybridizing the Taguchi method and chaos artificial bee colony (CABC). The algorithm is thus called HTCABC. First, the chaos search algorithm and adaptive bound method is adopted to improve the ABC performance and convergence rate. Then, the Taguchi method and crossover operation are incorporated into the CABC to produce good food sources, thus accelerating the search capacity. The Taguchi method has also been utilized to establish a proper balance between the exploration and exploitation by incorporating the information from the best global solution into the solution search equation. Third, the natural phenomenon of the elite strategy is adopted and the recruitment of new scout bees is used for HTCABC, which can have a rapid convergence rate maintain the diversity of the population, and escape from local optima. Additionally, there is no complex parameter setting in the algorithm design. Therefore, the HTCABC can be a more robust, quickly convergent and more accurate optimal solution. Finally, the algorithm is examined by using a set of benchmarks and the proposed approach is effectively applied to solve the parameter identification of a chaotic system. Simulation results show that the proposed algorithm is more efficient than the existing algorithm reported in the literature.*

Keywords: Taguchi method, Chaos search, Artificial bee colony algorithm

1. Introduction.

It is well-known that many practical dynamic systems can be modeled as optimization problems. In recent years, the population-based optimization algorithm has inspired new resources for optimization of problem solving, such as genetic algorithm (GA) [1,2], particle swarm optimization (PSO) [5-5], differential evolution (DE) algorithm [6], and artificial bee colony (ABC) [7-9]. The ABC algorithm invented recently by Karaboga is a biologically-inspired optimization algorithm, which has been compared with the algorithms of GA, PSO, evolutionary algorithms (EA), and DE for optimizing multi-modal numerical functions. Moreover, the ABC algorithm has been run with different colony sizes and limit values and produces the better performance among the algorithms presented in the literature [10-12]. The major advantage of the ABC algorithm is that it possesses both the global and local searching capability in each iteration. In addition, except for the population number and maximum evaluation number, the ABC uses one control parameter, which is called "limit". In the ABC algorithm, onlookers and employed bees carry out the exploitation process in the search space, and the scouts control the exploration process. One of the employed bees is selected and classified as the scout bee by a control parameter called limit. Essentially, the ABC algorithm is very simple and robust.

However, like most population-based algorithm, the ABC algorithm also has some drawbacks. The convergence speed of the ABC algorithm is slower than DE and PSO in solving the unimodal problem and the ABC algorithm is poor at exploitation in handling the complex multimodal problem because of its stochastic nature. Furthermore, some research results on the optimization problem show that it can obtain a good enough solution through the improved ABC algorithm [13-16]. In [13], Alatas introduced different chaotic maps into the ABC algorithms (CABC) to avoid the local optimum. The global searching capability of the CABC was not satisfactory because it suffered from different benchmark problems. In [14], Zhu and Kwong proposed an improved ABC algorithm called gbest-guided ABC (GABC) algorithm by incorporating the information from the global solution into the solution search equation of the ABC algorithm to improve the exploitation. In the GABC algorithm the gbest term parameter of the modified solution search equation plays an important role in controlling the exploration and exploitation of the new candidate solution search. However, the control parameter of the modified solution search equation must be tested and selected to obtain the best performance when facing with different modal/dimension benchmark problems. Kang *et al.* [15] used the Rosenbrock's rotational direction method (RM) to implement the exploitation phase and proposed the Rosenbrock ABC algorithm (RABC). By RABC, the quality of solutions depends on three different control parameters values in the exploitation phase. However, the RABC must find a proper combination of different parameter values to improve the quality of solutions for different modal/dimension benchmark problems. Gao and Liu [16] proposed an improved ABC (IABC) by adding two solution search equations. However, the control parameter and the selective probability of the two improved solution search equations must be tested and chosen when dealing with different modal/dimension benchmark problems.

Recently, some researchers have proposed evolutionary algorithms, GA, PSO or IA are able to be further promoted by using the Taguchi method and chaos operator [17-22]. The Taguchi method is a robust parameter design method for optimizing the product and process conditions by minimizing the effect of the causes of the variation [23]. Therefore, the Taguchi method can play a role in attaining more robust solutions for the evolutionary algorithm. The Taguchi method is incorporated into the crossover operation of GAs [17,18]. The reasoning ability of the Taguchi-based crossover can systematically select better genes to achieve a crossover and, consequently, enhance the GA. In [19], modifying the velocity equation of PSO by utilizing the Taguchi method can establish a proper balance between the social and the cognitive to provide a fast convergence toward the best solution.

Chaos is a universal phenomenon of nonlinear dynamic systems. Taking advantage of the ergodic and stochastic properties of chaotic variables, a chaotic search algorithm can escape from local optima. Liao and Tsao [20] presented a hybrid chaos search immune algorithm, GA, and fuzzy system (CIGAFS) method. Using the CIGAFS can accelerate the search speed and increase the global search ability for solving short-term thermal generating unit commitment (UC) problems. He *et al.* [21] introduced chaos into the CLONALG [24] to enhance the global searching ability and overcome the prematurity of the algorithm. Zuo and Fan [22] developed the chaos search immune algorithm (CISA), which employed the evolutionary model of the CLONALG and the chaotic optimization mechanism of COA [25] to obtain good global and local searching capabilities.

Motivated by the literature, we combine the ABC algorithm, chaos operator, and Taguchi method in this paper to form a new hybrid intelligent algorithm, namely HTCABC. The advantages of the proposed HTCABC are summarized as follows.

i) Combining the Taguchi experimental design method and the crossover operation of GA can enhance HTCABC so that the HTCABC can be more robust, statistically sound, and fast convergent. ii) Combining the Taguchi method and the CABC will have exploitation and exploration capability. iii) The adaptive bound CABC algorithm is proposed so that the proper bounds of the chaotic search space can be obtained automatically in the training process to speed up search of the CABC. iv) The concept of elite strategy in preserving superior bees is adopted in HTCABC and colony bees in a new generation are created, not only by recruitment of scouts, but also by HTCABC. v) The parameters of HTCABC are fixed for different problems, further reducing the difficulties in parameter setting and enhancing the adaptability of HTCABC. Thus, the proposed method not only has a rapid convergence rate but also improve the searching ability. Additionally, parameter setting is simple in HTCABC design.

The paper is organized as follows. Section 2, a brief introduction to the ABC algorithm is described. In Section 3, the learning algorithm of HTCABC for global numerical optimization is described. Section 4 proves the performance of the proposed HTCABC algorithm comparison with existing ABC, CABC3, GABC, RABC, and IABC by testing it for some well known benchmark functions. Moreover, we apply the HTCABC approach to solving the complex multimodal problem of the parameter identification of a chaotic system. Finally, conclusions are drawn in Section 5.

2. **Artificial Bee Colony Algorithm.**

The artificial bee colony algorithm is a new population-based optimization approach that imitates the intelligent behavior of real honey bees [26]. In the ABC algorithm, the colony of artificial bees comprises three groups of bees: employed bees, onlookers and scouts. The employed bees search for food around the food sources from their memory and then provide the onlookers with food information. The onlookers wait in the nest and conduce to select more advantageous food sources through sharing the information from the employed bees; then they further search for food around the selected food source. A bee completing a random search is called a scout. In other words, the food search of bees is cooperatively performed by the exchange of information among the bees. By simulating the foraging behaviors of a honey bee swarm, the pseudo-code for the ABC algorithm [10-12] can be described in Algorithm 1. In the ABC algorithm, a solution represents a food source and each food source is exploited by one employed bee.

Algorithm 1: Pseudo-code for ABC algorithm

```
1.  Initialize algorithm and problem parameters (food source positions, $x_i$, $i = 1, \ldots,$
    $sn/2$).
2.  Evaluate the nectar amount of food sources.
3.  cycle = 1
4.  **repeat**
5.  Employed Bees Phase Calculation.
6.      **for** each employed bee
7.          Produce a new food source position $v_i$.
8.          Calculate the fitness value $fit(i)$.
9.          Apply a greedy selection mechanism between $x_i$ and $v_i$.
10.     **end for**
11. Calculate the probability values $p_i$ for the solution.
12. Onlooker Bees Phase Calculation.
13.     **for** each onlooker bee
14.         Choose a food source depending on $p_i$.
15.         Produce a new food source position $v_i$.
```

```
16.         Calculate the fitness value $fit(i)$.
17.         Apply a greedy selection mechanism between $x_i$ and $v_i$.
18.     end for
19. Scout Bees Phase Calculation.
20. If the scout solution is better than the employed solution, the employed solution is
    replaced by a new random source position.
21. Memorize the best solution achieved so far.
22. cycle = cycle + 1.
23. until cycle = Maximum Cycle Number.
```

In the initialization phase, the ABC algorithm generates a randomly distributed initial population of sn solutions, where sn denotes the size of the population. Half of the population includes employed bees, while the other half comprises onlooker bees. Each solution x_i $(i = 1, 2 \ldots, sn/2)$ is a D-dimensional vector $(x_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,D}])$, where D is the number of optimization parameters. In the ABC algorithm, the nectar amount is the value of the fitness. In the employed bees phase each employed bee searches for a new food source v_i in the neighborhood of its current source x_i, which denotes the i-th solution in the population. The new candidate solution is calculated using the following search equation:

$$v_i^j = x_i^j + \phi_i^j (x_i^j - x_r^j) \tag{1}$$

where $r \in (1, 2, \ldots, sn)$ and $j \in (1, 2, \ldots, D)$ are randomly chosen indexes, and $r \neq i$. ϕ_i^j is a random number between $[-1, 1]$. Then, a greedy selection is done between x_i and v_i that completes the updating process.

In the onlooker bees phase, each onlooker chooses a food source with a probability related to the nectar amount of the food source shared by the employed bees. The probability is calculated by the following expression:

$$p_i = \frac{fit_i}{\sum_{n=1}^{EN} fit_n} \tag{2}$$

where fit_i is the fitness value of the solution, i is evaluated by its employed bee, and EN is the number of food sources, which is equal to the number of employed bees ($EN = sn/2$). The search process used in the onlooker bees phase is the same as that in the employed bees one.

In the scout bees phase, if a food source cannot be improved through a predetermined number of cycles, called "limit", it is removed from the population, and the employed bee of that food source becomes a scout. The scout bee finds a new random food source position using the equation below

$$x_i^j = x_{\min}^j + rand(0, 1)(x_{\max}^j - x_{\min}^j) \tag{3}$$

where $x_{\min}^j$ and $x_{\max}^j$ are lower and upper bounds of parameter j, respectively.

These steps are repeated through a predetermined number of cycles, called Maximum Cycle Number (MCN), or until a termination is satisfied.

3. Hybrid Taguchi-Chaotic Artificial Bee Colony Algorithm for Global Numerical Optimization.

In this section, we describe the basic concepts of the Taguchi method, chaotic theory and how to efficiently solve the global numerical optimization problem with continuous variables by using the HTCABC. The pseudo-code of the HTCABC algorithm proposed for solving the global numerical optimization problem is given in Algorithm 2:

Algorithm 2: Pseudo-code main body of HTCABC algorithm

```
1. Initialization.
```

```
2. Evaluation.
3. cycle = 1.
4. **repeat**
5. Generation of diverse employed bees by crossover operation.
6. Generation of better employed bees and the best employed bee by the Taguchi
   method.
7. Enhanced employed bees phase calculation.
8. Enhanced onlooker bees phase calculation.
9. Enhanced scout bees phase calculation.
10. Memorize the best solution achieved so far (Elites).
11. Recruitment.
12. cycle = cycle + 1.
13. **until** cycle = Maximum Cycle Number.
```

3.1. **Problem statement.** Generally, the unconstrained global optimization problem is considered

$$\min f(X), \quad X = [x^1, x^2, \dots, x^D] \quad \text{s.t. } x^j \in [x^j_{\min}, x^j_{\max}], \quad j = 1, 2, \dots, D \tag{4}$$

where f is the objection function, and X is the decision solution vector consisting of D variables $x^j \in \Re^D$ bounded by lower ($x^j_{\min}$) and upper limits ($x^j_{\max}$).

3.2. **The detail steps of Algorithm 2.**

Step 1. Generation of the initial population

In the HTCABC, a problem with the objection function value (fitness) is treated as the nectar amount of a food source, while the solution to this problem represents a food source. Each food source is exploited by one employed bee. The real number representation for potential solutions is adopted to simplify the food source operator definitions. The initialization procedure produces sn solutions x_M ($M = 1, 2 \dots, sn$). Then suppose the M-th solutions $x_M = [x^1, x^2, \dots, x^j, \dots, x^D]$ is a D-dimensional vector as a food source position to represent a solution to the optimization problem.

Algorithm 3: Initialization step of the HTCABC algorithm

```
1. **for** $M = 1$ to $sn$ do
2.     **for** $j = 1$ to $D$ do
3.         Generate $x_M$ solution
4.
           $x^j_M = x^j_{\min} + rand(0,1)(x^j_{\max} - x^j_{\min})$          (5)
           where $x^j_{\min}$ and $x^j_{\max}$ are the lower and upper bounds of the parameter $j$,
           respectively.
5.     **end for**
6. **end for**
```

Step 2. Fitness evaluation

For the minimization problem,

$$fitness = offset - J \tag{6}$$

where J is a smaller performance index, i.e., $J = f^*$ (the global optimum value). The offset of fitness value is set to be 50 or 10^3 for minimum optimization.

According to above statements, the best employed bee with the largest fitness representing the optimal solution to the problem is obtained.

Step 3. Generation of diverse employed bees by crossover operation.

A good initial solution is fundamental for computational performance of a search algorithm. The search algorithm might encounter premature convergence in optimization multimodal problems if a sufficient diversity is not provided within the initial population. In the ABC algorithm the initial population is not considered to be feasible. The crossover operation is performed to generate a high diversity of employed bees to avoid early convergence. In the ABC algorithm, there is no explicit crossover unlike in GA. However, the transformation of good information between the employed bees is completed by adopting crossover operation which is quite important to accelerate the local converging speed of the ABC algorithm.

The crossover operation used here is an arithmetical operator derived from [27], which calculates the linear combination to generate new employed bees. For example, selecting two employed bees x_1 and x_2 using the roulette wheel selection may produce two new employed bees, $\tilde{x}_1$ and $\tilde{x}_2$, which is calculated as follows:

$$\tilde{x}_1^j = a \cdot x_1^j + (1-a) \cdot x_2^j \text{ and } \tilde{x}_2^j = (1-a) \cdot x_1^j + a \cdot x_2^j, \quad j = 1, 2, \ldots, D \tag{7}$$

where a is a random value and $a \in [0, 1]$. In this work, the value of the parameter $\tilde{x}_1^j$ and $\tilde{x}_2^j$ exceeding its boundary is set to its boundaries.

Generation of diverse employed bees by a crossover operation of the HTCABC algorithm is presented in Algorithm 4.

Algorithm 4: Crossover operation of the HTCABC algorithm

```
1.  for $M = 1$ to $sn$ do
2.  Select the two employed bees $x_1$ and $x_2$ based on the roulette wheel selection
3.      if rand < $P_c$ ($P_c$: crossover rate, $P_c = 0.7$)
4.          for $j = 1$ to $D$ do
5.              Produce two new employed bees, $\tilde{x}_1$ and $\tilde{x}_2$ according to Equation (7)
                    $\tilde{x}_1^j, \tilde{x}_2^j \in [x_{\min}^j, x_{\max}^j]$
6.          end for
7.      else
8.          $\tilde{x}_1 = x_1$ and $\tilde{x}_2 = x_2$
9.      end if
10. end for
```

Step 4. Generation of better employed bees and the best employed bee by the Taguchi method.

In the HTCABC, the Taguchi method is inserted between the crossover and chaos of artificial bee colony (CABC) in order to generate representative employed bees to become better employed bees. Moreover, the Taguchi method is adopted in order to improve the exploitation of the ABC algorithm. The details are described in the next section.

The Taguchi method is a robust parameter design method for optimizing the product and process conditions by minimizing the effect of the causes of the variation. In this paper two major tools of the Taguchi method orthogonal array (OA) are used as well as the signal-to-noise ratio (SNR). The orthogonal array of the Taguchi method is used to study a large number of decision variables with a small number of experiments. The SNR is a quality measuring characteristic in the field of communication engineering, which was introduced in the field of Design of Experiments (DOE) by Taguchi [31].

The Taguchi concept is based on SNR measuring the statistical variation of each experimental number and assigning the optimal level of control factors whereas the orthogonal array determines the combination of factor levels for each experimental run ensuring a balanced comparison of levels. Then, the better combinations of decision variables are decided by the orthogonal arrays and SNRs. The details regarding the Taguchi method can be obtained in [28-31]. In this research, a two-level orthogonal array is used. There are Q factors, where Q is the number of design factors (variables), and each factor has two levels. To establish an orthogonal array of Q factors with two levels, a two level orthogonal array must be used generalized by $L_m(2^{m-1})$, where $m = 2^h$ presents the number of employed bees experiments that is the number of rows in the orthogonal array, h is a positive integer greater than 1, the number of levels is 2, $(m-1)$ represents the number of columns in the orthogonal array, and $Q \leq m-1$. If $Q < m-1$, only the first Q columns are used, while the other $m-1-Q$ columns are ignored. The size of the orthogonal array to be used depends on the problem in this study. For example, there are ten factors with two levels for each factor. We only need ten columns to allocate these factors, and $L_{16}(2^{15})$ is enough for this purpose because it has fifteen columns.

In order to determine the optimal level of each factor, the SNR has been used. It allows a contribution only from levels which have a maximum value. The SNR (η) refers to the mean-square-deviation of the objective function. Taguchi defined η to be, for the cases of the larger, the better characteristic and the smaller, the better characteristic, respectively, $\eta = -10\log\left((1/m)\sum_{t=1}^{m}(1/w_t^2)\right)$ and $\eta = -10\log\left((1/m)\sum_{t=1}^{m}w_t^2\right)$, which is measured in decibels, where $\{w_1, w_2, \ldots, w_N\}$ denotes a set of characteristics. A detailed description can be found in [28-31]. We modify the aforesaid equations to be $\eta_t = 0.01 \times w_t$ if the objection function is to be maximized (the larger-the-better) or minimized (the smaller-thebetter), respectively. Let w_t denote the fitness value of the objective function of experiment t, where $t = 1, 2, \ldots, m$. The effects of the various factors (variables) can be defined as follows:

$$E_{fl} = \sum \eta_t \quad \forall t: \text{ factor } f \text{ is at level } l \tag{8}$$

The two tools of the Taguchi method, two-level orthogonal and SNR, are employed in this study. For a two-level problem, if $E_{f1} > E_{f2}$, the optimal level is level 1 for factor f. Otherwise, level 2 is the optimum one. After the optimal levels for each factor are selected, one can also obtain the optimal employed bees.

Generation of better employed bees and the best employed bee by the Taguchi method of the HTCABC algorithm is presented in Algorithm 5.

Algorithm 5: Taguchi method of HTCABC algorithm

Select a suitable two-level orthogonal array $L_m(2^{m-1})$ as required by HTCABC. Choose two employed bees (named as level 1 and level 2) randomly, each comprising D factors ($D = Q$) which correspond to $\tilde{x}^j$ ($j = 1 \sim D$) and formulate a matrix of order $m \times Q$.

```
1.  for $M = 1$ to $sn$ do
2.      for $t = 1$ to number of experiment $m$ do
3.          Evaluate fitness value of experiment number $t$ by using Equation (6).
4.          Evaluate SNR $\eta_t = 0.01 \times w_t$ of experiment number $t$.
5.      end for
6.      for $f = 1$ to number of factor $Q$ do
7.          Evaluate effects of the factors $E_{f1}$ and $E_{f2}$ of factor $f$ by using Equation (8).
8.          if $E_{f1} > E_{f2}$
9.              Optimal level = level 1.
10.         else
11.             Optimal level = level 2.
12.         end if
13.     end for
14. Generation of better employed bees to enhance the employed phase. The better
    employed bees $\tilde{x}'_M = [\tilde{x}'^1_M, \tilde{x}'^2_M, \ldots, \tilde{x}'^j_M]$ s.t. $\tilde{x}'^j_M \in [x^j_{\min}, x^j_{\max}]$, $j = 1, 2, \ldots, D$.
15. end for
```

Step 5. Enhanced employed bees phase calculation.

In the ABC algorithm an employed bee discovers a new food source from the position of the food source in his memory depending on the local information (visual information), bee swarms are as an employed bees without exchanging any information among bees. However, in nature, share of information among bees is the most important circumstance in the formation of collective knowledge. In step 4, the honey bees are enhanced and become "better employed bees" through the Taguchi method. Afterwards, sort the fitness values in descending order among the better employed bees, and the high fitness is the better. Then, the better employed bees are divided into a top-third part and a foot-two thirds part. The foot-two thirds part is considered as the better employed bees whose food sources have been abandoned as their bees become scouts, while the top-one third part is considered as the "enhanced employed bees" which continue in the exploration and exploitation of new food sources by the modified solution search equation described in Equation (9).

After all the enhanced employed bees complete the search process, they share the nectar information of the food sources and their position information with the onlooker bees. The number of onlooker bees is selected to be equal to the enhanced employed bees.

Inspired by the global best guided ABC (GABC) [14], in HTCABC, in order to improve the exploitation of the ABC algorithm, the solution search equation Equation (1) is replaced by Equation (9) as follows:

$$\tilde{v}_e^j = \tilde{x}_e'^j + \gamma_e^j(\tilde{x}_e'^j - \tilde{x}_g'^j) + \lambda\left(gb^j - \tilde{x}_e'^j\right) \tag{9}$$

where $\tilde{x}_e'^j$ denotes the j-th element of the enhanced employed bees, $e \in (1, 2, \ldots, sn/3)$. $g \in (1, 2, \ldots, sn/3)$ and $j \in (1, 2, \ldots, D)$ are randomly chosen indexes, and $e \neq g$. γ_e^j is a random number between $[-1, 1]$. gb^j is the jth element of the global best solution, λ is a constant, and $\lambda = 1.5$. Then, a greedy selection is done between $\tilde{x}_e'$ and $\tilde{v}_e'$, which completes the updating process. As can be seen from Equation (9), the global best term by using the Taguchi method can drive the new candidate solution towards the global best solution, which is the modified solution search equation described by Equation (1), so that the HTCABC outperforms the ABC algorithm at exploitation. The enhanced employed bees procedure of the HTCABC algorithm is presented in Algorithm 6.

Algorithm 6: Enhanced employed bees phase of the HTCABC algorithm

```
1. for $e = 1$ to $sn/3$
2.     for $j = 1$ to $D$
3.         Produce a new food source $\tilde{v}_e$ for the enhanced employed bees of the food
           source $\tilde{x}_e$ by using Equation (9).
4.     end for
5. Calculate the value $fit(e)$ by using Equation (6).
6. Apply a greedy selection mechanism between $\tilde{v}_e$ and $\tilde{x}_e$.
7. The new food source $\tilde{v}_e = [\tilde{v}_e^1, \tilde{v}_e^2, \ldots, \tilde{v}_e^j]$ s.t. $\tilde{v}_e^j \in \left[x_{\min}^j, x_{\max}^j\right]$, $j = 1, 2, \ldots, D$.
8. end for
```

Step 6. Enhanced onlooker bees phase calculation.

In this study, the onlookers tend to select good food sources from those found by the enhanced employed bees, and then they further search for the food within the neighborhood of the selected food source by using chaos local search [22]. Chaos local search is performed in the neighborhood of the solution space of the onlookers by replacing some lower nectar food source. First, in order to improve the search ability and accelerate the search speed of the chaos local search, an adaptive lower $x_{\min}^j$ and upper limits $x_{\max}^j$ bound method is proposed.

The adaptive bound method is as follows:

$$x^j_{\min} = \max\{x^j_{\min}, x^j_b - \varepsilon(x^j_{\max} - x^j_{\min})\},$$
$$0 < \varepsilon < 1,\ x^j_b \in \left[x^j_{\min}, x^j_{\max}\right],\ j = 1, 2, \ldots, D \tag{10}$$
$$x^j_{\max} = \min\{x^j_{\max}, x^j_b + \varepsilon(x^j_{\max} - x^j_{\min})\},$$
$$0 < \varepsilon < 1,\ x^j_b \in \left[x^j_{\min}, x^j_{\max}\right],\ j = 1, 2, \ldots, D \tag{11}$$

where the step size ε is a random number, which adjusts the j-th element of the enhanced employed bees $\tilde{v}^j_e$ of the food source search range by the j-th element of the current best enhanced employed bees x^j_b.

The chaos local search is described as follows:

The chaos local search was executed in the onlookers of the food source neighborhoods to generate new food sources. First, the solution space of the onlookers of the food source $\tilde{v}^j_e$ is mapped to the chaos space $[0, 1]$ as follows:

$$\hat{v}^j_k = \frac{x^j_{\min} - \tilde{v}^j_e}{x^j_{\min} - x^j_{\max}},\ k = 1, 2, \ldots, sn/3 \text{ and } j = 1, 2, \ldots, D \tag{12}$$

where $\hat{v}^j$ is the j-th chaotic variable of the onlookers.

Then suppose the k-th onlookers $\hat{v}_k$ $(1 \le k \le sn/3)$ is expressed by a vector $\hat{v}_k = (\hat{v}^1_k, \hat{v}^2_k, \ldots, \hat{v}^D_k)^T$, then the new onlookers $\bar{v}_k = (\bar{v}^1_k, \bar{v}^2_k, \ldots, \bar{v}^D_k)^T$ generated by chaotic search in the neighborhood of $\hat{v}_k$ is obtained by

$$\bar{v}^j_k = \hat{v}^j_k + \alpha_k(2z^{p+1}_j - 1), \quad \bar{v}^j_k \in [0, 1]^D, \quad j = 1, 2, \ldots, D \tag{13}$$

where α_k is the size of the neighborhood that is around these onlookers, which is determined by

$$\alpha_k = \exp\left(-\frac{\rho(O_n - k)}{O_n}\right) \tag{14}$$

where O_n is the number of onlookers $(O_n = sn/3)$, and $\rho = 4.5$ is the control factor determining the neighborhood size. Equation (13) presents that for high fitness onlookers, a small step length is used to search roughly around the onlookers. $z^{p+1}_j = (z^{p+1}_1, z^{p+1}_2, \ldots, z^{p+1}_D)^T$ is the value of the j-th chaotic variable at the p-th chaotic iteration (Logistic mappings) and is expressed as

$$z^{p+1}_j = \mu_j z^p_j(1 - z^p_j),\ j = 1, 2, \ldots, D,\ p = 1, 2, \ldots, I_K \tag{15}$$

where j is the serial number of the chaotic variables, and $\mu_j = 4$.

The number of chaotic iterations in the neighborhood of the onlooker $\hat{v}_k$ can be calculated by

$$I_k = round\left(\frac{\delta \cdot sn}{3k}\right) \tag{16}$$

where $round(\cdot)$ denotes the operator whose argument is rounded toward the closest integer, and $\delta = 0.3$ is the multiplying factor determining the number of chaotic iterations. The enhanced onlooker bees procedure for the HTCABC algorithm is presented in Algorithm 7.

Algorithm 7: Enhanced onlooker bees phase of HTCABC algorithm

```
1. for $k = 1$ to $sn/3$
2.     for $p = 1$ to $I_K$
3.         for $j = 1$ to $D$
4.         $\tilde{v}$ is mapped to the chaos space $\hat{v}$ based on Equation (12).
```

```
5.          Produce a new food source $\bar{v}$ for the onlookers of the food source $\hat{v}$ by using
            chaos local search Equation (13).
6.          end for
7.      end for
8.  Map $\bar{v}$ from the chaos space to the solution space and calculate the value $fit(k)$ by
    using Equation (6).
9.  Apply a greedy selection mechanism between $\bar{v}$ and $\tilde{v}$.
10. The new food source $\bar{v}_k = [\bar{v}_k^1, \bar{v}_k^2, \ldots, \bar{v}_k^j]$ s.t. $\bar{v}_k^j \in \left[x_{\min}^j, x_{\max}^j\right]$, $j = 1, 2, \ldots, D$.
11. end for
```

Step 7. Enhanced scout bees phase calculation.

In this study, the scout bees are the foot-two thirds part of the better employed bees, and they continue to search around the food source by using chaos global search [22]. This is different from the standard ABC algorithm, where the scouts are random employed bees. Chaos global search is performed in the whole of the solution space of the better employed bees by replacing some lower nectar food source. Suppose the sth scout bees $\tilde{x}_s\,(sn/3 + 1 \leq s \leq sn)$ is expressed by a vector $\tilde{x}_s = (\tilde{x}_s^1, \tilde{x}_s^2, \ldots, \tilde{x}_s^j, \ldots, \tilde{x}_s^D)^T$. The enhanced scout bees procedure for the HTCABC algorithm is presented in Algorithm 8.

Algorithm 8: Enhanced scout bees phase of the HTCABC algorithm

```
1.  for $s = sn/3 + 1$ to $sn$ (the size of the Enhanced scout bees)
2.      for $U = 1$ to $I_c$ ($I_c$ : number of chaotic iterations, $I_c = 10$)
3.          for $j = 1$ to $D$
4.              $\tilde{x}_s$ is mapped to the chaos space $\hat{x}_s$ based on Equation (17).
5.
                $$\hat{x}_s^j = \frac{x_{\min}^j - \tilde{x}_s^j}{x_{\min}^j - x_{\max}^j}, \quad s = sn/3 + 1, \ldots, sn \text{ and } j = 1, 2, \ldots, D \tag{17}$$
6.              Produce a new food source $\bar{x}_s$ for the scouts of the food source $\hat{x}_s$ by using
                chaos global search Equation (18).
7.
                $$\bar{x}_{s,j}^{U+1} = \mu_j \hat{x}_{s,j}^U \left(1 - \hat{x}_{s,j}^U\right), \quad j = 1, 2, \ldots, D,\ U = 1, 2, \ldots, I_c \tag{18}$$
8.          end for
9.      end for
10. Map $\bar{x}_s$ from the chaos space to the solution space and calculate the value $fit(s)$ by
    using Equation (6).
11. Apply a greedy selection mechanism between $\bar{x}_s$ and $\tilde{x}_s$.
12. The new food source $\bar{x}_s = [\bar{x}_s^1, \bar{x}_s^2, \ldots, \bar{x}_s^j]$ s.t. $\bar{x}_s^j \in [x_{\min}^j, x_{\max}^j]$, $j = 1, 2, \ldots, D$.
13. end for
```

Step 8. Memorize the best solution achieved so far (Elites).

In each generation, HTCABC evaluates the bees' fitness values in the same population, and the top two thirds of the best-performing ones are selected. These bees are regarded as elite bees which are effective in finding good food sources.

Step 9. Recruitment (Diversity introduction).

This inserts $sn/3$ colony bees into the next generation to ensure diversity in the population during the search period, and to avoid an excessively long convergence time. In this step, all the bees without any prior knowledge play the role of detecting bees. After a search process for food sources, the detecting bees can convert into any kind of bees in accordance with the amount of nectar in food source.

Step 10. Test the terminate conditions

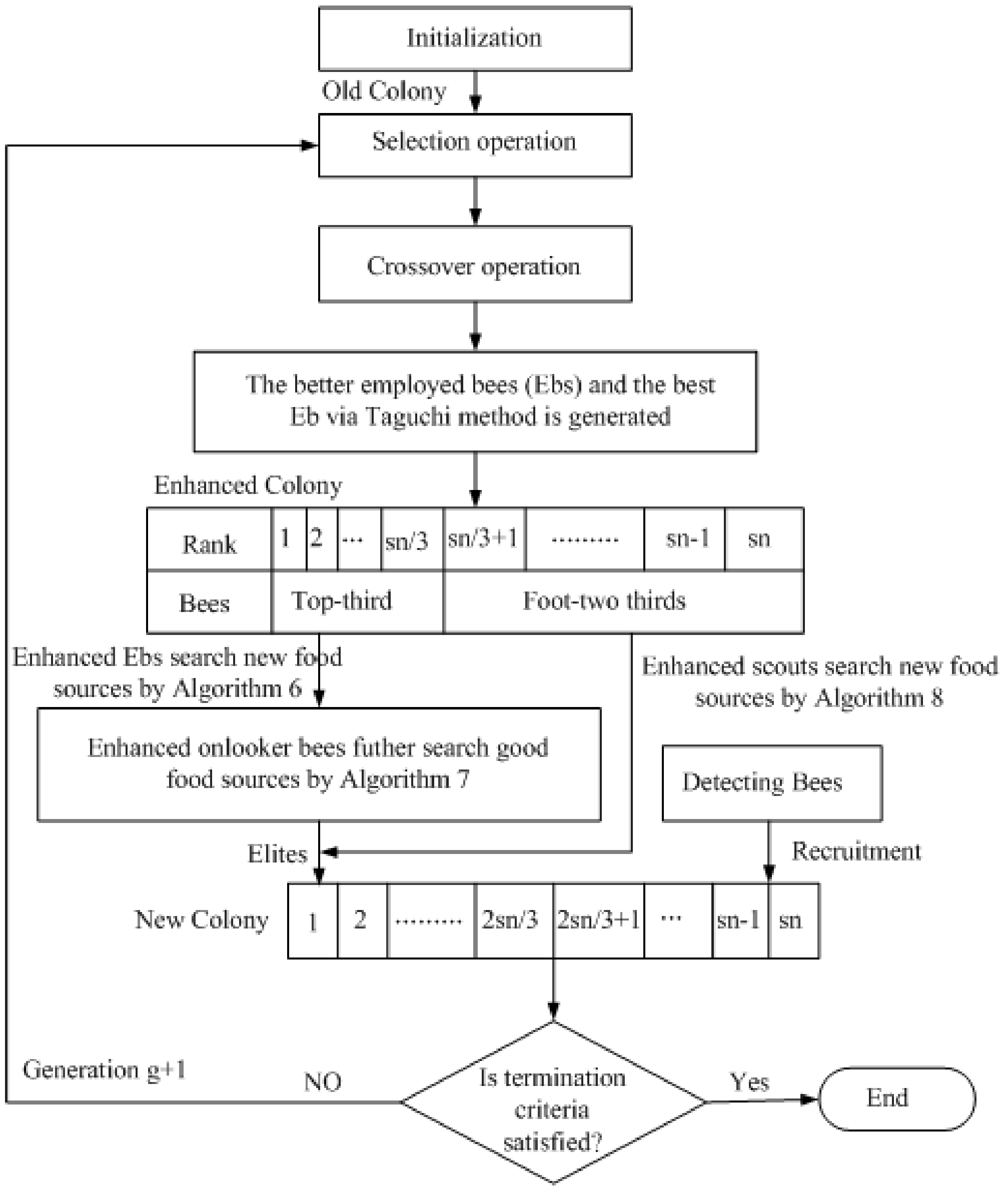

Initialization
Old Colony
Selection operation
Crossover operation
The better employed bees (Ebs) and the best Eb via Taguchi method is generated
Enhanced Colony
Rank
1
2
···
sn/3
sn/3+1
·········
sn-1
sn
Bees
Top-third
Foot-two thirds
Enhanced Ebs search new food sources by Algorithm 6
Enhanced scouts search new food sources by Algorithm 8
Enhanced onlooker bees futher search good food sources by Algorithm 7
Detecting Bees
Elites
Recruitment
New Colony
1
2
·········
2sn/3
2sn/3+1
···
sn-1
sn
Generation g+1
NO
Is termination criteria satisfied?
Yes
End

FIGURE 1. Flow chart for the HTCABC algorithm

Proceed to step of evaluation of each bee until a stopping criterion is met, usually a maximal generation or finding an ideal optimum set by the user. Figure 1 illustrates the flow chart for the HTCABC algorithms.

4. **Experimental Study and Results Analysis.**

4.1. **Experiments.** The experiments in the following sub-section illustrate a comparison of the HTCABC algorithm with different ABC algorithms. The different versions of ABC algorithms including ABC, CABC3, GABC, RABC, and IABC are examined. In addition, the mean and standard deviation of ABC, CABC3, GABC, RABC, and IABC are obtained from [11,13-16] directly. In order to evaluate the performance of the HT-CABC algorithm, including 20 global optimization problems collected from [13-16,32] are presented in Table 1 and Table 2.

Many different kinds of issues such as unimodal, multimodal, separable, non-separable and multidimensional problems are included. To obtain good results for these functions, the search strategy must efficiently comprise the exploratory and exploitative parts.

Test strategies and metrics: Considering the common test strategies in the literature [11,13-16,22], we compare the performances of ABC, CABC3, GABC, RABC, and IABC using the metrics below.

(a) According to the termination criterion, the number of successes converges to the global optimum (NS) at a total number of runs. Generally, the termination criterion achieves and stops the evolution when the algorithm is equal to the optimal value or finds a function error value less than a given value (e-6 $\sim$ e-200) before achieving the maximum number of generations. To decrease the effect of the stochastic nature of the algorithms on those metrics, the outcome of each benchmark is the average of 30 $\sim$ 100 runs.

(b) Average number of evaluations (AE): We compare the convergence speed by measuring the average number of evaluations to the optimal value. This definition means that a smaller AE has a higher convergence speed. The AE is defined as follows:

$$AE = \frac{\text{Total number of evaluation}}{\text{Total number runs}} \tag{19}$$

(c) Average function values of the best solutions (AV): We define the average function values of the best solutions by these algorithms in total number of runs to compare the accuracy of the solution gained. The AV is defined as follows:

$$AV = \frac{\text{Function values of the best solutions}}{\text{Total number runs}} \tag{20}$$

(d) Standard deviation of the best values (SD): We define the standard deviation of the best values to compare the accuracy of the solution gained.

(e) Success rate (SR): We define the success rate as measuring the algorithm successfully to reach the NS for each benchmark. The SR is defined as follows:

$$SR = \frac{\text{NS}}{\text{Total number runs}} \tag{21}$$

4.1.1. *Study 1: comparison of ABC and HTCABC.* ABC Settings: In [11], the ABC algorithm produces the best performance among the DE, PSO, and EA algorithms [33]. Moreover [11] has been run with different colony sizes (20, 40 and 100) and limit values ($0.1 \times n_e \times D$, $0.5 \times n_e \times D$, $n_e \times D$, and without scout. n_e: employed bee number, D: dimension of the problem). As mentioned before, in this study, the ABC algorithm of the best colony sizes and limit values for the same function is selected by the average of the best function values compared with the HTCABC algorithm. For f_1, the colony sizes = 100, the limit value = $n_e \times D$, and $n_e = n_o = 50$ are selected. For f_2, the colony sizes = 20, the limit value = $0.5 \times n_e \times D$, and $n_e = n_o = 10$ are selected. For f_3, the colony sizes = 40, the limit value = $n_e \times D$, and $n_e = n_o = 20$ are selected. For f_4, the colony sizes = 20, the limit value = $n_e \times D$, and $n_e = n_o = 10$ are selected. For f_5, the colony sizes = 40, without a scout, and $n_e = n_o = 20$ is selected. The number of onlooker bees (n_o) taken was equal to the number of employed bees (n_e), which were 50% of the colony and the number of scout bees was selected to be at most one for each cycle. The maximum number of cycles was taken as 200 for f_1, 1000 for f_2, 5000 for f_4, 2500 for f_3 and f_5 in order to equalize the total number of evaluations as 20,000 for the first two functions and 100,000 for the other three functions, respectively.

HTCABC Settings: The HTCABC parameters were chosen as $sn = 60$, $P_c = 0.7$, $\lambda = 1.5$, $\rho = 4.5$, $\mu = 4$, and $\delta = 0.3$. The maximum number of cycles was taken as 50 for f_1, f_2, f_3, f_4, and 1,600 for f_5, and the total number of evaluation as 3,000 for the first four functions and 96,000 for the other functions, respectively. Each of the benchmark ranges and dimensions were the same as in the ABC algorithm.

Each of the benchmarks was repeated for 30 runs, and the average function values of the best solutions found were recorded when the algorithm achieved the maximum number of generations. The AV, SD, and AE of the function values obtained by ABC and HTCABC are given in Table 3. Results analysis was seen from the results presented in Table 3.

Table 1. Benchmarks function used in experiments. R: Range, D: Dimension, C: Characteristic, U: Unimodal, M: Multimodal, S: Separable, N: Non-Separable.

No.	C	Function	Formulation	Min. value	R and D of Algorithms
1	MN	Schaffer	$0.5+\frac{\sin^2\left(\sqrt{x_1^2+x_2^2}\right)-0.5}{\left(1+0.001\left(x_1^2+x_2^2\right)\right)^2}$	0	ABC: $[-100,100]$, 2 GABC: $[-100,100]$, 3
2	US	Sphere	$\sum_{i=1}^{D} x_i^2$	0	ABC: $[-100,100]$, 5 GABC: $[-100,100]$, 60 RABC: $[-100,100]$, 30 IABC: $[-100,100]$, 30
3	MN	Griewank	$\frac{1}{4000}\left(\sum_{i=1}^{D}(x_i-100)^2\right)$ $-\left(\prod_{i=1}^{D}\cos\left(\frac{x_i-100}{\sqrt{i}}\right)\right)+1$	0	ABC: $[-600,600]$, 50 CABC: $[-50,50]$, 10 GABC: $[-100,100]$, 60 RABC: $[-600,600]$, 30 IABC: $[-600,600]$, 30
4	MS	Rastrigin	$\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	0	ABC: $[-5.12,5.12]$, 50 CABC: $[-50,50]$, 10 GABC: $[-5.12,5.12]$, 60 RABC: $[-5.12,5.12]$, 30 IABC: $[-5.12,5.12]$, 30
5	UN	Rosenbrock	$\sum_{i=1}^{D-1} 100(x_{i+1}-x_i^2)^2+(x_i-1)^2$	0	ABC: $[-50,50]$, 50 CABC: $[-2.48,2.48]$, 2 GABC: $[-50,50]$, 3 RABC: $[-30,30]$, 100 IABC: $[-30,30]$, 100
6	MN	Ackley	$20+e-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)$ $-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)$	0	GABC: $[-32.768,32.768]$, 60 RABC: $[-32,32]$, 30 IABC: $[-32,32]$, 30
7	MS	Bohachevsky 1	$x_1^2+2x_2^2-0.3\cos(3\pi x_1)$ $-0.4\cos(4\pi x_2)+0.7$	0	RABC: $[-100,100]$, 2
8	MN	Bohachevsky 2	$x_1^2+2x_2^2$ $-0.3\cos(3\pi x_1)\cos(4\pi x_2)+0.3$	0	RABC: $[-100,100]$, 2
9	MN	Branin RCOS	$\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2$ $+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	$5/(4\pi)$	RABC: $[-5,15]$, 2
10	MN	Goldstein and Price	$\left[1+(x_1+x_2+1)^2(19-14x_1\right.$ $\left.+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]$ $\times\left[30+(2x_1-3x_2)^2(18-32x_1\right.$ $\left.+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	33	RABC: $[-2,2]$, 2

In all the function, the HTCABC algorithm produced the better performance and the lowest total number of evaluations than ABC. In addition, the HTCABC algorithm does not have to do any parameter selecting or testing including colony sizes, limit values,

Table 2. Benchmarks function used in experiments. R: Range, D: Dimension, C: Characteristic, U: Unimodal, M: Multimodal, S: Separable, N: Non-Separable.

No.	C	Function	Formulation	Min. value	R and D of Algorithms
11	MN	Six-hump Camel Back	$4x_1^2 - 2.1x_1^4 + 1/3x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	-1.03163	RABC: $[-5, 5]$, 2
12	MN	Shekel's Foxholes	$\left(\frac{1}{500}\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right)^{-1}$ The constants a_{ij} can be seen in [15].	0.998004	RABC: $[-65.536, 65.536]$, 2
13	MS	Shubert	$\left(\sum_{i=1}^{5} i\cos((i+1)x_1+i)\right)\left(\sum_{i=1}^{5} i\cos((i+1)x_2+i)\right)$	-186.7309	RABC: $[-10, 10]$, 2
14	US	Step	$\sum_{i=1}^{D}(x_i+0.5)^2$	0	IABC: $[-100, 100]$, 30
15	UN	Schwefel 2.22	$\sum_{i=1}^{D}\lvert x_i\rvert + \prod_{i=1}^{D}\lvert x_i\rvert$	0	IABC: $[-10, 10]$, 30
16	UN	Schwefel 1.2	$\sum_{i=1}^{D}\left(\sum_{j=1}^{D} x_i\right)^2$	0	IABC: $[-100, 100]$, 30
17	UN	Schwefel 2.21	$\max_i\{\lvert x_i\rvert, 1 \le i \le D\}$	0	IABC: $[-100, 100]$, 30
18	MS	Schwefel	$-\sum_{i=1}^{D}\left(x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	-12569.5	IABC: $[-500, 500]$, 30
19	MN	Weiestrass	$\sum_{i=1}^{D}\left\{\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right]\right\}$ $-D\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right]$ $a=0.5,\ b=3, k_{\max}=20$	0	IABC: $[-0.5, 0.5]$, 30
20	MS	Noncontinuous Rastrigin	$10D+\sum_{i=1}^{D}\left(y_i^2-10\cos(2\pi y_i)\right)$ $y_i = \begin{cases} x_i, & \lvert x_i\rvert < 1/2 \\ round(2x_i)/2, & \lvert x_i\rvert \ge 1/2 \end{cases}$	0	IABC: $[-5.12, 5.12]$, 30

Table 3. AV, SD, and AE comparisons of ABC and HTCABC

f		ABC [11]	HTCABC
1	AV	5.80E-09	0
	SD	1.78E-08	0
	AE	20,000	3,000
2	AV	3.52E-17	0
	SD	1.21E-17	0
	AE	20,000	3,000
3	AV	3.57E-12	0
	SD	1.36E-11	0
	AE	100,000	3,000
4	AV	3.58E-14	0
	SD	1.84E-13	0
	AE	100,000	3,000
5	AV	8.9114E-03	9.3813E-04
	SD	4.3152E-02	1.3657E-03
	AE	100,000	96,000

number of employed bees, or number of onlooker bees. Therefore, the HTCABC algorithm is simple, robust, and has a rapid convergence rate.

4.1.2. *Study 2: comparison of CABC3 and HTCABC.* CABC3 Settings: In [13], on the Gauss map, the CABC3 algorithm produces the best performance among the ABC, CABC1, and CABC2 algorithms when the algorithm finds a function error value less than a given value (e-6) before achieving the maximum number of generations (500). As mentioned before, in this study, the CABC3 algorithm is selected to compare with the HTCABC algorithm.

In this study, the parameters of the CABC3 algorithm is identical to [13]. In the CABC3 algorithm, the Gauss map is chosen, the colony size is 20, the limit parameter is 40, and the maximum number of generations is 500. So, the total objective function evaluation number is 10,000.

HTCABC Settings: All the HTCABC parameters were chosen to be the same as in the Section 4.1.1. The maximum number of cycles was taken as 50, and the total number of evaluations as 3,000. Each of the benchmark ranges and dimensions were the same as in the CABC3 algorithm.

Each of the benchmarks was repeated for 100 runs, and the NS was recorded when the algorithm found a function error value less than a given value (e-6) before achieving the maximum number of generations. The NS and SR of the function values obtained by CABC3 and HTCABC are given in Table 4.

Results analysis was seen from the results presented in Table 4. In all the functions, the HTCABC algorithm has a more powerful global search capability than CABC3 and there are no any chaotic maps selected. Therefore, the HTCABC algorithm is robust and possesses the global search capability.

Table 4. NR and SR comparisons of CABC3 and HTCABC

f		CABC3 [13]	HTCABC
3	NS	26	100
	SR	0.26	1
4	NS	86	100
	SR	0.86	1
5	NS	6	100
	SR	0.06	1

4.1.3. *Study 3: comparison of GABC and HTCABC.* GABC Settings: In [14], the GABC algorithm was tested with different the parameters C ($C = 0.5, 1.0$, or 1.5) and were all superior to the ABC algorithms. And, as a whole, the GABC algorithm with $C = 1.5$ had the best performance among the tested algorithms.

In this study, the parameters of the GABC algorithm are identical to [14], where the parameter $C = 1.5$, the colony sizes = 80, the limit value = $n_e \times D$, and $n_e = n_o = 40$ are selected to compare with the HTCABC algorithm. The maximum number of cycles was taken as 5000 and the total number of evaluation as 400,000.

HTCABC Settings: All the HTCABC parameters were chosen to be the same as in the Section 4.1.1. The maximum number of cycles was taken as 50 for f_1, f_2, f_3, f_4, and f_6, 3,000 for f_5, and the total number of evaluations as 18,000 for the f_5 and 3,000 for the other functions, respectively. Each of the benchmark ranges and dimensions were the same as in the GABC algorithm.

Each of the benchmarks was repeated for 30 runs, and the average function values of the best solutions found were recorded when the algorithm found a function error value less than a given value (e-20) before achieving the maximum number of generations. The AV SD, and AE of the function values obtained by GABC and HTCABC are given in Table 5. Results analysis as seen from the results presented in Table 5. In all the functions, the HTCABC algorithm produced a better performance and the lowest total number of evaluations than GABC. In addition, the HTCABC algorithm does not have to do any parameter selecting or testing including of colony sizes, limit values, number of employed bees, number of onlooker bees, or parameter C. Therefore, the HTCABC algorithm can just converge to the global optimal solution quickly and the optimal solution is the most accurate.

4.1.4. *Study 4: comparison of RABC and HTCABC.* In [15], the RABC algorithm is compared with ABC and other algorithms; RABC is faster, robust and accurate. Therefore, in this study, the RABC algorithm is selected to compare with the HTCABC algorithm.

RABC Settings: In this study, the parameters of the RABC algorithm are identical to [15], where the three control parameters $a = 20$, $n_l = 15$, and $n_c = 5n$, the colony sizes is 50, the limit values $n_e \times D$ and $n_e = n_o = 25$ are selected to compare with the HTCABC algorithm. The maximum number of cycles is taken as 3,000 for f_2, 40,000 for f_5, 1,000 for f_3, f_4 and f_6, 6,000 for $f_7 \sim f_{13}$, and the total number of evaluations is chosen as 150,000 for f_2, 2,000,000 for f_5, 50,000 for f_3, f_4, and f_6 and 300,000 for other functions, respectively.

HTCABC Settings: All the HTCABC parameters are chosen to be the same as those in Section 4.1.1. The maximum number of cycles is taken as 50 for f_2, f_3, f_4, f_6, f_7, f_8, and f_{12}, 500 for f_{11}, 34,000 for f_5, 3,000 for f_9, f_{10}, and f_{13}, and the total number of evaluations is 30,000 for f_{11}, 2,040,000 for f_5, 180,000 for f_9, f_{10}, and f_{13}, and 3,000 for other functions, respectively. Each of the benchmark ranges and dimensions are the same as those in the RABC algorithm.

Each of the benchmarks is repeated for 50 runs, and the average function values of the best solutions are recorded when the algorithm figures out a function error value less than an extremely small value (e-200) before achieving the maximum number of generations. The AV, SD, and AE of the function values obtained by RABC and HTCABC are given in Table 6. One can find that HTCABC performs much better than the RABC on almost all the functions, while the RABC performs better than the HTCABC on f_5. However, the HTCABC can also find closer-to-optimal solution on the high-dimension function f_5. Besides, the HTCABC does not have to do any control parameter selection or trial including colony sizes, limit values, number of employed bees, number of onlooker bees, or parameters a, n_l and n_c. Therefore, the HTCABC possesses superior performance in convergence speed and robustness in comparison with the RABC.

4.1.5. *Study 5: comparison of IABC and HTCABC.* In [16], the IABC algorithm is compared with CABC, GABC and RABC, where IABC is more efficient than these ABCs. In this study, the IABC algorithm is also selected to compare with the HTCABC algorithm.

IABC Settings: In this study, the parameters of the IABC algorithm are identical to those in [16], where the control parameters $f_2 : M = 25$, $f_3 : M = 3$, $f_4 : M = 1$, $f_5 : M = 1$, $f_6 : M = 10$, the colony size is 50, the selective probability $p = 0.25$, the limit values $n_e \times D$ and $n_e = n_o = 25$ are selected to compare with the HTCABC algorithm. The maximum number of cycles is taken as 40,000 for f_5 and 1,000 for other functions and the total number of evaluation is 2,000,000 for f_5 and 50,000 for other functions, respectively.

HTCABC Settings: All the HTCABC parameters are chosen to be the same as those in Section 4.1.1. The maximum number of cycles is taken as 50 for f_2, f_3, f_4, f_6, f_{14}, f_{15}, f_{16}, f_{17}, f_{19} and f_{20}, 800 for f_{18}, 5,000 for f_5 and the total number of evaluations is 300,000 for f_5, 48,000 for f_{18}, and 3,000 for other functions, respectively. Each of the benchmark ranges and dimensions are the same as those in the IABC algorithm.

Each of the benchmarks is performed 50 runs, and the average function values of the best solutions are recorded when the algorithm figures out a function error value less than an extremely small value (e-200) before achieving the maximum number of generations.

Table 5. AV, SD, and AE comparisons of GABC and HTCABC

f		GABC [12]	HTCABC
1	AV	1.850371E-18	0
	SD	1.031E-17	0
	AE	400,000	272
2	AV	1.433867E-15	0
	SD	1.375E-16	0
	AE	400,000	432
3	AV	7.549516E-16	0
	SD	4.127E-16	0
	AE	400,000	404
4	AV	3.524291E-13	0
	SD	1.243E-13	0
	AE	400,000	314
5	AV	2.659E-03	2.990E-05
	SD	2.220E-03	6.770E-05
	AE	400,000	18,000
6	AV	1.000088E-13	8.88180E-16
	SD	6.089E-15	4.012E-31
	AE	400,000	318

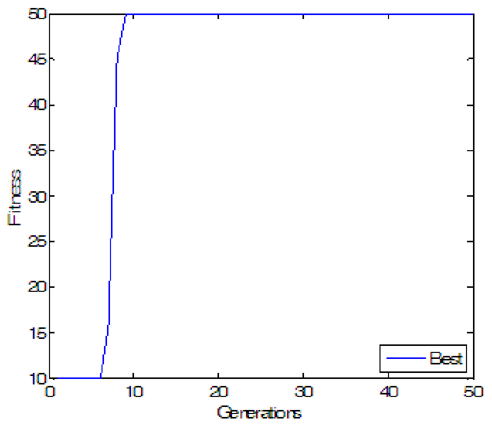
Table 6. AV, SD, and AE comparisons of RABC and HTCABC

f	RABC [15,16]			HTCABC		
	AV	SD	AE	AV	SD	AE
2	9.1E-61	2.1E-60	150,000	0	0	426
3	8.7E-08	2.1E-08	50,000	0	0	382
4	2.3E-02	5.1E-01	50,000	0	0	316
5	6.40E-23	3.80E-22	2,000,000	6.83E-09	5.39E-07	2,040,000
6	9.6E-07	8.3E-07	50,000	8.88E-16	4.01E-31	332
7	0	0	300,000	0	0	276
8	0	0	300,000	0	0	292
9	0.397887	3.3650E-16	300,000	0.397887	1.1292E-16	119,340
10	3.000000	8.9720E-16	300,000	3.000000	4.5168E-16	121,168
11	−1.03163	2.3093E-16	300,000	−1.03163	1.7752E-16	16,508
12	0.998004	5.6075E-16	300,000	0.998004	3.3876E-16	2,184
13	−186.73	7.4204E-14	300,000	−186.73	2.8907E-14	123,998

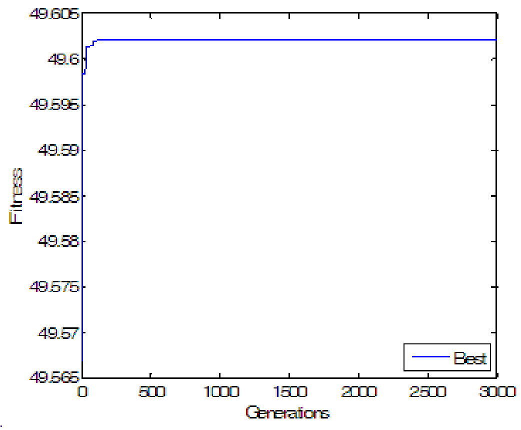
The AV, SD, and AE of the function values obtained by IABC and HTCABC are given in Table 7, where the HTCABC algorithm produces a better performance and the lowest total number of evaluations than IABC.

In addition, the HTCABC algorithm does not have to do any parameter selecting or testing for colony sizes, limit values, number of employed bees, number of onlooker bees, or parameters M and p. Therefore, the HTCABC algorithm can speed up convergence rate and achieve the accuracy results simultaneously.

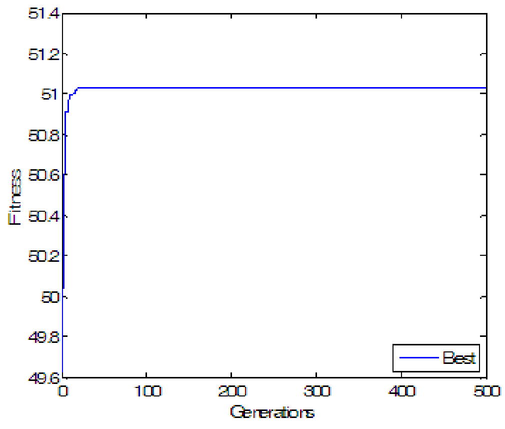
In addition, the best fitness versus generations of 6 functions (f_3, f_9, f_{11}, f_{12}, f_{19}, and f_{20}) are plotted in Figure 2. One can tell from these curves that HTCABC algorithm makes the solution converge to the optimal one sufficiently fast.



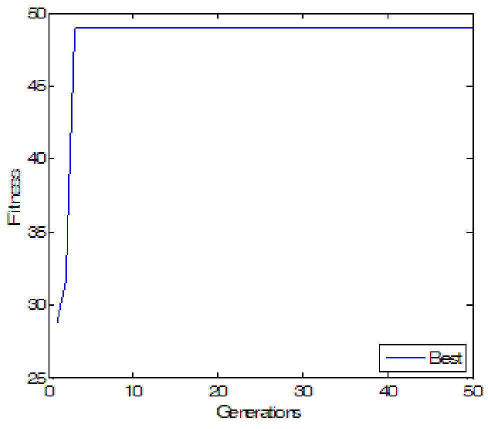
(a)



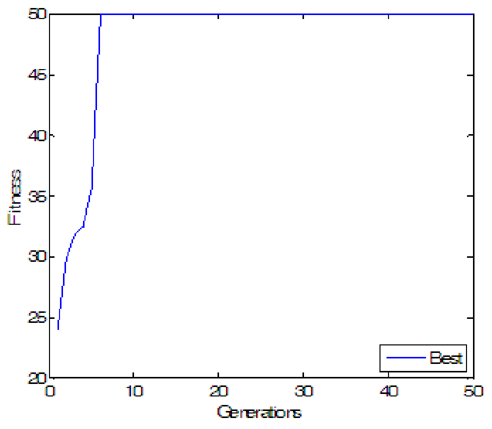
(b)



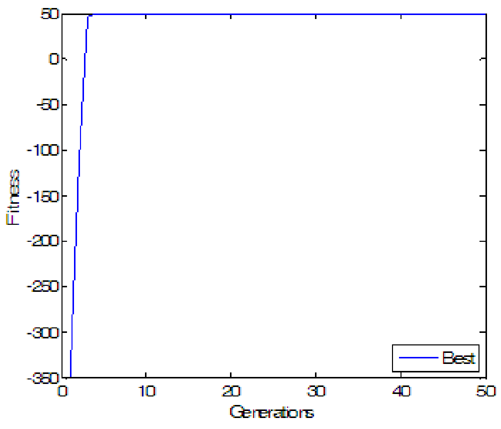
(c)



(d)



(e)



(f)

FIGURE 2. Sample graphs (the best fitness versus generations) for the HT-CABC algorithm: (a) f_3 Griewank, (b) f_9 Branin, (c) f_{11} Six-hump Camel Back, (d) f_{12} Shekel's Foxholes, (e) f_{19} Weiestrass and (f) f_{20} Noncontinuous Rastrigin

Table 7. AV, SD, and AE comparisons of IABC and HTCABC

f	IABC [16]			HTCABC		
	AV	SD	AE	AV	SD	AE
2	6.75E-57	1.72E-56	50,000	0	0	426
3	0	0	50,000	0	0	382
4	0	0	50,000	0	0	316
5	4.75E-03	4.22E-02	2,000,000	7.35E-08	6.49E-07	300,000
6	3.87E-14	8.52E-15	50,000	8.88E-16	4.01E-31	332
14	0	0	50,000	0	0	542
15	4.96E-30	1.56E-29	50,000	0	0	208
16	6.47E-03	9.25E-03	50,000	0	0	242
17	6.63E-03	1.92E-02	50,000	0	0	176
18	−12569.5	3.92E-12	50,000	−12569.5	2.55E-12	43,062
19	6.90E-04	1.39E-03	50,000	0	0	280
20	4.50e-15	5.79E-15	50,000	0	0	298

4.2. **Simulations on Lorenz system.** In this section, we evaluate the performance of the proposed HTCABC algorithm via the Lorenz system. The Lorenz system is described as follows:

$$\begin{cases} \dot{x}_1 = a_1 \cdot (x_2 - x_1) \\ \dot{x}_2 = a_2 \cdot x_1 - x_1 \cdot x_3 - x_2 \\ \dot{x}_3 = x_1 \cdot x_2 - a_3 \cdot x_3 \end{cases} \tag{22}$$

where $a_1 = 10$, $a_2 = 28$, and $a_3 = 8/3$.

The state X_0 of the Lorenz system is chosen as $X_0 = [x_1(0)\ x_2(0)\ x_3(0)]^T = [0.1\ 0.1\ 0]^T$. The length of data L is set to be 300. The performance index is defined by

$$J = \frac{1}{L} \sum_{d=1}^{L} \left\| X_d - \hat{X}_d \right\| \tag{23}$$

where L denotes the length of the data used for parameter estimation, and X_d and $\hat{X}_d$ are real and estimated values at time d, respectively.

The maximum generation number and the population size are 100 and 60, respectively. The searching spaces of the parameters a_1, a_2, and a_3 belong to the intervals $[9, 11]$, $[20, 30]$, $[2, 3]$, respectively [34]. In addition, because parameter identification is a multi-dimensional optimization problem, these three parameters in Lorenz system are unknown. HTCABC Settings: All the HTCABC parameters are chosen to be the same as those in Section 4.1.1. The algorithm is executed for 20 runs, and the statistical results obtained by HTCABC is listed in Table 8, where all the results obtained by the HTCABC are consistent with the true values. Therefore, the HTCABC algorithm is effective for parameter identification for a chaotic Lorenz system. It is obvious from Figure 3 that the convergence speed of the HTCABC is very fast and all the parameters a_1, a_2, and a_3 converge to the matching true values rapidly (less than 30 generations). So the HTCABC is really efficient for parameter identification of a chaotic system.

4.2.1. *Effect of noise.* Suppose Gaussian white noise in $[-0.1, 0.1]$ is added to every state variable. Setting the population size as 60, we make HTCABC 20 runs. From Table 9 and Figure 4, it is clear that the HTCABC still can achieve good results when noises are present. The error between the true value and estimated value is less than e-2 level. Moreover, the HTCMIABC is still of fast convergence property, and all the parameters are approaching to their true values within 40 generations.

TABLE 8. Simulation results for three-dimensional parameter identification

	Best	Average	Worst
a	10.000000	10.000000	10.000000
b	28.000000	28.000000	28.000000
c	2.666667	2.666667	2.666667
J	0	0	0

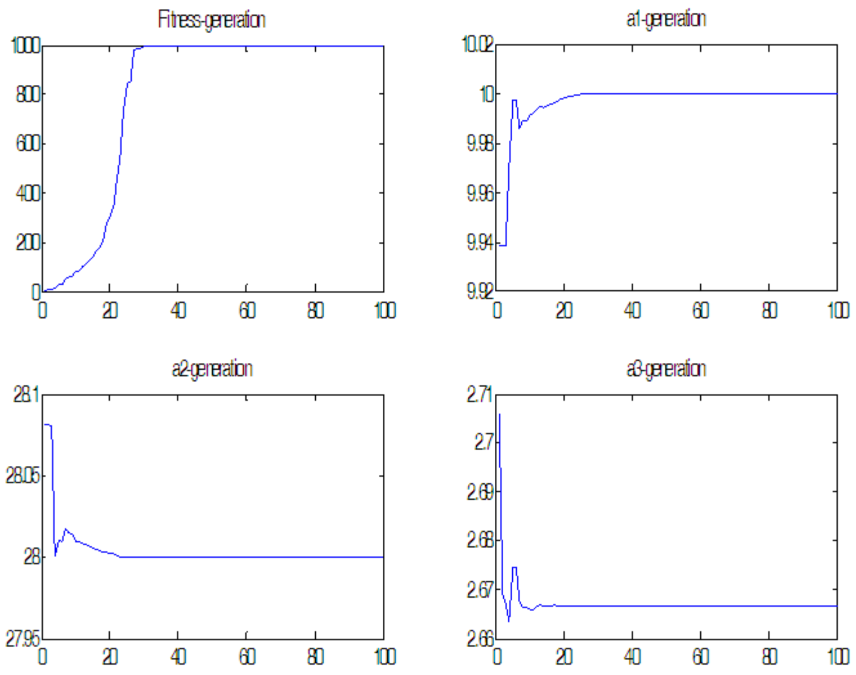


FIGURE 3. Evolving process of parameter identification of Lorenz system

TABLE 9. Simulation results for three-dimensional parameter identification with noise

	Best	Average	Worst
a_1	10.00086	10.00053	10.00209
a_2	27.99689	27.99896	27.99790
a_3	2.66640	2.66683	2.66727
J	7.40E-2	7.68E-2	7.87E-2

4.2.2. *Online chaotic system parameter identification.* To show the effectiveness of the proposed algorithm in tracking a time-varying parameter, some sudden changes are also applied to the system parameters. The population size is chosen as 60 and the maximum generation number is selected as 150. To form a data set, the sampling frequency is set to 100 kHz such that 100 pairs of data are sampled within 0.0001 ms in each period. The ranges of the parameters a_1, a_2, and a_3 are changing to the intervals $[4, 14]$, $[24, 90]$, $[1.5, 4.5]$, respectively [35]. The changes of the system parameters are in three different parts. In part (I), the nominal parameters are given. In part (II), a_1 is changed from 10 to 13 while the other parameters are kept unchanged. In part (III), a_1 is kept unchanged whereas a_2 is varied from 28 to 30 and a_3 is varied from 2.6667 to 3, respectively [36].

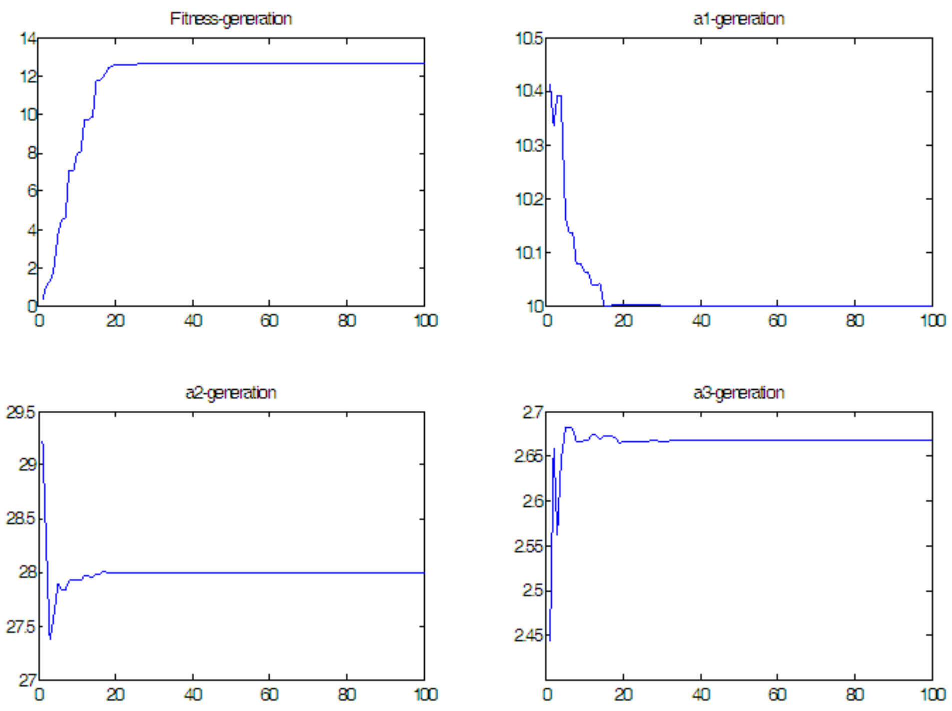


FIGURE 4. Evolving process of parameter identification of Lorenz system (with noise)

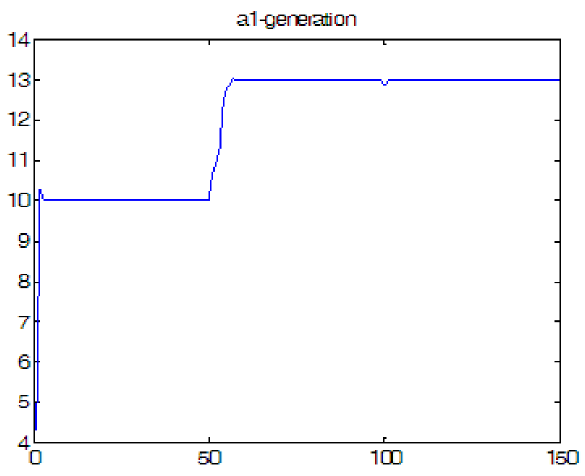


FIGURE 5. HTCABC process for online identification of a_1

Figures 5-7 demonstrate that the proposed algorithm can track any change in parameters. These three simulation results are also tabulated in Table 10.

5. **Conclusions.**

The HTCABC has been presented in this paper; the hybrid algorithm, combines the advantages of the Taguchi method, crossover operation, adaptive bound chaos search algorithm, and ABC algorithm. The Taguchi method is a robust design approach, which draws on many ideas, from statistical experimental design to plan experiments for obtaining dependable information about variables, so it can achieve the bee colony distribution closest to the target and produce robust results. So the Taguchi

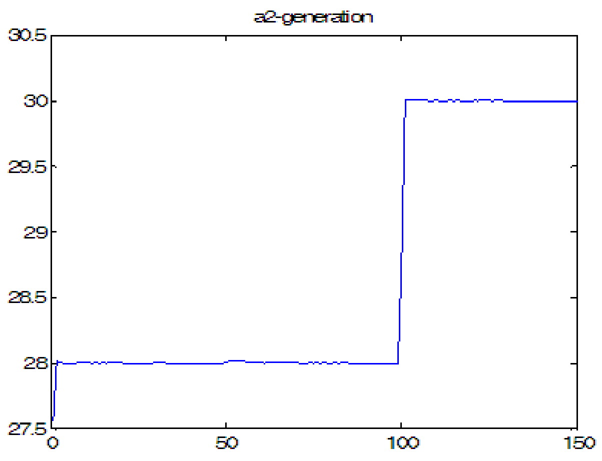


Figure 6. HTCABC process for online identification of a_2

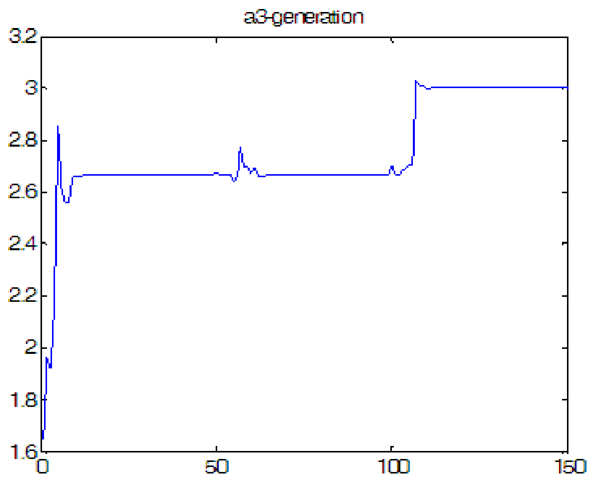


Figure 7. HTCABC process for online identification of a_3

Table 10. Simulation results of online identification of Lorenz parameter by HTCABC

	Part (I)		Part (II)		Part (III)	
	Actual	Predicted	Actual	Predicted	Actual	Predicted
a_1	10.0000	10.0000	13.0000	13.0000	13.0000	13.0000
a_2	28.0000	28.0000	28.0000	28.0000	30.0000	30.0000
a_3	2.6667	2.6667	2.6667	2.6667	3.0000	3.0000

method is incorporated into the crossover operation to select better bees to enhance the global search and convergence capability of the ABC algorithm. The Taguchi method has also been utilized by incorporating the information from the best global solution into the solution search equation of the ABC algorithm to improve the capability of the exploration and exploitation. Moreover, the adaptive bound chaos search algorithm, elitist mechanism, and recruitment of new food sources are adopted to the ABC algorithm, which can have a rapid convergence rate and maintain the diversity of the colony so as to escape from local optima. Additionally, there is no complex parameter selecting in the algorithm design. Therefore, the proposed HTCABC possesses the merits of global exploration, quick convergence, and robustness. These 20 benchmark experiments illustrate that the proposed HTCABC can find much more accurate optimal solutions and be more efficient than other ABCs. From the identification problem of a chaotic system, the proposed HTCABC algorithm can correctly and effectively identify the system parameters in the presence of noises and parameter variations.

Acknowledgment. The support of this work in part by the National Science Council of Taiwan under NSC98-2221-E-006-212-MY3 is gratefully acknowledged.

REFERENCES

[1] T. K. Liu, J. T. Tsai and J. H. Chou, Improved genetic algorithm for jop-shop scheduling problem, *Int. J. Adv. Manuf. Technol.*, vol.27, no.9-10, pp.1021-1029, 2006.

[2] P. Z. Lin, W. Y. Wang, T. T. Lee and C. H. Wang, On-line genetic algorithm-based fuzzy-neural sliding mode controller using improved adaptive bound reduced-form genetic algorithm, *International Journal of Systems Science*, vol.40, pp.571-585, 2009.

[3] J. Kennedy and R. Eberhart, Particle swarm optimization, *Proc. of IEEE International Conference on Neural Networks*, vol.4, pp.1942-1948, 1995.

[4] G. Ciuprina, D. Ioan and I. Munteanu, Use of intelligent-particle swarm optimization in electromagnetic, *IEEE Trans. Magn.*, vol.38, no.2, pp.1037-1040, 2002.

[5] A. Ratnaweera, S. K. Halgamuge and H. C. Watson, Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Trans. on Evolutionary Computation*, vol.8, no.3, pp.240-255, 2004.

[6] K. V. Price, R. M. Storn and J. A. Lampinen, Differential evolution: A practical approach to global optimization, *Springer Natural Computing Series*, 2005.

[7] C. S. Zhang, D. T. Ouyang and J. X. Ning, An artificial bee colony approach for clustering, *Expert Systems with Applications*, vol.37, no.7, pp.4761-4767, 2010.

[8] P.-W. Tsai, J.-S. Pan, B.-Y. Liao and S.-C. Chu, Enhanced artificial bee colony optimization, *International Journal of Innovative Computing, Information and Control*, vol.5, no.12(B), pp.5081-5092, 2009.

[9] N. Karaboga, A new design method based on artificial bee colony algorithm for digital IIR filters, *Journal of the Franklin Institute*, vol.346, pp.328-348, 2009.

[10] D. Karaboga and 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, 2007.

[11] D. Karaboga and B. Basturk, On the performance of artificial bee colony (ABC) algorithm, *Applied Soft Computing*, vol.8, no.1, pp.687-697, 2008.

[12] D. Karaboga and B. Akay, A comparative study of artificial bee colony algorithm, *Applied Mathematics and Computer*, vol.214, pp.108-312, 2009.

[13] B. Alatas, Chaotic bee colony algorithms for global numerical optimization, *Expert Systems with Applications*, vol.37, no.8, pp.5682-5687, 2010.

[14] G. P. Zhu and S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, *Applied Mathematics and Computation*, vol.217, no.6, pp.3166-3173, 2010.

[15] F. Kang, J. J. Li and Z. Y. Ma, Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions, *Information Sciences*, vol.181, no.16, pp.3508-3531, 2011.

[16] W. F. Gao and S. Y. Liu, Improved artificial bee colony algorithm for global optimization, *Information Processing Letters*, vol.111, no.17, pp.871-882, 2011.

[17] J. T. Tsai, T. K. Liu and J. H. Chou, Hybrid Taguchi-genetic algorithm for global numerical optimization, *IEEE Transactions on Evolutionary Computation*, vol.8, no.4, pp.365-377, 2004.

[18] J. T. Tsai, T. K. Liu, W. H. Ho and J. H. Cho, An improved genetic algorithm for job-shop scheduling problems using Taguchi-based crossover, *International Journal of Advanced Manufacturing Technology*, vol.38, no.9-10, pp.987-994, 2008.

[19] V. V. Kumar, M. K. Pandey, M. K. Tiwari and D. Ben-Arieh, Simultaneous optimization of parts and operations sequences in SSMS: A chaos embedded Taguchi particle swarm optimization approach, *Journal of Intelligent Manufacturing*, vol.21, no.4, pp.335-353, 2010.

[20] G. C. Liao and T. P. Tsao, Using chaos search immune genetic and fuzzy system for short-term unit commitment algorithm, *International Journal of Electrical Power & Energy Systems*, vol.28, pp.1-12, 2006.

[21] H. He, F. Qian and W. Du, A chaotic immune algorithm with fuzzy adaptive parameters, *Asia-Pacific Journal of Chemical Engineering*, vol.3, pp.695-705, 2008.

[22] X. Q. Zuo and Y. S. Fan, A chaos search immune algorithm with its application to neuro-fuzzy controller design, *Chaos Solitons & Fractals*, vol.30, pp.94-109, 2006.

[23] S. H. Park, *Robust Design and Analysis for Quality Engineer*, Chapman & Hall, London, U.K., 1996.

[24] L. N. de Castro and F. J. Von Zuben, Learning and optimization using the clonal selection principle, *IEEE Transactions on Evolutionary Computation*, vol.6, pp.239-251, 2002.

[25] B. Li and W. S. Jiang, Optimizing complex functions by chaos search, *Cybernetics and Systems*, vol.29, pp.409-419, 1998.

[26] D. Karaboga, An idea based on honey bee swarm for numerical optimization, *Tech. Rep. TR06*, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.

[27] Z. Michalewice, *Genetic Algorithm + Data Structures = Evolution Programs*, Spinger-Verlag, Berlin, Germany, 1996.

[28] M. S. Phadke, *Quality Engineering Using Robust Design*, Prentice-Hall, Englewood Cliffs, NJ, 1989.

[29] P. J. Ross, *Taguchi Techniques for Quality Engineering*, McGraw-Hill, Singapore, 1989.

[30] D. C. Montgomery, *Design and Analysis of Experiment*, Wiley, New York, 1991.

[31] G. Taguchi, S. Chowdhury and S. Taguchi, *Robust Engineering*, McGraw-Hill, New York, 2000.

[32] T. Krink, B. Filipic, G. B. Fogel and R. Thomsen, Noisy optimization problem – A particular challenge for differential evolution? *Proc. of Congress on Evolutionary Computation*, Piscataway, NJ, USA, pp.332-339, 2004.

[33] D. E. Goldberg, *Genetic Algorithm in Search, Optimization, and Machine Learning*, Addison-Wesley Pub. Co., 1989.

[34] L. Wang, Y. Xu and L. P. Li, Parameter identification of chaotic systems by hybrid Nelder-Mead simplex search and differential evolution algorithm, *Expert Systems with Application*, vol.38, no.4, pp.3238-3245, 2011.

[35] G. Alvarez, F. Montoya, M. Romera and G. Pastor, Breaking parameter modulated chaotic secure communication system, *Chaos, Solitons and Fractals*, vol.21, pp.783-787, 2004.

[36] H. Modares, A. Alfi and M. M. Fateh, Parameter identification of chaotic dynamic systems through an improved particle swarm optimization, *Expert Systems with Application*, vol.37, no.5, pp.3714-3720, 2010.