An improved AEA algorithm with Harmony Search(HSAEA) and its application in reaction kinetic parameter estimation

Shunlong Ma, Yuehua Dong, Zhixiang Sang, Shaojun Li*

Key Laboratory of Advanced Control and Optimization for Chemical Processes (East China University of Science and Technology), Ministry of Education, Shanghai, 200237, China

1. Introduction

Alopex (Algorithm of Pattern Extraction), firstly proposed by Harth in 1974, is a stochastic and parallel optimization algorithm used to solve combinatorial optimization and pattern matching problems [1]. It has the characteristic of both stochastic and heuristic searching methods. The algorithm is inspired by the influence of previous change of independent variables with respect to the objective function. Meanwhile, with the effect of the process parameter called “temperature”, the algorithm changes gradually from the stochastic search process into a deterministic one, which helps to find the best solution quickly. In recent decades, much attention has been paid to this algorithm. The heuristic idea of Alopex has been successfully applied to the field of engineering optimization. Such as, it was used to make all the interconnection strengths update synchronously and optimize neural network controller [2]. Alopex had also been applied to the sports medicine domain [3].

However, the basic Alopex algorithm is a single point iterative algorithm, namely only one point can be updated every iteration, which makes the algorithm work less efficiently [4]. Xue et al. had used Alopex strategy to improve other evolutionary algorithms. Such as, a hybrid genetic-alopex algorithm (GAA) [5] was proposed to enhance the algorithm to converge at the global optimum. The GAA had been used to optimize waste interception and allocation networks [5], Li et al. developed a PSO-Alopex algorithm by introducing Alopex into particle swarm optimization (PSO) algorithm to train a BP neural network [6]. Xu et al. improved differential algorithm (DE) based on Alopex to overcome the disadvantages of DE [7]. Inspired by the traditional evolutionary algorithm, especially swarm intelligence, Li and Li proposed an Alopex-based evolutionary algorithm (AEA), with combination of the computation method of Alopex and the concept of swarm evolutionary [8]. In AEA, two populations are generated every generation, and then they evolve through Alopex operation. In order to overcome the demerit of basic Alopex that it was somewhat blind to set the annealing temperature artificially, a new strategy about adaptive adjustment of the annealing temperature was proposed. The annealing temperature calculated and changed automatically with the parameters of the evolutionary process. The testing results [8] on benchmark functions showed that the AEA was approximately equal to DE and better than HGA3 (an improved GA algorithm by Deep and Das [9]) and PSO.

In this paper, an improved algorithm (HSAEA) which combines HS and AEA is developed. Experiment results show that the new algorithm is an effective optimizing method. The remainder of the paper is organized as follows. Theoretical backgrounds including AEA algorithm, EDAs and HS algorithm are reviewed in Section 2. Section 3 details the improved algorithm, HSAEA. Section 4
analyzes the testing results with 22 benchmark functions of HSAAE by comparing with AEA. Then the applications of the parameters estimation for a heavy oil thermal cracking three lumps model and Homogeneous mercury (Hg) oxidation are demonstrated in Section 5. Lastly, Section 6 outlines our conclusion and future work.

2. Theoretical backgrounds

2.1. Alopex-based evolutionary algorithm (AEA)

The Alopex-based evolutionary algorithm (AEA) was firstly proposed by Li and Li [8]. This algorithm merges the advantages of evolutionary algorithms and Alopec. A trial individual is generated by adding (or subtracting) the weighted difference vector between two chosen individuals to (or from) the first individual according to a probability. If the trial individual yields a better result than the first individual, it will substitute for the first one. The key idea of AEA is that the moving direction of trial individual is determined by the cross-correlation of the two chosen individuals.

Take a minimization of the function as an example. Suppose that there are two populations $G^i_t$ and $G^{2}_t$ which have the same dimension of the individuals at iterative time $t$. $X^i_t = (x^i_{1t}, x^i_{2t}, \ldots, x^i_{Nt})$ and $Y^j_t = (y^j_{1t}, y^j_{2t}, \ldots, y^j_{Nt})$ are the $i$-th individuals in $G^i_t$ and $G^{2}_t$, respectively. The equations related can be described as follows:

$$\Delta F^i_j = F(X^i_t) - F(Y^j_t)$$ \hspace{1cm} (1)

where, $i=1,2, \ldots, K$ ($K$ is the individual number in one population), $R(.)$ is the objective function of optimizing problem. $\Delta F^i_j$ is the difference of the two objective function values.

$$C^i_{yj} = |x^i_{yt} - y^j_{yt}| \cdot \Delta F^i_j$$ \hspace{1cm} (2)

where, $j=1,2, \ldots, N$ ($N$ is the number of variable dimension), $C^i_{yj}$ is the correlation-coefficient between the variables and the corresponding function value.

$$T^t = \frac{1}{KN} \sum_{i=1}^{K} \sum_{j=1}^{N} C^i_{yj}$$ \hspace{1cm} (3)

$$P^t_{yj} = \frac{1}{1 + e^{-T^t}}$$ \hspace{1cm} (4)

$$\Delta^t_{yj} = \begin{cases} 1, & \text{if } P^t_{yj} \geq \text{rand}(0, 1) \\ -1, & \text{otherwise} \end{cases}$$ \hspace{1cm} (5)

where, $T^t$ is the annealing temperature and $P^t_{yj}$ is the probability. $e^{-}$ is the Natural base. $\Delta^t_{yj}$, determined by the probability $P^t_{yj}$, denotes the variable changing direction. “rand(0,1)” is a random number which subjects to uniform distribution between 0 and 1.

$$x^t_{yt} = x^i_{yt} + \left( x^i_{yt} - y^j_{yt} \right) \cdot \text{rand}(0, 1) \cdot \Delta^t_{yj}$$ \hspace{1cm} (6)

$$x^{t+1}_{yt} = \begin{cases} x^i_{yt}, & \text{if } F(x^t_{yt}) < F(x^i_{yt}) \\ x^i_{yt}, & \text{otherwise} \end{cases}$$ \hspace{1cm} (7)

where, $x^t_{yt}$ is the trial variable, and $x^{t+1}_{yt}$ is the variable in $G^i_t$ at iterative time $(t+1)$.

AEA is one kind of swarm intelligence evolutionary algorithms. The two populations are generated every generation, and executed the operations according to Eq. (1)–Eq. (7). The computation procedure of the algorithm is described as follows: Suppose $G^i_t$ is a population at time $t$. $G^{2}_t$, with the same dimension of the individuals as $G^i_t$, is generated by rearranging the individual order in population $G^i_t$. Calculate a correlation-coefficient by multiplying the difference between the two variables and their objective function differences. The probability $P^t_{yj}$ which determines the directions where all variables of every individual in population $G^2_t$ move is calculated. Because of the probability, each variable of individual does not always move to the gradient descent direction, and it may have the chance to move to the opposite direction. In this case, the algorithm owns better ability to climb the hill, which may give the algorithm more chance to escape from local minima. The detail steps of the algorithm can be referenced in [8].

2.2. Estimation of distribution algorithms (EDAs)

Estimation of distribution algorithm (EDA) was first proposed by Mühlenbein and Paß in 1996 [10]. It is a population-based stochastic heuristics search strategy, and used as an extension of genetic algorithms. In EDAs, a number of individuals are created and computed every generation until an optimal solution is achieved. Unlike other evolutionary algorithms, EDAs have some unique advantages which are described as follows. Firstly, EDAs evolve from one generation to the next by estimating the probability distribution of candidate solutions and sampling the built model. Then the evolutionary trends of all individuals can be described at a macro level. Secondly, EDAs provide new tools to solve some complex optimum problems. Because EDAs can clearly represent the dependencies between individuals by using a probability model, they are more efficient to solve the multi-variable coupled and nonlinear optimum problems. Thirdly, EDA is a combination of statistical learning theory and random optimization algorithm. The hybrid of EDAs with other algorithms will provide a new idea for the research of optimization algorithm [11].

Generally, the evolutionary process of EDA can be described as follows. Firstly, an initial population is generated with a fixed number of individuals. The objective function values of all individuals are calculated. Then a subset of individuals is selected from the initial population to form an intermediate population. The probability distribution of individuals given in intermediate population is estimated. Finally, a new population is generated by sampling from the distribution of the intermediate population. The steps above are repeated until a stop criterion is verified.

The two major problems in EDAs are how to build a probability model and how to sample individuals. Bayesian networks and multivariate Gaussian distribution are common probabilistic models which are used in EDAs [12]. And Monte-Carlo method [13] is one of the most widely used sampling strategies. This method will sort all the individuals in intermediate population by their objective function values. Then the individuals with better functions values are in front and have more chance to be selected. In this way, the data dependencies and the structure found in the best individuals can be transferred into new population. EDAs have a wide range of applications, such as in the graph matching [14] and the Bayesian networks [15]. In this paper, it is used to generate the compared population in AEA.

2.3. Harmony search algorithm (HS)

Harmony Search algorithm, imitating the improvisation process of musicians, was firstly introduced and developed by Geem et al. [16]. When a musician improvises, the aesthetic value referred to as the fitness function is determined by a set of pitches produced by the music instruments (variables) involved. The musician seeks to produce aesthetically pleasing harmony (the optimum solution) as determined by his/her aesthetic perception inferred from rehearsals (iterations). In spite of its youth, HS has been used in many practical applications and is attracting more attentions. A global HS creates new harmonies using the global best harmony
in the HM [17]. A local HS periodically regroups the HM, applies HS independently within each group, and uses the group-best harmony to create new harmonies for each group [18]. In addition, HS has also been successfully hybridized with other metaheuristic methods to collaboratively boost the optimization performance. Such as, Chakraborty et al. proposed an improved variant of the classical Harmony Search metaheuristic by blending with it a difference vector-based mutation operator borrowed from the DE family of algorithms [19]. The pseudo-code of HS can be described as follows:

```
Harmony search (HS)
Begin
  Create and randomly initialize a harmony memory (HM) with HM size (HMS)
  Repeat MaxImp times
    Begin
      Improvise a new harmony $X_{t, new}$ from the HM as following
      for ($j = 1$ to $N$) do
        if ($r_j < \text{HMCR}$) then
          $X_{t, new,j} = X_{t,j}$
        else
          $X_{t, new,j} = X_{t,j} + r_j \text{BW}$
        endif
      endfor
      Update the HM as $X_{t, worst} = X_{t, new} \text{if } f(X_{t, new}) < f(X_{t, worst})$
    End
  End Repeat
End HS
```

In the pseudo-code of HS, the variable $a$ is an integer number in the range of $1$ to HMS. The variables $r_1, r_2, r_3, r$ are random numbers between $0$ and $1$. Pitch adjusting rate (PAR) and bandwidth (BW) have a profound effect on the performance of the HS. Mahdavi and co-workers [17] proposed an improved harmony search (IHS) which dynamically updates PAR and BW according to Eq. (8) and Eq. (9):

$$\text{PAR}(t) = \text{PAR}_{\text{min}} + \frac{\text{PAR}_{\text{max}} - \text{PAR}_{\text{min}}}{t_{\text{max}}}t$$  \hspace{1cm} (8)$$

$$\text{BW}(t) = \text{BW}_{\text{min}} \exp \left( \frac{\ln(BW_{\text{min}}/BW_{\text{max}})}{t_{\text{max}}} t \right)$$  \hspace{1cm} (9)$$

where, $t$ is the current number of iterations and $t_{\text{max}}$ is the maximum number of iterations. $\text{PAR}_{\text{min}}$ and $\text{BW}_{\text{min}}$ are the minimum and maximum adjusting rates, respectively. $\text{BW}_{\text{max}}$ and $\text{PAR}_{\text{max}}$ are the minimum and maximum bandwidths, respectively. A large number of experiments and studies show that the IHS has better optimization performance than the HS in most cases [20]. In this paper, the IHS was introduced to improve the AEA algorithm.

3. The improvement of AEA algorithm

3.1. The improvement of the compared population $G_1^t$

As mentioned above, two populations $G_1^t$ and $G_2^t$ are generated at each iteration time. The heuristic information used in AEA is obtained by computing the two populations. So the quality of the two populations has an effect on searching efficiency. In AEA, the initial population $G_1^0$ is randomly generated according to the population size and variable dimensions. The compared population $G_2^t$ is generated by rearranging the individuals order in population $G_1^t$. The individuals in two populations are the same. Due to the randomness and finiteness of the individuals in $G_1^t$, it is easy to trap into local optima.

In this paper, EDA is employed to generate the compared population $G_2^t$. In order to explain the process clearly, take a minimization of the function as an example. Firstly, the function values of all individuals in $G_1^t$ are calculated. The individuals are sorted by their objective function values in ascending order. The sorted individuals are numbered consecutively from $1$ to $K$. Secondly, estimate the probability distribution of the individuals given in $G_1^t$. The probability value of one individual in $G_1^t$ is calculated by using its number to divide the sum of the numbers. This calculation can be described as follows:

$$PM(i) = \frac{\text{NUM}(i)}{\sum_{i=1}^{K}\text{NUM}(i)}$$  \hspace{1cm} (10)$$

where, NUM($i$) and PM($i$) are the number and probability value of the $i$-th individual in $G_1^t$, respectively. By this model, the individuals with smaller objective function values will have larger probability values.

Finally, generate the compared population $G_2^t$. The initial and compared populations can be described as $G_1^t = \{X_1^t, X_2^t, \ldots, X_N^t\}$ and $G_2^t = \{X_1^t, X_2^t, \ldots, X_N^t\}$. Where, $DX_1^t$ and $DY_1^t$ denote the variables in the $i$-th dimension of the two populations, respectively. The variables in $DY_1^t$ are generated by sampling from the corresponding $DX_1^t$. Each variable in $DX_1^t$ has the same probability value with its corresponding individual. The Monte-Carlo method is used to sample the variables in $DX_1^t$, so the variables with large probability values will have more chance to be selected into the compared population. For example, the variables in $DY_1^t$ are generated by sampling from the $DX_1^t$. The variables in $DX_1^t$ are selected with different chances. The operations above are repeated until all the variables in $G_1^t$ are generated.

The individuals in $G_2^t$ are different from those in $G_1^t$. EDA transfers the better variables which are found in the best individuals into the compared population. The diversity of the compared population is increased. On the other hand, the compared population $G_2^t$ may miss the best individuals in $G_1^t$. In order to avoid this problem, some of the worst individuals in $G_2^t$ are replaced by the best individuals of the same number in $G_1^t$. The number of the replacement is $K/2$. This method is generally known as “elitist strategy” [21–22]. The pseudo-code of generating the compared population $G_2^t$ is described as follows:

```
EDA
Begin
  Generate an initial population $G_1^t$ randomly
  Calculate the objective functions values of all individuals in $G_1^t$
  Sort the individuals in $G_1^t$ in ascending order and number them from $1$ to $K$
  Calculate the probability values PM($i$) of each individual given in $G_1^t$
  Generate the variables in $DY_1^t$ by sampling from the corresponding $DX_1^t$
  Repeat until all variables in $G_2^t$ are generated
  Replace $K/2$ worse individuals in $G_2^t$ with the same number of better individuals in $G_1^t$
End
```
3.2. The improvement of the moving step length of AEA

In AEA, the searching process is similar to Simulated Annealing (SA) algorithm to some extent. Two parameters “T” and “P” are used to control the step length and directions. Take a minimization of the function as an example, if the objective function value decreases at the previous moving direction, it is expected that the variables can keep moving along the same direction. This operation makes the algorithm have greater opportunity to accelerate convergence. Meanwhile, because of the probability, there is still opportunity to search the opposite direction of the anterior search to avoid trapping in local minima.

But it is not reasonable that the range of step length is the same for the variables moving forward and backward. Actually, AEA is a hybrid algorithm of deterministic search and stochastic search. Each of the two searches plays a key role at a certain moment, which is determined by the probability “P”. The deterministic search is similar to the gradient descent method. In order to accelerate the convergence process and search more efficiently, individuals are expected to walk along the gradient descent direction continually. In this case, the algorithm searches in a smaller range to avoid missing the global optimum. It must be noted that the gradient descent direction is not always the right direction to the global optimum but to a local optimum. In order to help the algorithm jump out of local minima as soon as possible, the search direction would have a certain probability to be the opposite of gradient descent. In this case, a larger range for individuals moving would be provided to ensure that they can escape from local minima. Otherwise, it is easy to swing back and forth in the bottom of one valley, and finally converge to a local minimum. The individuals can also jump out of a local minimum with more iterations, but it is not beneficial to global search. In this paper, the change of independent variables with respect to the objective function is considered. The Eq. (5) can be set as follows:

$$
\Delta t^I = \begin{cases} 
\alpha^I_t, & \text{if } P^I_t \geq \text{rand(0, 1)} \\
-\beta^I, & \text{otherwise}
\end{cases}
$$

where, $\alpha^I$ and $\beta^I$ are both real number, and $0 < \alpha^I < 1, \beta^I \geq 1$.

Furthermore, at the beginning of the evolutionary process, individuals in population are scattered in a large search scope. They move towards the direction of global optimum fast. An individual may step backward for continuous several times with the probability. The factor $\beta^I$ which is used to control the stepping backward should be a little larger. At the same time, the factor $\alpha^I$ which is used to control the stepping forward should be a little smaller. In this case, a larger range for individuals moving would be provided to ensure that they can escape from local minima. Otherwise, it is easy to swing back and forth in the bottom of one valley, and finally converge to a local minimum. The individuals can also jump out of a local minimum with more iterations, but it is not beneficial to global search. In this paper, the change of independent variables with respect to the objective function is considered. The Eq. (5) can be set as follows:

$$
\alpha^I_t = \alpha^I_0 + (\alpha^I_0 - \alpha^I_1) \cdot \frac{t}{t_{\text{max}}}
$$

$$
\beta^I_t = \beta^I_0 + (\beta^I_0 - \beta^I_1) \cdot (1 - \frac{t}{t_{\text{max}}})
$$

where, $t$ is iteration time, $t_{\text{max}}$ is the maximal iteration times. $\alpha^I_0$ and $\alpha^I_1$ are the low bound and high bound of $\alpha^I$, respectively. $\beta^I_0$ and $\beta^I_1$ are the low bound and high bound of $\beta^I$, respectively. After a number of experiments, we set $\alpha^I_0 = 0.1, \alpha^I_1 = 0.7; \beta^I_0 = 1.3, \beta^I_1 = 1.7$. It is clear that in the whole iterative process, $\alpha^I$ increases from 0.1 to 0.7, while $\beta^I$ decreases from 1.7 to 1.3. Both of them can be calculated by using the Eq. (12) and Eq. (13), respectively.

3.3. AEA algorithm with harmony search (HSAEA)

In the original AEA, the heuristic information is obtained from the change of independent variables with respect to the objective functions. In order to generate a trial individual, the weighted difference vector between two individuals is added (or subtracted) to (or from) the first individual with a probability. In this way, the convergence speed is improved. But how, the individuals are easy to trap into local optima. In this paper, the HS algorithm is employed to overcome the problem.

In the process of generating a vector, the different evolutionary algorithms always have the same characteristic that only seldom parent vectors are considered. For example, in genetic algorithm (GA), only two individuals from old population are considered to produce a new vector [23]. In particle swarm optimization (PSO), each particle adjusts its flying according to its own flying experience and its companions’ flying experience [24]. In Differential Evolution (DE), a donor vector is generated by adding the weighted difference of the two parameter vectors [25]. All of them consider seldom information. In this paper, the HS algorithm is employed to generate a new vector through considering all of the existing vectors. It contrasts sharply with the conventional evolutionary approaches. The use of HS algorithm will increase the robustness and flexibility of the underlying search mechanism and ensures better solutions [26].

Hence, HS algorithm is employed to improve the search mechanism in AEA. In this paper, it is used to improve the swarm after Alopex operator in every generation. As discussed above, the detailed implementation steps of HSAEA algorithm can be described as follows:

**Step 1:** Set the number of individuals in population ($K$), the number of variable dimension ($N$), function optimization goal, maximal iteration times, and HS parameters: Maxlmp, HMCr (Harmony memory consideration rate), PAR$_{\text{max}}$, PAR$_{\text{min}}$, BW$_{\text{max}}$ and BW$_{\text{min}}$.

**Step 2:** Set iteration time $t = 1$ and initialize the population $G_1^t$ within the search scope randomly.

**Step 3:** Calculate the individual’s objective function values in population $G_1^t$.

**Step 4:** Estimate the probability distribution of individuals given in $G_1^t$ by Eq. (10). The initial population and compared population can be respectively described as

$$
G_1^t = (Dx_1^1, Dx_2^1, \ldots, Dx_N^1) \quad \text{and} \quad G_2^t = (DY_1^1, DY_2^1, \ldots, DY_N^1).
$$

The variables in $DY_1^1$ are generated by sampling from the corresponding $DX_j^1$. Repeat the operations above until all variables in $G_2^t$ are generated. Then replace K/2 worst individuals in $G_1^t$ with the same number of best individuals in $G_2^t$.

**Step 5:** Suppose $G_1^t = (X_1^1, X_2^1, \ldots, X_N^1)$ and $G_2^t = (Y_1^1, Y_2^1, \ldots, Y_N^1)$ denote the populations $G_1^t$ and $G_2^t$. For the variable $X_i^1(i = 1, 2, \ldots, N)$ of the individual $X_j^1(i = 1, 2, \ldots, K)$ from population $G_1^t$ and the corresponding variable $Y_j^1(i = 1, 2, \ldots, N)$ of the individual $Y_j^1(i = 1, 2, \ldots, K)$ from population $G_2^t$, calculate the difference $(x_{ij}^1 - y_{ij}^1)$, and then compute difference $|F(X_j^1) - F(Y_j^1)|$ of the two objective function values.

**Step 6:** Calculate the cross-correlation $C_{ij}^t(i = 1, 2, \ldots, K; j = 1, 2, \ldots, N)$ between the variable $X_{ij}^1(i = 1, 2, \ldots, K; j = 1, 2, \ldots, N)$ and its corresponding variable $Y_{ij}^1(i = 1, 2, \ldots, K; j = 1, 2, \ldots, N)$ according to Eq. (2).

**Step 7:** According to Eq. (3) and Eq. (4), calculate the annealing temperature $T$ and the probability $P_{ij}^t(i = 1, 2, \ldots, K; j = 1, 2, \ldots, N)$.

**Step 8:** Calculate the value of $\alpha^I$ and $\beta^I$, according to Eq. (12) and Eq. (13), respectively.
4. Validation with benchmark functions

4.1. The compare of HSAEA and AEA

In literature [8], the researchers conducted a set of tests with one improved Genetic Algorithm (HGA3) [9], Particle Swarm Optimization (PSO) [27], Differential Evolution (DE) [28] and AEA. The results in literature [8] show that AEA is approximately equal to HGA3 and DE on the successful runs. As for ABVF (the average best function value) which is the characterization of individual quality, AEA is better than PSO, DE and HGA3. In this paper, the performance of HSAEA is tested and compared only with AEA. Table 1 lists a set of 22 unconstrained benchmark functions selected from the literature [9]. In Table 1, both the mathematical formulas and the bounds of variables for each function are described. In fact, these benchmark functions are commonly used for comparing evolutionary algorithms because they have many different types of characteristics which can help to evaluate the performance of the algorithms from different aspects. In this paper, all the algorithms used for the experiments are programmed in MATLAB. And all the experiments are carried out on a LENOVO computer (T400) with an Intel(R) Core(TM) 2 Duo CPU P8400 with frequency 2.26 GHz, 3GB RAM under Windows 7 operation system.

In order to make a fair comparison, a group of experiments are conducted as the same as literature [8], in which the variable dimension for all the benchmark functions are fixed at 10 (N = 10). And all the numbers of individuals in population for different functions are fixed at 100. Additionally, HS parameters are shown in Table 2.

Here, a total of 100 runs are conducted with randomly initial populations for each test function. According to the literature [9], a run is regarded as a successful one if the value obtained by the algorithm is within 1% accuracy of the known optimal individual. But particularly for three functions including Rosenbrock, Schwefel, and Neumaier, the accuracy is set as 15, 0.4 and 2, respectively. The termination condition is a maximum of 2000 generations or there is no improvement observed in the best individual in consecutive 100 generations. In Table 3, there lists the rate of success (RS), the average best function value (ABVF) and standard deviations (SD) of 100 independent runs of the two algorithms. Particularly, the ABVF and SD are accounted for successful runs only, while unsuccessful runs are not considered and the never succeed one finally is denoted as (-) instead. Note that the results obtained by AEA are cited from the Li and Li’s literature [8].

From Table 3, the best results for these functions are assigned by the bold-face. To evaluate the overall performance, for the two algorithms, the numbers of no worse results (NNWR) compared with the other are listed at the bottom of the table. It is clear that HSAEA algorithm outperforms AEA algorithm on most functions. The number of no worse ABVF obtained by HSAEA is 20, while that by AEA is 10 only. Then the standard SD which represents the stability of the two algorithms is considered. The numbers of no worse SD obtained by HSAEA and AEA are 19 and 8, respectively. These results show that HSAEA is more efficient and more stable than AEA in optimization. At last but not least, the numbers of no worse RS obtained by the two algorithms are 21 and 18, which show that HSAEA has a higher successful rate to find solutions.

Meanwhile, HSAEA especially works better on three functions which are difficult for AEA to find the global optimum. The characters of the three functions are described as follows. Ackley function (f1) is a continuous, multimodal test function, which is obtained by modulating an exponential function with cosine wave of moderate amplitude [29]. Due to the reason that it traps into local optima easily, this function can be used to test algorithms [30]. Griewank function (f2) has a global minimum value 0 and a lot of local minima [31]. It is generally used to test the convergence of optimization algorithms. Rastrigin function (f3) is generated by adding cosine modulation to the sphere function. Then it is highly multimodal and difficult to find the global optimum for algorithms. All of the three functions are suitable to be benchmark functions. On the other hand, because the global optimum of the Rosenbrock function (f4) is in a long, narrow and parabolic shaped flat valley [32], it is difficult for both HSAEA and AEA to find an optimal solution of the function.
<table>
<thead>
<tr>
<th>Functions</th>
<th>Name</th>
<th>Function</th>
<th>Bounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>Ackley</td>
<td>$-20 \exp \left(-0.02 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}\right) - \exp \left(\frac{1}{n} \sum_{i=1}^{n} \cos(2 \pi x_i)\right) + 20 + e$</td>
<td>$[-30,30]$</td>
</tr>
<tr>
<td>$f_2$</td>
<td>Cosine mixture</td>
<td>$0.1 n + \sum_{i=1}^{n} x_i^2 - 0.1 \sum_{i=1}^{n} \cos(5 \pi x_i)$</td>
<td>$[-1,1]$</td>
</tr>
<tr>
<td>$f_3$</td>
<td>Exponential</td>
<td>$1 - \left(\exp \left(-0.5 \sum_{i=1}^{n} x_i^2\right)\right)$</td>
<td>$[-1,1]$</td>
</tr>
<tr>
<td>$f_4$</td>
<td>Griewank</td>
<td>$1 + \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{\pi x_i}{\sqrt{n}}\right)$</td>
<td>$[-600,600]$</td>
</tr>
<tr>
<td>$f_5$</td>
<td>Levy and Montalvo-1</td>
<td>$\frac{10}{\pi} \left(10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \left[1 + 10 \sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right), \quad y_1 = 1 + \frac{1}{\pi} (x_1 + 1)$</td>
<td>$[-10,10]$</td>
</tr>
<tr>
<td>$f_6$</td>
<td>Levy and Montalvo-2</td>
<td>$0.1(\sin^2(3 \pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3 \pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2 \pi x_n)])$</td>
<td>$[-5.5]$</td>
</tr>
<tr>
<td>$f_7$</td>
<td>Paviani</td>
<td>$45.778 + \sum_{i=1}^{n} \left[\ln(x_i - 2)^2 + \ln(10 - x_i)^2\right] - \left(\prod_{i=1}^{n} x_i\right)^{0.2}$</td>
<td>$[2.10]$</td>
</tr>
<tr>
<td>$f_8$</td>
<td>Rastrigin</td>
<td>$10n + \sum_{i=1}^{n} \left[100(x_i^2 - 10 \cos(2 \pi x_i)\right) + (x_i - 1)^2\right]$</td>
<td>$[-5.12,5.12]$</td>
</tr>
<tr>
<td>$f_9$</td>
<td>Rosenbrock</td>
<td>$\sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i)^2 + (x_i - 1)^2\right] + (x_n - 1)^2\right]$</td>
<td>$[-30,30]$</td>
</tr>
<tr>
<td>$f_{10}$</td>
<td>Schwefel</td>
<td>$418.9829n - \sum_{i=1}^{n} x_i \sin\left(\sqrt{</td>
<td>x_i</td>
</tr>
<tr>
<td>$f_{11}$</td>
<td>Sinosoidal</td>
<td>$3.5 - \left[2.5 \prod_{i=1}^{n} \sin\left(x_i - \frac{x_i}{\pi}\right) + \prod_{i=1}^{n} \sin\left(5 (x_i - \frac{x_i}{\pi})\right)\right]$</td>
<td>$[0,\pi]$</td>
</tr>
<tr>
<td>$f_{12}$</td>
<td>Zakharov's</td>
<td>$\sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} x_i\right)^2 + \left(\sum_{i=1}^{n} x_i\right)^4$</td>
<td>$[-5.12,5.12]$</td>
</tr>
<tr>
<td>$f_{13}$</td>
<td>Sphere</td>
<td>$\sum_{i=1}^{n} x_i^2$</td>
<td>$[-5.12,5.12]$</td>
</tr>
<tr>
<td>$f_{14}$</td>
<td>Axis Parallel hyper ellipsoid</td>
<td>$\sum_{i=1}^{n}</td>
<td>x_i</td>
</tr>
<tr>
<td>$f_{15}$</td>
<td>Schewefel-3</td>
<td>$\sum_{i=1}^{n}</td>
<td>x_i</td>
</tr>
<tr>
<td>$f_{16}$</td>
<td>Neumaier3</td>
<td>$\frac{n \sin^{(n+1)/(n-1)}}{\pi^{(n+1)}} + \sum_{i=1}^{n} (x_i - 1)^2 - \sum_{i=1}^{n} x_i x_{i-1}$</td>
<td>$[-100,100]$</td>
</tr>
<tr>
<td>$f_{17}$</td>
<td>Salonmon</td>
<td>$1 - \cos(2 \pi</td>
<td>x</td>
</tr>
<tr>
<td>$f_{18}$</td>
<td>Ellipsoidal</td>
<td>$\sum_{i=1}^{n} (x_i - i)^2$</td>
<td>$[-10,10]$</td>
</tr>
<tr>
<td>$f_{19}$</td>
<td>Schaffer1</td>
<td>$0.5 + \left(\sin \sqrt{\sum_{i=1}^{n} x_i^2} - 0.5\right) / \left(1 + 0.001 \left(\sum_{i=1}^{n} x_i^2\right)^{2}\right)$</td>
<td>$[-100,100]$</td>
</tr>
<tr>
<td>$f_{20}$</td>
<td>Brown3</td>
<td>$\sum_{i=1}^{n} \left[x_i^2 \left(\sum_{j=1}^{i-1} x_j^2 \right) + \left(\sum_{j=1}^{i-1} x_j^2 \right)^2\right]$</td>
<td>$[-1.4]$</td>
</tr>
<tr>
<td>$f_{21}$</td>
<td>New function</td>
<td>$\sum_{i=1}^{n} (0.2x_i^2 + 0.1x_i^2 \sin 2x_i)$</td>
<td>$[-10,10]$</td>
</tr>
<tr>
<td>$f_{22}$</td>
<td>Cigar</td>
<td>$x_i^2 + 100,000 \sum_{i=2}^{n} x_i^2$</td>
<td>$[-10,10]$</td>
</tr>
</tbody>
</table>
4.2. The Wilcoxon signed test

In order to deeply verify the significant difference between HSAEA and AEA, the Wilcoxon signed ranks test is employed [33]. It is a nonparametric test for pairwise comparisons. In this paper, the ABFVs represent the characterization of individuals quality and are used as the performance scores of the two algorithms. The main steps of Wilcoxon test are described as follows. Firstly, \( d_i \) is calculated as the difference of ABFVs between the two algorithms on \( i \)-th out of 21 functions (the function \( f_{17} \) can’t be found an optimal solution by using HSAEA and AEA, so it is ignored). Secondly, the differences are ranked according to their absolute values. If some differences have the same absolute value, they will use the average rank of them. For example, there are three differences which have the same absolute value and they are assigned ranks from 1 to 3. Then they will use the average rank 2. The differences of ABFVs between HSAEA and AEA, and their ranks are shown in Table 4.

From Table 4, the number of the differences which have the same absolute value (0) is 9. One of them will be ignored [33] and the surplus use the average rank of them, which is equal to 4.5. Then the other differences are ranked from 9 to 20. Finally, the ranks of the functions in which the HSAEA outperforms the AEA are summed. This sum is denoted as \( R^+ \). In the same way, \( R^- \) denotes the sum of the ranks for the opposite. In addition, the ranks of the functions in which the two algorithms have the same performances are split evenly among the two sums. The equations for the calculation can be set as follows:

\[
R^+ = \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i) \tag{14}
\]

\[
R^- = \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i) \tag{15}
\]

In this paper, the \( R^+ \) and \( R^- \) are 172 and 38, respectively. Then the \( T \) (test statistic) is assigned the smaller of the two sums, \( T = 38 \). It will be compared with the value of the distribution of Wilcoxon for 20 degrees of freedom [34]. If the value of \( T \) is smaller, the HSAEA outperforms the AEA with the \( p \)-value associated. The \( p \)-value can be calculated as 0.017 by using the R software package [35]. As a result, HSAEA has a significant improvement over AEA, with a level of significance \( \alpha = 0.05 \).

### Table 4

<table>
<thead>
<tr>
<th>Function</th>
<th>AEA</th>
<th>HSAEA</th>
<th>DI</th>
<th>Sign</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>4.57E−15</td>
<td>4.23E−15</td>
<td>3.40E−16</td>
<td>+</td>
<td>17</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Ignored</td>
<td></td>
</tr>
<tr>
<td>( f_3 )</td>
<td>1.11E−18</td>
<td>0</td>
<td>1.11E−18</td>
<td>+</td>
<td>16</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>5.12E−04</td>
<td>3.08E−04</td>
<td>2.04E−04</td>
<td>+</td>
<td>19</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>4.71E−32</td>
<td>4.71E−32</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_6 )</td>
<td>1.35E−32</td>
<td>1.35E−32</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_7 )</td>
<td>-4.70E−04</td>
<td>-4.70E−04</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_8 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_9 )</td>
<td>4.56E−01</td>
<td>2.07</td>
<td>-1.614</td>
<td>0.20</td>
<td></td>
</tr>
<tr>
<td>( f_{10} )</td>
<td>1.27E−04</td>
<td>1.27E−04</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_{11} )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_{12} )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_{13} )</td>
<td>1.53E−43</td>
<td>2.28E−69</td>
<td>1.53E−43</td>
<td>+</td>
<td>15</td>
</tr>
<tr>
<td>( f_{14} )</td>
<td>8.53E−101</td>
<td>6.84E−238</td>
<td>8.53E−101</td>
<td>+</td>
<td>11</td>
</tr>
<tr>
<td>( f_{15} )</td>
<td>8.12E−101</td>
<td>4.13E−236</td>
<td>8.12E−101</td>
<td>+</td>
<td>10</td>
</tr>
<tr>
<td>( f_{16} )</td>
<td>6.86E−70</td>
<td>3.55E−157</td>
<td>6.86E−70</td>
<td>+</td>
<td>13</td>
</tr>
<tr>
<td>( f_{17} )</td>
<td>1.31E−08</td>
<td>9.38E−09</td>
<td>3.72E−09</td>
<td>+</td>
<td>18</td>
</tr>
<tr>
<td>( f_{18} )</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>-</td>
</tr>
<tr>
<td>( f_{19} )</td>
<td>9.72E−03</td>
<td>9.72E−03</td>
<td>0</td>
<td>0.45</td>
<td></td>
</tr>
<tr>
<td>( f_{20} )</td>
<td>8.74E−101</td>
<td>7.36E−239</td>
<td>8.74E−101</td>
<td>+</td>
<td>12</td>
</tr>
<tr>
<td>( f_{21} )</td>
<td>7.89E−101</td>
<td>4.85E−239</td>
<td>7.89E−101</td>
<td>+</td>
<td>9</td>
</tr>
<tr>
<td>( f_{22} )</td>
<td>6.76E−57</td>
<td>8.19E−145</td>
<td>6.76E−57</td>
<td>+</td>
<td>14</td>
</tr>
</tbody>
</table>

### Table 3

Comparison of the optimization result after 2000 generations.

<table>
<thead>
<tr>
<th>Functions</th>
<th>AEA (from Li and Li(2011))</th>
<th>HSAEA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RS</td>
<td>ABFV</td>
</tr>
<tr>
<td>( f_1 )</td>
<td>54</td>
<td>4.57E−15</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>100</td>
<td>1.11E−18</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>83</td>
<td>5.12E−04</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>100</td>
<td>4.71E−32</td>
</tr>
<tr>
<td>( f_6 )</td>
<td>100</td>
<td>1.35E−32</td>
</tr>
<tr>
<td>( f_7 )</td>
<td>100</td>
<td>-4.70E−04</td>
</tr>
<tr>
<td>( f_8 )</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>( f_9 )</td>
<td>100</td>
<td>4.56E−01</td>
</tr>
<tr>
<td>( f_{10} )</td>
<td>100</td>
<td>1.27E−04</td>
</tr>
<tr>
<td>( f_{11} )</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>( f_{12} )</td>
<td>100</td>
<td>1.53E−43</td>
</tr>
<tr>
<td>( f_{13} )</td>
<td>100</td>
<td>8.53E−101</td>
</tr>
<tr>
<td>( f_{14} )</td>
<td>100</td>
<td>8.12E−101</td>
</tr>
<tr>
<td>( f_{15} )</td>
<td>100</td>
<td>6.86E−70</td>
</tr>
<tr>
<td>( f_{16} )</td>
<td>100</td>
<td>1.31E−08</td>
</tr>
<tr>
<td>( f_{17} )</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>( f_{18} )</td>
<td>100</td>
<td>9.72E−03</td>
</tr>
<tr>
<td>( f_{19} )</td>
<td>100</td>
<td>8.74E−101</td>
</tr>
<tr>
<td>( f_{20} )</td>
<td>100</td>
<td>7.89E−101</td>
</tr>
<tr>
<td>( f_{21} )</td>
<td>100</td>
<td>6.76E−57</td>
</tr>
<tr>
<td>( f_{22} )</td>
<td>18</td>
<td>10</td>
</tr>
</tbody>
</table>
5. Reaction kinetic parameter estimation using HSAEA

HSAEA has achieved good results on the 22 unconstrained benchmark functions. Then the practical performance of HSAEA will be tested in reaction kinetic parameter estimation. In chemical process, many parameters which are important to the modeling process are non-measurable. Meanwhile, the number of them will increase with the complexity of the physic-chemical model [36]. In this case, a method which can give the best estimation of the non-measurable parameters is needed. HSAEA is a stochastic heuristic optimization algorithm which has ability to search in a large space. In order to test the search precision and search efficiency of HSAEA, two types of parameter estimation are employed in this paper.

5.1. Parameter estimation for a heavy oil thermal cracking three lumps model

Cracking of heavy oil is a very complex chemical process. In order to simplify the reaction system, a heavy oil thermal cracking three lumps model is given by Song et al. [37]. The model is set up merging a large number of single compounds in the reaction system into a number of virtual components, according to the principle of similar kinetic characteristics.

The material transformation process of heavy oil thermal cracking three lumps model is shown in Fig. 1. There are two hypothesizes as follows:

1. Heavy oil raw material H is methylbenzene in the residue with temperature more than 510 °C. The heavy middle fractions W generated from Heavy oil pyrolytic reaction can be converted to pyrolysis gas, light fractions and condensation product L, simultaneously.

2. The secondary reaction that heavy middle fractions W are converted to condensation product L(W → L) is first order reaction.

Given the condition of the same temperature, and the overall reaction order is 1, according to the reaction mechanism, dynamic equation of reaction can be derived as follows:

\[
x_L = \frac{K_{WP0} \cdot x_L^i}{K_L} - \frac{1}{1 - \frac{x_L}{K_W}} \frac{K_{WP0} \cdot x_L^i}{K_L} + \frac{1}{n_L} \frac{1}{K_{WP0} \cdot x_L^i} \\
\times \left\{ \frac{1}{K_{WP0} \cdot x_L^i} \cdot \left[ 1 - \frac{1}{n_L} \right] + \frac{1}{n_L} \cdot \left[ 1 - x_L \right] \right\} 
\]

where \( x \) and \( T \) are the input variables and \( x_L \) is the system output. There are eight parameters of the model to be estimated: \( K_{WP0}, K_{WP}, K_{WP0}, K_{WP}, n_W, n_L \).

In literature, researchers employed many algorithms to implement on parameter estimation of this model, including SGA, EGA [37], CPSO [38], APSO [39] and so on. In order to make a fair comparison, all of 56 groups of data provided by Song et al. [37] are used as modeling samples and test samples, the maximal generation of HSAEA is set as 500, and the objective function is selected as:

\[
\min Arer = \frac{1}{m} \sum_{i=0}^{m-1} \left( \frac{x_L(i) - \hat{x}_L(i)}{\hat{x}_L(i)} \right) 
\]

where, \( \hat{x}_L(i) \) is obtained by using Eq. (16), \( m \) is the number of samples, \( Arer \) is the average self-checking relative error.

In Table 5, the parameter estimation results given by various algorithms, including the values of the eight parameters and the average self-checking relative errors, are listed. From Table 5, it can be seen the average self-checking relative error (Arer) gained by HSAEA is 5.27%, smaller than all the other algorithms mentioned, which means that HSAEA can offer a fairly high fitting precision of the model.

5.2. Kinetic parameters estimation for homogeneous mercury oxidation

Mercury emissions from coal-fired power plants are highly dependent upon mercury speciation [40]. Thus, much attention has been paid to research efficient means to remove Hg from power plant flue gases.

Commonly, Mercury in the flue gas is classified in three forms: elemental mercury (Hg⁰), oxidized mercury (Hg²⁺) and particulate bound mercury (Hg⁰). The particulate bound mercury is usually trapped by ash collection devices within power plants, such as electrostatic precipitators (ESP), mechanical hoppers, or bag houses. Elemental mercury is relatively inert and difficult to capture because of its non-reactivity. It is also volatile at high temperatures and insoluble in water. Conversely, oxidized mercury is very water soluble and has an affinity for adsorbing onto particulate matter such as fly ash or on metal surfaces in the duct. As a result of these physical and chemical properties of Hg⁰ and Hg²⁺, the removal of mercury is enhanced when elemental Hg is converted to its oxidized form [41].

In detail, the reaction mechanism, according to Agarwal and Stenger [41], is a five-reaction system to analyze mechanism of Hg oxidation. During the five reactions, two of them are reversible and three are irreversible.

In order to model or optimize the process, ten parameters need to be determined. The ten parameters are \( A_1, A_2, A_3, A_4, A_5, A_1, E_2, E_3, E_4, E_5 \), which are used in five reaction kinetic equations. Where, \( A_1 \) is the pre-exponential factor, \( E_1 \) is the activation energy of reaction kinetic equation \( i \) (\( i = 1, 2, \ldots, 5 \)). The main purpose of this optimization problem is to determine the best values of these parameters.

Table 5
Parameter estimation results of heavy oil thermal cracking three lumps model given by various algorithms.

<table>
<thead>
<tr>
<th>Method</th>
<th>( K_{WP0} )</th>
<th>( E_p )</th>
<th>( K_{WP0} )</th>
<th>( E_p )</th>
<th>( K_{WP0} )</th>
<th>( E_p )</th>
<th>( n_w )</th>
<th>( n_L )</th>
<th>Arer (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SGA</td>
<td>4.68</td>
<td>1257</td>
<td>5.155</td>
<td>1850</td>
<td>4.197</td>
<td>3776</td>
<td>1.488</td>
<td>1.191</td>
<td>7.7</td>
</tr>
<tr>
<td>EGA</td>
<td>4.032</td>
<td>1166</td>
<td>5.058</td>
<td>3676</td>
<td>4.228</td>
<td>2382</td>
<td>1.57</td>
<td>1.357</td>
<td>6.28</td>
</tr>
<tr>
<td>CPSO</td>
<td>0.112</td>
<td>-743</td>
<td>1.176</td>
<td>295</td>
<td>3.659</td>
<td>-817</td>
<td>2.915</td>
<td>1.202</td>
<td>6.13</td>
</tr>
<tr>
<td>APSO</td>
<td>0.151</td>
<td>-1013</td>
<td>4.032</td>
<td>1770</td>
<td>4.303</td>
<td>-400</td>
<td>2.97</td>
<td>1.605</td>
<td>5.85</td>
</tr>
<tr>
<td>HSAEA</td>
<td>5.303</td>
<td>2388.3</td>
<td>2.9616</td>
<td>493.88</td>
<td>1.2086</td>
<td>-240.34</td>
<td>5.5449</td>
<td>8.8673</td>
<td>5.69</td>
</tr>
<tr>
<td></td>
<td>1.974</td>
<td>95.582</td>
<td>2.231</td>
<td>65.033</td>
<td>4.424</td>
<td>95.463</td>
<td>5.13</td>
<td>56.683</td>
<td>5.27</td>
</tr>
</tbody>
</table>
so that difference between the values of reaction rates calculated from the five reaction kinetic equations, and those measured experimentally is minimized. Thus the objective function is described by

$$\text{min EQS} = \frac{1}{m} \sum_{i=1}^{m} (\hat{c}_i - c_i)^2 \quad (18)$$

where, EQS denotes the average of the squares of differences; $c_i$ is the experimental Hg conversion; $\hat{c}_i$ is the Hg conversion calculated according to the five reaction kinetic equations; $m$ is the number of data points.

Then, HSEA is employed to determine the kinetic parameters for homogeneous mercury oxidation in flue gas with the data measured experimentally and reported by Agarwal et al. [42]. To take a fair comparison, the maximal number of generations is also set as 1000. The objective function is used as a guideline to determine the best optimal parameters from the data set. The results are compared with those obtained by “fmin search algorithm”, a deterministic method which EQS uses the Nelder–Mead simplex algorithm [41], and ISDE [43] are listed in Table 6. The minimum obtained by HSEA is 69.709. It is better than the reported results, 88.134 by Agarwal et al. [41] and 73.746 by ISDE. Hence, HSEA is found to be more profitable operating than those reported in earlier literatures.

In general, from the promising results of the two applications above, it can be concluded that HSEA is competent to solve these real complex nonlinear cases.

6. Conclusions

AEA is proved to be an efficient and a powerful search and optimization algorithm in scientific and engineering fields. In the AEA algorithm, the compared population and the moving step length play key roles in the algorithm’s performance. However the setting methods of them are not reasonable. In this paper, the compared population is generated by EDA. The moving step length is improved to be gradually adjusted during different phases of evolution. And more importantly, HS is introduced to improve the quality of swarm of every generation. Some benchmark functions and two real cases on parameter estimation are used to test the performance of AEA. The promising results demonstrate that HSEA is superior to AEA in almost all indicators related.

Despite of these promising results, there is still much room for improvement, such as, our future work is mainly focused on more reasonable controlling of the annealing temperature “T”, further improvement of the generation method of the compared population $G_2$, how to solve the variable-interdependent problems such as Rosenbrock function, and so on.

Acknowledgments

The authors would like to appreciate the National Natural Science Foundation of China (under project No.20976048 and No.21176072), Shanghai Leading Academic Discipline Project (under Project No.B504) and the Fundamental Research Funds for the Central Universities, for their financial support.

References

demic Publisher, Boston, 1987.


