Neurocomputing 138 (2014) 392–402

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A new improved krill herd algorithm for global numerical optimization

Lihong Guo [a], Gai-Ge Wang [b,*], Amir H. Gandomi [c,1], Amir H. Alavi [d,2], Hong Duan [e]

[a] Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China
[b] School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, Jiangsu, 221116, China
[c] Department of Civil Engineering, University of Akron, Akron, OH 443253905, USA
[d] Department of Civil and Environmental Engineering, Engineering Building, Michigan State University, East Lansing, MI 48824, USA
[e] Education College, Shihezi University, Shihezi, Xinjiang, 832000, China

ARTICLE INFO

Article history:
Received 13 October 2012
Received in revised form
12 January 2013
Accepted 23 January 2014
Communicated by: A. Abraham
Available online 15 February 2014

Keywords:
Global optimization problem
Krill herd
Exchange information
Multimodal function

ABSTRACT

This study presents an improved krill herd (IKH) approach to solve global optimization problems. The main improvement pertains to the exchange of information between top krill during motion calculation process to generate better candidate solutions. Furthermore, the proposed IKH method uses a new Lévy flight distribution and elitism scheme to update the KH motion calculation. This novel meta-heuristic approach can accelerate the global convergence speed while preserving the robustness of the basic KH algorithm. Besides, the detailed implementation procedure for the IKH method is described. Several standard benchmark functions are used to verify the efficiency of IKH. Based on the results, the performance of IKH is superior to or highly competitive with the standard KH and other robust population-based optimization methods.

1. Introduction

In computer science, mathematics, and computational science, the process of optimization is searching for a vector in a function that produces an optimal solution. All of feasible values are available solutions and the extreme value is optimal solution. In general, optimization algorithms have been applied to solve optimization problems [1]. A simple classification way for optimization algorithms is considering the nature of the algorithms. The optimization algorithms can be divided into two main categories: deterministic algorithms, and stochastic algorithms. Deterministic algorithms using gradient such as hill-climbing have a rigorous move, and will generate the same set of solutions if the iterations commence with the same initial starting point. On the other hand, stochastic algorithms without using gradient often generate different solutions even with the same initial value. However, generally speaking, the final values, though slightly different, will converge to the same optimal solutions within a given accuracy. The recently nature-inspired meta-heuristic algorithms perform powerfully and efficiently in solving modern nonlinear numerical global optimization problems. To some extent, all meta-heuristic algorithms strive for making balance between randomization (global search) and local search [2].

Inspired by nature, the strong meta-heuristic algorithms are applied to solve NP-hard problems such as parameter estimation [3], system identification [4], WSN dynamic deployment [5], UCAV path planning [6,7], test-sheet composition [8], and water, geotechnical and transport engineering [9,10]. During the 1950s and 1960s, computer scientists studied the possibility of conceptualizing evolution as an optimization tool and this generated a subset of gradient free approaches named genetic algorithms (GAs) [11,12]. Since then many other nature-inspired meta-heuristic algorithms have emerged, such as ant colony optimization (ACO) [13], differential evolution (DE) [14,15], bat algorithm (BA) [16,17], harmony search (HS) [18], and particle swarm optimization (PSO) [19].

Recently, Gandomi and Alavi [20] proposed krill herd (KH) algorithm which is based on the simulation of the herding behavior of krill individuals in nature. In KH, the objective function for the krill movement is determined by the minimum distances of each individual krill from food and from highest density of the herd. The time-dependent position of the krill individuals is comprised of three main components: (i) movement induced by other individuals (ii) foraging motion, and (iii) random physical diffusion. One of remarkable advantage of the KH algorithm is that

* Corresponding author.
E-mail addresses: guolh@ciomp.ac.cn (L. Guo), gaigewang@163.com, gaigewang@gmail.com (G.-G. Wang), a.h.gandomi@gmail.com (A. H. Gandomi), ah_alavi@hotmail.com, alavi@msu.edu (A. H. Alavi), duanh0618@163.com (H. Duan).
[1] http://gozips.uakron.edu/~ag72/
[2] http://www.egr.msu.edu/~alavi/

http://dx.doi.org/10.1016/j.neucom.2014.01.023
0925-2312

the derivative information is not necessary because it uses a stochastic random search rather than a gradient search. The other important advantage of the KH algorithm is its simplicity. Comparing with other population-based meta-heuristic algorithms, this new approach requires few control variables, in essence only a single parameter C_t (time interval) to regulate (apart from the population size). This feature makes KH easy to implement, more robust, and very appropriate for parallel computation.

KH is a powerful algorithm in exploitation (i.e., local search) but at times it may trap into some local optima so that it cannot perform global search well [20]. For KH, the search depends completely on random walks, so a fast convergence cannot be guaranteed. In order to better KH in optimization problems, two methods have been proposed [21,22], which introduces mutation scheme into KH to add the diversity of population.

Cuckoo search (CS) is another new meta-heuristic search algorithm based on the obligate brood parasitic behavior of some cuckoo species in combination with the Lévy flight behavior of some birds and fruit flies [23]. Walton et al. [24] improved the basic CS algorithm and introduced modified CS (MCS) method. The fist improvement was to adopt the size of the Lévy flight step size instead of a constant step size used in the CS method. The second improvement was to add information exchange between the eggs in an effort to accelerate convergence to the best solutions. In MCS, a fraction of the eggs with the best fitness were put together to form a group of top eggs. Also, all the cuckoos could exchange information through top eggs [24].

The described information exchange concept is introduced to the KH algorithm to develop an improved KH (IKH) method. The main goal is to speed up the algorithm convergence and therefore to provide a more efficient tool for a wider range of practical applications while preserving the attractive characteristics of the basic KH method. Besides, IKH adopts a new Lévy flight distribution and elitism scheme to update the KH motion calculation. The proposed approach is evaluated on 14 standard benchmark functions. Experimental results show that IKH performs better than the basic KH, GA, BA, CS, DE, HS, PSO, probability-based incremental learning (PBIL), and artificial bee colony (ABC) optimization methods.

The structure of this paper is organized as follows: Section 2 describes global numerical optimization problem and the basic KH algorithm in brief. The proposed IKH approach is presented in detail in Section 3. Subsequently, Section 4 presents the validity verification of IKH against different benchmark functions and various optimization algorithms. Finally, Section 5 consists of the conclusion and proposals for future work.

2. Preliminary

In this section, we will provide a brief background on the optimization problem and KH algorithm.

2.1. Optimization problem

In computer science, mathematics, and management science, optimization means the selection of an optimal solution from some set of feasible alternatives. In general, an optimization problem includes minimizing or maximizing a function by systematically selecting input values from a given feasible set and calculating the value of the function. More generally, optimization consists of finding the optimal values of some objective function within a given domain, including a number of different types of domains and different types of objective functions.

A global optimization problem can be described as follows:

Given: a function f: $S \to R$ from some set S to the real numbers

Sought: a parameter x_0 in S such that $f(x_0) \le f(x)$ for all x in S ("minimization") or such that $f(x_0) \ge f(x)$ for all x in S ("maximization").

Such a formulation is named a numerical optimization problem. Many theoretical and practical problems may be modeled in this general framework. In general, S is some subset of the Euclidean space R^n, often specified by a group of constraints, equalities or inequalities that the components of S have to satisfy. The domain S of f is named the search space, while the elements of S are named feasible solutions or candidate solutions. In general, the function f is called an objective function, utility function (maximization), or cost function (minimization). An optimal solution is an available solution that is the extreme of (minimum or maximum) the objective function.

Conventionally, the standard formulation of an optimization problem is stated in accordance with minimization. In general, unless both the feasible region and the objective function are convex in a minimization problem, there may be more than one local minima. A global minimum x^* is defined as a point for which the following expression

$$f(x^*) \le f(x) \tag{1}$$

holds [21].

A variety of algorithms have been proposed to solve nonconvex problems. Among them, heuristics algorithms can evaluate approximate solutions to some optimization problems, as described in introduction.

2.2. Krill herd algorithm

KH is a novel meta-heuristic swarm intelligence optimization method for solving optimization problems [20]. This method is based on the simulation of the herding of the krill swarms in response to specific biological and environmental processes. The time-dependent position of an individual krill in two-dimensional surface is determined by three main actions described as follows:

(i) movement induced by other krill individuals;
(ii) foraging action; and
(iii) random diffusion.

In KH, the Lagrangian model is used in a d-dimensional decision space as shown in Eq. (2) [20].

$$\frac{dX_i}{dt} = N_i + F_i + D_i \tag{2}$$

where N_i is the motion induced by other krill individuals; F_i is the foraging motion, and D_i is the physical diffusion of the ith krill individuals.

2.2.1. Motion induced by other krill individuals

The direction of motion induced, α, is approximately evaluated by the target swarm density (target effect), a local swarm density (local effect), and a repulsive swarm density (repulsive effect). For a krill individual, this movement can be defined as [20]:

$$N_i^{new} = N^{\max}\alpha_i + \omega_n N_i^{old} \tag{3}$$

where

$$\alpha_i = \alpha_i^{local} + \alpha_i^{target} \tag{4}$$

and $N^{\max}$ is the maximum induced speed, ω_n is the inertia weight of the motion induced in [0, 1], N_i^{old} is the last motion induced, α_i^{local} is the local effect provided by the neighbors and α_i^{target} is the target direction effect provided by the best krill individual. According to the experimental values of the maximum induced speed, we set $N^{\max}$ to 0.01 (ms^{-1}) in our study [20].

2.2.2. Foraging motion

The foraging motion is influenced by the two main factors. One factor is the food location and the other one is the previous experience about the food location. For the ith krill individual, this motion can be expressed as follows [20]:

$$F_i = V_f \beta_i + \omega_f F_i^{old} \tag{5}$$

where

$$\beta_i = \beta_i^{food} + \beta_i^{best} \tag{6}$$

and V_f is the foraging speed, ω_f is the inertia weight of the foraging motion between 0 and 1, F_i^{old} is the last foraging motion, β_i^{food} is the food attractive and β_i^{best} is the effect of the best fitness of the ith krill so far. In our study, we set V_f to 0.02 [20].

In KH, the virtual center of food concentration is approximately calculated according to the fitness distribution of the krill individuals, which is inspired from "*center of mass*". The center of food for each iteration is estimated as follows [20]:

$$X^{food} = \frac{\sum_{i=1}^{N} (1/K_i) X_i}{\sum_{i=1}^{N} 1/K_i} \tag{7}$$

where K_i represents the fitness or the objective function value of the ith krill individual.

2.2.3. Physical diffusion

The physical diffusion of the krill individuals is considered to be a random process. This motion can be expressed in terms of a maximum diffusion speed and a random directional vector. It can be formulated as follows [20]:

$$D_i = D^{\max} \delta \tag{8}$$

where $D^{\max}$ is the maximum diffusion speed, and δ is the random directional vector and its arrays are random values in $[-1, 1]$. The better the position of the krill is, the less random the motion is. The effects of the motion induced by other krill individuals and foraging motion gradually decrease with increasing the time (iterations). Thus, another term (Eq. (9)) [20] is added to Eq. (8). This term linearly decreases the random speed with the time and performs on the basis of a geometrical annealing schedule [25]:

$$D_i = D^{\max}\left(1 - \frac{1}{I_{max}}\right)\delta \tag{9}$$

2.2.4. Main procedure of the KH algorithm

In general, the defined motions frequently change the position of a krill individual toward the best fitness. The foraging motion and the motion induced by other krill individuals contain two global and two local strategies. These are working in parallel which make KH a powerful algorithm. Using different effective parameters of the motion during the time, the position vector of a krill individual during the interval t to $t+\Delta t$ is expressed by the following equation [20]:

$$X_i(t+\Delta t) = X_i(t) + \Delta t \frac{dX_i}{dt} \tag{10}$$

It should be noted that Δt is one of the most important constants and should be carefully set according to the optimization problem. This is because this parameter works as a scale factor of the speed vector.

In addition, to improve the performance of the KH, genetic reproduction mechanisms are incorporated into the algorithm. The introduced adaptive genetic reproduction mechanisms are crossover and mutation which are inspired from the classical DE algorithm.

Algorithm 1. Krill herd algorithm.

Begin
    Step 1: Initialization. *Set the generation counter* $G=1$; *initialize the population P of NP krill individuals randomly and each krill corresponds to a potential solution to the given problem; set the foraging speed* V_f, *the maximum diffusion speed* D^{max}, *and the maximum induced speed* N^{max}.
    Step 2: Fitness evaluation. *Evaluate each krill individual according to its position.*
    Step 3: While *the termination criteria is not satisfied* ***or*** $G < MaxGeneration$ ***do***
        Sort the population/krill from best to worst.
        for $i=1{:}NP$ (*all krill*) ***do***
            Perform the following motion calculation.
                Motion induced by the presence of other individuals
                Foraging motion
                Physical diffusion
            Implement the genetic operators.
            Update the krill individual position in the search space.
            Evaluate each krill individual according to its position.
        end for i
        Sort the population/krill from best to worst and find the current best.
        $G=G+1$.
    Step 4: ***end while***
    Step 5: *Post-processing the results and visualization.*
End.

Various krill-inspired algorithms can be developed by idealizing the motion characteristics of the krill individuals. Generally, the KH algorithm can be described by the following steps:

(I) Data structures: define the simple bounds; determine the algorithm parameter(s) etc.
(II) Initialization: randomly create the initial population in the search space.
(III) Fitness evaluation: evaluate each krill individual according to its position.
(IV) Motion calculation:
- Motion induced by other krill individuals,
- Foraging motion
- Physical diffusion

(V) Perform the genetic operators
(VI) Updating: update the krill individual position in the search space.
(VII) Repeating: go to step III until the stop criteria is reached.
(VIII) Post-processing the results and visualization.

The basic representation of the KH can be summarized as shown in Algorithm 1. More details about the three main motions and KH algorithm can be found in [20].

3. The proposed IKH approach

Similar to the strategy proposed in [24] for improving CS, this study introduces an information exchange concept between top krill during the motion calculation process. This results in improving the IKH method while it keeps the attractive features of the original KH method.

Algorithm 2. The algorithm of exchanging information among top krill.

Begin
Step **1:** *Set max Lévy flight step size A and golden ratio* φ.
Step **2:** ***for*** $i=1$*:NoTop (all top krill)* ***do***
Current krill at position x_i
Pick another krill from the top krill at random x_j
if $x_i = x_j$ ***then***
Calculate Lévy flight step size$\varphi \leftarrow A/G^2$
Perform Lévy flight from x_i *to generate new krill* x_k
Evaluate the fitness f_k *for krill* x_k
Choose a random krill l from all krill
if $(f_k > f_l)$
Move krill k towards l;
end if
else
$dx = |x_i - x_j|/\varphi$
Move distance dx from the worst krill to the best krill to find x_k
Evaluate the fitness f_k *for krill* x_k
Choose a random krill l from all krill
if $(f_k > f_l)$ ***then***
Move krill k towards l;
end if
end if
Step 3: ***end for i***
End.

For the KH algorithm, the search depends completely on random walks, thus a fast convergence cannot be guaranteed. To cope with this issue, new features are added to KH.

The first improvement is adding Lévy flight to KH with the step size α. Moreover, in the IKH, the value of α declines as the procedure proceeds (increasing generations). This is done for the similar reasons that the inertia constant is declined in the PSO [19] and basic KH algorithm [20], *i.e*., to stimulate more localized searching as the krill, get closer to the solution.

Algorithm 3. Improved KH algorithm.

Begin
Step **1:** ***Initialization.*** *Set the generation counter* $t=1$; *initialize the population P of NP krill individuals randomly and each krill corresponding to a potential solution to the given problem; set the foraging speed* V_f*, the maximum diffusion speed* D^{max}*, and the maximum induced speed* N^{max}*; set max Lévy flight step size A, golden ratio* φ *and the fraction of krill placed in the top krill group* p_a*; set elitism parameter KEEP: how many of the best krill to keep from one generation to the next, here* $KEEP = \lceil NP * (1-p_a) \rceil$
Step **2:** ***Fitness evaluation****. Evaluate each krill individual according to its position.*
Step **3:** ***While*** *the termination criteria is not satisfied or* $t < MaxGeneration$ ***do***
Sort the population/krill from best to worst.
Store the KEEP best krill as KEEPKRILL.
for $i=1$: $\lceil NP * p_a \rceil$*(all top frill)* ***do***
Perform the following motion calculation.
Motion induced by the presence of other individuals
Forage motion
Exchange information between top krill by Algorithm 2
Update the krill individual position in the search space.
Evaluate each krill individual according to its position.
end for i
Replace the KEEP worst krill with the KEEP best krill stored in KEEPKRILL.
Sort the population/krill from best to worst and find the current best.
$t = t+1$;
Step 4: ***end while***
Step **5:** *Post-processing the results and visualization;*
End.

The second improvement is to add information exchange between the krill in an attempt to accelerate the convergence speed to the best solution. In KH, there is no information exchange between krill and the searches are implemented independently in essence, *i.e*., different krill work almost independently [20]. In the IKH, portions of the krill with the best fitness are made up of a group of top krill. For every top krill, a second krill in this group is selected randomly and a new krill is then produced on the line connecting these two top krill. The distance along this line at which the new krill is situated is calculated, using the inverse of the golden ratio $\varphi = (1+\sqrt{5})/2$, such that it is closer to the krill with the best fitness. In the case that both krill have the same fitness, the new krill is produced at the middle point. In this step, there is a possibility that, the same krill is selected twice. In this case, a local Lévy flight search is carried out from the randomly selected krill with step size $\alpha = A/G^2$. The detailed procedure for exchanging information between top krill involved in the improved krill herd algorithm is presented in Algorithm 2, and the basic framework of improved krill herd algorithm can be simply described as shown in Algorithm 3. In IKH, $\lceil NP * p_a \rceil$ is an integer number whose value is not more than $NP * p_a$. From Algorithm 2 and Algorithm 3, it is clear that there is only one parameter, the parameter of the fraction of krill to make up the top krill, which needs to be regulated in IKH. Through testing on benchmarking functions, it was found that setting the parameter of the fraction of krill placed in the top krill group to 0.25 produced the best results through a series of simulation experiments.

The third improvement is the addition of elitism scheme into IKH. As considered for other population-based optimization algorithms, we typically incorporate some sort of elitism in order to retain the best solutions in the population. This prevents the best solutions from being corrupted by motion calculation operator. Note that we use an elitism approach to save the property of the krill that has the best fitness in the IKH process. Hence, even if the motion calculation operation ruins its corresponding krill, we have saved it and can revert back to it if needed.

4. Simulation experiments

In this section, we test the performance of the proposed metaheuristic IKH algorithm through a series of experiments. In the following experiments, IKH1, IKH2, and IKH3 are corresponding to the three improvements discussed in Section 3, respectively.

To allow a fair comparison of running times, all the experiments were conducted on a PC with a Pentium IV processor running at 2.0 GHz, 512 MB of RAM and a hard drive of 160 Gbytes. Our implementation was compiled using MATLAB R2012a (7.14) running under Windows XP3. No commercial KH tool was used in the following experiments.

4.1. General performance of IKH

In order to explore the benefits of IKH, we compared with nine other population-based optimization methods, which are ABC, BA, CS, DE, GA, HS, PBIL, and PSO. [26] is a classical swarm intelligence method based on the smart behavior of honey bee swarm. BA [27] is a novel powerful meta-heuristic optimization method inspired by the echolocation behavior of bats with varying pulse rates of emission and loudness. CS [23,28] is a meta-heuristic optimization approach inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds. DE [14,29,30] is a simple but excellent optimization method that uses the difference between two solutions to probabilistically adapt a third solution. An ES [31,32] is an algorithm that generally distributes equal importance to mutation and recombination, and that allows two or more parents to reproduce an offspring. A GA [11] is a search heuristic that mimics the process of natural evolution. HS [33] is a new meta-heuristic method inspired by behavior of musician' improvisation process. PBIL [34] is a type of genetic algorithm where the genotype of an entire population (probability vector) is evolved rather than individual members. PSO [19,35] is also a swarm intelligence algorithm which is based on the swarm behavior of fish, and bird schooling in nature. It is worth mentioning that according to Gandomi and Alavi [20], the KH II (KH with crossover operator) has the best performance. Therefore, in this study, we use KH II as the basic KH algorithm.

In all experiments, we will use the same parameters for KH and IKH that are the fraction of krill placed in the top krill group p_a=0.25, the foraging speed V_f=0.02, the maximum diffusion speed $D^{\max}$=0.005, the maximum induced speed $N^{\max}$=0.01. In addition, the inertia weights (ω_n, ω_f) are equal to 0.9 at the beginning of the search to emphasize exploration. These two parameters are linearly decreased to 0.1 at the end to encourage exploitation [20]. For DE, GA, PBIL and PSO, we set the parameters as follows [36]. For ABC, the number of colony size (employed bees and onlooker bees) *NP*=50, the number of food sources *Food Number*=*NP*/2, maximum search times *limit*=100 (a food source which could not be improved through "limit" trials is abandoned by its employed bee). For BA, we set loudness A=0.95, pulse rate r=0.5, and scaling factor ε=0.1; for CS, a discovery rate p=0.25. For HS, we set harmony memory accepting rate=0.75, and pitch adjusting rate=0.7.

Well-defined problem sets are favorable for evaluating the performance of optimization methods proposed in this paper. Based on mathematical functions, benchmark functions can be applied as objective functions to perform such tests. The properties of these benchmark functions can be easily achieved from their definitions. Fourteen different benchmark functions are applied to verify our proposed meta-heuristic IKH algorithm. The benchmark functions described in Table 1 are standard testing functions. The properties of the benchmark functions are given in Table 2. The modality property means the number of the best solutions in the search space. More details of all the benchmark functions can be found in [37].

We set population size *NP*=50 and maximum generation *Maxgen*=50 for each algorithm. We ran 100 Monte Carlo simulations of each algorithm on each benchmark function to get representative performances. Tables 3 and 4 illustrate the results of the simulations. Table 3 shows the average minima found by each algorithm, averaged over 100 Monte Carlo runs. Table 4 shows the absolute best minima found by each algorithm over 100 Monte Carlo runs. In other words, Table 3 shows the average performance of each algorithm, while Table 4 shows the best

Table 1
Benchmark functions.

No	*Name*	*Definition*
F01	Ackley	$f(\vec{x})=20+e-20\cdot e^{-0.2\cdot\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}}-e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$
F02	Fletcher–Powell	$f(\vec{x})=\sum_{i=1}^{n}(A_i-B_i)^2, A_i=\sum_{j=1}^{n}(a_{ij}\ \sin\ \alpha_j+b_{ij}\ \cos\ \alpha_j)$ $B_i=\sum_{j=1}^{n}(a_{ij}\ \sin\ x_j+b_{ij}\ \cos\ x_j)$
F03	Griewank	$f(\vec{x})=\sum_{i=1}^{n}\frac{x_i^2}{4000}-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$
F04	Penalty #1	$f(\vec{x})=\frac{\pi}{30}\Big\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\bullet[1+10\sin^2(\pi y_{i+1})]$ $+(y_n-1)^2\Big\}+\sum_{i=1}^{n}u(x_i,10,100,4),\ \ y_i=1+0.25(x_i+1)$
F05	Penalty #2	$f(\vec{x})=0.1\Big\{\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2\bullet[1+\sin^2(3\pi x_{i+1})]$ $+(x_n-1)^2[1+\sin^2(2\pi x_n)]\Big\}+\sum_{i=1}^{n}u(x_i,5,100,4)$
F06	Quartic *with noise*	$f(\vec{x})=\sum_{i=1}^{n}(i\cdot x_i^4+U(0,1))$
F07	Rastrigin	$f(\vec{x})=10\cdot n+\sum_{i=1}^{n}(x_i^2-10\cdot\ \cos(2\pi x_i))$
F08	Rosenbrock	$f(\vec{x})=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$
F09	Schwefel 2.26	$f(\vec{x})=418.9829\times D-\sum_{i=1}^{D}x_i\ \sin(\lvert x_i\rvert^{1/2})$
F10	Schwefel 1.2	$f(\vec{x})=\sum_{i=1}^{n}\left(\sum_{j=1}^{i}x_j\right)^2$
F11	Schwefel 2.22	$f(\vec{x})=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$
F12	Schwefel 2.21	$f(\vec{x})=\max_i\{\lvert x_i\rvert, 1\le i\le n\}$
F13	Sphere	$f(\vec{x})=\sum_{i=1}^{n}x_i^2$
F14	Step	$f(\vec{x})=6\cdot n+\sum_{i=1}^{n}\lfloor x_i\rfloor$

*In benchmark function F02, the matrix elements $\mathbf{a}_{n\times n},\mathbf{b}_{n\times n}\in(-100,100),\boldsymbol{\alpha}_{n\times 1}\in(-\pi,\pi)$ are drawn from uniform distribution.
*In benchmark functions F04 and F05, the definition of the function $u(x_i,a,k,m)$ is as follows:

$$u(x_i,a,k,m)=\begin{cases} k(x_i-a)^m, & x_i>a \\ 0, & -a\le x_i\le a \\ k(-x_i-a)^m, & x_i<-a \end{cases}$$

performance of each algorithm. The best value achieved for each test problem is shown in bold. Note that the normalizations in the tables are based on different scales, so values cannot be compared between the two tables. Each of the functions in this study has 20 independent variables (*i.e.*, $d=20$).

From Table 3, we see that, on average, IKH1 and IKH2 perform slightly different and are the most effective at finding objective function minimum on six of the 14 benchmarks (F06, F07, F10, F11, F13-F14 and F02-F05, F08, F14). IKH3 is the second most effective, performing best on two of the 14 benchmarks (F01 and F12); while

Table 2
Properties of benchmark functions, *lb* denotes lower bound, *ub* denotes upper bound, *opt* denotes optimum point.

No.	*Function*	*lb*	*ub*	*opt*	*Continuity*	*Modality*
F01	Ackley	−32.768	32.768	0	Continuous	Multimodal
F02	Fletcher–Powell	$-\pi$	π	0	Continuous	Multimodal
F03	Griewangk	−600	600	0	Continuous	Multimodal
F04	Penalty #1	−50	50	0	Continuous	Multimodal
F05	Penalty #2	−50	50	0	Continuous	Multimodal
F06	Quartic *with noise*	−1.28	1.28	1	Continuous	Multimodal
F07	Rastrigin	−5.12	5.12	0	Continuous	Multimodal
F08	Rosenbrock	−2.048	2.048	0	Continuous	Unimodal
F09	Schwefel 2.26	−512	512	0	Continuous	Multimodal
F10	Schwefel 1.2	−100	100	0	Continuous	Unimodal
F11	Schwefel 2.22	−10	10	0	Continuous	Unimodal
F12	Schwefel 2.21	−100	100	0	Continuous	Unimodal
F13	Sphere	−5.12	5.12	0	Continuous	Unimodal
F14	Step	−5.12	5.12	0	Discontinuous	Unimodal

Table 3
Mean normalized optimization results in 14 benchmark functions. The values shown are the minimum objective function values found by each algorithm, averaged over 100 Monte Carlo simulations.

	ABC	BA	CS	DE	GA	HS	IKH1	IKH2	IKH3	KH	PBIL	PSO
F01	3.08	4.49	2.54	2.85	3.91	4.46	1.11	1.16	**1.00**	1.23	4.54	3.75
F02	1.44	7.51	1.17	2.15	2.26	5.06	1.04	**1.00**	1.98	7.51	4.99	4.65
F03	9.25	50.73	3.56	5.05	9.64	46.64	1.01	**1.00**	1.40	7.44	55.33	18.97
F04	3.6E5	1.9E7	1.5E3	7.1E4	1.9E5	1.4E7	5.53	**1.00**	3.5E3	1.2E6	1.8E7	1.4E6
F05	914.27	2.6E4	33.73	340.31	411.80	1.9E4	1.01	**1.00**	24.63	2.0E3	2.3E4	2.9E3
F06	85.26	1.4E3	10.36	34.89	75.12	1.2E3	**1.00**	1.26	10.46	202.64	1.4E3	231.00
F07	2.58	7.35	3.01	4.28	4.55	6.50	**1.00**	1.02	2.61	5.02	7.05	5.07
F08	8.96	54.64	2.86	7.30	13.41	42.93	1.08	**1.00**	3.15	14.41	52.03	14.97
F09	1.90	4.39	2.05	2.44	**1.00**	3.68	1.41	1.44	2.36	4.18	3.82	3.66
F10	9.26	18.68	2.48	11.00	8.33	11.69	**1.00**	1.02	5.53	9.97	12.05	8.65
F11	2.99	12.80	2.66	3.26	5.87	9.75	**1.00**	1.11	4.53	11.66	9.82	7.28
F12	6.48	6.95	2.75	5.28	5.39	6.62	1.17	1.09	**1.00**	1.41	6.78	5.27
F13	11.41	66.67	4.68	6.43	23.79	60.99	**1.00**	1.02	1.80	9.72	70.54	25.08
F14	7.46	42.69	3.06	3.86	7.14	37.56	**1.00**	**1.00**	1.02	6.07	45.14	14.87
Total	0	0	0	0	1	0	6	6	2	0	0	0

*The values are normalized so that the minimum in each row is 1.00. These are not the absolute minima found by each algorithm, but the average minima found by each algorithm.

Table 4
Best normalized optimization results in 14 benchmark functions. The values shown are the minimum objective function values found by each algorithm.

	ABC	BA	CS	DE	GA	HS	IKH1	IKH2	IKH3	KH	PBIL	PSO
F01	4.26	7.32	3.10	4.36	5.72	7.22	1.42	1.37	**1.00**	1.14	7.50	5.94
F02	2.43	10.45	1.54	3.01	1.60	10.64	1.06	**1.00**	3.35	12.98	6.53	7.85
F03	7.06	36.55	3.29	5.16	6.24	60.35	1.02	**1.00**	1.23	8.38	66.77	22.73
F04	3.2E3	7.8E6	7.92	7.5E3	15.08	1.2E7	1.11	**1.00**	13.01	1.2E6	2.5E7	6.0E5
F05	5.7E4	5.0E6	530.77	8.2E3	2.9E3	5.2E6	1.27	**1.00**	868.43	9.9E5	1.2E7	6.8E5
F06	13.92	3.6E3	23.02	133.00	21.16	5.4E3	1.49	**1.00**	30.25	765.35	7.1E3	399.65
F07	3.33	12.03	4.66	7.44	6.81	10.90	1.02	**1.00**	3.32	8.45	12.76	7.57
F08	7.17	26.15	2.07	7.06	7.39	40.91	**1.00**	1.02	3.23	14.27	41.82	12.59
F09	3.61	9.75	3.56	5.05	**1.00**	8.29	2.05	2.40	4.62	8.58	8.43	6.90
F10	26.22	26.28	4.19	31.86	16.39	32.18	1.96	**1.00**	13.94	27.80	26.58	23.26
F11	4.29	18.73	1.98	4.76	6.76	16.12	**1.00**	1.29	4.51	16.12	16.34	8.99
F12	12.37	12.43	4.52	11.80	8.78	12.37	1.72	1.47	**1.00**	2.11	12.93	8.27
F13	8.66	64.92	6.16	11.16	19.90	65.99	1.13	**1.00**	2.07	15.02	134.42	40.88
F14	13.05	91.79	6.39	8.68	3.19	82.29	1.52	**1.00**	1.39	13.21	110.11	32.75
Total	0	0	0	0	1	0	2	9	2	0	0	0

*The values are normalized so that the minimum in each row is 1.00. These are the absolute best minima found by each algorithm.

GA is the third most effective, performing best on one benchmark (F09) when multiple runs are made. For the best solutions, Table 4 shows that IKH2 has the best performance on nine of the 14 benchmarks (F02-F07, F10, F13, and F14). IKH1 and IKH3 are the second most effective, performing the best on two of the 14 benchmarks F08, F11 and F01, F12 when multiple runs are made respectively. GA is the third most effective, performing the best on only one benchmark F09 when multiple runs are made. In addition, statistical analysis on these values obtained by the 10 methods on 14 benchmark functions based on the Friedman's test [38] reveals that the differences in the obtained average and best function minima are statistically significant ($p=1.64\times10^{-17}$ and $p=7.06\times10^{-17}$, respectively) at the confidence level of 5%.

Furthermore, to further prove the merits of the proposed IKH method, convergence plots of ABC, BA, CS, DE, GA, HS, IKH1, IKH2, IKH3, KH, PBIL, and PSO are illustrated in this section. However, here only some most representative benchmarks are illustrated Figs. 1–7. The solutions shown in Figs. 1–7 are the average objective function minimum obtained from 100 Monte Carlo simulations, which are the accurate objective function solution, not normalized. In addition, note that the best solutions of the benchmarks (F04, F05, and F14) are represented in the form of the semi-logarithmic convergence plots. We use KH short for KH II in the legend of Figs. 1–7.

Fig. 1 shows the results obtained for the 12 methods when the F01 Ackley function is applied. This is a multimodal function with a narrow global minimum basin ($F01_{min}=0$) and many minor local optima. From Fig. 1, we can draw the conclusion that, IKH3 is significantly superior to the other algorithms during the process of optimization, while IKH1 and IKH2 performs the second and the third best in this multimodal benchmark function, respectively.

Fig. 2 shows the results for F04 Penalty #1 function. From Fig. 2, apparently, IKH2 outperforms all other methods in this example. At last, IKH1 converges to the value that is very close to IKH2's. While, IKH1 performs the fourth best that is inferior to CS.

Fig. 3 shows the results for F05 Penalty #2 function. From Fig. 3, for this multimodal function, very similar to F02 Penalty #1 function, IKH2 performs the best that outperforms all other methods during the process of optimization, and IKH1 performs the second best that slightly inferior to IKH2 among 12 methods.

Fig. 4 shows the optimization results for the F07 Rastrigin function, which is a complex multimodal problem with a unique global minimum of $F07_{min}=0$ and a large number of local optima. When attempting to solve F07, methods may easily trap into a local optimum. Hence, a method capable of maintaining a larger diversity is likely to produce better results. As can be seen in Fig. 4, there is little difference between the performance of IKH1 and IKH2. However, carefully studying Table 3 and Fig. 4, we can conclude that, IKH1 performs slightly better than IKH2 in this multimodal function. For the other algorithms, similar to IKH1 and IKH2, there is little difference between the performance of ABC and IKH3. In effect, ABC performs slightly better than IKH3 in this multimodal function.

Fig. 5 shows the results for F10 Schwefel 1.2 function. From Fig. 5, similar to F07 Rastrigin function as shown in Fig. 4, the figure shows that there is little difference between the performance of IKH1 and IKH2. However, carefully studying Table 3 and Fig. 5, we can conclude that, IKH1 performs slightly better than IKH2 in this relative simple unimodal benchmark function. For other algorithms, CS works very well, because it ranks 3 among twelve methods.

Fig. 6 shows the results for F12 Schwefel 2.21 function. It is obvious that IKH1, IKH2, IKH3 and KH perform the best and significantly outperform other algorithms. However, carefully studying Table 3 and Fig. 6, we can conclude that, IKH3 is superior to IKH2, IKH1 and KH in the whole optimization progress.

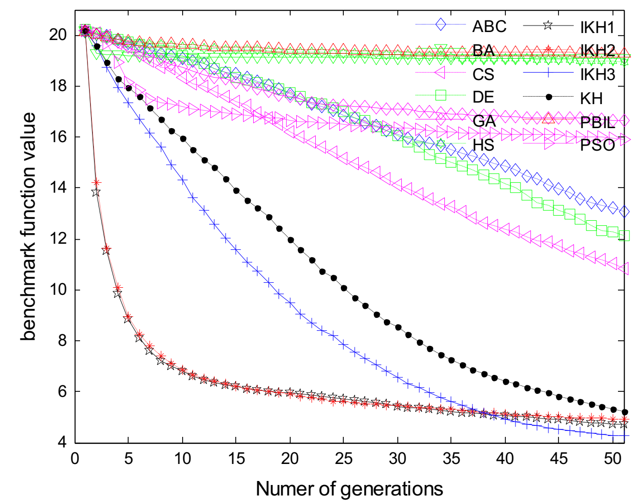

Fig. 1. Comparison of the performance of the different methods for the F01 Ackley function.

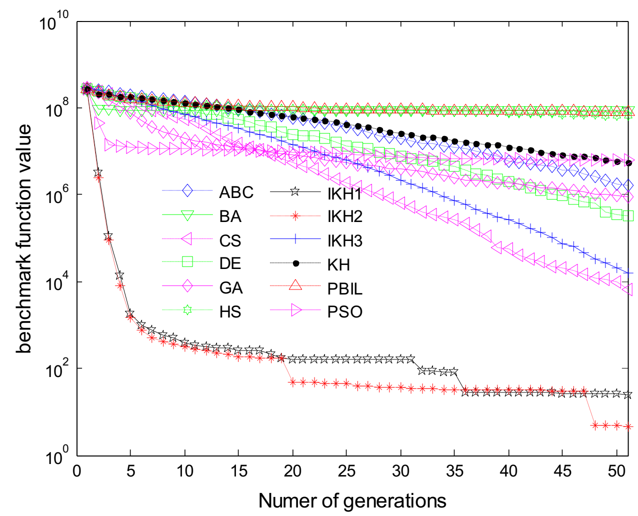

Fig. 2. Comparison of the performance of the different methods for the F04 Penalty #1 function.

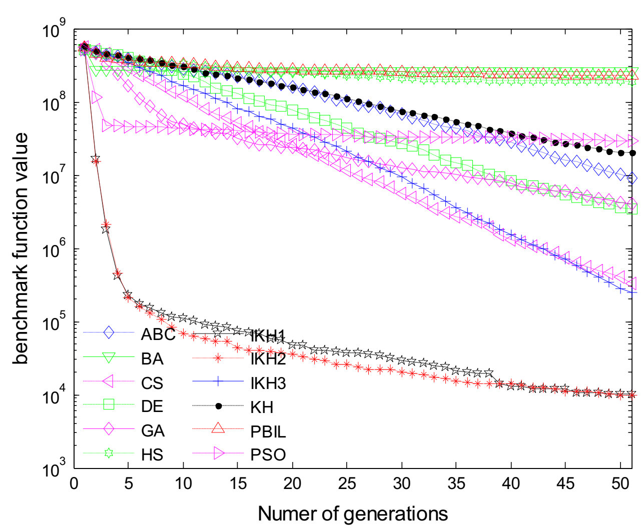

Fig. 3. Comparison of the performance of the different methods for the F05 Penalty #2 function.

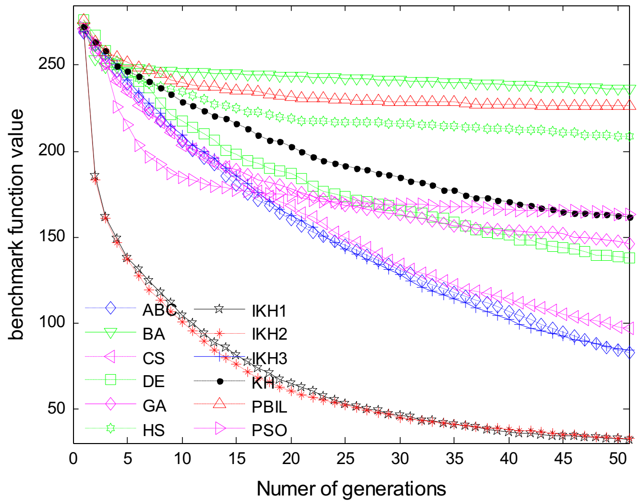

Fig. 4. Comparison of the performance of the different methods for the F07 Rastrigin function.

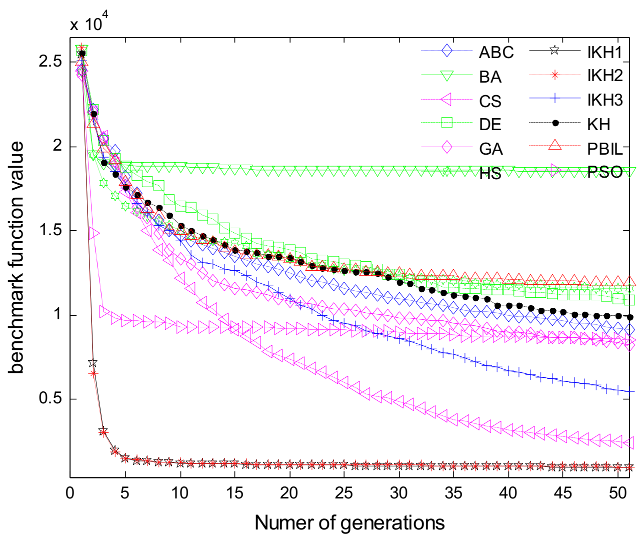

Fig. 5. Comparison of the performance of the different methods for the F10 Schwefel 1.2 function.

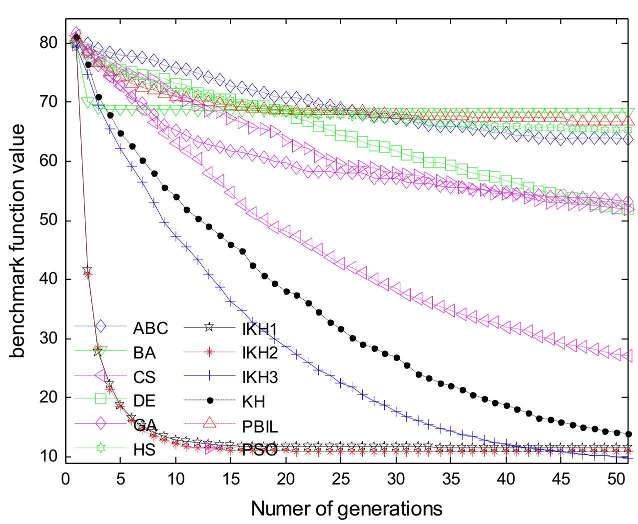

Fig. 6. Comparison of the performance of the different methods for the F12 Schwefel 2.21 function.

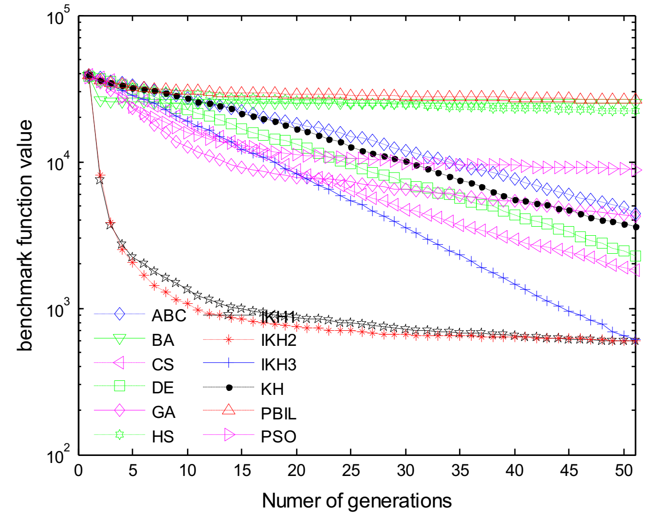

Fig. 7. Comparison of the performance of the different methods for the F14 Step function.

Fig. 7 shows the results for F14 Step function. Apparently, IKH1 and IKH2 perform almost the same during the whole optimization process. Eventually, they converge to the same final optimal values; while, IKH3 is only inferior to IKH1 and IKH2, and converges to the value that is very close to the IKH1 and IKH2.

From above-analyses about the Figs. 1–7 and Table 3, we can arrive at a conclusion that, on average, among 12 optimization methods, our proposed IKH1 an IKH2 approach perform the best and most effectively when solving the global numerical optimization problems and significantly outperforms the other ten approaches. Generally speaking, IKH3 and KH are only inferior to IKH1 an IKH2, and perform the second best among 12 methods. CS and ABC perform the third best only inferior to the IKH1, IKH2, IKH3 and KH.

4.2. *Influence of control parameter*

The choice of the control parameters is of vital importance for different problems. To compare the different effects on the parameter of the fraction of fireflies placed in the top krill group p_a, we ran 100 Monte Carlo simulations of the IKH2 algorithm on the above problem to get the best performances. All other parameter settings are kept unchanged (unless noted otherwise in the following paragraph). The results are recorded in Tables 5–6 after 100 Monte Carlo runs. Among them, Table 5 shows the best minima found by IKH2 algorithm over 100 Monte Carlo runs. Table 6 shows the average minima found by the IKH2 algorithm, averaged over 100 Monte Carlo runs. In other words, Tables 5 and 6 show the best and average performance of IKH2 algorithm respectively. In each table, the last row is the total number of functions on which IKH2 performs the best with specific parameters.

Tables 5 and 6 recorded the results performed on the benchmark problems with the fraction of krill placed in the top krill group p_a=0, 0.1, 0.2, 0.9, 1.0. From Tables 5 and 6, it can be seen that: (i) for the three benchmark functions F01, F02, F03 and F08, IKH2 performs slightly differently, that is to say, these three benchmark functions are insensitive to the parameter p_a. (ii) For benchmark functions F04–F07, F09, F11–F14, IKH2 performs better on smaller p_a (<0.5). (iii) However, there are very few benchmark functions that IKH2 performs better on bigger p_a (>0.5). As can be observed from Tables 5 and 6 , IKH2 performs the best in most benchmarks when p_a is equal or very close to 0.2 and 0.3. Hence, we set p_a=0.25 in other experiments. In addition, statistical

Table 5
Best normalized optimization results in 14 benchmark functions with different p_a. The numbers shown are the best results found after 100 Monte Carlo simulations of IKH2 algorithm.

	p_a										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F02	2.9E3	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F03	6.03	2.3	11.00	2.31	2.31	2.31	2.31	2.31	2.31	2.31	2.31
F04	1.69	1.69	106.66	**1.00**	1.69	1.13	1.69	1.69	1.69	1.69	1.69
F05	7.28	2.31	2.31	6.8E3	**1.00**	2.31	2.31	2.31	2.31	2.31	2.31
F06	**1.00**	52.17	61.84	34.69	10.38	56.16	4.24	32.21	8.03	46.64	2.45
F07	4.55	1.36	**1.00**	2.31	2.31	11.61	2.31	2.31	2.31	2.31	2.31
F08	4.62	2.61	**1.00**	**1.00**	**1.00**	1.01	8.47	**1.00**	**1.00**	**1.00**	**1.00**
F09	305.76	7.28	2.31	**1.00**	2.31	2.31	2.31	508.16	2.31	2.31	2.31
F10	192.19	6.03	1.36	**1.00**	2.31	2.31	2.31	2.31	1.1E3	2.31	2.31
F11	759.73	**1.00**	751.20	751.20	325.79	751.20	751.20	751.20	751.20	2.4E3	751.20
F12	23.60	**1.00**	46.71	14.40	37.65	14.40	14.40	14.40	14.40	14.40	27.86
F13	**1.00**	18.55	2.29	31.79	48.74	23.34	53.81	18.73	40.81	29.05	53.81
F14	109.99	1.36	**1.00**	2.31	2.31	6.03	2.31	2.31	2.31	2.31	2.31
	3	4	6	6	4	2	2	3	3	3	3

*The values are normalized so that the minimum in each row is 1.00. These are not the absolute minima found by each algorithm, but the average minima found by each algorithm.

Table 6
Mean normalized optimization results in 14 benchmark functions with different p_a. The numbers shown are the best results found after 100 Monte Carlo simulations of IKH2 algorithm.

	p_a										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F02	180.37	2.40	**1.00**	1.63	2.37	1.90	2.17	2.11	3.07	2.78	1.78
F03	1.80	1.60	**1.00**	1.01	**1.00**	**1.00**	**1.00**	**1.00**	1.03	1.01	1.01
F04	5.2E3	125.22	9.4E3	**1.00**	1.21	1.13	1.02	1.05	27.85	2.21	13.95
F05	220.69	4.16	**1.00**	23.33	2.82	5.45	1.32	2.14	2.59	3.80	1.41
F06	**1.00**	2.9E5	1.6E3	3.9E3	2.9E5	31.18	31.16	31.17	31.17	31.18	31.16
F07	4.31	5.1E3	**1.00**	51.28	123.01	9.2E3	1.04	1.78	1.02	1.05	1.01
F08	10.15	118.71	8.9E3	81.06	49.92	119.73	9.0E3	**1.00**	1.08	1.08	1.04
F09	82.69	4.4E4	1.10	**1.00**	42.75	26.63	62.92	4.7E3	1.18	1.41	1.23
F10	110.21	616.66	1.9E3	3.4E3	46.43	32.44	21.03	47.90	3.5E3	**1.00**	1.26
F11	37.91	**1.00**	1.8E3	14.58	25.57	1.8E3	1.2E3	740.76	1.8E3	1.3E5	14.76
F12	2.44	8.9E3	9.0E3	**1.00**	9.0E3	95.89	119.24	81.86	50.37	120.85	9.1E3
F13	**1.00**	8.14	1.7E5	9.7E3	1.88	3.38	186.26	231.59	158.98	97.82	234.73
F14	85.17	2.0E3	651.82	50.59	**1.00**	3.6E3	44.82	39.44	48.92	34.62	20.77
	3	2	5	4	3	2	2	3	1	2	1

*The values are normalized so that the minimum in each row is 1.00. These are the absolute best minima found by each algorithm.

analysis on these values obtained by the IKH2 with the fraction of krill placed in the top krill group p_a on 14 benchmark functions based on the Friedman's test reveals that the differences in the obtained average and best function minima across various chaotic maps are statistically significant ($p = 7.6 \times 10^{-16}$ and $p = 6.7 \times 10^{-16}$, respectively) at the confidence level of 5%.

4.3. Discussion

For all of the standard benchmark functions that have been considered, IKH performs better than or at least highly competitive with the standard KH and other acclaimed state-of-the-art population-based algorithms. The IKH performs excellently and efficiently because of its ability to simultaneously carry out a local search, still searching globally at the same time. It succeeds in doing this due to the information exchange between the top krill and global search via Lévy flights concurrently. A similar behavior may be performed in the PSO by using multi-swarm from a particle population initially. However, IKH's advantages include performing simpy and easily, and using only one parameter to regulate.

Benchmark evaluation is a good way for verifying the performance of the meta-heuristic algorithms, but it also has limitations. First, we did not make any special effort to tune the optimization algorithms in this section. Different tuning parameter values in the optimization algorithms might result in significant differences in their performance. Second, real-world optimization problems may not have much of a relationship to benchmark functions. Third, benchmark tests might result in different conclusions if the grading criteria or problem setup change. In our work, we examined the mean and best results obtained with a certain population size and after a certain number of generations. However, we might arrive at different conclusions if (for example) we change the generation limit, or look at how many generations it takes to reach a certain function value, or if we change the population size. In spite of these caveats, the benchmark results shown here are promising for IKH, and indicate that this novel method might be able to find a niche among the plethora of population-based optimization algorithms.

In this work, 14 benchmark functions are used to evaluate the performance of our approach. In future, will test our approach on more problems, such as the high-dimensional ($d \geq 20$) CEC 2010 test suit [39] and the real-world problems. Moreover, we will compare IKH with other EAs. In addition, we only consider the unconstrained function optimization in this work. Our future work consists of adding the diversity rules into IKH for constrained optimization problems, such as constrained real-parameter optimization CEC 2010 test suit [40].

5. Conclusion and future work

This paper proposed an improved meta-heuristic IKH method for optimization problem. A novel type of KH has been presented which introduces three improvements to the original method. The first improvement is adding Lévy flight to the KH with the step size α, similar to declined inertia constant in the PSO [19] and basic KH algorithm [20]. This strategy can inspire more localized searching as the krill get closer to the solution. Information exchange between the top krill is added to the method as the second improvement in an effort to accelerate the convergence speed to the best solution. In IKH, portions of the krill with the best fitness are made up of a group of top krill. The third improvement is the addition of elitism scheme into IKH. This prevents the optimal krill from being corrupted by three motion calculation operators. The IKH attempts to take merits of the KH and exchange information in order to avoid all krill getting trapped in inferior local optimal regions. This new method can speed up the global convergence rate without losing the strong robustness of the basic KH. From the analysis of the experimental results, it can be concluded that the proposed IKH method uses the information in past solutions more efficiently when compared to the other population-based optimization algorithms such as ABC, BA, CS, DE, GA, HS, KH, PBIL, and PSO. Based on the results, IKH significantly improves the performance of KH on most multimodal and unimodal problems. In addition, IKH is simple and easy to implement.

Acknowledgments

This work was supported by the State Key Laboratory of Laser Interaction with Material Research Fund under Grant no. SKLLIM0902-01 and the Key Research Technology of Electric-discharge Non-chain Pulsed DF Laser under Grant no. LXJJ-11-Q80.

References

[1] S.M. Farahani, A.A. Abshouri, B. Nasiri, M. Meybodi, Some hybrid models to improve Firefly algorithm performance, Int. J. Artif. Intell. 8 (2012) 97–117.
[2] X.S. Yang, Nature-Inspired Metaheuristic Algorithms, Luniver Press, Frome, 2010.
[3] H.-C. Lu, M.-H. Chang, C.-H. Tsai, Parameter estimation of fuzzy neural network controller based on a modified differential evolution, Neurocomputing 89 (2012) 178–192.
[4] X. Hong, S. Chen, The system identification and control of Hammerstein system using non-uniform rational B-spline neural network and particle swarm optimization, Neurocomputing 82 (2012) 216–223.
[5] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, Dynamic deployment of wireless sensor networks by biogeography based optimization algorithm, J. Sens. Actuator Netw. 1 (2012) 86–96.
[6] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, A modified firefly algorithm for UCAV path planning, Int. J. Hybrid Inf. Technol. 5 (2012) 123–144.
[7] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, J. Wang, A hybrid meta-heuristic DE/CS algorithm for UCAV path planning, J. Inform. Comput. Sci. 9 (2012) 4811–4818.
[8] H. Duan, W. Zhao, G. Wang, X. Feng, Test-sheet composition using analytic hierarchy process and hybrid metaheuristic algorithm TS/BBO, Math. Probl. Eng. 2012 (2012) 1–22.
[9] X.S. Yang, A.H. Gandomi, S. Talatahari, A.H. Alavi, Metaheuristics in Water, Geotechnical and Transport Engineering, Elsevier, Waltham, MA, 2013.
[10] A.H. Gandomi, X.S. Yang, S. Talatahari, A.H. Alavi, Metaheuristic Applications in Structures and Infrastructures, Elsevier, Waltham, MA, 2013.
[11] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine learning, Addison-Wesley, New York, 1998.
[12] H. He, Y. Tan, A two-stage genetic algorithm for automatic clustering, Neurocomputing 81 (2012) 49–59.
[13] R. Vatankhah, S. Etemadi, A. Alasty, G.-R. Vossoughi, M. Boroushaki, Active leading through obstacles using ant-colony algorithm, Neurocomputing 88 (2012) 67–77.
[14] R. Storn, K. Price, Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359.
[15] A.H. Gandomi, X.-S. Yang, S. Talatahari, S. Deb, Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization, Comput. Math. Appl. 63 (2012) 191–200.
[16] G. Wang, L. Guo, A novel hybrid bat algorithm with harmony search for global numerical optimization, J. Appl. Math. 2013 (2013) 1-21. doi:10.1155/2013/696491.
[17] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, Path planning for UCAV using bat algorithm with mutation, Sci. World J. 2012 (2012) 1–15.
[18] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, M. Shao, Hybridizing harmony search with biogeography based optimization for global numerical optimization, J. Comput. Theor. Nanosci. 10 (2013) 2318-2328.
[19] J. Kennedy, R. Eberhart, Particle swarm optimization, In: Proceedings of the IEEE International Conference on Neural Networks, IEEE, Perth, Australia, 1995, pp. 1942–1948.
[20] A.H. Gandomi, A.H. Alavi, Krill Herd: a new bio-inspired optimization algorithm, Commun. Nonlinear Sci. Numer. Simul. 17 (2012) 4831–4845.
[21] G. Wang, L. Guo, H. Wang, H. Duan, L. Liu, J. Li, Incorporating mutation scheme into krill herd algorithm for global numerical optimization, Neural Comput. Appl. 24 (2014) 853-871.
[22] G. Wang, L. Guo, A.H. Gandomi, L. Cao, A.H. Alavi, H. Duan, J. Li, Lévy-flight krill herd algorithm, Math. Probl. Eng. (2012).
[23] X.S. Yang, S. Deb, Engineering optimisation by cuckoo search, Int. J. Math. Model. Numer. Optim 1 (2010) 330–343.
[24] S. Walton, O. Hassan, K. Morgan, M.R. Brown, Modified cuckoo search: a new gradient free optimisation algorithm, Chaos Soliton Fract. 44 (2011) 710–718.
[25] S. Kirkpatrick, C.D. Gelatt , M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[26] D. Karaboga, B. Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, J. Glob. Optim. 39 (2007) 459–471.
[27] X.S. Yang, A.H. Gandomi, Bat algorithm: a novel approach for global engineering optimization, Eng. Comput. 29 (2012) 464–483.
[28] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, M. Shao, A hybrid meta-heuristic DE/CS algorithm for UCAV three-dimension path planning, Sci. World J. 2012 (2012) 1–11.
[29] I. Triguero, J. Derrac, S. García, F. Herrera, Integrating a differential evolution feature weighting scheme into prototype generation, Neurocomputing 97 (2012) 332–343.
[30] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, M. Shao, Path planning for uninhabited combat aerial vehicle using hybrid meta-heuristic DE/BBO algorithm, Adv. Sci. Eng. Med. 4 (2012) 550–564.
[31] T. Back, Evolutionary Algorithms in Theory and Practice, Oxford University Press, Oxford, 1996.
[32] H. Beyer, The Theory of Evolution Strategies, Springer, New York, 2001.
[33] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search, Simulation 76 (2001) 60–68.
[34] B. Shumeet, Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning, Carnegie Mellon University, Pittsburgh PA, 1994.
[35] Y. Sun, L. Zhang, X. Gu, A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems, Neurocomputing 98 (2012) 76–89.
[36] G. Wang, L. Guo, H. Duan, H. Wang, A new improved firefly algorithm for global numerical optimization, J. Comput. Theor. Nanosci. 11 (2014) 477-485.
[37] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster,, IEEE. Trans. Evol. Comput. 3 (1999) 82–102.
[38] M. Friedman, A comparison of alternative tests of significance for the problem of m rankings, Ann. Math. Stat. 11 (1940) 86–92.
[39] K. Tang, X. Li, P.N. Suganthan, Z. Yang, T. Weise, Benchmark Functions for the CEC'2010 Special Session and Competition on Large Scale Global Optimization, Nature Inspired Computation and Applications Laboratory, USTC, Hefei, China, 2010.
[40] R. Mallipeddi, P. Suganthan, Problem definitions and evaluation criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization,, Nanyang Technological University, Singapore, 2010.

Dr. Lihong Guo received her Ph.D. from Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China. Her areas of interest include computer vision, and computer application. She is a recognized Supervisor to guide the Ph.D. scholars in Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, China.

Gaige Wang obtained his bachelor degree in computer science and technology from Yili Normal University, Yining, Xinjiang, China, in 2007. His master degree was in the field of intelligent planning and planning recognition at Northeast Normal University, Changchun, China. In 2010 he began working on his Ph.D for developing target threat evaluations by employing computational intelligence techniques at Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, China. He is currently a researcher in School of Computer Science and Technology at Jiangsu Normal University, Xuzhou, China. Gai-Ge Wang has published over 20 journal papers and conference papers indexed by SCI/EI. His research interests are metaheuristic optimization methods and its applications in engineering.

Amir H. Gandomi is the pioneer of Krill Herd Algorithm proposed. He was selected as an elite by In 2008 Iranian National Institute of Elites. He used to be a lecturer in Tafresh University and serve as a researcher in National Elites Foundation. He is currently a researcher in Department of Civil Engineering at the University of Akron, OH. Amir H. Gandomi has published over 80 journal papers and several discussion papers, conference papers and book chapters. He has two patents and has published three books in Elsevier.

Amir H. Alavi is the pioneer of Krill Herd Algorithm proposed in 2012. He used to be a lecturer in Eqbal Lahoori Institute of Higher Education and serve as a researcher School of Civil Engineering at IUST. He is currently a researcher in Department of Civil and Environmental Engineering at the Michigan State University, MI. He has published over 90 research papers and book chapters. He has two patents and has published two books in Elsevier. His research interests are engineering modeling and optimization.

Hong Duan obtained her bachelor degree in educational technology from Yili Normal University, Yining, Xinjiang, China, in 2009. In 2010 she began working on her masters for developing test-sheet composition by employing computational intelligence techniques at Northeast Normal University, Changchun, China. She is currently a researcher in Education College at Shihezi University, Shihezi, China.