Neural Comput & Applic
DOI 10.1007/s00521-013-1433-8

ORIGINAL ARTICLE

Animal migration optimization: an optimization algorithm inspired by animal migration behavior

Xiangtao Li · Jie Zhang · Minghao Yin

Received: 8 January 2013 / Accepted: 20 May 2013

Abstract In this paper, we intend to propose a new heuristic optimization method, called animal migration optimization algorithm. This algorithm is inspired by the animal migration behavior, which is a ubiquitous phenomenon that can be found in all major animal groups, such as birds, mammals, fish, reptiles, amphibians, insects, and crustaceans. In our algorithm, there are mainly two processes. In the first process, the algorithm simulates how the groups of animals move from the current position to the new position. During this process, each individual should obey three main rules. In the latter process, the algorithm simulates how some animals leave the group and some join the group during the migration. In order to verify the performance of our approach, 23 benchmark functions are employed. The proposed method has been compared with other well-known heuristic search methods. Experimental results indicate that the proposed algorithm performs better than or at least comparable with state-of-the-art approaches from literature when considering the quality of the solution obtained.

Keywords AMO · Animal migration optimization · Global numerical optimization · Exploration · Exploitation

1 Introduction

Optimization problems play an important role in both industrial application fields and the scientific research world. During the last decade, learning from nature system, many computational methods have been proposed to solve optimization problems. Usually, these methods begin from an initial set of variables and then run the process until obtaining the global optimal solutions or the maximum of the objective function. Among them, genetic algorithm may be the first and popular algorithm inspired by natural genetic variation and natural selection [1, 2]. Particle swarm algorithm, proposed by Eberhart and Kennedy in 1995, was inspired by the social behavior of bird flocking or fish school [3, 4]. Artificial bee colony was developed by Karaboga in 2005, which simulate the foraging behavior of bee swarm [5, 6]. Ant colony algorithm was another optimization algorithm inspired by the foraging behavior of ant colonies. Biogeography-based optimization is a recently proposed evolutionary algorithm, inspired by the migration behavior of island species [7]. Cuckoo search algorithm is inspired by the obligate brood parasitic behavior of some cuckoo species in combination with the Lévy flight behavior of some birds and fruit flies [8, 9]. These algorithms have been applied to various research areas and have gained a lot of success [10–20]. However, up to now, there is no algorithm that performs well in all the fields. Some algorithms perform much better for some particular problems, while worse for other problems. Until now, how to design a new heuristic optimization algorithm for optimization problem is still an open problem [21].

In this paper, a new heuristic optimization algorithm called as animal migration optimization algorithm, inspired by the behavior of animal migration, is proposed. This algorithm can be divided into two processes. In the first process, the algorithm simulates how the groups of animals move from the current position to the new position. During this process, each individual should obey three main rules. In the latter process, the algorithm simulates how some animals leave the group and some join the group during the migration.

X. Li · J. Zhang · M. Yin (✉)
School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China
e-mail: Minghao.Yin1@gmail.com

Published online: 15 June 2013

The rest of this paper is organized as follows: in Sect. 2, we will review the animal migration or herd behavior. In Sect. 3, we will introduce the animal migration optimization algorithm. Benchmark problems and corresponding experimental results are shown in Sect. 4. In the last section, we conclude this paper and point out some future research directions.

2 Animal migration model

In animal behavior ecology, migration is a widespread phenomenon in the animal kingdom has been studied intensively. The migration is persistent and straightened-out movement affected by the animal's own locomotory exertions carrying them to new habitats. It depends on some temporary inhibition of station keeping responses but promotes their eventual disinhibition and recurrence. Animal migration is the relatively long-distance movement of individuals, usually on a seasonal basis. It is a ubiquitous phenomenon that can be found in all major animal groups, such as birds, mammals, fish, reptiles, amphibians, insects, and crustaceans. The trigger for the migration may be local climate, local availability of food, and the season of the year, and so on. The typical image of migration is of northern landbirds, such as swallows and birds of prey, making long flights to the tropics [22].

In the migration process, the simplest mathematical models of animal aggregations generally instruct the individual to follow three rules: (1) move in the same direction as your neighbors; (2) remain close to your neighbors; and (3) avoid collisions with your neighbors. Recent studies of starling flocks have shown that each bird modifies its position related to the six or seven animals directly around it, no matter how close or how far away those animals are [23, 24]. Interactions between flocking starlings are thus based on a topological rule rather than a metric rule.

In this paper, based on these rules, we proposed a new swarm intelligent algorithm, called as animal migration optimization, according to these rules. The key idea is implemented by means of concentric "zones" around each animal. In the zone of repulsion, the focal animal will seek to distance itself from its neighbors to avoid collision. Slightly further away, the focal animal will seek to align its direction of motion with its neighbors in the zone of alignment. In the outermost zone of attraction, the focal animal will seek to move toward a neighbor.

3 AMO: animal migration optimization algorithm

For simplicity in describing our new animal migration optimization algorithm, we now use the following two idealized assumptions:

1. The leader animal with high quality of position will be retained in the next generation.
2. The number of available animals is fixed, and the animal will be replaced by a new individual with a probability Pa. In this case, the animal will leave the group, and then a new animal will join the group.

In animal migration optimization, there are mainly two processes: migration process and population updating process. In the first process, the algorithm simulates how the groups of animals move from the current position to the new position. During this process, each individual should obey three main rules. In the latter process, the algorithm simulates how some animals leave the group and some join the group during the migration. Suppose that we have a global optimization problem and a population of candidate individuals. The individual is denoted by a D-dimensional real coded vector.

During the initialization process, the algorithm begins with a randomly initiated population which utilizes NP D-dimension parameter vector within constrained by the prescribed minimum and maximum bounds, where NP denotes the size of the population:

$$\begin{aligned} \overrightarrow{X}_{\min} &= \{x_{1,\min}, x_{2,\min}, \ldots, x_{D,\min}\} \\ \overrightarrow{X}_{\max} &= \{x_{1,\max}, x_{2,\max}, \ldots, x_{D,\max}\} \end{aligned} \tag{1}$$

Therefore, we can initialize the jth component of the ith vector as

$$x_{j,i,0} = x_{j,\min} + \text{rand}_{i,j}[0,1] \cdot (x_{j,\max} - x_{j,\min}) \tag{2}$$

where $\text{rand}_{i,j}[0, 1]$ is a uniformly distribution random number between 0 and 1, $i = 1, \ldots,$ NP and $j = 1, \ldots, D$.

During the migration phase, an animal should obey three rules: (1) avoid collisions with your neighbors; (2) move in the same direction as your neighbors; and (3) remain close to your neighbors. For the first rule, we require that the position of each individual in the group should be different. For the latter two rules, we require that the individual should move to a new position according to the current positions of its neighbors. To define concept of the local neighborhood of an individual, we use a *ring* topology scheme, as has been schematically illustrated in Fig. 1. For the sake of simplicity, we set the length of the neighborhood to be five for each dimension of the individual. Note that in our algorithm, the neighborhood topology is static and is defined on the set of indices of vectors. If the index of animal is i, then the neighborhood consists of animal having indices $i - 2$, $i - 1$, i, $i + 1$, $i + 2$ as shown in Fig. 1. If the index of animal is 1, the neighborhood consists of animal having indices NP $-$ 1, NP, 1, 2, and 3. If the index of animal is 2, the neighborhood consists of animal having indices NP, 1, 2, 3, and 4. If the index of animal is NP, the

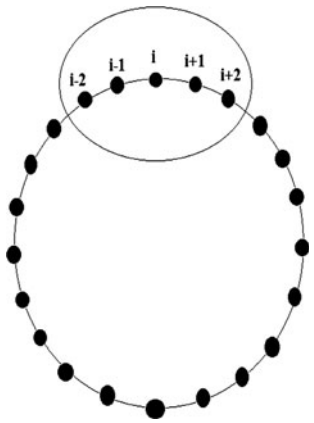

Fig. 1 The concept of the local neighborhood of an individual

neighborhood consists of animal having indices NP − 2, NP − 1, NP, 1, and 2. If the index of animal is NP − 1, the neighborhood consists of animal having indices NP − 3, NP − 2, NP − 1, NP, and 1. Once the neighborhood topology has been constructed, we select one neighbor randomly and update the position of the individual according to this neighbor, as can been seen in the following formula:

$$X_{i,G+1} = X_{i,G} + \delta \cdot (X_{\text{neighborhood},G} - X_{i,G}) \quad (3)$$

where $X_{\text{neighborhood},G}$ is the current position of the neighborhood, and δ can be changed according to different real-world problems. In this paper, δ is produced by using a random number generator controlled by a Gaussian distribution.

During the population updating process, some animals will leave the group, and then some new animal will join in the new population. We assume that the number of available animals is fixed, and the animals will be replaced by some new individual with a probability *Pa*. The probability is used according to the quality of the fitness. For the best fitness, the probability *Pa* is 1. For the worst fitness, the probability is 1/NP.

This process can be shown as follows:

For i=1 to NP
  For j=1 to D
    If rand >Pa
      $X_{i,G+1} = X_{r1,G} + rand \cdot (X_{best,G} - X_{i,G}) + rand \cdot (X_{r2,G} - X_{i,G})$
    End if
  End for
End for

where r_1, $r_2 \in [1, \ldots, \text{NP}]$ are randomly chosen integers, and $r_1 \neq r_2 \neq i$. After producing the new solution $X_{i,G+1}$, it will be evaluated and compared with the $X_{i,G}$, If the objective fitness of $X_{i,G+1}$ is smaller than the fitness of $X_{i,G}$, $X_{i,G+1}$ is accepted as a new basic solution; otherwise, $X_{i,G}$ would be obtained.

The standard animal migration optimization algorithm can be described as the followings:

Procedure Algorithm description of AMO algorithm

begin
  Set the generation counter G=0; and randomly initialize a population of *NP* animal X_i.
    Evaluate the fitness for each individual in *P*.
    while stopping criteria is not satisfied do
    for i= 1 to *NP* **do**
      for j=1 to *D* **do**
        $X_{i,G+1} = X_{i,G} + \delta \cdot (X_{neighborhood,G} - X_{i,G})$
      end for
    end for
      for i=1 to *NP* **do**
      Evaluate the offspring $X_{i,G+1}$
      If $X_{i,G+1}$ is better than X_i **then**
        $X_i = X_{i,G+1}$
      end if
    end for
    for i=1 to *NP*
      for j=1 to *D*
        select randomly $r1 \neq r2 \neq i$
        If rand>*Pa* **then**
          $X_{i,G+1} = X_{r1,G} + rand \cdot (X_{best,G} - X_{i,G}) + rand \cdot (X_{r2,G} - X_{i,G})$
          End if
        End for
      End for
      for i=1 to *NP* **do**
        Evaluate the offspring $X_{i,G+1}$
        If $X_{i,G+1}$ is better than X_i **then**
          $X_i = X_{i,G+1}$
      end if
    end for
    Memorize the best solution achieved so far
  end while
end

4 Experimental results

To evaluate the performance of our algorithm, we applied it to 23 standards benchmark functions. These functions have been widely used in the literature. Since we do not make any modification of these functions, they are given in Table 1. Among these benchmarks, functions *f1–f13* are multidimensional problems. Functions *f1–f5* are unimodal functions, and *f5* is multimodal function for $D > 3$. The *f06* is the step function which has one minimum and is discontinuous. Function *f07* is a noisy quadratic function where random [0, 1] is a uniformly distributed random number in [0, 1]. The following seven functions are multimodal test functions. For these functions, the number of local minima increases exotically with the problem dimensions. They apparently belong to the most difficult class of problem for many optimization problems. For the unimodal function, researchers are more interesting in the

Table 1 Benchmark functions based in our experimental study

Test function	D	Range	Optimum
$f_{01} = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]$	0
$f_{02} = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$[-10, 10]$	0
$f_{03} = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2$	30	$[-100, 100]$	0
$f_{04} = \max_i \{ \lvert x_i \rvert, 1 \le i \le D \}$	30	$[-100, 100]$	0
$f_{05} = \sum_{i=1}^{D-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$	30	$[-30, 30]$	0
$f_{06} = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100, 100]$	0
$f_{07} = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]$	0
$f_{08} = \sum_{i=1}^{D} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	$[-500, 500]$	-418.9829*n
$f_{09} = \sum_{i=1}^{D} \left[x_i^2 - 10 \cos(2\pi x_i) + 10 \right]$	30	$[-5.12, 5.12]$	0
$f_{10} = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos 2\pi x_i \right) + 20 + e$	30	$[-32, 32]$	0
$f_{11} = \frac{1}{400} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	30	$[-600, 600]$	0
$f_{12} = \frac{\pi}{D} \left\{ 10 \sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_i + 1)] + (yD - 1)^2 + \sum_{i=1}^{D} u(x_i, 10, 100, 4) \right\}$ $y_i = 1 + \frac{x_i + 1}{4} \quad u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30	$[-50, 50]$	0
$f_{13} = 0.1 \left\{ 10 \sin^2(\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_i + 1)] + (yD - 1)^2 \right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	30	$[-50, 50]$	0
$f_{14} = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right]^{-1}$	2	$[-65.53, 65.53]$	0.998004
$f_{15} = \sum_{i=1}^{11} \left[a_i - \frac{x_1 \left(b_i^2 + b_i x_i \right)}{b_i^2 + b_1 x_3 + x_4} \right]^2$	4	$[-5, 5]$	0.0003075
$f_{16} = 4x_1^2 - 2.1x_i^4 + \frac{1}{3} x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	-1.0316285
$f_{17} = \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6 \right)^2 + 10 \left(1 - \frac{1}{8\pi} \right) \cos x_1 + 10$	2	$[-5, 10]$*$[0, 15]$	0.398
$f_{18} = \left[1 + (x_1 + x_2 + 1)^2 \left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2 \right) \right]$ $\times \left[30 + (2x_1 - 3x_2)^2 \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2 \right) \right]$	2	$[-5, 5]$	3
$f_{19} = -\sum_{i=1}^{4} c_i \exp\left(\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2 \right)$	3	$[0, 1]$	-3.86
$f_{20} = -\sum_{i=1}^{4} c_i \exp\left(\sum_{j=1}^{6} a_{ij} (x_j - p_{ij})^2 \right)$	6	$[0, 1]$	-3.32
$f_{21} = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^T + c_i \right]$	4	$[0, 10]$	-10.1532
$f_{22} = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i \right]^{-1}$	4	$[0, 10]$	-10.4029
$f_{23} = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i \right]^{-1}$	4	$[0, 10]$	-10.5364

convergence rates instead of the final results of optimization. For multimodal functions, the final results are more important than the convergence rate of different algorithms. Then, ten multimodal test functions with fix dimension which have only a few local search minima are used in our experimental study. Table 1 has shown the details of these functions. So far, these problems have been widely used as benchmarks for study with different methods by many researchers.

4.1 Experimental setup

All algorithms are coded in MATLAB 7.0, and experiments are performed on a Pentium 3.0 GHz Processor with 1.0 GB of memory. The source code can be obtained from the first author upon request.

4.2 Algorithm comparison

In the test conducted in this section, success of AMO algorithm introduce in this paper has been compared with the success of the algorithms PSO [3], DE [25–27], RCBBO [7], GSA [28], FA [29], CS [8, 30, 31] and ABC [5] for the mean solution of numerical optimization problem. For all test functions, maximum number of Fitness Evaluation(Max_NFFEs): The maximum number of generations: 1,500 for *f1*, *f6*, *f10*, *f12*, and *f13*, 2,000 for *f2* and *f11*, 3,000 for *f7*, *f8*, *f9*, and 5,000 for *f3*, *f4*, *f5*. 400 for *f15*, 100 for *f14*, *f16*, *f17*, *f19*, *f21*, *f22*, and *f23*, *30* for *f18*, *200* for *f20*.

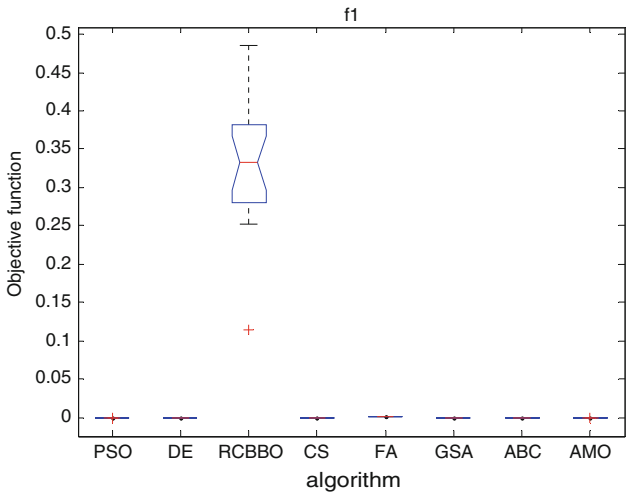

Fig. 2 ANOVA tests of the global minimum values, which are computed by using the PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for *f1*

Table 2 Minimization result of benchmark functions *f1–f7* for different algorithms

Functions	Algorithm	PSO [3]	DE [25]	RCBBO [7]	CS [8]	FA [29]	GSA [28]	ABC [5]	AMO
f1	Mean	3.3340e−10	5.6016e−14	0.3737	5.6565e−06	0.0017	3.3748e−18	2.9860e−20	8.6464e−40
	StdDev	7.0387e−10	4.4053e−14	0.1181	2.8611e−06	4.0608e−04	8.0862e−19	2.1455e−20	1.0435e−39
	Rank	5	4	8	6	7	3	2	1
f2	Mean	6.6598e−11	4.7348e−10	0.1656	0.0020	0.0453	8.92115e−09	1.4213e−15	8.2334e−32
	StdDev	9.2553e−11	1.7759e−10	0.0342	8.0959e−04	0.0338	1.33404e−09	5.5340e−16	3.4120e−32
	Rank	3	4	8	6	7	5	2	1
f3	Mean	2.9847	2.8038e−11	1.5972e + 03	0.0014	0.0182	0.1126	2.4027e + 03	8.8904e−04
	StdDev	2.2778	3.6788e−11	833.6020	6.0987e−04	0.0064	0.1266	656.9600	8.7256e−04
	Rank	6	1	7	3	4	5	8	2
f4	Mean	7.9997	0.2216	7.9738	3.2388	0.0554	9.9302e−10	18.5227	2.8622e−05
	StdDev	2.5351	0.2430	2.6633	0.6644	0.0101	1.1899e−10	4.2477	2.3468e−05
	Rank	7	4	6	5	3	1	8	2
f5	Mean	46.9202	0.2657	64.6907	8.0092	38.1248	20.0819	0.0441	4.1817
	StdDev	38.0312	1.0293	36.2782	1.9188	30.3962	0.1722	0.0707	2.1618
	Rank	7	2	8	4	6	5	1	3
f6	Mean	3.6925e−10	4.5028e−14	0.3695	5.4332e−06	0.0017	3.3385e−18	3.0884e−20	0
	StdDev	6.3668e−10	2.3309e−14	0.1115	2.2446e−06	4.1593e−04	5.6830e−19	4.0131e−20	0
	Rank	5	4	8	6	7	3	2	1
f7	Mean	0.0135	0.0042	0.0030	0.0096	0.0082	0.0039	0.0324	0.0017
	StdDev	0.0041	0.0014	0.0012	0.0028	0.0093	0.0013	0.0059	4.7058e−04
	Rank	7	4	2	6	5	3	8	1
Average rank		5.71	3.29	6.71	4.86	5.43	3.43	4.43	1.57
Overall rank		7	2	8	5	6	3	4	1

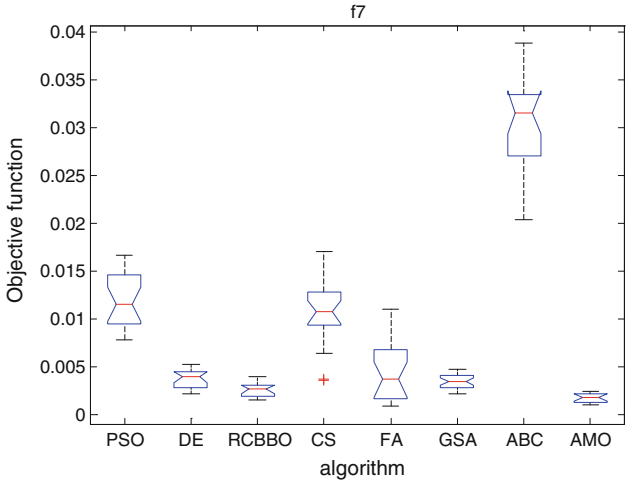

Fig. 3 ANOVA tests of the global minimum values, which are computed by using the PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for *f7*

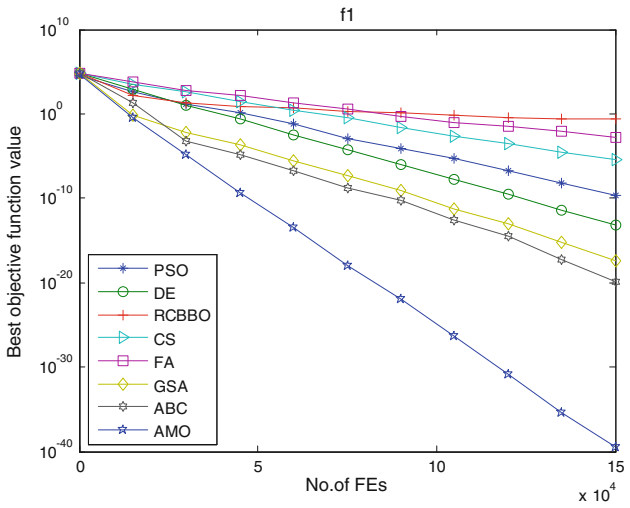

Fig. 4 Comparison of performance of PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for minimization of *f1*

The setting values of algorithm control parameters of the mentioned algorithms are given below.

PSO setting: weight factor $\omega = 0.6$ and $c_1 = c_2 = 2$. The population size is 100 [11].

DE setting: $F = 0.5$ and CR $= 0.9$ in accordance with the suggestions given in [25], the population size is 100.

ABC setting: limit $= 50D$ has been used as recommended in [5], the population size is 50 because of this algorithm has two phases.

RCBBO setting: Maximum immigration rate: $I = 1$, Maximum emigration rate: $E = 1$, and Mutation probability: $m_{\max} = 0.005$ have been used as recommended in [7], the population size is 100.

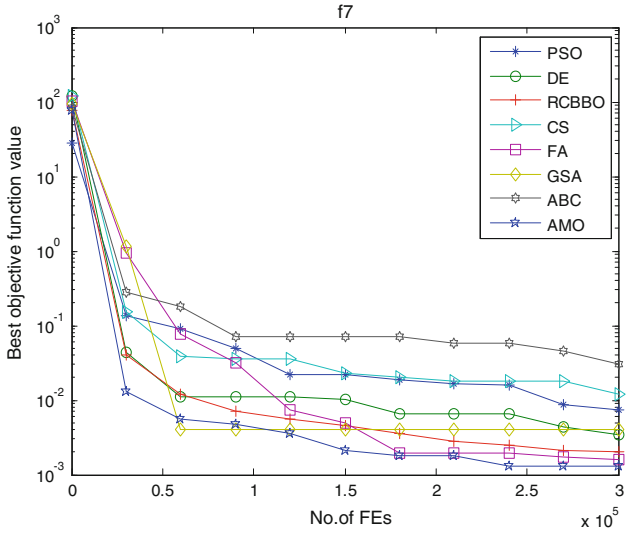

Fig. 5 Comparison of performance of PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for minimization of *f7*

CS setting: $\beta = 1.5$ and $p_o = 1.5$ have been used as recommended in [8], the population size is 50 because of this algorithm has two phases.

GSA setting: $G_o = 100$, $\alpha = 20$, K_o is set to NP and is decreased linearly to 1 have been used as recommended in [28].the population size is 100.

FA setting: $\alpha_0 = 0.5$, $\beta_0 = 0.2$, and $\gamma = 1$ have been used as recommended in [29].the population size is 100.

AMO setting: the population size is 50 because of this algorithm has two phases.

The global minimum values of each of the benchmark functions used in this paper have been solved 25 times by the mentioned algorithms using a different initial population at every turn.

4.3 Unimodal functions

In order to show the effectiveness of our proposed AMO approach, we compare it with the PSO, DE, RCBBO, CS, GSA, FA, and ABC. In the experiment, the mean results of 25 independent runs for *f1–f7* are summarized in Table 2. Functions *f1–f7* are unimodal functions. Figures 2 and 3 show the graphical analysis results of the ANOVA tests. For the unimodal function, the convergence rate of search algorithm is more important for unimodal function than the final results because there are other methods which are specifically designed to optimize unimodal functions. As can be seen in Table 2, we first rank the algorithm from the smallest mean solution to the highest solution. Then, we average the ranks based on these seven functions and obtain the average rank. In the last, we rank the average rank and obtain the overall rank. From the rank of each function, we can find that AMO provides better results than

Table 3 Minimization result of benchmark functions *f8–f13* for different algorithms

Functions	Algorithm	PSO [3]	DE [25]	RCBBO [7]	CS [8]	FA [29]	GSA [28]	ABC [5]	AMO
f8	Mean	−8.8278e+03	−1.1276e+04	−1.2568e+04	−9.1492e+03	−6.2238e+03	−3.0499e+03	−1.2507e+04	−1.2569e+04
	StdDev	611.1590	1.8135e+03	0.5758	2.53143e+02	7.7230e+02	3.3886e+02	61.1186	1.2384e−07
	Rank	6	4	2	5	7	8	3	1
f9	Mean	18.2675	134.6789	0.0385	51.2202	23.5213	7.2831	0	0
	StdDev	4.7965	28.8598	0.0154	8.1069	8.3683	1.8991	0	0
	Rank	5	8	3	7	6	4	1	1
f10	Mean	3.8719e−06	7.4739e−08	0.1947	2.3750	0.0094	1.4717e−09	1.1946e−09	4.4409e−15
	StdDev	2.8604e−06	3.1082e−08	0.0461	1.1238	0.0014	1.4449e−10	5.0065e−10	0
	Rank	5	4	7	8	6	3	2	1
f11	Mean	0.0168	0	0.2765	4.4900e−05	0.0025	0.01265	0	0
	StdDev	0.0205	0	0.0796	8.9551e−05	4.6910e−04	0.02160	0	0
	Rank	7	1	8	4	5	6	1	1
f12	Mean	0.0083	4.7114e−15	0.0020	0.5071	8.8694e−06	2.0358e−20	1.1928e−21	1.5705e−32
	StdDev	0.0287	3.2597e−15	0.0023	0.2662	2.7999e−06	4.5322e−21	1.0783e−21	2.8080e−48
	Rank	7	4	6	8	5	3	2	1
f13	Mean	4.6694e−07	3.1598e−14	0.0218	4.6965e−04	1.2812e−04	5.6991e−33	2.2990e−20	1.3498e−32
	StdDev	1.3713e−06	2.2825e−14	0.0096	2.9932e−04	4.1539e−05	6.2589e−33	2.2886e−20	2.8080e−48
	Rank	5	4	8	7	6	1	3	2
Average rank		5.83	4.17	5.67	6.5	5.67	4.17	2	1.17
Overall rank		7	3	5	8	5	3	2	1

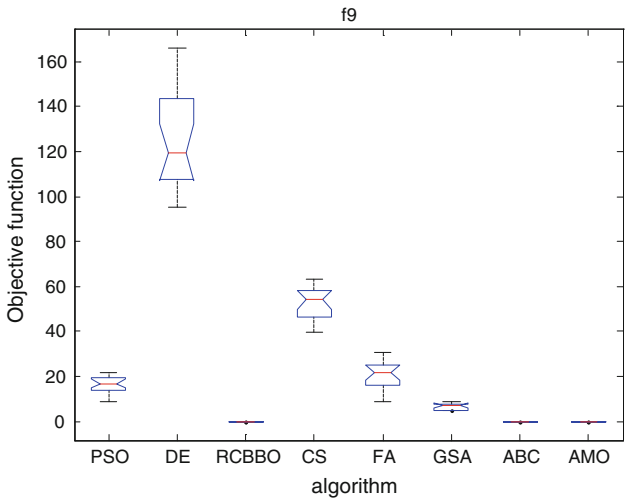

Fig. 6 ANOVA tests of the global minimum values, which are computed by using the PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for *f9*

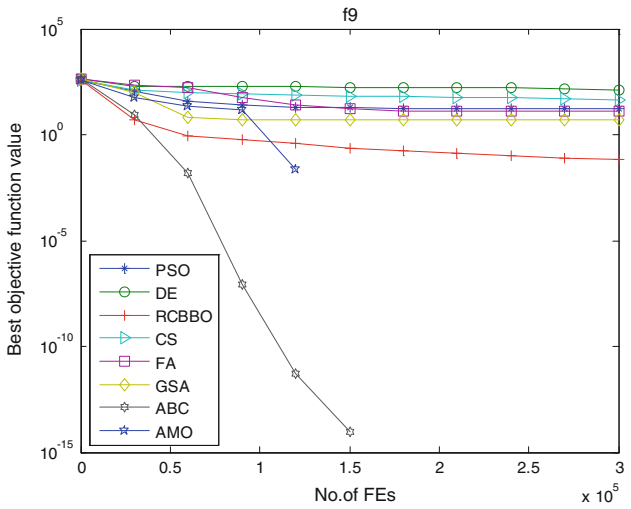

Fig. 8 Comparison of performance of PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for minimization of *f9*

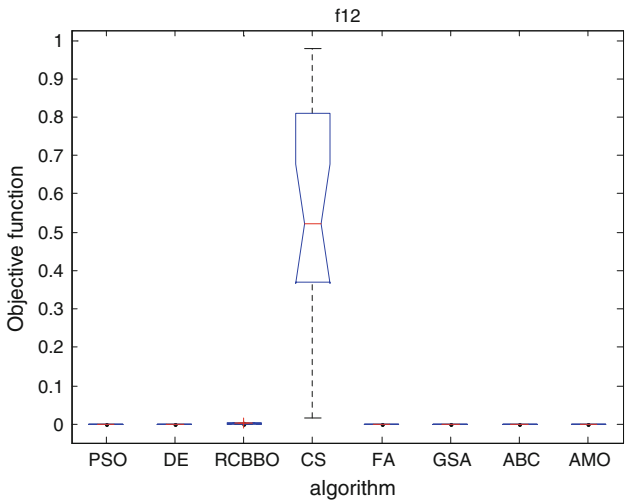

Fig. 7 ANOVA tests of the global minimum values, which are computed by using the PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for *f12*

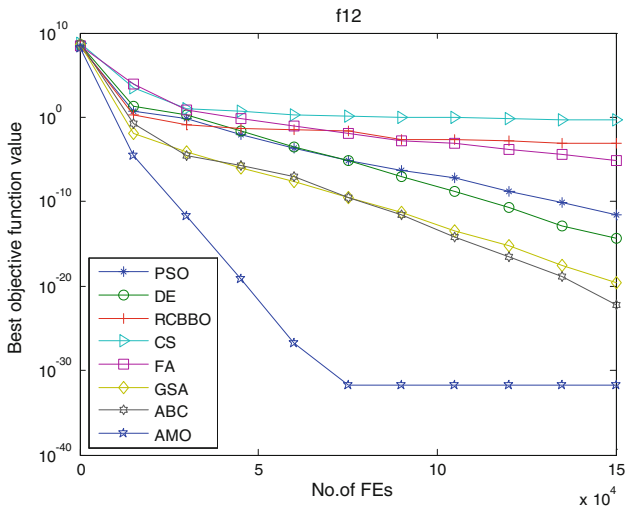

Fig. 9 Comparison of performance of PSO, DE, RCBBO, CS, FA, GSA, ABC and AMO for minimization of *f12*

other algorithms except *f3–f5*. For the functions *f3–f5*, AMO cannot give the best solution. For *f03*, the DE algorithm gives the better results. For *f4*, GSA algorithm is a winner. For *f5*, ABC can give the best solution. However, AMO can obtain the best rank for the overall rank. The difference in performance between them occurs with these unimodal functions. Also, the convergence rate of AMO can be shown in Figs. 4 and 5. As can be seen in these figures, we can conclude that AMO has the faster convergence rate.

4.4 Multimodal high-dimensional functions

For multimodal functions *f8–f13* with many local minima, the final results are more important because of this function can reflect the algorithm's ability to escape from poor local optima and obtain the near-global optimum. We have tested the experiments on *f8–f13* where the number of local minima increases exponentially as long as the dimension of the function increases. Table 3 summarizes the average results of 25 independent runs for the selected functions. Figures 6 and 7 show the graphical analysis results of the ANOVA tests. We also first rank the algorithm from the smallest mean solution to the highest solution. Then, we average the ranks based on these seven functions and obtain the average rank. In the last, we rank the average rank and obtain the overall rank. From the rank of each function, we can find that AMO provides better results than other algorithms. For *f13*, the GSA can perform better

Table 4 Minimization result of benchmark functions *f14–f23* for different algorithms

Functions	Algorithm	PSO [3]	DE [25]	RCBBO [7]	CS [8]	FA [29]	GSA [28]	ABC [5]	AMO
f14	Mean	0.9980	0.9980	0.9981	0.9981	3.0273	5.9533	0.9980	0.9980
	StdDev	1.0968e−14	9.8526e−16	4.0691e−04	4.8277e−04	1.5853	3.4819	3.7921e−16	3.3858e−12
	Rank	3	2	5	5	7	8	1	4
f15	Mean	6.5867e−04	4.5400e−04	0.0028	5.0310e−04	0.0010	0.0048	7.4715e−04	3.9738e−04
	StdDev	2.2775e−04	3.4262e−04	0.0044	1.1180e−04	4.6002e−04	0.0033	2.1481e−04	4.4503e−05
	Rank	4	2	7	3	6	8	5	1
f16	Mean	−1.0316	−1.0316	−1.0312	−1.03163	−1.0314	−1.03163	−1.0316	−1.0316
	StdDev	1.2775e−12	6.6855e−14	8.1085e−04	1.58211e−07	0.0011	4.7536e−16	1.1269e−14	5.2006e−11
	Rank	4	3	7	6	8	1	2	5
f17	Mean	0.3979	0.3979	0.3984	0.3979	0.3979	0.3979	0.3979	0.3979
	StdDev	8.2239e−12	0	6.7654e−04	3.2449e−06	3.0709e−08	0	5.3819e−08	0
	Rank	4	1	8	7	5	1	6	1
f18	Mean	3.0001	3.0000	3.0491	3.0013	3.0123	3.7403	3.0000	3.0018
	StdDev	7.9627e−05	1.2230e−10	0.0650	0.0026	0.0526	1.6055	2.6164e−05	0.0055
	Rank	3	1	7	4	6	8	2	5
f19	Mean	−3.8628	−3.8628	−3.8627	−3.8628	−3.8613	−3.8625	−3.8628	−3.8628
	StdDev	3.1270e−12	2.3042e−15	1.4077e−04	1.4043e−05	0.0037	3.8767e−04	1.3654e−10	1.3669e−15
	Rank	3	2	6	5	8	7	4	1
f20	Mean	−3.2554	−3.2174	−3.2833	−3.3210	−3.2741	−3.3220	−3.3220	−3.3220
	StdDev	0.0602	0.0394	0.0563	7.5186e−04	0.0723	4.7967e−16	8.5733e−10	5.0850e−06
	Rank	7	8	5	4	6	1	2	3
f21	Mean	−7.6393	−10.1532	−5.4381	−9.7256	−6.5633	−4.784	−10.1528	−10.0592
	StdDev	3.2359	8.1382e−06	3.6051	0.2846	3.8155	1.4761	0.0013	0.2491
	Rank	5	1	7	4	6	8	2	3
f22	Mean	−7.3602	−10.4029	−7.7759	−9.8624	−10.4027	−6.5797	−10.4012	−10.3899
	StdDev	3.3035	2.5631e−06	3.3125	0.3291	7.0965e−04	3.8073	0.0046	0.0297
	Rank	7	1	6	5	2	8	3	4
f23	Mean	−8.9611	−10.5364	−9.3514	−9.7534	−10.2297	−8.2651	−10.5339	−10.4990
	StdDev	2.8381	3.9747e−06	2.6288	0.4913	1.5332	2.8868	0.0054	0.1428
	Rank	7	1	6	5	4	8	2	3
Average rank		4.8	2.3	6.4	4.0	5.8	5.6	2.7	3.0
Overall rank		5	1	8	4	7	6	2	3

solution than our AMO algorithm. In general, the performance of AMO is highly competitive with other algorithms, especially for the high-dimensional problems. Figures 8 and 9 show the AMO converges faster than other algorithms due to its better exploration ability.

4.5 Multimodal low-dimensional functions

For *f14–f23* with only a few local minima, the dimension of the function is also small. In this case, it is hard to judge the performances of individual algorithms. The major difference compared with functions *f8–f13* is that functions *f14–f23* appear to be simpler than *f8–f13* due to their low dimensionalities and a smaller number of local minima. In the experiment, the mean results of 25 independent runs are summarized in Table 4. All algorithms were able to easily find optimal solutions for these functions. However, we still rank these algorithms. From the Table 4, we can find the DE and AMO can provide better solutions than other algorithm. For 9 out of 10 functions, there are not significant difference between the AMO, ABC, and DE approaches and other algorithm. Only for function *f15*, AMO is significant better than other algorithms in terms of the final result. For three functions (i.e., *f21–f23*), DE algorithm outperforms the other approaches.

5 Conclusions

In this paper, we propose a new swarm intelligent algorithm based on the animal migration model to solve the global optimization problems with continuous variables. In our paper, the algorithm consists two main parts. In the first process, the algorithm simulates how the groups of animals move from the current position to the new position. During this process, each individual should obey three main rules. In the latter process, the algorithm simulates how some animals leave the group and some join the group during the migration. To verify the performance of our algorithm, 23 benchmark functions chosen from literature are employed. The results show that the proposed algorithm clearly outperforms some evolution algorithms from literature.

In this paper, we only consider the global optimization. The algorithm can be extended to solve other problem such as constrained optimization problems or multiobjecitve optimization problem.

Acknowledgments This research is fully supported by Opening Fund of Top Key Discipline of Computer Software and Theory in Zhejiang Provincial Colleges at Zhejiang Normal University under Grant No. ZSDZZZZXK37 and the Fundamental Research Funds for the Central Universities Nos. 11CXPY010.

References

1. Melanie M (1999) An introduction to genetic algorithms. MIT Press, Massachusetts
2. Sivanandam SN, Deepa SN (2008) Introduction to genetic algorithms. Springer, Berlin
3. Kennedy J, Eberhart R (1995) Particle swarm optimization. Proc Int Conf Neural Netw 4:1942–1948
4. Engelbrecht AP (2005) Fundamentals of computational swarm intelligence. Wiley, New Jersey
5. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Global Optim 39(3):459–471
6. Karaboga D, Basturk B (2008) On the performance of artificial bee colony (ABC) algorithm. Appl Soft Comput 8(1):687–697
7. Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12(6):702–713
8. Yang XS, Deb S (2009) Cuckoo search via Levy flights, in: world congress on nature & biologically inspired computing (NaBIC 2009). IEEE Publication, USA, pp 210–214
9. Yang XS, Deb S (2010) Engineering optimisation by cuckoo search. Int J Math Modell Numer Optim 1(4):330–343
10. Horn J, Nafpliotis N, Goldberg DE (1994) A niched Pareto genetic algorithm for multiobjective optimization. Evol Comput 1:82–87
11. Clerc M, Kennedy J (2002) The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Trans Evol Comput 6:58–73
12. Mohan BC, Baskaran R (2011) Energy aware and energy efficient routing protocol for adhoc network using restructured artificial bee colony system. Commun Comput Inf Sci 169(3):473–484
13. Rao RV, Patel VK (2011) Optimization of mechanical draft counter flow wet-cooling tower using artificial bee colony algorithm. Energy Convers Manage 52(7):2611–2622
14. Karaboga D, Ozturk C, Karaboga N, Gorkemli B (2012) Artificial bee colony programming for symbolic regression. Inf Sci 209:1–15
15. Li X, Yin M (2011) Hybrid differential evolution with biogeography-based optimization for design of a reconfigurable antenna array with discrete phase shifters. Int J Antennas Propag 2011. Article ID 685629
16. Li X, Wang J, Zhou J, Yin M (2011) A perturb biogeography based optimization with mutation for global numerical optimization. Appl Math Comput 218(2):598–609
17. Li X, Yi M (2012) Multi-operator based biogeography based optimization with mutation for global numerical optimization. Comput Math Appl 64(9):2833–2844
18. Walton S, Hassan O, Morgan K, Brown MR (2011) Modified cuckoo search: a new gradient free optimisation algorithm. Chaos Solitons Fractals 44:710–718
19. Gandomi AH, Yang XS, Alavi AH (2011) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Engineering with Computers, 27, July
20. Layeb A (2011) A novel quantum inspired cuckoo search for knapsack problems. Int J Bio Inspir Comput 3:297–305
21. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1:67–82
22. Dyer JRG, Ioanno CC, Morrell LJ, Croft DP, Couzin ID, Waters DA, Krause J (2008) Consensus decision making in human crowds. Anim Behav 75(2): 461–470
23. Braha D (2012) Global civil unrest: contagion, self-organization, and prediction. PLoS ONE 7(10):e48596. doi:10.1371/journal.pone.0048596
24. Ballerini M, Cabibbo N, Candelier R, Cavagna A, Cisbani E, Giardina I, Lecomte V, Orlandi A, Parisi G, Procaccini A, Viale

M, Zdravkovic V (2008) Interaction ruling animal collective behavior depends on topological rather than metric distance: evidence from a field study. Proc Natl Acad Sci USA 105(4):1232–1237. arXiv:0709.1916. Bibcode 2008PNAS.105.1232B. doi:10.1073/pnas.0711437105. PMC 2234121. PMID 18227508.//www.ncbi.nlm.nih.gov/pmc/articles/PMC2234121/

25. Storn R, Price K (1997) Differential evolution-a simple and efficient heuristic for global optimization over continuous space. J Global Optim 11:341–359
26. Li X, Yin M (2013) An opposition-based differential evolution algorithm for permutation flow shop scheduling based on diversity measure. Adv Eng Softw 55:10–31
27. Li X, Yin M (2012) Application of differential evolution algorithm on self-potential data. PLoS ONE 7(12):e51199
28. Rashedi Esmat, Nezamabadi-pour Hossein (2009) Saeid Saryazdi GSA: a gravitational search algorithm. Inf Sci 179:2232–2248
29. Yang XS (2010), Firefly algorithm, L′evy flights and global optimization. In: Bramer M, Ellis R, Petridis M (eds) Research and development in intelligent systems XXVI. Springer London, pp 209–218
30. Li X, Yin M (2012) Parameter estimation for chaotic systems by Cuckoo search algorithm using orthogonal learning method. Chin Phys B 5:50507
31. Li X, Wang JN, Yin M (2013) Enhancing the performance of cuckoo search algorithm using orthogonal learning method. Neural Comput Appl. doi:10.1007/s00521-013-1354-6