# A New Bio-inspired Algorithm: Chicken Swarm Optimization

**Conference Paper** · October 2014

DOI: 10.1007/978-3-319-11857-4_10



**4 authors**, including:

Xian-Bing Meng
Central South University
**5** PUBLICATIONS **138** CITATIONS

SEE PROFILE

# A New Bio-inspired Algorithm: Chicken Swarm Optimization

Xianbing Meng[1,2], Yu Liu[2], Xiaozhi Gao[1,3], and Hengzhen Zhang[1]

[1] College of Information Engineering, Shanghai Maritime University, 1550 Haigang Avenue, Shanghai, 201306, P.R. China
[2] Chengdu Green Energy and Green Manufacturing R&D Center, 355 Tengfei Road No. 2, Chengdu, 610200, P.R. China
[3] Department of Electrical Engineering and Automation, Aalto University School of Electrical Engineering, Otaniementie 17, FI-00076 Aalto, Finland
x.b.meng12@gmail.com, yu.liu@vip.163.com

**Abstract.** A new bio-inspired algorithm, Chicken Swarm Optimization (CSO), is proposed for optimization applications. Mimicking the hierarchal order in the chicken swarm and the behaviors of the chicken swarm, including roosters, hens and chicks, CSO can efficiently extract the chickens' swarm intelligence to optimize problems. Experiments on twelve benchmark problems and a speed reducer design were conducted to compare the performance of CSO with that of other algorithms. The results show that CSO can achieve good optimization results in terms of both optimization accuracy and robustness. Future researches about CSO are finally suggested.

**Keywords:** Hierarchal order, Chickens' behaviors, Swarm intelligence, Chicken Swarm Optimization, Optimization applications.

## 1 Introduction

Bio-inspired meta-heuristic algorithms have shown proficiency of solving a great many optimization applications [1, 2]. They exploit the tolerance for imprecision and uncertainty of the optimization problems and can achieve acceptable solutions using low computing cost. Thus the mate-heuristic algorithms, like Particle Swarm Optimization (PSO) [3], Differential Evolution (DE) [2], Bat Algorithm (BA) [1], have attracted great research interest for dealing with optimization applications.

New algorithms are still emerging, including krill herb algorithm [4], and social spider optimization [5] et al. All these algorithms extract the swarm intelligence from the laws of biological systems in nature. However, to learn from the nature for developing a better algorithm is still in progress.

In this paper, a new bio-inspired optimization algorithm, namely Chicken Swarm Optimization (CSO) is proposed. It mimics the hierarchal order in the chicken swarm and the behaviors of the chicken swarm. The chicken swarm can be divided into several groups, each of which consists of one rooster and many hens and chicks. Different chickens follow different laws of motions. There exist competitions between different chickens under specific hierarchal order.

Y. Tan et al. (Eds.): ICSI 2014, Part I, LNCS 8794, pp. 86–94, 2014.

The rest of paper is organized as follows. Section 2 introduces the general biology of the chicken. The details about the CSO are discussed in Section 3. The simulations and comparative studies are presented in section 4. Section 5 summaries this paper with some conclusions and discussions.

## 2 General Biology

As one of the most widespread domestic animals, the chickens themselves and their eggs are primarily kept as a source of food. Domestic chickens are gregarious birds and live together in flocks. They are cognitively sophisticated and can recognize over 100 individuals even after several months of separation. There are over 30 distinct sounds for their communication, which range from clucks, cackles, chirps and cries, including a lot of information related to nesting, food discovery, mating and danger. Besides learning through trial and error, the chickens would also learn from their previous experience and others' for making decisions [6].

A hierarchal order plays a significant role in the social lives of chickens. The preponderant chickens in a flock will dominate the weak. There exist the more dominant hens that remain near to the head roosters as well as the more submissive hens and roosters who stand at the periphery of the group. Removing or adding chickens from an existing group would causes a temporary disruption to the social order until a specific hierarchal order is established [7].

The dominant individuals have priority for food access, while the roosters may call their group-mates to eat first when they find food. The gracious behavior also exists in the hens when they raise their children. However, this is not the case existing for individuals from different groups. Roosters would emit a loud call when other chickens from a different group invade their territory [8].

In general, the chicken's behaviors vary with gender. The head rooster would positively search for food, and fight with chickens who invade the territory the group inhabits. The dominant chickens would be nearly consistent with the head roosters to forage for food. The submissive ones, however, would reluctantly stand at the periphery of the group to search for food. There exist competitions between different chickens. As for the chicks, they search for the food around their mother.

Each chicken is too simple to cooperate with each other. Taken as a swarm, however, they may coordinate themselves as a team to search for food under specific hierarchal order. This swarm intelligence can be associated with the objective problem to be optimized, and inspired us to design a new algorithm.

## 3 Chicken Swarm Optimization

Given the aforementioned descriptions, we can develop CSO mathematically. For simplicity, we idealized the chickens' behaviors by the following rules.

(1) In the chicken swarm, there exist several groups. Each group comprises a dominant rooster, a couple of hens, and chicks.

(2) How to divide the chicken swarm into several groups and determine the identity of the chickens (roosters, hens and chicks) all depend on the fitness values of the chickens themselves. The chickens with best several fitness values would be acted as roosters, each of which would be the head rooster in a group. The chickens with worst several fitness values would be designated as chicks. The others would be the hens. The hens randomly choose which group to live in. The mother-child relationship between the hens and the chicks is also randomly established.

(3) The hierarchal order, dominance relationship and mother-child relationship in a group will remain unchanged. These statuses only update every several ($G$) time steps.

(4) Chickens follow their group-mate rooster to search for food, while they may prevent the ones from eating their own food. Assume chickens would randomly steal the good food already found by others. The chicks search for food around their mother (hen). The dominant individuals have advantage in competition for food.

Assume *RN, HN, CN and MN* indicate the number of the roosters, the hens, the chicks and the mother hens, respectively. The best *RN* chickens would be assumed to be roosters, while the worst *CN* ones would be regarded as chicks. The rest are treated as hens. All $N$ virtual chickens, depicted by their positions $x_{i,j}^t$ $(i \in [1,\cdots,N], j \in [1,\cdots,D])$ at time step $t$, search for food in a $D$-dimensional space. In this work, the optimization problems are the minimal ones. Thus the best *RN* chickens correspond to the ones with *RN* minimal fitness values.

### 3.1 Movement of the Chickens

The roosters with better fitness values have priority for food access than the ones with worse fitness values. For simplicity, this case can be simulated by the situation that the roosters with better fitness values can search for food in a wider range of places than that of the roosters with worse fitness values. This can be formulated below.

$$x_{i,j}^{t+1} = x_{i,j}^t * (1 + Randn(0,\sigma^2)) \ . \tag{1}$$

$$\sigma^2 = \begin{cases} 1\,, if\ f_i \le f_k\,, \\ \exp\left(\frac{(f_k - f_i)}{|f_i| + \varepsilon}\right), \ otherwise, \end{cases} k \in [1,N], k \ne i \ . \tag{2}$$

Where *Randn* $(0,\ \sigma^2)$ is a Gaussian distribution with mean 0 and standard deviation $\sigma^2$. $\varepsilon$, which is used to avoid zero-division-error, is the smallest constant in the computer. $k$, a rooster's index, is randomly selected from the roosters group, $f$ is the fitness value of the corresponding $x$.

As for the hens, they can follow their group-mate roosters to search for food. Moreover, they would also randomly steal the good food found by other chickens, though they would be repressed by the other chickens. The more dominant hens would have advantage in competing for food than the more submissive ones. These phenomena can be formulated mathematically as follows.

$$x_{i,j}^{t+1} = x_{i,j}^t + S1 * Rand * \left(x_{r1,j}^t - x_{i,j}^t\right) + S2 * Rand * \left(x_{r2,j}^t - x_{i,j}^t\right) \ . \tag{3}$$

$$S1 = \exp(\ (f_i - f_{r1})\ /(abs(f_i) + \varepsilon)\ ) \ . \tag{4}$$

$$S2 = \exp(\ (f_{r2} - f_i))\ . \tag{5}$$

Where *Rand* is a uniform random number over [0, 1]. $r1 \in [1, \cdots, N]$ is an index of the rooster, which is the *i*th hen's group-mate, while $r2 \in [1, \cdots, N]$ is an index of the chicken (rooster or hen ), which is randomly chosen from the swarm. $r1 \neq r2$.

Obviously, $f_i > f_{r1}, f_i > f_{r2}$, thus *S2* <1< *S1*. Assume *S1*=0, then the *i*th hen would forage for food just followed by other chickens. The bigger the difference of the two chickens' fitness values, the smaller *S2* and the bigger the gap between the two chickens' positions is. Thus the hens would not easily steal the food found by other chickens. The reason that the formula form of *S1* differs from that of *S2* is that there exist competitions in a group. For simplicity, the fitness values of the chickens relative to the fitness value of the rooster are simulated as the competitions between chickens in a group. Suppose *S2*=0, then the *i*th hen would search for food in their own territory. For the specific group, the rooster's fitness value is unique. Thus the smaller the *i*th hen's fitness value, the nearer *S1* approximates to 1 and the smaller the gap between the positions of the *i*th hen and its group-mate rooster is. Hence the more dominant hens would be more likely than the more submissive ones to eat the food.

The chicks move around their mother to forage for food. This is formulated below.

$$x_{i,j}^{t+1} = x_{i,j}^{t} + FL * \left(x_{m,j}^{t} - x_{i,j}^{t}\right)\ . \tag{6}$$

Where $x_{m,j}^{t}$ stands for the position of the $i$th chick's mother $(m \in [1, N])$. *FL* $(FL \in (0,2))$ is a parameter, which means that the chick would follow its mother to forage for food. Consider the individual differences, the *FL* of each chick would randomly choose between 0 and 2.

Chicken Swarm Optimization. Framework of the CSO

```
Initialize a population of N chickens and define the related parameters;
Evaluate the N chickens' fitness values, t=0;
While (t < Max_Generation)
  If (t % G == 0)
    Rank the chickens' fitness values and establish a hierarchal order in the swarm;
    Divide the swarm into different groups, and determine the relationship between the chicks and mother hens in a group; End if
  For i = 1 : N
    If i == rooster  Update its solution/location using equation (1); End if
    If i == hen      Update its solution/location using equation (3); End if
      If i == chick  Update its solution/location using equation (6); End if
    Evaluate the new solution;
    If the new solution is better than its previous one, update it;
  End for
End while
```

### 3.2 Parametric Analysis

There exist six parameters in CSO. Humans keep chickens primarily as a source of food. As the food themselves, only hens can lay eggs, which can also be the source of

food. Hence keeping hens is more beneficial for human than keeping roosters. Thus *HN* would be bigger than *RN*. Given the individual differences, not all hens would hatch their eggs simultaneously. Thus *HN* is also bigger than *MN*. Though each hen can raise more than one chick, we assume the population of adult chickens would surpass that of the chicks, *CN*. As for *G*, it should be set at an appropriate value, which is problem-based. If the value of *G* is very big, it's not conducive for the algorithm to converge to the global optimal quickly. While if the value of *G* is very small, the algorithm may trap into local optimal. After the preliminary test, $G \in [2,20]$ may achieve good results for most problems.

Furthermore, the formula of the chick's movement can be associated with the corresponding part in DE. If we set *RN* and *MN* at 0, thus CSO essentially becomes the basic mutation scheme of DE. Hence the partial conclusions from the DE [2] can be used. In practice, $FL \in [0.4, 1]$ usually perform well.

# 4 Validation and Comparison

## 4.1 Benchmark Problems Optimization

Twelve popular benchmark problems [9, 10] (shown in Table 1) are used to verify the performance of the CSO compared with that of PSO, DE and BA. The statistical results have been obtained, based on 100 independent trials, in all the case studies. The number of iterations is 1,000 in each trial. For a fair comparison, all of the common parameters of these methods, such as the population size, dimensions and maximum number of generations, are set to be the same. The related parameters of these algorithms are showed at Table 2.

**Table 1.** Twelve benchmark problems

| Problem name | ID | Dimension | Bounds | Optimum |
|---|---|---|---|---|
| High Conditioned Elliptic | F1 | 20 | [-100,100] | 0 |
| Bent Cigar | F2 | 20 | [-100,100] | 0 |
| Discus | F3 | 20 | [-100,100] | 0 |
| Ackley | F4 | 20 | [-32,32] | 0 |
| Griewank | F5 | 20 | [-600,600] | 0 |
| Sphere | F6 | 20 | [-100,100] | 0 |
| Step | F7 | 20 | [-100,100] | 0 |
| Powell Sum | F8 | 20 | [-1,1] | 0 |
| Rastrigin | F9 | 20 | [-5,10] | 0 |
| Axis parallel hyper-ellipsoid | F10 | 20 | [-5.12,5.12] | 0 |
| Brown | F11 | 20 | [-1,4] | 0 |
| Exponential | F12 | 20 | [-1,1] | -1 |

**Table 2.** The related parameter values

| Algorithm | Parameters |
|---|---|
| PSO | $c1$=$c2$=1.49445, $w$ = 0.729 |
| DE | $CR$ = 0.9, $F$ = 0.6 |
| BA | $\alpha = \gamma = 0.9,\ f_{min} = 0,\ f_{max} = 2,\ A_0 \in [0,2], r_0 \in [0,1]$ |
| CSO | *RN*=0.2**N*, *HN*=0.6**N*, *CN*=*N*-*RN*-*HN*, *MN*=0.1**N*, *G* = 10, $FL \in [0.5, 0.9]$ |

There are many variants of PSO, DE and BA. In this work, the basic BA and the standard PSO are chosen. As for DE, the DE/rand/1/bin scheme is selected. Table 3 displays the statistical comparison of the four algorithms on twelve benchmark problems. It clearly shows that CSO is superior to PSO, DE and BA on all these problems in terms of accuracy, efficiency and robustness.

The superiority of CSO over PSO, BA and DE should be the case. If we set $RN = CN = 0$, and let $S1$, $S2$ be the parameters like $c1$ and $c2$ in PSO, thus CSO will be similar to the standard PSO. Hence CSO can inherit many advantages of PSO and DE. Moreover, the chickens' swarm intelligence can be efficiently extracted in CSO. Given the diverse laws of the chickens' motions and cooperation between the multi-groups, the search space can be efficiently explored. Under the specific hierarchal order, the whole chicken swarm may behave like a team to forage for food, which can be associated with the objective problems to be optimized. All of these merits enhance the performance of CSO.

**Table 3.** Statistical comparison of CSO with PSO, DE and BA

| Problem | Algorithm | Best | Mean | Worst | Std. |
|---|---|---|---|---|---|
| | PSO | 12600.53084 | 49808.05813 | 101417.29666 | 2124.09 |
| | CSO | 0 | 0 | 0 | 1.89943e-60 |
| F1 | BA | 3782.10211 | 30996.60571 | 84110.41779 | 30996.6057 |
| | DE | 0 | 0 | 0 | 5.99605e-12 |
| | PSO | 0 | 1300 | 10000 | 339.7 |
| | CSO | 0 | 0 | 0 | 1.35815e-62 |
| F2 | BA | 1780594.354 | 2636386.576 | 3804547.807 | 36993.4 |
| | DE | 0 | 0.00001 | 0.0001 | 1.88465e-6 |
| | PSO | 0 | 0 | 0 | 4.66505e-33 |
| | CSO | 0 | 0 | 0 | 9.37294e-66 |
| F3 | BA | 101.92698 | 2256.99578 | 6307.26214 | 132.731 |
| | DE | 0 | 0 | 0 | 3.22023e-12 |
| | PSO | 0 | 0 | 0 | 1.31168e-16 |
| | CSO | 0 | 0 | 0 | 6.12169e-17 |
| F4 | BA | 1.48288 | 2.59402 | 3.07403 | 0.02297 |
| | DE | 0 | 0.35702 | 11.64977 | 0.17096 |
| | PSO | 0 | 0 | 0 | 5.36538e-8 |
| | CSO | 0 | 0 | 0 | 0 |
| F5 | BA | 0.004 | 2.82906 | 15.42094 | 0.296984 |
| | DE | 0 | 0 | 0 | 1.05399e-12 |
| | PSO | 0 | 0 | 0 | 1.8988e-35 |
| | CSO | 0 | 0 | 0 | 4.10796e-70 |
| F6 | BA | 1.867408 | 2.94197 | 4.18701 | 0.0491192 |
| | DE | 0 | 0 | 0 | 1.94304e-12 |
| | PSO | 0 | 0 | 0 | 0 |
| | CSO | 0 | 0 | 0 | 0 |
| F7 | BA | 1 | 3.41 | 6 | 0.103105 |
| | DE | 0 | 0 | 0 | 0 |
| | PSO | 0 | 0 | 0 | 4.78569e-60 |
| | CSO | 0 | 0 | 0 | 0 |
| F8 | BA | 0 | 0 | 0 | 5.02596e-8 |
| | DE | 0 | 0 | 0 | 0 |

**Table 3.** (*Continued*)

| Problem | Algorithm | Best | Mean | Worst | Std. |
|---|---|---|---|---|---|
| | PSO | 10.94454 | 21.26284 | 41.78822 | 0.60048 |
| | CSO | 0 | 0 | 0 | 0 |
| F9 | BA | 88.44729 | 121.99296 | 167.60654 | 1.57913 |
| | DE | 8.41884 | 22.70527 | 43.9751 | 0.706825 |
| | PSO | 0 | 0 | 0 | 2.13096e-37 |
| | CSO | 0 | 0 | 0 | 1.59801e-71 |
| F10 | BA | 24.34652 | 39.73613 | 63.15 | 0.749752 |
| | DE | 0 | 0 | 0 | 4.15726e-14 |
| | PSO | 0 | 1.29 | 8 | 0.171595 |
| | CSO | 0 | 0 | 0 | 5.71381e-72 |
| F11 | BA | 4.22819 | 5.92480 | 7.52686 | 0.0726235 |
| | DE | 0 | 0 | 0 | 2.37888e-15 |
| | PSO | -1 | -1 | -1 | 9.58157e-18 |
| | CSO | -1 | -1 | -1 | 0 |
| F12 | BA | -0.41494 | -0.20415 | -0.12952 | 0.0052408 |
| | DE | -1 | -1 | -1 | 1.75511e-16 |

### 4.2 Speed Reducer Design

Design of the speed reducer [11] (as shown in Fig. 1) is to design a gearbox, which can be rotated at its most efficient speed. The gearbox is described by the face width $b(x_1)$, module of teeth $m(x_2)$, number of teeth in the pinion $z(x_3)$, length of the first shaft between bearings $h1(x_4)$, length of the second shaft between bearings $h2(x_5)$, diameter of the first shaft $d1(x_6)$, and diameter of the first shaft $d2(x_7)$. The optimization in the design of the speed reducer is to minimize its total weight, subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts, and stresses in the shafts. This problem can be formulated as follows.

Minimize $f(\vec{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$

Subject to $g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3}$ $g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_3^2}$ $g_3(\vec{x}) = \frac{1.93x_4^3}{x_2x_6^4x_3}$ $g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_7^4x_3}$ $g_5(\vec{x}) = \frac{1}{110x_6^3}\sqrt{(\frac{745x_4}{x_2x_3})^2 + 16.9 \times 10^6}$ $g_6(\vec{x}) = \frac{1}{85x_7^3}\sqrt{(\frac{745x_5}{x_2x_3})^2 + 157.5 \times 10^6}$ $g_7(\vec{x}) = \frac{x_2x_3}{40}$ $g_8(\vec{x}) = \frac{5x_2}{x_1}$ $g_9(\vec{x}) = \frac{x_1}{12x_2}$ $g_{10}(\vec{x}) = \frac{1.5x_6+1.9}{x_4}$ $g_{11}(\vec{x}) = \frac{1.1x_7+1.9}{x_5}$

Where $2.6 \le x_1 \le 3.6, 0.7 \le x_2 \le 0.8, 17 \le x_3 \le 28, 7.3 \le x_4 \le 8.3, 7.8 \le x_5 \le 8.3, 2.9 \le x_6 \le 3.9, 5 \le x_7 \le 5.5,\ g_i(\vec{x}) \le 1(i = 1, 2, 3, \cdots 11)$.

Table 4 summarizes a comparison of the results achieved by CSO and other algorithms. It clearly shows that CSO's results outperform all the results achieved by the six methods in terms of both optimization accuracy and robustness. The best solution achieved by CSO is $\vec{x}$ = (3.5, 0.7, 17, 7.308, 7.802, 3.35, 5.287) with $f(\vec{x})$ = 2996.60481329. The constraint values are $\vec{g}$ = (-0.07, -0.2, -0.5, -0.9, -2.33e-6, -1.06e-5,-0.7,-5.06e-5,-0.58,-0.05,-0.01), which indicates that the solution is feasible.

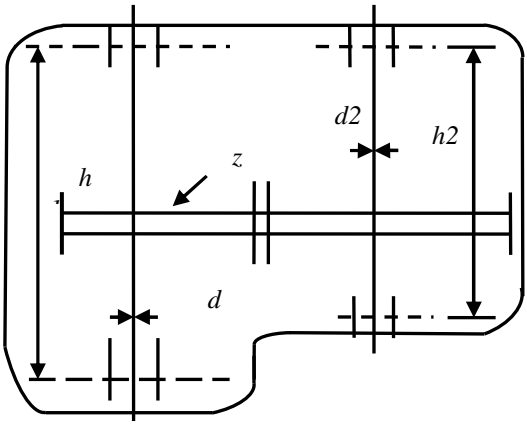


**Fig. 1.** Speed reducer

**Table 4.** Optimization results of the speed reducer design

| | Robert, et al [11] | | | Mezura, et al [12] | Akay, et al [13] | Gandomi et al [14] | CSO |
|---|---|---|---|---|---|---|---|
| | SF1(lBest) | SFI(square) | SFI(gBest) | | | | |
| Best | 3000.737 | 2996.974 | 2998.991 | 2999.264 | 2997.05841 | 3000.981 | 2996.605 |
| Mean | 3015.026 | 3000.278 | 3011.062 | 3014.759 | 2997.05841 | 3007.2 | 2997.764 |
| Worst | 3044.332 | 3007.301 | 3034.973 | N/A | N/A | 3009 | 3007.258 |
| Std. | 9.95 | 2.804 | 9.105 | 11.0 | 0 | 4.963 | 0.165 |

## 5 Discussions

Mimicking the chickens' behaviors, a new bio-inspired algorithm, namely Chicken Swarm Optimization was proposed for optimization problems. The performance of CSO is compared with that of the PSO, DE and BA on twelve benchmark problems. Experiments show that CSO outperforms the PSO, DE and BA in terms of both optimization accuracy and robustness. Moreover, CSO can efficiently solve the speed reducer design, which endues the CSO with a promising prospect of further studying.

One of the reasons that CSO has very promising performance is that CSO inherits major advantages of many algorithms. PSO and the mutation scheme of DE are the special cases of the CSO under appropriate simplifications. What is more significant for the superiority of the CSO is that the chickens' swarm intelligence can be efficiently extracted to optimize problems. The chickens' diverse movements can be conducive for the algorithm to strike a good balance between the randomness and determinacy for finding the optima. The whole chicken swarm consists of several groups, namely multi-swarm. Through integration of the hierarchal order, chickens of the different groups may behave as a team and coordinate themselves to forage for food. Thus CSO can behave intelligently to optimize problems efficiently.

The innovation in this paper not only lies in efficiently extracting the chickens' swarm intelligence to optimize problems, but also making CSO innate multi-swarm method. Multi-swarm technique is usually used to enhance performance of the population-based algorithm. As an innate multi-swarm algorithm, various multi-swarm

techniques can be used to develop the different variants of CSO. Thus CSO has good extensibility. Moreover, from the parametric analysis, the population of the hens is the biggest in the swarm. Thus the performance of CSO largely depends on how the hens' swarm intelligence can be extracted to optimize problems. The motion of the hens can be adaptively controlled according to the fitness value of the problem itself. With the dynamical hierarchal order, the hens swarm can be updated. Hence CSO has the self-adaptive ability to solve the optimization problems.

More comprehensive analyses on the CSO are still need to be investigated in the future. Moreover, we can consider there exist several roosters in a group and dynamically adjust the population of the hens and chicks in each group. It's also significant to tune the related parameters for enhancing the algorithm performance, and design the variants of the CSO to solve many optimization applications.

**Acknowledgments.** This research work was funded by the New Academic Staffs Program of Shanghai Maritime University under Grant GK2013089.

## References

1. Yang, X.S.: Bat algorithm: literature review and applications. International Journal of Bio-inspired Computation 5(3), 141–149 (2013)
2. Das, S., Suganthan, P.N.: Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15(1), 4–31 (2011)
3. Jordehi, A.R., Jasni, J.: Parameter selection in particle swarm optimization: A survey. Journal of Experimental & Theoretical Artificial Intelligence 25(4), 527–542 (2013)
4. Gandomi, A.H., Alavi, A.H.: Krill herd: A new bio-inspired optimization algorithm. Communications in Nonlinear Science and Numerical Simulation 17, 4831–4845 (2012)
5. Cuevas, E., Cienfuegos, M., Zaldivar, D., Cisneros, M.: A swarm optimization algorithm inspired in the behavior of the social-spider. Expert Systems with Applications 40, 6374–6384 (2013)
6. Smith, C.L., Zielinski, S.L.: The Startling Intelligence of the Common Chicken. Scientific American 310(2) (2014)
7. Grillo, R.: Chicken Behavior: An Overview of Recent Science, `http://freefromharm.org/chicken-behavior-an-overview-of-recent-science`
8. Chicken, `http://en.wikipedia.org/wiki/Chicken`
9. Tan, Y., Li, J.Z., Zheng, Z.Y.: ICSI, Competition on Single Objective Optimization (2014), `http://www.ic-si.org/competition/ICSI.pdf`
10. Yang, X.S.: Nature-inspired optimization algorithm. Elsevier (2014)
11. Robert, R., Mostafa, A.: Embedding a social fabric component into cultural algorithms toolkit for an enhanced knowledge-driven engineering optimization. International Journal of Intelligent Computing and Cybernetic 1(4), 563–597 (2008)
12. Mezura, M.E., Hernandez, O.B.: Modified bacterial foraging optimization for engineering design. In: Proceedings of the Artificial Neural Networks in Engineering Conference, vol. 19, pp. 357–364. Intelligent Engineering Systems Through Artificial Neural Networks (2009)
13. Akay, B., Karaboga, D.: Artificial bee colony algorithm for large-scale problems and engineering design optimization. Journal of Intelligent Manufacturing 23(4), 1001–1014 (2012)
14. Gandomi, A.H., Yang, X.S., Alavi, A.H.: Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. Engineering with Computers 29, 17–35 (2013)