Bat Algorithm: Literature Review and Applications

Article *in* International Journal of Bio-Inspired Computation · August 2013

DOI: 10.1504/IJBIC.2013.055093

Xin-She Yang
Middlesex University

Bat Algorithm: Literature Review and Applications

Xin-She Yang

School of Science and Technology, Middlesex University, The Burroughs, London NW4 4BT, United Kingdom.

Xingshi He

School of Science, Xian Polytechnic University, No. 19 Jinhua South Road, Xian 710048, China

Abstract: Bat algorithm (BA) is a bio-inspired algorithm developed by Xin-She Yang in 2010 and BA has been found to be very efficient. As a result, the literature has expanded significantly in the last three years. This paper provides a timely review of the bat algorithm and its new variants. A wide range of diverse applications and case studies are also reviewed and summarized briefly here. In addition, we also discuss the essence of an algorithm and the links between algorithms and self-organization. Further research topics are also discussed.

Keywords: Algorithm; bat algorithm; cuckoo search; firefly algorithm; eagle strategy; nature-inspired algorithm; optimisation; metaheuristics.

Reference to this paper should be made as follows: Yang, X.-S., and He, X., (2013) 'Bat Algorithm: Literature review and applications', *Int. J. Bio-Inspired Computation*, Vol. 5, No. 3, pp.141–149.

1 Introduction

Modern optimisation algorithms are often nature-inspired, typically based on swarm intelligence. The ways for inspiration are diverse and consequently algorithms can be many different types. However, all these algorithms tend to use some specific characteristics for formulating the key updating formulae. For example, genetic algorithms were inspired by Darwinian evolution characteristics of biological systems, and genetic operators such as crossover, mutation and selection of the fittest are used. Solutions in genetic algorithms are represented as chromosomes or binary/real strings. On the other hand, particle swarm optimisation (PSO) was based on the swarming behaviour of birds and fish, and this multi-agent system may have emergent characteristics of swarm or group intelligence (Kennedy and Eberhart, 1995). Many variants of PSO and improvements exist in the literature, and many new metaheuristic algorithms have been developed (Cui, 2009; Yang, 2010; Yang and Deb, 2010b; Yang et al., 2011; Yang et al., 2013).

Algorithms such as genetic algorithms and PSO can be very useful, but they still have some drawbacks in dealing with multimodal optimization problems. One major improvement is the firefly algorithm (FA) which was based on the flashing characteristics of tropical fireflies (Yang, 2008a; Yang, 2013b). The attraction behaviour, light intensity encoding, and distance dependence provide a surprising capability to enable firefly algorithm to handle nonlinear, multimodal optimization problems efficiently. Furthermore, cuckoo search (CS) was based on the brooding behaviour of some cuckoo species (Yang and Deb, 2009; Yang and Deb, 2010b; Yang and Deb, 2013; Gandomi et al, 2013b) which was combined with Lévy flights. The CS algorithm is efficient because it has very good convergence behaviour that can be proved using Markovian probability theory. Other methods such as eagle strategy are also very effective (Yang and Deb, 2010a; Gandomi et al, 2012). In many cases, efficient randomisation techniques can help to enhance the performance of an algorithm (Yang, 2011b; Gandomi et al., 2013a).

As a novel feature, bat algorithm (BA) was based on the echolocation features of microbats (Yang, 2010), and BA uses a frequency-tuning technique to increase the diversity of the solutions in the population, while at the same, it uses the automatic zooming to try to balance exploration and exploitation during the search process by mimicking the variations of pulse emission rates and loudness of bats when searching for prey. As a result, it proves to be very efficient with a typical quick start. Obviously, there is room for improvement. Therefore, this paper intends to review the latest developments of the bat algorithm. The paper is organized as follows: Section 2 introduces the self-organization characteristics of algorithms. Section 3 introduces the basic behaviour of echolocation and the standard formulation of the bat algorithm. Section 4 provides a brief description of the variants of BA, and Section 5 highlights the diverse applications of bat algorithm and its variants. Finally, Section 6 provides some discussions and topics for further research.

2 Magic Formula for Algorithms?

2.1 Essence of An Algorithm

In essence, an algorithm is a procedure to generate outputs from given inputs. Numerically speaking, an optimization algorithm generates a new solution $\boldsymbol{x}^{t+1}$ to a given problem from a known solution $\boldsymbol{x}^t$ at iteration or time t. In general, we have

$$\boldsymbol{x}^{t+1} = A(\boldsymbol{x}^t, p(t)), \tag{1}$$

where A is a nonlinear mapping from a given solution, or d-dimensional vector, $\boldsymbol{x}^t$ to a new solution vector $\boldsymbol{x}^{t+1}$. The algorithm A has k algorithm-dependent parameters $p(t) = (p_1, p_2, ..., p_k)$ that can be time-dependent and can thus be tuned if necessary.

2.2 Self-Organizing Systems

Self-organization may occur in many systems, from physical and chemical to biological and artificial systems. Emergent phenomena such as Releigh-Bénard convection, Turing pattern formation, and organisms and thunderstorms can all be called self-organization (Ashby, 1962; Keller, 2009). Though there is no universal theory for self-organizing processes, some aspects of self-organization can partly be understood using theories based on nonlinear dynamical systems, far-from-equilibrium multiple interacting agents (Prigogine and Nicolois, 1967), and closed-system under unchanging laws (Ashby, 1962). As pointed out by cyberneticist and mathematician Ross Ashby, every isolated determinate dynamic system, obeying unchanging laws, will ultimately develop some sort of 'organisms' that are adapted to their 'environments' (Ashby, 1962).

Going to equilibrium is trivial for simple systems. However, for a complex system, if its size is so large that its equilibrium states are just a fraction of the vast number of possible states, and if the system is allowed to evolve long enough, some self-organized structures may emerge. The changes in environments can apply pressure on the system to re-organize and adapt to such changes. If the system have sufficient perturbations or noise, often working at the edge of the chaos, some spontaneous formation of structures will emerge as the systems move, far-from-equilibrium, and select some states, thus reducing the uncertainty or entropy.

Mathematically speaking, the state set S of a complex system such as a machine, may change from initial states $S(\psi)$ to other states $S(\phi)$, subject to the change of a parameter set $\alpha(t)$ which can be time dependent. That is,

$$S(\psi) \stackrel{\alpha(t)}{\longrightarrow} S(\phi), \tag{2}$$

where $\alpha(t)$ must come from external conditions such as the heat flow in Raleigh-Bénard convection, not from the states S themselves. Obviously, $S + \alpha(t)$ can be considered as a larger, closed system (Ashby, 1962). In this sense, self-organization is equivalent to a mapping from some high-entropy states to low-entropy states.

An optimization algorithm can be viewed as a complex, dynamical system. If we can consider the convergence process as a self-organizing process, then there are strong similarities and links between self-organizing systems and optimization algorithms.

2.3 Algorithms as Self-Organization

To find the optimal solution $\boldsymbol{x}_*$ to a given optimization problem S with often an infinitely number of states is to select some desired states ϕ from all states ψ, according to some predefined criterion D. We have

$$S(\psi) \stackrel{\boldsymbol{A}(t,D,\boldsymbol{p})}{\longrightarrow} S(\phi(x_*)), \tag{3}$$

where the final converged state ϕ corresponds to an optimal solution $\boldsymbol{x}_*$ to the problem of interest. The selection of the system states in the design space is carried out by running the optimization algorithm $\boldsymbol{A}$. The behavior of the algorithm is controlled by $\boldsymbol{p}$, the initial solution $\boldsymbol{x}^{t=0}$ and the stopping criterion D. We can view the combined $S + \boldsymbol{A}$ as a complex system with a self-organizing capability.

The change of states or solutions of the problem of interest is controlled by the algorithm $\boldsymbol{A}$. In many classical algorithms such as hill-climbing, gradient information is often used to select states, say, the minimum value of the landscape, and the stopping criterion can be a given tolerance or accuracy, or zero gradient, etc.

Alternatively, an algorithm can act like a tool to tune a complex system. If an algorithm does not use any state information of the problem, then it is more likely to be versatile to deal with many types of problems. However, such black-box approaches can also imply that the algorithm may not be efficient as it could be for a given type of problem. For example, if the optimization problem is convex, algorithms that use such convexity information will be more efficient than the ones that do not use such information. In order to select states/solutions efficiently, the information from the search process should be used to enhance the search process. In many cases, such information is often fed into the selection mechanism of an algorithm. By far the most widely used selection mechanism is to identify and keep the best solution found so far. That is, some form of 'survival of the fitness' is used.

Optimization algorithms can very diverse. There are several dozens of widely used algorithms. The main characteristics of different algorithms will only depend on the actual, often highly nonlinear or implicit, forms of $\boldsymbol{A}(t)$ and their parameters $\boldsymbol{p}(t)$.

In many situations concerning optimization, the generation and verification of the new solutions can often involve computationally expensive computer simulations or even measurements of the physical system. In such cases, the expensive model of the system under consideration is often replaced by its cheaper representation, so-called surrogate model, and the algorithm A uses that model to produce a new solution. The parameters $p(t)$ may then include variables that are used to align the surrogate with the expensive model to make it reliable representation of the latter (Koziel and Yang, 2011).

2.4 An Ideal Algorithm?

In an ideal world, we hope to start from any initial guess solution and wish to get the best solution in a single step. That is, to use the minimal computational effort. In other words, this is essentially saying that the algorithm simply has to tell what the best answer is to any given problem in a single step! You may wonder if such an algorithm exists. In fact, the answer is yes, for a very specific type of problem — quadratic, convex problems.

We know Newton-Raphsons method is a root-finding algorithm. It can find the roots of $f(x) = 0$. As the minimum or maximum of a function $f(x)$ has to satisfy the critical condition $f(x) = 0$, therefore, this optimization problem now becomes a problem of finding the roots of $f(x)$. Newton-Raphson method provides the following iteration formula

$$x_{t+1} = x_t - \frac{f'(x_t)}{f''(x_t)}. \tag{4}$$

For a quadratic function, for example, $f(x) = x^2$, if we start from a fixed location, $x_0 = a$ at $t = 0$, we have $f'(a) = 2a$ and $f''(a) = 2$. Then, we have

$$x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)} = a - \frac{2a}{2} = 0, \tag{5}$$

which is exactly the optimal solution $f_{\min} = 0$ at $x^* = 0$. This solution is also globally optimal. That is to say, we have found the global optimum in a single step. In fact, for any quadratic functions that are also convex, Newton-Raphson is an ideal algorithm. However, the world is not convex and certainly not quadratic, real-world problems are often highly nonlinear, and therefore there is no ideal algorithm in general.

For non-deterministic polynomial-time (NP) hard problems, or NP-hard problems, there is no known efficient algorithm at all. Such hard problems require a huge amount of research efforts to search for specific techniques that are still not satisfactory in practice. These challenges can also be a driving force for more active research.

2.5 The Magic Formulae?

The ultimate aim for optimization and algorithm researchers is to find a magic formula or method that works for many problems, like the Newton-Raphson method for quadratic functions. We wish it could work like a 'magic' to provide the best solution for any problem in a few steps. However, such formulae may never exist.

As optimization algorithms are iterative, an algorithm to solve a given problem Q can be written as the following generic formula

$$\boldsymbol{x}^{t+1} = g(\boldsymbol{x}^t, p, Q), \tag{6}$$

which forms a piece-wise trajectory in the search space. This algorithm depends on a parameter p, starting with initial guess $\boldsymbol{x}_0$. This iterative path will depend on the problem (Q) or its objective function $f(\boldsymbol{x})$. However, as algorithms nowadays tend to use multiple agents as those in swarm intelligence, we have to extend the above equation to a population of n agents/solutions

$$[\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, ..., \boldsymbol{x}_n]^{t+1}$$

$$= g([\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, ..., \boldsymbol{x}_n]^t, [p_1, p_2, p_3, ..., p_k]^t, Q), \tag{7}$$

which has a population size of n and depends on k different algorithm-dependent parameters. Each iteration will produce n new, often different, solutions $[\boldsymbol{x}_1, ..., \boldsymbol{x}_n]$. Modern metaheuristic algorithms have stochastic components, which means some of these k parameters can be drawn from some probability distributions. If we wish to express the randomness more explicitly, we can rewrite the above as

$$[\boldsymbol{x}_1, \boldsymbol{x}_n, ..., \boldsymbol{x}_n]^{t+1} = g([\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]^t,$$

$$[p_1, ..., p_k]^t, [\epsilon_1, ..., \epsilon_m]^t, Q), \tag{8}$$

where m is the number of random variables that are drawn from some probability distributions such as uniform, Gaussian or Lévy distributions (Yang, 2008a; Yang, 2008b; Yang, 2008c; Yang, 2013a; Yang et al., 2013). In some cases as those in cuckoo search, these random variables can also be drawn from a Lévy distribution (Yang and Deb, 2009; Yang and Deb, 2010b).

Though there is no magic formula, each algorithm strives to use fewer iterations (or smaller t) as possible. The only difference among algorithms is the exact form of $g(.)$. In fact, sometimes, the procedure $g(.)$ can be divided into many sub-steps or procedures with different branches, so that these branches can be used in a random manner during iterations, and one good example is the Eagle Strategy that uses a two-stage iterative strategy (Yang and Deb, 2010a). That is the essence of all contemporary swarm intelligence and bio-inspired metaheuristic algorithms.

3 The Standard Bat Algorithm

The standard bat algorithm, developed by Xin-She Yang in 2010, was based on the echolocation or bio-sonar characteristics of microbats (Yang, 2010). Before we outline the details of the algorithm, let us briefly introduce the echolocation.

3.1 Echolocation of Microbats

There are about 1000 different species of bats (Colin, 2000). Their sizes can vary widely, ranging from the tiny bumblebee bats of about 1.5 to 2 grams to the giant bats with a wingspan of about 2 m and may weight up to about 1 kg. Most bats uses echolocation to a certain degree; among all the species, microbats use echolocation extensively, while megabats do not.

Microbats typically use a type of sonar, called, echolocation, to detect prey, avoid obstacles, and locate their roosting crevices in the dark. They can emit a very loud sound pulse and listen for the echo that bounces back from the surrounding objects (Richardson, 2008). Their pulses vary in properties and can be correlated with their hunting strategies, depending on the species. Most bats use short, frequency-modulated signals to sweep through about an octave, and each pulse lasts a few thousandths of a second (up to about 8 to 10 ms) in the frequency range of 25kHz to 150 kHz. Typically, microbats can emit about 10 to 20 such sound bursts every second, and the rate of pulse emission can be sped up to about 200 pulses per second when homing on their prey. Since the speed of sound in air is about $v = 340$ m/s, the wavelength λ of the ultrasonic sound bursts with a constant frequency f is given by $\lambda = v/f$, which is in the range of 2mm to 14mm for the typical frequency range from 25kHz to 150 kHz. Interestingly, these wavelengths are in the same order of their prey sizes.

Though in reality microbats can also use time delay between their ears and loudness variations to sense three-dimensional surroundings, we are mainly interested in some features of echolocation so that we can some link them with the objective function of an optimization problem, which makes it possible to formulate a smart, bat algorithm.

3.2 Bat Algorithm

Based on the above description and characteristics of bat echolocation, Xin-She Yang (2010) developed the bat algorithm with the following three idealised rules:

1. All bats use echolocation to sense distance, and they also 'know' the difference between food/prey and background barriers in some magical way;
2. Bats fly randomly with velocity $\boldsymbol{v}_i$ at position $\boldsymbol{x}_i$ with a frequency f (or wavelength λ) and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$, depending on the proximity of their target;
3. Although the loudness can vary in many ways, we assume that the loudness varies from a large (positive) A_0 to a minimum constant value $A_{\min}$.

For simplicity, we do not use ray tracing in this algorithm, though it can form an interesting feature for further extension. In general, ray tracing can be computationally extensive, but it can be a very useful feature for computational geometry and other applications. Furthermore, a given frequency is intrinsically linked to a wavelength. For example, a frequency range of [20kHz, 500kHz] corresponds to a range of wavelengths from 0.7mm to 17mm in the air. Therefore, we can describe the changes either in terms of frequency f or wavelength λ to suit different applications, depending on the ease of implementation and other factors.

3.3 Bat Motion

Each bat is associated with a velocity $\boldsymbol{v}_i^t$ and a location $\boldsymbol{x}_i^t$, at iteration t, in a d-dimensional search or solution space. Among all the bats, there exists a current best solution $\boldsymbol{x}_*$. Therefore, the above three rules can be translated into the updating equations for $\boldsymbol{x}_i^t$ and velocities $\boldsymbol{v}_i^t$:

$$f_i = f_{\min} + (f_{\max} - f_{\min})\beta, \tag{9}$$

$$\boldsymbol{v}_i^t = \boldsymbol{v}_i^{t-1} + (\boldsymbol{x}_i^{t-1} - \boldsymbol{x}_*) f_i, \tag{10}$$

$$\boldsymbol{x}_i^t = \boldsymbol{x}_i^{t-1} + \boldsymbol{v}_i^t, \tag{11}$$

where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution.

As mentioned earlier, we can either use wavelengths or frequencies for implementation, we will use $f_{\min} = 0$ and $f_{\max} = O(1)$, depending on the domain size of the problem of interest. Initially, each bat is randomly assigned a frequency which is drawn uniformly from $[f_{\min}, f_{\max}]$. For this reason, bat algorithm can be considered as a frequency-tuning algorithm to provide a balanced combination of exploration and exploitation. The loudness and pulse emission rates essentially provide a mechanism for automatic control and auto-zooming into the region with promising solutions.

3.4 Variations of Loudness and Pulse Rates

In order to provide an effective mechanism to control the exploration and exploitation and switch to exploitation stage when necessary, we have to vary the loudness A_i and the rate r_i of pulse emission during the iterations. Since the loudness usually decreases once a bat has found its prey, while the rate of pulse emission increases, the loudness can be chosen as any value of convenience, between $A_{\min}$ and $A_{\max}$, assuming $A_{\min} = 0$ means that a bat has just found the prey and temporarily stopped emitting any sound. With these assumptions, we have

$$A_i^{t+1} = \alpha A_i^t, \tag{12}$$

and

$$r_i^{t+1} = r_i^0[1 - \exp(-\gamma t)], \tag{13}$$

where α and γ are constants. In essence, here α is similar to the cooling factor of a cooling schedule in simulated annealing. For any $0 < \alpha < 1$ and $\gamma > 0$, we have

$$A_i^t \to 0, \tag{14}$$

and

$$r_i^t \to r_i^0, \ \text{ as } \ t \to \infty. \tag{15}$$

In the simplest case, we can use $\alpha = \gamma$, and we have used $\alpha = \gamma = 0.9$ to 0.98 in our simulations.

3.5 How to Discretize

The standard bat algorithm is for continuous optimization. In order to deal with combinatorial problems effectively, some modifications are needed. Nakamura et al. (2012) extended the standard bat algorithm to the so-called binary bat algorithm (BBA) for feature selection. A key step is to convert continuous-valued positions of bats into binary values using a sigmoid funciton

$$S(\boldsymbol{x}_i^t) = \frac{1}{1 + \exp[-\boldsymbol{x}_i^t]}, \tag{16}$$

which leads to

$$\boldsymbol{x}_i^t = \begin{cases} 1 \text{ if } S(\boldsymbol{x}_i^t) > \sigma, \\ \\ 0 \text{ otherwise,} \end{cases} \tag{17}$$

where σ is a random variable that can be drawn from a uniform distribution $U(0, 1)$. This transformation will generate only binary states in a vast Boolean lattice, and consequently it can deal with feature selection very effectively (Nakumura et al., 2012).

4 Variants of Bat Algorithm

The standard bat algorithm has many advantages, and one of the key advantages is that it can provide very quick convergence at a very initial stage by switching from exploration to exploitation. This makes it an efficient algorithm for applications such as classifications and others when a quick solution is needed. However, if we allow the algorithm to switch to exploitation stage too quickly by varying A and r too quickly, it may lead to stagnation after some initial stage. In order to improve the performance, many methods and strategies have been attempted to increase the diversity of the solution and thus to enhance the performance, which produced a few good and efficient variants of bat algorithm.

From a quick literature survey, we found the following bat algorithm variants:

- *Fuzzy Logic Bat Algorithm (FLBA)*: Khan et al. (2011) presented a variant by introducing fuzzy logic into the bat algorithm, they called their variant fuzzy bat algorithm.
- *Multiobjective bat algorithm (MOBA)*: Yang (2011a) extended BA to deal with multiobjective optimization, which has demonstrated its effectiveness for solving a few design benchmarks in engineering.
- *K-Means Bat Algorithm (KMBA)*: Komarasamy and Wahi (2012) presented a combination of K-means and bat algorithm (KMBA) for efficient clustering.
- *Chaotic Bat Algorithm (CBA)*: Lin et al. (2012) presented a chaotic bat algorithm using Lévy flights and chaotic maps to carry out parameter estimation in dynamic biological systems.
- *Binary bat algorithm (BBA)*: Nakamura et al. (2012) developed a discrete version of bat algorithm to solve classifications and feature selection problems.
- *Differential Operator and Lévy flights Bat Algorithm (DLBA)*: Xie et al. (2013) presented a variant of bat algorithm using differential operator and Lévy flights to solve function optimization problems.
- *Improved bat algorithm (IBA)*: Jamil et al. (2013) extended the bat algorithm with a good combination of Lévy flights and subtle variations of loudness and pulse emission rates. They tested the IBA versus over 70 different test functions and proved to be very efficient.

There are other improvements and variants of bat algorithm. For example, Zhang and Wang (2012) used mutation to enhance the diversity of solutions and then used for image matching. In addition, Wang et al. (2012) also introduced mutation to the bat algorithm, and Wang and Guo (2013) hybridized bat algorithm with harmony search and have produced a hybrid bat algorithm for numerical optimization of function benchmarks.

On the other hand, Fister Jr et al. (2013) developed a hybrid bat algorithm using differential evolution as a local search part of bat algorithm, while Fister et al. (2013) incorporate quaternions into bat algorithm and presented a quaternion bat algorithm (QBA) for computational geometry and large-scale optimization problems with extensive rotations. It can be expected that more variants are still under active research.

5 Applications of Bat Algorithm

The standard bat algorithm and its many variants mean that the applications are also very diverse. In fact, since the original bat algorithm has been developed (Yang, 2010), bat algorithms have been applied in almost every area of optimisation, classifications, image processing, feature selection, scheduling, data mining and others. In the rest of the paper, we will briefly highlight some of the applications (Yang, 2010; Parpinelli and Lopes, 2011; Yang et al., 2012a; Yang, 2012; Yang, 2013; Gandomi et al., 2013).

5.1 Continuous Optimisation

Among the first set of applications of bat algorithm, continuous optimisation in the context of engineering design optimisation has been extensively studied, which demonstrated that BA can deal with highly nonlinear problems efficiently and can find the optimal solutions accurately (Yang, 2010; Yang and Gandomi, 2012; Yang, 2012; Yang et al., 2012a). Case studies include pressure vessel design, car side design, spring and beam design, truss systems, tower and tall building design and others. Tsai et al. (2011) solved numerical optimisation problems using bat algorithm.

In addition, Bora et al. (2012) optimised the brushless DC wheel motors using bat algorithm with superior results. BA can also handle multiobjective problems effectively (Yang, 2011a).

5.2 Combinatorial Optimisation and Scheduling

From computational complexity point of view, continuous optimization problems can be considered as easy, though it may be still very challenging to solve. However, combinatorial problems can be really hard, often non-deterministic polynomial time hard (NP-hard). Ramesh et al. (2013) presented a detailed study of combined economic load and emission dispatch problems using bat algorithm. They compared bat algorithm with ant colony algorithm (ABC), hybrid genetic algorithm and other methods, and they concluded that bat algorithm is easy to implement and much superior to other algorithms in terms of accuracy and efficiency.

Musikapun and Pongcharoen (2012) solved multi-stage, multi-machine, multi-product scheduling problems using bat algorithm, and they solved a class of NP hard problems with a detailed parametric study. They also implied that that the performance can be improved by about 8.4% using optimal set of parameters.

5.3 Inverse Problems and Parameter Estimation

Yang et al. (2012b) use the bat algorithm to study topological shape optimization in microelectronic applications so that materials of different thermal properties can be placed in such a way that the heat transfer is most efficient under stringent constraints. It can also be applied to carry out parameter estimation as an inverse problem. If an inverse problem can be properly formulated, then bat algorithm can provide better results than least-squares methods and regularization methods.

Lin et al. (2012) presented a chaotic Lévy flight bat algorithm to estimate parameters in nonlinear dynamic biological systems, which proved the effectiveness of the proposed algorithm.

5.4 Classifications, Clustering and Data Mining

Komarasamy and Wahi (2012) studied K-means clustering using bat algorithm and they concluded that the combination of both K-means and BA can achieve higher efficiency and thus performs better than other algorithms.

Khan et al. (2011) presented a study of a clustering problem for office workplaces using a fuzzy bat algorithm. Khan and Sahari (2012a) also presented a comparison study of bat algorithm with PSO, GA, and other algorithms in the context for e-learning, and thus suggested that bat algorithm has clearly some advantages over other algorithms. Then, Khan and Sahari (2012b) also presented a study of clustering problems using bat algorithm and its extension as a bi-sonar optimization variant with good results.

On the other hand, Mishra et al. (2012) used bat algorithm to classify microarray data, while Natarajan et al. (2012) presented a comparison study of cuckoo search and bat algorithm for Bloom filter optimization. Damodaram and Valarmathi (2012) studied phishing website detection using modified bat algorithm and achieved very good results.

Marichelvam and Prabaharan (2012) used bat algorithm to study hybrid flow shop scheduling problems so as to minimize the makespan and mean flow time. Their results suggested that BA is an efficient approach for solving hybrid flow shop scheduling problems. Faritha Banu and Chandrasekar (2013) used a modified bat algorithm to record deduplication as an optimisation approach and data compression technique. Their study suggest that the modified bat algorithm can perform better than genetic programming.

5.5 Image Processing

Abdel-Rahman et al. (2012) presented a study for full body human pose estimation using bat algorithm, and they concluded that BA performs better than particle swarm optimization (PSO), particle filter (PF) and annealed particle filter (APF).

Du and Liu (2012) presented a variant of bat algorithm with mutation for image matching, and they indicated that their bat-based model is more effective and feasible in imagine matching than other models such as differential evolution and genetic algorithms.

5.6 Fuzzy Logic and Other Applications

Reddy and Manoj (2012) presented a study of optimal capacitor placement for loss reduction in distribution systems using bat algorithm. It combined with fuzzy logic to find optimal capacitor sizes so as to minimize the losses. Their results suggested that the real power loss can be reduced significantly.

Furthermore, Lemma et al. (2011) used fuzzy systems and bat algorithm for exergy modelling, and later Tamiru and Hashim (2013) applied bat algorithm to study fuzzy systems and to model exergy changes in a gas turbine.

At the time of writing when we searched the Google scholar and other databases, we found other papers on bat algorithm that were either just accepted or conference presentations. However, these papers do not yet have enough details to be included in this review. In fact, as the literature is expanding, more and more papers on bat algorithm are emerging, a further timely review will be needed within the next two years.

6 Discussions and Conclusions

Likely many metaheuristic algorithms, bat algorithm has the advantage of simplicity and flexibility. BA is easy to implement, and such a simple algorithm can be very flexible to solve a wide range of problems, as we have seen in the above review.

6.1 Why Bat Algorithm is Efficient

A natural question is: why bat algorithm is so efficient? There are many reasons for the success of bat-based algorithms. By analysing the key features and updating equations, we can summarize the following three key points/features:

- *Frequency tuning*: BA uses echolocation and frequency tuning to solve problems. Though echolocation is not directly used to mimic the true function in reality, frequency variations are used. This capability can provide some functionality that may be similar to the key feature used in particle swarm optimization and harmony search. Therefore, BA may possess the advantages of other swarm-intelligence-based algorithms.

- *Automatic zooming*: BA has a distinct advantage over other metaheuristic algorithms. That is, BA has a capability of automatically zooming into a region where promising solutions have been found. This zooming is accompanied by the automatic switch from explorative moves to local intensive exploitation. As a result, BA has a quick convergence rate, at least at early stages of the iterations, compared with other algorithms.

- *Parameter control*: Many metaheuristic algorithms used fixed parameters by using some, pre-tuned algorithm-dependent parameters. In contrast, BA uses parameter control, which can vary the values of parameters (A and r) as the iterations proceed. This provides a way to automatically switch from exploration to exploitation when the optimal solution is approaching. This gives another advantages of BA over other metaheuristic algorithms.

In addition, preliminary theoretical analysis by Huang et al. (2013) suggested that BA has guaranteed global convergence properties under the right condition, and BA can also solve large-scale problems effectively.

6.2 Further Research Topics

However, there are still some important issues that require more research. These key issues are: parameter-tuning, parameter control and speedup of convergence.

Firstly, parameter-tuning is important for any metaheuristic algorithm to work properly. In almost all cases, the performance of an algorithm is largely dependent on the parameters of the algorithm. To find the best parameter settings, detailed parametric studies have to be carried out. It is not known yet if there is a method to automatically tune parameters for an algorithm to achieve the optimal performance for a given set of problems. This should be an important topic for further research.

Secondly, associated with the parameter tuning, there is an important issue of parameter control. In many algorithms, the parameter settings are fixed, and these settings will not vary during the iterations. It could be advantageous and sometime necessary to vary the values of algorithm-dependent parameters during the iterative search process. How to vary or control these parameters is another, higher level, optimisation problem, which needs further studies. For the bat algorithm, we have introduced the basic parameter control strategy, there is still room for improvement. An open question is that: what is the best control strategy so as to switch from exploration to exploitation at the right time?

Finally, even though the bat algorithm and other algorithms are efficient, it is still possible to improve and enhance their performance further. However, how to speed up the convergence of an algorithm is still a very challenging question. It is hoped this this paper can inspire more research in the near future. Future research should focus on the theoretical understanding of metaheuristic algorithms and large-scale problems in real-world applications (Yang, 2005; Koziel and Yang, 2011; Yang and Koziel, 2011; Yang et al., 2012b).

References and Notes

Abdel-Rahman, E. M., Ahmad, A. R., Akhtar, S., (2012). A metaheurisic bat-inspired algorithm for full body human pose estimation, in: *Ninth Conference on Computer and Robot Vision*, pp. 369–375.

Ashby W. R., (1962). Princinples of the self-organizing sysem, in: *Pricinples of Self-Organization: Transactions of the University of Illinois Symposium* (Eds H. Von Foerster and G. W. Zopf, Jr.), Pergamon Press, London, UK. pp. 255–278.

Bora, T. C., Coelho, L. S., Lebensztajn, L., (2012). Bat-inspired optimization approach for the brushless DC wheel motor problem, *IEEE Trans. Magnetics*, Vol. 48, No. 2, 947-950 (2012).

Colin, T., (2000). *The Varienty of Life*. Oxford University Press, Oxford.

Cui, Z. H., and Cai, X. J. (2009). Integral particle swarm optimisation with dispersed accelerator information, *Fundam. Inform.*, Vol. 95, No. 3, 427–447.

Damodaram, R., Valarmathi, M. L., (2012). Phishing website detection and optimization using modified bat algorithm, *Int. J. Engineering Research and Applications*, Vol. 2, No. 1, pp. 870–876.

Du, Z. Y., Liu B., (2012). Image matching using a bat algorithm with mutation, *Applied Mechanics and Materials*, Vol. 203, No. 1, pp. 88–93.

Faritha Banu, A., Chandrasekar, C., (2012). An optimized appraoch of modified bat algorithm to record deduplication, *Int. Journal of Computer Applications*, Vol. 62, No. 1, pp. 10–15.

Fister Jr., I., Fister, D., and Yang, X. S., (2013). A hybrid bat algorithm, *Elekrotehniški Vestnik* (English Edition), (2013, submitted).

Fister, I., Fister Jr., I., Yang, X. S., and Brest, J., (2013). On the representation of individual s using quaternions in swarm intelligence and evolutionary computation, *IEEE Trans. Evol. Computation*, (2013, submitted).

Gandomi, A. H., Yang, X. S., Talatahari, S., and Deb, S., (2012). Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization, *Computers & Mathematics with Applications*, vol. 63, no. 1, pp. 191–200.

Gandomi, A. H., Yun, G. J., Yang, X. S., Talatahari, S., (2013a). Chao-enhanced accelerated particle swarm optimization, *Communications in Nonlinear Science and Numerical Simulation*, Vol. 18, No. 2, pp. 327–340.

Gandomi, A. H., Yang, X. S., Alavi, A. H., Talatahari, S. (2013b). Bat algorithm for constrained optimization tasks, *Neural Computing and Applications*, http://link.springer.com/article/10.1007

Huang, G. Q., Zhao, W. J., and Lu, Q. Q., (2013). Bat algorithm with global convergence for solving large-scale optimization problem, Application Research of Computers, vol. 30, no. 3, 1-10 (in Chinese).

Jamil, M., Zepernic, H.-J., and Yang, X. S., (2013). Improved bat algorithm for global optimization, *Applied Soft Computing*, (2013, submitted).

Khan, K., Nikov, A., Sahai A., (2011). A fuzzy bat clustering method for ergonomic screening of office workplaces, S3T 2011, *Advances in Intelligent and Soft Computing*, 2011, Volume 101/2011, No. 1, pp. 59–66.

Khan, K., and Sahai, A., (2012a). A comparison of BA, GA, PSO, BP and LM for training feed forward neural networks in e-learning context, *Int. J. Intelligent Systems and Applications (IJISA)*, Vol. 4, No. 7, pp. 23–29.

Khan, K., and Sahai, A., (2012b). A fuzzy c-means bi-sonar-based metaheuristic optimization algorithm, *INt. J. of Interactive Multimedia and Artificial Intelligence*, Vol. 1, no. 7, pp. 26–32.

Keller E. F., (2009). Organisms, machines, and thunderstorms: a history of self-organization, part two. Complexity, emergenece, and stable attractors, *Historical Studies in the Natural Sciences*, Vol. 39, no. 1, pp. 1–31.

Komarasamy, G., and Wahi, A., (2012). An optimized K-means clustering techniqueusing bat algorithm, *European J. Scientific Research*, Vol. 84, No. 2, pp. 263-273.

Koziel, S., and Yang, X. S., (2011). *Computational Optimization, Methods and Algorithms*, Springer, Heidelberg, Germany.

Lemma, T. A., Bin Mohd Hashim, F., (2011). Use of fuzzy systems and bat algorithm for exergy modelling in a gas turbine generator, IEEE Colloquium on Humanities, Science and Engineering (CHUSER'2011), 5-6 Dec. 2011, pp. 305–310.

Lin, J. H., Chou,C. W., Yang, C. H.,Tsai, H. L., (2012). A chaotic Levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems, *J. Computer and Information Technology*, Vol. 2, No. 2, pp. 56–63.

Marichelvam, M. K., and Prabaharam, T., (2012). A bat algorithm for realistic hybrid flowshop schedulihng problems to minimize makespan and mean flow time, *ICTACT Journal on Soft Computing*, Vol. 3, No. 1, pp. 428–433.

Mishra, S., Shaw, K., Mishra, D., (2012). A new meta-heuristic bat inspired classification approach for microarray data, *Procedia Technology*, Vol. 4, No. 1, pp. 802–806.

Musikapun, P., Pongcharoen, P., Solving multi-stage multi-machine multi-product scheduling problem using bat aglorithm, (2012). 2nd International Conference on Management and Artificial Intelligence (IPEDR),Vol. 35, IACSIT Press, Singapore, pp. 98–102.

Nakamura, R. Y. M., Pereira, L. A. M., Costa, K. A., Rodrigues, D., Papa, J. P., Yang, X. S., (2012). BBA: A binary bat algorithm for feature selection, in: 25th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), 22-25 Aug. 2012, IEEE Publication, pp. 291-297.

Natarajan, A., Subramanian, S., Premalatha, K., (2012). A comparative study of cuckoo search and bat algorithm for Bloom filter optimisation in spam filtering, *Int. J. Bio-Inspired Computation*, Vol. 4, No. 2, pp. 89–99.

Parpinelli, R. S., and Lopes, H. S., (2011). New inspirations in swarm intelligence: a survey, *Int. J. Bio-Inspired Computation*, Vol. 3, No. 1, pp. 1–16.

Prigogine I. and Nicolois G., (1967). On symmetry-breaking instabilities in dissipative systems, *J. Chemical Physics*, Vol. 46, No. 4, pp. 3542–50.

Ramesh, B., Mohan, V. C. J., Reddy, V. C. V., (2013). Application of bat algorithm for combined economic load and emission dispatch, *Int. J. of Electricl Engineering and Telecommunications*, Vol. 2, No. 1, pp. 1–9.

Reddy, V. U., Manoj, A., (2012). Optimal capacitor placement for loss reduction in distribution systems using bat algorithm, *IOSR Journal of Engineering*, Vol. 2, No. 10, pp. 23–27.

Richardson, P., (2008). *Bats*. Natural History Museum, London.

Tamiru, A. L., Hashim, F. M., (2013). Application of bat algorithm and fuzzy systems to model exergy changes in a gas turbine, in: *Artificial Intelligence, Evolutionary Computing and Metaheuristics* (Eds. X. S. Yang), Studies in Computational Intelligence, Vol. 427, Springer, Heidelberg, pp. 685–719.

Tsai, P. W., Pan, J. S., Liao, B. Y., Tsai, M. J., Istanda, V., (2011). Bat algorithm inspired algorithm for solving numerical optimization problems, *Applied Mechanics and Materials*, Vol. 148-149, No. 1, pp.134–137.

Wang, G. G, Guo, L. H., Duan, H., Liu, L, Wang, H. Q., (2012). A bat algorithm with mutation for UCAV path planning, *Scientific World Journal*, Vol. 2012, 15 pages. doi:10.1100/2012/418946 http://www.hindawi.com/journals/tswj/2012/418946/

Wang, Gaige, and Guo, Lihong, (2013). A novel hybrid bat algorithm with harmony search for global numerical optimization, *Journal of Applied Mathematics*, Vol. 2013, No. 2013, pp. 696491–21. http://www.hindawi.com/journals/jam/2013/696491/

Xie, J., Zhou, Y. Q., Chen,H., A novel bat algorithm based on differential operator and Lévy flights trajectory, *Computational Intelligence and Neuroscience*, Vol. 2013, Article ID: 453812 DOI:www.hindawi.com/journals/cin/aip/453812.pdf

Yang, X. S., (2005). Modelling heat transfer of carbon nanotubes, *Modelling and Simulation in Materials Science and Engineering*, Vol. 13, No. 6, pp. 893–902.

Yang, X. S., (2008a). *Nature-Inpsired Metaheursitic Algorithms*, Luniver Press, Frome, UK.

Yang, X. S., (2008b). *Introduction to Mathematical Optimization: From Linear Programming to Metaheuristics*, Cambridge International Science Publishing, Cambridge, UK.

Yang, X. S., (2008c). *Introduction to Computational Mathematics*, World Scientific Publishing Co. Inc., Singapore.

Yang, X. S., (2010). A New Metaheuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization (NISCO 2010) (Eds. Cruz, C.; González, J. R.; Pelta, D. A.; Terrazas, G), Studies in Computational Intelligence Vol. 284, Springer Berlin, pp. 65–74.

Yang, X. S. and Koziel, S., (2011). *Computational Optimization and Applications in Engineering and Industry*, Studies in Computational Intelligence, Vol. 359, Springer, Heidelberg.

Yang, X. S., and Deb, S., (2010a). Eagle strategy using Lévy walk and firefly algorithms for stochastic optimization, Nature Inspired Cooperative Strategies for Optimization (NICSO2010), (Eds. Cruz, C.; González, J. R.; Pelta, D. A.; Terrazas, G), Studies in Computational Intelligence Vol. 284, pp. 101–111.

Yang, X. S. and Deb, S. (2010b). Engineering optimisation by cuckoo search, *International Journal of Mathematical Modelling and Numerical Optimisation*, Vol. 1, No. 4, 330–343.

Yang, X. S., (2011a). Bat algorithm for multi-objective optimisation, *Int. J. Bio-Inspired Computation*, Vol. 3, No. 5, pp. 267–274.

Yang, X. S., (2011b). Review of meta-heuristics and generalised evolutionary walk algorithm, *Int. J. of Bio-Inspired Computation*, Vol. 3, No. 2, pp. 77–84.

Yang, X. S. and Deb, S. (2009). Cuckoo search via Lévy flights, in: *Proc. of World Congress on Nature & Biologically Inspired Computing* (NaBic 2009), IEEE Publications, USA, pp. 210–214.

Yang, X. S. and Deb, S., (2013). Cuckoo search: Recent advances and applications, *Neural Computing and Applications*, Vol. 22, March 2013 (in press). Online First, DOI 10.1007/s00521-013-1367-1.

Yang, X. S., Deb, S., and Fong, S., (2011). Accelerated particle swarm optimization and support vector machine for business optimization and applications, in: Networked Digital Technologies 2011, Communications in Computer and Information Science, **136**, pp. 53–66.

Yang, X. S., (2012). Metaheuristic optimization with applications: Demonstration via bat algorithm, in: Proceedings of 5th Bioinspired Optimization Methods and Their Applications (BIOMA2012) (Eds. B.Filipic and J. Silc), 24-25 May 2012, Bohini, Slovenia, pp. 23–34.

Yang, X. S., (2013a). Bat algorithm and cuckoo search: a tutorial, in: *Artificial Intelligence, Evolutionary Computing and Metaheuristics* (Eds. X. S. Yang), Studies in Computational Intelligence, Vol. 427, pp. 421–434.

Yang, X. S., (2013b). Multiobjective firefly algorithm for continuous optimization, *Engineering with Computers*, Vol. 29, No. 2, pp. 175–184.

Yang, X. S., Cui, Z. H., Xiao, R. B., Gandomi, A. H., Karamanoglu, M., (2013). *Swarm Intelligence and Bio-Inpsired Computation: Theory and Applications*, Elsevier, London, (2013).

Yang, X. S. and Gandomi, A. H., (2012). Bat algorithm: a novel approach for global engineering optimization, *Engineering Computations*, Vol. 29, No. 5, pp. 464–483.

Yang, X. S., Gandomi, A. H., Talatahari, S., Alavi, A. H., (2012a). *Metaheuristics in Water, Geotechnical and Transport Engineering*, Elsevier, London, UK and Waltham, USA.

Yang, X. S., Karamanoglu, M., Fong, S., (2012b). Bat aglorithm for topology optimization in microelectronic applications, in: IEEE Int. Conference on Future Generation Communication Technology (FGCT2012), British Computer Society, 12-14 Dec 2012, London, pp. 150–155.

Zhang, J. W., and Wang, G. G., (2012). Image matching using a bat algorithm with mutation, *Applied Mechanics and Materials* (Editted by Z. Y. Du and Bin Liu), Vol. 203, No. 1, pp. 88–93.