Hybridizing Bat Algorithm with Modified Pitch-Adjustment Operator for Numerical Optimization Problems

Waheed Ali H. M. Ghanem[1, 2, 3] and Aman Jantan[1]

[1]School of Computer Science, Universiti Sains Malaysia, Pulau Pinang, Malaysia
[2]Faculty of Education-Saber, University of Aden, Aden, Yemen
[3]Faculty of Engineering, University of Aden, Aden, Yemen

Email: waheed.ghanem@gmail.com
Email: aman@cs.usm.my

Abstract. This article introduces a new metaheuristic approach that is a hybrid of two known algorithms, for solving global optimization problems. The proposed algorithm is based on the Bat Algorithm (BA), which is inspired by the micro-bat echolocation phenomenon, and addresses the problems of local-optima trapping and low precision using an adjusted mutation operator from the Harmony Search (HS) algorithm. The proposed Hybrid Bat Harmony (HBH) algorithm attempts to balance the good exploitation process of BA with a fast exploration feature inspired by HS. The design of HBH is introduced and its performance is evaluated against fourteen of the standard benchmark functions, and compared to that of the standard BA and HS algorithms and to another recent hybrid algorithm (HS/BA). The obtained results show that the new HBH method is indeed a promising addition to the arsenal of metaheuristic algorithms and can outperform the original BA and HS algorithms.

Keywords: Bat algorithm; Harmony search algorithm; Global Optimization problem; Pitch adjustment operator.

1 Introduction

Innumerous problems in the real life involve a set of possible solutions, from which the one with the best quality is termed as the optimal solution, and the process of searching for such a solution is known as (mathematical) optimization. The quality of solutions is represented by the ability to maximize or minimize a certain function, called the objective function, while the pool of possible solutions that can satisfy the required objective is called the search space. One can traverse all possible solutions, examine the result of the objective function in each case, and select the best solution. However, many real problems are intractable using this exhaustive search strategy. In these problems, the search space expands exponentially with the input size, and exact optimization algorithms are impractical. The historical alternative in such situations is to resort to heuristics, similar to simple rules of thumb that humans would utilize in a search process. Heuristic algorithms implement such heuristics to explore the otherwise prohibitively large search space, but they do not guarantee finding the actual optimal solution, since not all

areas of the space are examined. However, a close solution to the optimal is returned, which is "good enough" for the problem at hand.
The next step would be to generalize those heuristics in higher level algorithmic frameworks that are problem independent, and that provide strategies to develop heuristic optimization algorithms. The latter are known as metaheuristics [1]. Early metaheuristics were based on the concept of evolution, where the best solutions among a set of candidate solutions are selected in successive iterations, and new solution are generated by applying genetic operators such as crossover and mutation to the parent solutions.
Similar to and including evolutionary algorithms, many metaheuristics were based on a metaphor, inspired by some physical or biological processes. Many recent metaheuristics mimic the biological swarms in performing their activities; in particular, the important tasks of foraging, preying and migration. Popular examples of developed metaheuristic algorithms in this category include Particle Swarm Optimization (PSO) [2], which is inspired by the movement of swarms of birds or fishes; Ant Colony Optimization (ACO) [3, 4], which is inspired by the foraging behavior of ants, where ants looking for food sources in parallel employ the concept of pheromone to indicate the quality of the found solutions; and Artificial Bee Colony (ABC) algorithm, inspired by the intelligent foraging behavior of honey bees [5, 6].
The idea of deriving metaheuristics from natural-based metaphors proved so appealing that much more of such algorithms have been, and continue to be developed. A few more examples include Cuckoo Search (CS) [7, 8], Biogeography-Based Optimization (BBO) [9], Animal Migration Optimization (AMO) [10], Chicken Swarm Optimization (CSO) [11], Grey Wolf Optimization (GWO) [12], Krill Herd (KH) [13], and Monarch Butterfly Optimization (MBO) [14]. The Bat Algorithm (BA) [15] also belongs to the metaheuristics that are based on animal behavior; inspired by the echolocation behavior of bats in nature. On the other hand, several metaphor-based metaheuristics are derived from physical phenomena such as Simulated Annealing (SA) [16] which is inspired by the annealing process of a crystalline solid. The Harmony Search (HS) algorithm [17] belongs to this category, and is inspired by the process of improvising musical harmonies by musicians in an orchestra.
The aforementioned metaheuristics are classified as stochastic optimization techniques. To avoid searching the whole solution space, they include a randomization component to explore new solution areas. Though these random operators are essential, they can introduce two types of problems. First, if the randomization is too strong, the metaheuristic algorithm might keep moving between candidate solutions, loosely examining each localized region and failing to exploit promising solutions and find the best solution. Second, if the search process is too localized, exploiting the first found good solutions very well but failing to explore more regions, the algorithm might indeed miss the real optimal solution (called the global optimum), and trap into some local optima.

The perfect balance between *exploitation* and *exploration* is essential to all metaheuristics. In fact, it is whether and how this balance is achieved that distinguishes most metaheuristics from each other, and forms a source of new attempts to improve existing algorithms, possibly by hybridizing ideas from more than one metaheuristic strategy. The work in this paper follows this path, focusing on two of the known metaheuristics: the BA and the HS algorithms. The most similar attempt in the literature is the hybrid metaheuristic method of Harmony Search/Bat Algorithm (HS/BA) [18]. HS/BA tries to improve the tendency of BA to trap into local optima by adding a pitch adjustment operation in HS serving as a mutation operator during the process of the bat exploration, in an attempt to increase its diversity [18].

Motivated by the mutation operator and by the HS/BA algorithms, we introduce in this work a new hybrid algorithm that improves the diversity of BA by adding a mutation operator. Unlike HS/BA we do not employ the same pitch adjustment as the original HS algorithm. Rather, we employ a custom operator that we consistently found superior to other mutations in our research, during numerous experiments with the algorithms. We name the resulting algorithm the Hybrid Bat Harmony (HBH) algorithm, and evaluate its performance compared to the original BA, HS, and HS/BA.

The rest of this article is organized as follows. Section 2 introduces the proposed HBH method, while Section 3 explains the setup of experimental evaluation. Section 4 presents and discusses the obtained results, and finally Section 5 concludes the paper.

2 The Hybrid Bat Harmony Algorithm

This section introduces the Hybrid Bat Harmony (HBH) algorithm, which is based on the standard BA [15] and HS [17] algorithms. The main idea behind the new algorithm is to augment the BA with a very effective operator from the HS algorithm. In particular, the principle of pitch adjustment in HS is further modified and fine-tuned to increase the diversity of BA and allow for more mutations in the BA search, in order to jump out of potential local-optima traps.

In the standard BA all bats use echolocation to sense distance, and they also ‘know’ the difference between prey and obstacles in some way. Bats fly randomly with velocity vi at position xi with a frequency fi, varying the wavelength λ and loudness A0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission r ∈ [0, 1], depending on the proximity of their target. It is also assumed that the loudness varies from a large (positive) A0 to a minimum constant value Amin. The critical aspect of metaheuristic search mechanisms in solving optimization problems is the correct balance between exploitation and exploration as discussed in the introduction. The standard BA controls the capability of the exploration process by Equations (1), (2) and (3), and the capability of the exploitation process by Equation (4):

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{1}$$

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

$$v_i^t = v_i^{t-1} + (x_i^t - x_*)f_i \tag{2}$$

Where x_* is the current global best location (solution), which is located after comparing all the solutions among all the bats.

$$x_i^t = x_i^{t-1} + x_i^t \tag{3}$$

Initially, each bat is randomly given a frequency which is drawn uniformly from [$f$min, $f$max]. For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk where $\varepsilon \in [-1, 1]$ is a scaling factor which is a random number, while $A^t = < A_i^t >$ is the average loudness of all the bats at time step t.

$$x_{new} = x_{old} + \varepsilon A^t \quad (4)$$

Furthermore, the loudness A_i and the rate r_i of pulse emission are updated as follows:

$$A_i^{t+1} = \alpha A_i^t, \qquad r_i^{t+1} = r_i^0[1 - exp(-\gamma t)] \quad (5)$$

Where α and γ are constants.

However, BA at times falls in the trap of local optima after a quick convergence into a promising area. The main improvement by adding an operator from HS algorithm is to introduce more mutation into the elements of the current solution, either drawing from features of a previous good solution or from a random distribution according to the value of a random parameter called the Memory Consideration Rate (HMCR), which is inspired from the HS algorithm. Further, depending on another parameter called the Pitch Adjustment Rate (PAR), the algorithm might pull the search back to better position with respect to the best and worst solutions recorded so far, which proves very useful in case the BA traps in a local optimum. The modification of the pitch adjustment operator is listed in Algorithm 1 and in Equations (6) and (7):

$$x_{new}(d) = x_{old}(d) + bw\times(x_{worst}(d) - x_{old}(d))\times(2\times rand - 1) \quad (6)$$

$$x_{new}(d) = x_{old}(d) + bw\times(x_{old}(d) - x_{best}(d))\times(2\times rand - 1) \quad (7)$$

In the equations above, x_{new} is a new bat solution, x_{old} is the current solution, while variables x_{worst} and x_{best} represent the worst and best solutions ever found, respectively. d refers to a single dimension of the solution (an element of the solution vector). The bw is the control parameter called bandwidth, which is an arbitrary distance bandwidth for each generation, and rand is a random uniform real number between [0, 1].

Algorithm 1

```
Modified Pitch Adjustment Operation (Mutation Operator)
For $d$ = 1: $D$
  If (rand < HMCR) then                    //memory consideration
    If (rand < PAR) then                   //pitch adjustment
      $x_{new}$(d)=$x_{old}$(d)+2bw($x_{worst}$(d)-$x_{old}$(d))*(rand-1)      // Eq(6)
    Else
      $x_{new}$(d)=$x_{old}$(d)+2bw($x_{old}$(d)-$x_{best}$(d))*(rand-1)       // Eq(7)
    End if
  Else
    $x_{new}$($d$)=$x_{min}$(d)+rand×($x_{max}$(d)−$x_{min}$(d))   //random selection
  End if
End for $d$
```

The introduced mutation maintains the attractive features of the original bat algorithm, especially in terms of fast convergence, while allowing the algorithm to make use of more mutation towards a better diversity. Based on the aforementioned analyses, the pseudocode of the HBH algorithm is shown in Algorithm 2.

Algorithm 2

```
Begin
Stage 1: Initialization stage
  Initialize the population of NP bats;
Set the generation counter $t$ = 1; define loudness $A_i$, frequency $f_i$ position $\mathbf{x}_i$
and the initial velocities $v_i$;set pulse rate $r_i$ ($i$ = 1, 2… NP);
  Set the parameters and initialize the HM, HMCR, PAR and bw;
  Evaluate the quality $f$ for each bat determined by the objective function ($x$);
```

```
Stage 2: Update stage
  While ($t$< Maximum Generation)
  for$i$ = 1: NP (all bats) do
  Generate a new solution by adjusting frequency, and updating velocity and
position by (3), (4), and (5);
  I f (rand >$r_i$)
    Select a solution among the best solutions;
    Generate a local solution around the selected best solution;
  End if
  Algorithm 1                              // mutation operator
  Generate a new solution by flying randomly
  If (rand <$A_i$& $x_i$<$f$ ($x_*$))
    Accept the new solution;
    Increase $r_i$&reduce $A_i$;
  End if
  Rank the bats and find the current best $x_*$;
  End for
  $t$ = $t$ + 1;
  End while
  Process the results and visualize them
End.
```

3 Experimental Evaluation

In this section, we layout the experimental setup through which we have evaluated the proposed algorithm, HBH.

3.1 General setup

Hardware and software implementation

All the experiments were conducted on a laptop with an Intel Core 5i processor running at 2.4 GHz, and 8 GB of RAM. The software implementation of the proposed HBH algorithm was based on the implementation of BA and HS in [15, 17] and the description of HS/BA in [18].All software is compiled using MATLAB R2009b (V7.9.0.529) running under Windows 7.

Compared algorithms

To put the performance of HBH in perspective and illustrate its merits among similar metaheuristic methods, we compare its performance with the three closest techniques: the original BA and HS algorithms, which are the basic components of the proposed method, and HS/BA, which is the most related previous work on hybridizing BA with HS. This set of comparisons is benchmarked using a group of 14 global optimization functions.

Parameters

Table 1 lists the set of parameters used in all experiments. The listed parameters include those for the compared methods for the sake of reproducibility.

Table 1. Set of used parameters in all experiments

Metaheuristic	Parameter	Symbol / Abbr.	Value
HBH	Loudness	A	0.95

BA	Pulse rate	*r*	0.6
HS	Harmony memory consideration rate	*HMCR*	0.95
HS/BA	Pitch adjustment rate	*PAR*	0.1
	Bandwidth	*bw*	0.9

In all cases, the population size NP was set to 50, function dimension was set to 20, 50 and 100 in three sets of experiments and the maximum number of generations was 50. To mitigate the impact of randomness in individual runs, we report the results over a 100 implementation runs for each algorithm on each benchmark function (Tables 2).

3.2 Benchmark Function

This paper uses a set of 14 test functions for global numerical optimization. These functions are listed in Table 2 alongside their respective equations and properties.

Table 2. Benchmark global numerical functions used for evaluating optimization methods

No.	*Equation*	*Low*	*Up*	*Opt*
F01	$f(x)=\sum_{i=1}^{n}\lfloor\lvert x_i\rvert\rfloor$	-100	100	0
F02	$f(x)=\sum_{i=1}^{n} ix_i^4+rand[0,1)$	-1.28	1.28	0
F03	$f(x)=\sum_{i=1}^{n}[x_i^2-10\cos 2\pi x_i+10]$	-5.12	5.12	0
F04	$f(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	-600	600	0
F05	$f(X)=\frac{\pi}{n}\times\{10sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2\,[1+10sin^2(\pi y_{i+1})]+(y_n-1)^2\} + \sum_{i=1}^{n} u(x_i,10,100,4)$	-50	50	0
F06	$f(x)=\sum_{i=1}^{n}(x_i-1)^2-\sum_{i=2}^{n} x_i x_i-1$	-100	100	0
F07	$f(x)=-\sum_{i=1}^{n}\sin(x_i)sin^{2m}(ix_i^2/\pi)$	0	3.1416	-9.7
F08	$f(x)=\sum_{i=1}^{n} x_i^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^2+\left(\sum_{i=1}^{n}0.5ix_i\right)^4$	-5	10	0
F09	$f(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{i+1}$	-1	1	0
F10	$f(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2-48x_2+36x_1x_2+27x_2^2)]$	-2	2	3
F11	$f(x)=-\cos(x_1)\cos(x_2)\,exp(-(x_1-\pi)^2--(x_2-\pi)^2)$	-100	100	-1
F12	$f(x)=0.5+sin^2(x_1^2-x_2^2)-0.5/[1+0.001(x_1^2+x_2^2)]^2$	-100	100	0
F13	$f(x)=-\sum_{i=1}^{m}\left(\sum_{j=1}^{4}(x_j+c_{ji})^2+\beta_i\right)^{-1}$	0	10	-10.5
F14	$f(x)=-1/1.94\left[2.58+\sum_{i=1}^{4}\alpha_i\,exp\left(-\sum_{j=1}^{6}A_{ij}\,(x_j-P_{ij})^2\right)\right]$	0	1	-3.3

4 Results

Table 3 lists the optimization results when applying the 14 optimization test functions to HS, BA, HS/BA and our HBH methods. The listed values are the optimal value of the objective function achieved by each algorithm after iterating 50 generations. The Mean values in the table are averaged over 100 runs (each run constitutes 50 iterations) and listed along the standard deviation. The Min values, however, are the best results achieved by each algorithm at all. By the "best result" we mean the closest result to the actual optimal value of the function, as per Table 2.
It is evident from Table 3 that the HBH algorithm can reach a better optimum on average; at least with respect to the set of benchmark functions used in the experiments (HBH has better average results in the case of 12 out of 14 test functions). For ease of recognition, the best average result is marked with bold font and bordered cell. The Min values are shaded in grey to identify the absolute best minimum achieved for each function. Note that this value is meaningful because it happened that the minimum achieved values by the algorithms for the selected benchmark functions are closest to the real optimum. With respect to the set of test functions used in our evaluations, HBH could achieve the best result in 12 out of 14 cases.
On another perspective, we also graphed the optimization process of each algorithm (for each benchmark function) as the value of the so-far best solution versus the current iteration. That is, to show the search path in terms of selected best solution per iteration. The curve of this kind is expected to decline overall at a slope that reflects the convergence speed of the algorithm (there is no degradation during the process of any included metaheuristic algorithm, as the best solution is either improved or kept unchanged at all iterations). Therefore, these graphs can be called the convergence plots of the algorithms. Because of the large number of plots, we include hereby a representative samples of the convergence plots in Figure 1, which compares the convergence of HBH with the three most related metaheuristic techniques: BA, HS and HS/BA.

Table 3. The min, mean and standard deviation of test function values found by HS, BA, HS/BA and the proposed HBH algorithms, averaged over 100 experimental run. The best mean for each function is marked in bold font and bordered. The MIN value is the best optimization result found by each algorithm (closest value to the global optimum over all runs), and is shaded in grey. Functions are set with 20, 50 and 100 dimensions.

Function	Dim	HS			BAT			HS/BA			HBH		
		Min	Mean	Std. dev	Min	Mean	Std. dev	Min	Mean	Std. dev	Min	Mean	Std. dev
F1	20	9.9E+02	2.1E+03	5.7E+02	1.3E+04	3.8E+04	6.1E+03	4.1E-16	7.1E-01	7.0E+00	7.5E-23	2.5E+01	7.6E+01
	50	1.5E+04	2.1E+04	2.6E+03	3.3E+04	1.2E+05	1.3E+04	4.1E-16	7.1E-01	7.0E+00	0.0E+00	2.0E-01	9.3E-01
	100	6.6E+04	8.4E+04	6.8E+03	5.2E+04	2.7E+05	2.5E+04	1.9E-09	3.7E+01	2.9E+02	0.0E+00	6.0E-01	3.6E+00
F2	20	6.5E+00	8.3E+00	6.4E-01	1.8E+01	4.5E+01	9.2E+00	4.4E+00	5.9E+00	4.8E-01	4.7E+00	5.9E+00	4.3E-01
	50	3.6E+01	5.0E+01	5.4E+00	5.9E+01	3.9E+02	6.2E+01	4.4E+00	5.9E+00	4.8E-01	1.6E+01	1.8E+01	6.2E-01
	100	2.4E+02	3.3E+02	3.9E+01	2.0E+02	1.8E+03	2.3E+02	3.8E+01	4.0E+01	9.4E-01	3.6E+01	4.0E+01	1.0E+00
F3	20	3.0E+01	4.6E+01	8.0E+00	1.2E+02	1.7E+02	2.1E+01	6.3E-13	1.1E+01	1.2E+01	0.0E+00	7.4E+00	1.2E+01
	50	1.7E+02	2.6E+02	2.4E+01	4.4E+02	5.9E+02	3.6E+01	6.3E-13	1.1E+01	1.2E+01	0.0E+00	9.8E-01	6.3E+00
	100	1.7E+02	2.6E+02	2.4E+01	4.4E+02	5.9E+02	3.6E+01	6.3E-13	1.1E+01	1.2E+01	0.0E+00	9.8E-01	6.3E+00
F4	20	1.1E+01	2.0E+01	4.7E+00	1.1E+02	3.5E+02	5.1E+01	1.0E-14	3.9E+00	8.5E+00	0.0E+00	6.0E-01	8.1E-01
	50	1.5E+02	1.9E+02	2.1E+01	2.6E+02	1.1E+03	1.1E+02	1.0E-14	3.9E+00	8.5E+00	0.0E+00	1.3E-01	6.2E-01
	100	5.8E+02	7.5E+02	6.2E+01	6.2E+02	2.4E+03	2.2E+02	1.7E-10	2.9E+00	6.2E+00	0.0E+00	6.1E-02	3.1E-01

F5	20	4.0E+03	2.3E+05	2.3E+05	1.3E+07	2.6E+08	8.6E+07	1.5E-15	1.7E+00	3.3E+00	3.0E-23	6.8E-01	2.1E+00
	50	1.1E+07	3.4E+07	1.0E+07	9.4E+06	1.0E+09	2.1E+08	1.5E-15	1.7E+00	3.3E+00	9.4E-33	2.4E-03	2.0E-02
	100	1.7E+08	3.0E+08	5.8E+07	1.6E+08	2.7E+09	3.7E+08	3.1E-11	3.1E-01	9.4E-01	4.7E-33	7.3E-04	3.6E-03
F6	20	6.2E+02	2.8E+03	9.7E+02	8.8E+03	3.0E+04	5.7E+03	-3.9E+02	-3.8E+02	3.5E+01	-4.5E+02	-3.9E+02	1.9E+01
	50	1.8E+04	2.8E+04	4.3E+03	3.2E+04	1.0E+05	1.3E+04	-3.9E+02	-3.8E+02	3.5E+01	-2.6E+03	-2.5E+03	2.5E+01
	100	8.1E+04	1.0E+05	9.6E+03	6.5E+04	2.4E+05	2.4E+04	-9.9E+03	-9.9E+03	2.3E+01	-1.0E+04	-9.9E+03	1.5E+02
F7	20	-1.8E+01	-1.6E+01	5.4E-01	-1.2E+01	-8.6E+00	1.2E+00	-1.1E+01	-8.3E+00	7.4E-01	-1.4E+01	-1.0E+01	1.6E+00
	50	-3.5E+01	-3.1E+01	1.5E+00	-2.3E+01	-1.8E+01	2.2E+00	-1.1E+01	-8.3E+00	7.4E-01	-2.0E+01	-1.7E+01	2.2E+00
	100	-5.4E+01	-4.8E+01	2.1E+00	-3.8E+01	-3.0E+01	2.7E+00	-4.0E+01	-2.9E+01	7.4E+00	-4.0E+01	-2.8E+01	4.3E+00
F8	20	8.9E+01	1.9E+02	4.3E+01	1.5E+02	4.8E+06	2.2E+07	5.0E-12	4.9E+00	3.5E+01	1.6E-19	2.9E+00	2.4E+01
	50	5.1E+02	7.2E+02	1.1E+02	9.8E+02	1.3E+11	1.5E+11	5.0E-12	4.9E+00	3.5E+01	4.4E-38	4.5E-02	3.2E-01
	100	1.3E+03	2.5E+03	5.2E+03	1.7E+04	1.4E+14	8.0E+13	9.3E-03	2.2E+05	1.1E+06	4.0E-32	2.8E+02	2.7E+03
F9	20	2.2E-05	1.3E-03	1.7E-03	6.9E-06	1.8E-02	1.7E-02	1.2E-19	1.6E-11	3.5E-11	1.6E-24	7.1E-12	3.7E-11
	50	5.4E-04	5.1E-03	3.9E-03	7.3E-06	6.6E-02	5.8E-02	1.2E-19	1.6E-11	3.5E-11	3.3E-48	9.7E-07	8.3E-06
	100	1.8E-03	8.6E-03	5.5E-03	2.2E-05	1.1E-01	8.6E-02	6.1E-10	1.8E-05	3.2E-05	7.4E-46	1.8E-07	1.8E-06
F10	2	3.0E+00	1.1E+01	1.3E+01	3.0E+00	1.2E+01	1.8E+01	3.0E+00	9.0E+00	1.1E+01	3.0E+00	8.7E+00	1.5E+01
F11	2	-9.9E-01	-9.5E-02	2.3E-01	-1.4E-05	-1.5E-07	1.4E-06	-1.0E+00	-3.0E-01	4.4E-01	-1.0E+00	-3.7E-01	4.7E-01
F12	2	1.7E-04	8.2E-02	6.4E-02	1.5E-04	2.1E-01	1.1E-01	6.4E-15	7.1E-02	9.4E-02	4.4E-15	4.5E-02	6.8E-02
F13	4	-9.3E+00	-3.7E+00	1.4E+00	-7.7E+00	-1.8E+00	1.1E+00	-1.1E+01	-6.9E+00	3.0E+00	-1.1E+01	-7.6E+00	3.4E+00
F14	6	-3.0E+00	-3.0E+00	2.3E-02	-3.0E+00	-3.0E+00	4.5E-02	-3.0E+00	-3.0E+00	5.9E-02	-3.0E+00	-3.0E+00	3.9E-02

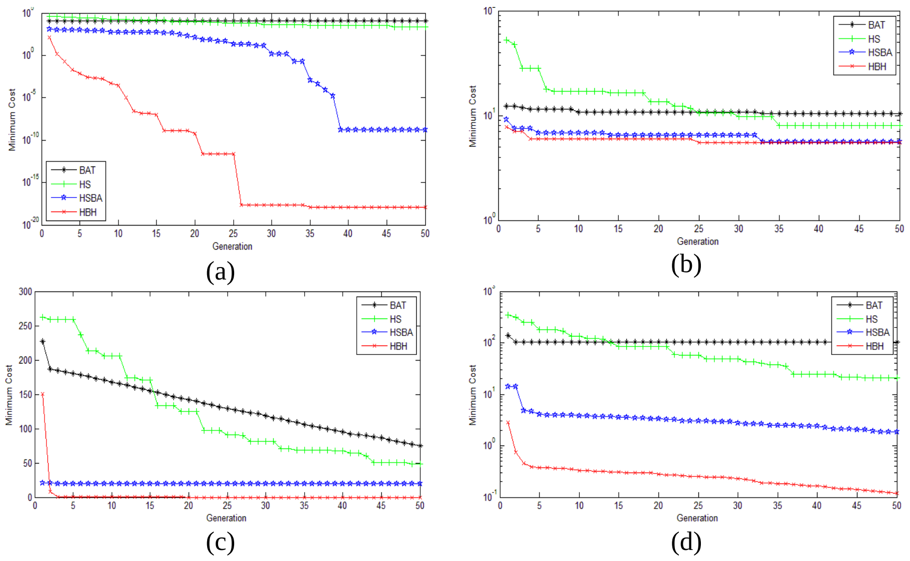

Fig.1. Performance of BA, HS, HS/BA and HBH algorithms for (a) F01 STEP, (b) F02 QUARTIC, (c) F03 RASTRIGIN and (d) F04 GRIEWANK benchmark functions.

Figures 1 (a-d) show that the HBH algorithm enjoys not only a superior overall performance in terms of the quality of the found optimal solution, but also a faster convergence especially in the earlier stages. Although the starting points of the algorithms are close to each other in the plots of the four testing functions in the figure, the proposed HBH method does not trap into a quick local optimum, unlike the original BA and HS algorithms for example.

5 Conclusion

This paper presented a new metaheuristic, combining the original Bat Algorithm with a mutation operator to increase its diversity. The introduced operator resembles the pitch adjustment operator from the Harmony Search metaheuristic but is modified so as to allow for a larger mutation rate while preserving the strength of BA in swift and efficient exploitation. Experimental evaluations against a set of 14 benchmark numerical optimization functions showed that the proposed HBH algorithm converges faster than other metaheuristics and achieve better or at least competitive performance in most cases. We believe that these results are very promising and hope to further explore the specific applications that will benefit from the merits of HBH the most, which is left for future experimentation.

Acknowledgments

This research was funded by Universiti Sains Malaysia under USM Fellowship 2016 [APEX (1002/CIPS/ ATSG4001)], Also partially supported by the Fundamental Rsearch Grant Scheme (FRGS) for "Content Based Analysis Framework for Better Email Forensic and Cyber Investigation" [203/PKOMP/6711426].

REFERENCE

1. Sörensen, K. and F.W. Glover, *Metaheuristics*, in *Encyclopedia of Operations Research and Management Science*. 2013, Springer. p. 960-970.
2. Eberhart, R.C. and J. Kennedy. *A new optimizer using particle swarm theory*. in *Proceedings of the sixth international symposium on micro machine and human science*. 1995. New York, NY.
3. Dorigo, M., M. Birattari, and T. Stützle, *Ant colony optimization.* Computational Intelligence Magazine, IEEE, 2006. 1(4): p. 28-39.
4. Dorigo, M., V. Maniezzo, and A. Colorni, *Ant system: optimization by a colony of cooperating agents.* Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 1996. 26(1): p. 29-41.
5. Karaboga, D., *An idea based on honey bee swarm for numerical optimization*. 2005, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.
6. Karaboga, D. and B. Basturk, *A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm.* Journal of global optimization, 2007. 39(3): p. 459-471.
7. Yang, X.-S., *Nature-inspired metaheuristic algorithms*. 2010: Luniver press.
8. Yang, X.-S. and S. Deb. *Cuckoo search via Lévy flights*. in *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. 2009. IEEE.
9. Simon, D., *Biogeography-based optimization.* Evolutionary Computation, IEEE Transactions on, 2008. 12(6): p. 702-713.
10. Li, X., J. Zhang, and M. Yin, *Animal migration optimization: an optimization algorithm inspired by animal migration behavior.* Neural Computing and Applications, 2014. 24(7-8): p. 1867-1877.
11. Meng, X., et al., *A new bio-inspired algorithm: chicken swarm optimization*, in *Advances in swarm intelligence*. 2014, Springer. p. 86-94.
12. Mirjalili, S., S.M. Mirjalili, and A. Lewis, *Grey wolf optimizer.* Advances in Engineering Software, 2014. 69: p. 46-61.
13. Gandomi, A.H. and A.H. Alavi, *Krill herd: a new bio-inspired optimization algorithm.* Communications in Nonlinear Science and Numerical Simulation, 2012. 17(12): p. 4831-4845.
14. Wang, G.-G., S. Deb, and Z. Cui, *Monarch butterfly optimization.* Neural Computing and Applications, 2015: p. 1-20.

15. Yang, X.-S., *A new metaheuristic bat-inspired algorithm*, in *Nature inspired cooperative strategies for optimization (NICSO 2010)*. 2010, Springer. p. 65-74.
16. Kirkpatrick, S. and M.P. Vecchi, *Optimization by simmulated annealing.* science, 1983. 220(4598): p. 671-680.
17. Geem, Z.W., J.H. Kim, and G. Loganathan, *A new heuristic optimization algorithm: harmony search.* Simulation, 2001. 76(2): p. 60-68.
18. Wang, G. and L. Guo, *A novel hybrid bat algorithm with harmony search for global numerical optimization.* Journal of Applied Mathematics, 2013.