Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 9(12): 35655–35677.
DOI:10.3934/math.20241692
Received: 19 September 2024
Revised: 11 November 2024
Accepted: 02 December 2024
Published: 23 December 2024

Research article

Improving diversification by a hybrid bat-Nelder-Mead algorithm and DDE for rapid convergence to solve global optimization

Enas Suhail[1,3], **Mahmoud El-Alem**[1], **Omar Bazighifan**[2,4,*] **and Ahmed Zekri**[1]

[1] Department of Mathematics and Computer Science, Faculty of Science, Alexandria University, Alexandria, Egypt
[2] Department of Mathematics, Faculty of Education, Seiyun University, Hadhramout, Yemen
[3] Department of Mathematics, Faculty of Science, Hadhramout University, Hadhramout, Yemen
[4] Jadara Research Center, Jadara University, Irbid 21110, Jordan

* **Correspondence:** Email: o.bazighifan@gmail.com, enas.omar_ PG@alexu.edu.eg.

Abstract: Delay differential equations and algorithms hold a crucial position in the exploration of some biological systems and several models in real-world applications. So, some algorithms contribute to improve mathematical models related to natural life problems and global optimization. A novel hybridization between the downhill Nelder-Mead simplex algorithm (NM) and the classic bat algorithm (BA) was presented. The classic BA suffers from premature convergence, which is due to its global search weakness. In this research, this weakness was overcome by the intervention of NM in the velocity updating formula of the particles as an additional term. This improvement distracts particles from the rapporteur route, toward only the best solution found, to discover the search space more accurately. Once this improvement detects a promising area, sequential expansions are performed to deeply explore the area. This mechanism provides rapid convergence for the algorithm. Deep analysis of the algorithm's behaviour was provided, and thoughtful experiments were conducted and evaluated utilizing several evaluation metrics together with the Wilcoxon signed rank test to accentuate the effectiveness and efficiency of the proposed algorithm.

Keywords: non-linear optimization; bat algorithm; Nelder-Mead algorithm; metaheuristics; numerical comparison
Mathematics Subject Classification: 65K10

1. Introduction

The discovery of several real-world applications is crucially related to delay differential equations and algorithms. Different algorithms are designed to solve global optimization problems. Global optimization problems are inherently intractable. Conventional algorithms (deterministic algorithms) are not efficient to solve global problems. One of the main reasons for this is that deterministic algorithms seek the exact global optimum among all feasible regions, which is a time-consuming procedure for problems of moderate to high dimension. Ideal surrogate for deterministic algorithms in solving these problems are algorithms with a stochastic element, called heuristic algorithms, as the randomness in the global search guides the search toward regions where a better solution is likely to be found. Metaheuristic algorithms are a type of heuristic applicable for a wide range of problems. In the past few decades several nature-inspired metaheuristics have been developed. One class of them is based on the social interactions of animal communities, called swarm intelligence (SI) algorithms, such as the genetic algorithm [1], particle swarm optimization (PSO) algorithm [2], ant colony optimization (ACO) [3], artificial fish swarm algorithm (AFSA) [4], and bat algorithm (BA) [5]. Recently, several other metaheuristic algorithms were proposed such as the liver cancer algorithm (LCA) [6], fata morgana algorithm (FATA) [7], snow geese algorithm (SGA) [8], and more.

Over time, diligent research has shown some defects that affect the performance of standard metaheuristics in solving certain problems. Accordingly, numerous impressive algorithms were reported that combine metaheuristics with different techniques or other algorithms to improve new versions that overcome the deficiencies in the component algorithms. In [9], the author combined the firefly algorithm (FA) with extremal optimization to tackle the defects of FA in slow convergence and falling into local minimum. In [10], the author built a framework for conducting an effective analysis for recurrent spontaneous abortion. This framework was presented by a combination between the joint self-adaptive sime mould algorithm (JASMA) with the common kernel learning support vector machine with maximum-margin hyperplane theory. BA was not excluded from this cutting-edge phenomenon. The BA has some advantages over other algorithms that make it promising and interesting such as its simplicity in structure, speed in processing, and ease in hybridization with other algorithms. Furthermore, the BA efficiently succeeded in dealing with several optimization problems in different fields such as power systems, economic load dispatch problems, image processing, and medical applications, see [11] for a detailed description. However, the exploration in the BA is weak as the BA focuses in the exploitation rather than exploration which makes it fall into local optimum. For that, many works proposed variants of the BA to improve its performance by introducing different techniques. For instance, the researcher in [12] incorporated a chaotic sequence and chaotic Levy flight schemes to enrich the searching behavior and improve the local and global search capabilities. In [13], the author introduced, first, the iterative local search in the local search of the BA to avoid falling into local minimum and, second, a stochastic inertia weight in the velocity updating formula of the BA. The researcher [14] incorporated quantum evolution and an annealing strategy into the BA to achieve a balance between intensification and diversification. Other works tend to combine the BA with other algorithms in which the combination is referred to as hybrid metaheuristics. This hybridization accentuates the mettle of its components and creates an algorithm capable of solving many hard optimization problems. For instance, [15] introduced a step in the harmony search (HS) algorithm in the BA and added an HS attribute as an operator. Author in [16] introduced a hybrid strategy that combines the genetic algorithm (GA) with BA to improve the efficiency of the global search of the standard BA. In their work, the mean magnitude of relative error was applied as an objective function. Other variants of the BA were proposed by combining the benefits of the BA with the local search Nelder-Mead algorithm (NM). In [17], the author started by following the BA routine to generate a new set of solutions. If any inferior solution was obtained, NM was performed by taking the inferior solution as the starting point of the simplex to refine it. While, in [18], the researcher took the best solution obtained after performing the BA routine as the starting point of the simplex in the NM algorithm. In [19], the NM method was used as a local search instead of the random walk method that is applied in the standard BA to refine the best solution found so far, which helped to accelerate the search process. In [20], the author replaced the random walk that is used as a local search in the standard BA by the pattern search method to increase the intensification ability. Furthermore, the NM method was employed in the final stage of the algorithm to improve the best found solution.

The motivation of this work is to tackle the main drawback of the BA, which is the weakness in exploration, by making changes in its structure, specifically in the updating process, using the NM algorithm rather than the technique used in [17–19], which is refining the solution found by the standard BA utilizing the NM method. For that, a novel hybridization is carried out utilizing the BA and NMA at which the global search ability of the standard BA is enhanced in the proposed algorithm (HBNMA) in premier iterations by the intervention of the NMA in the updating process of the solutions. This helped at first to distract solutions from heading only toward best solution found so far to avoid falling into local minimum. Second, it helped to penetrate deeply in a region where further potential best solutions may be found using sequential expansion steps that are defined in the NM algorithm. This can be seen in the Experiments and results section. By the lapse of iterations, the exploitation arises by the usual local search mechanism of the classic BA. This balance between exploration and exploitation enables HBNMA to solve complex global optimization problems. The contributions of this paper are as follows:

- The fast hybrid bat-NM algorithm is proposed with a new contribution of the NM in a hybridization. The NM in this work is used to tackle the main drawback of the BA by improving the updating process of the BA. The proposed algorithm is able to solve complex and high-dimensional global problems.
- Breaking the tradition of following a uniform route for all particles, in the proposed algorithm, particles determine their next step by choosing whether to follow the proposed hybrid updating process or the classic one of the standard BA according to which one is able to achieve better results.
- Precise analysis of the working mechanism of the proposed algorithm and thoughtful experiments are presented with statistical measurements to perfectly evaluate the merit of the HBNMA.

The rest of this paper is organized as follows: Section 2 gives an overview on the classic NM and classic BA. Furthermore Section 2 describes the proposed algorithm. In Section 3, experiments and numerical results are illustrated. The time complexity is described in Section 4. In Section 5, the study's conclusion is presented. Finally, future work is described in Section 6.

2. Materials and methods

2.1. Classic Nelder-Mead algorithm (NM)

In this section, the classic NMA is presented as stated in the original version [21]. The main thrust of the NM is to solve a multidimensional, unconstrained optimization with no need for derivatives.

$$\min f(x), \tag{2.1}$$

where the objective function $f : R^d \to R$ and $x \in R^d$.

This algorithm constructs a design framework called simplex with $d+1$ vertices for a d-dimensional problem, in which the worst vertex is replaced by a better one at each iteration. The surrogate vertex is obtained utilizing one or more procedures including reflection, expansion, contraction, or reduction.

The NM starts by initializing the points x_i (the vertices of the simplex). Next, the fitness value (objective function) f is evaluated for each point x_i. The points are reordered according to their objective function value to determine the best x_b, the second worst x_{w2}, and the worst x_w, which satisfy:

$$f(x_b) < ... < f(x_{w2}) < f(x_w).$$

After that, the centroid x_c of all points except x_w is calculated using:

$$x_c = \frac{1}{d} \sum_{i=1}^{d} x_i. \tag{2.2}$$

The NMA implements different procedures to decide the most effective step in a direction expressed utilizing a line segment joining x_c with x_w. The procedures are described in the following in which the values of the hyperparameters $\mu_r, \mu_e, \mu_{oc}, \mu_{ic}$, and μ_{re} that are used in these procedures are assigned to the same values as in [21].

(1) ***Reflection***:
This step indicates a reflection of the point x_w using the point x_c. The reflection point x_r is calculated utilizing the following equation:

$$x_r = x_c + (x_c - x_w). \tag{2.3}$$

(2) ***Expansion***:
This step expands the reflection step as long as an improvement is achieved. The expansion point x_e is calculated utilizing:

$$x_e = x_c + 2(x_c - x_w). \tag{2.4}$$

(3) ***Contraction***:
This step shrinks the size of the simplex utilizing either an outside contraction x_{oc} or inside contraction x_{ic}. The points x_{oc} and x_{ic} are calculated as follows:

$$x_{oc} = x_c + 0.5(x_c - x_w), \tag{2.5}$$

$$x_{ic} = x_c - 0.5(x_c - x_w). \tag{2.6}$$

(4) ***Reduction***:
This is the last resort after the failure of all previous attempts. In this step, all points x_i are updated toward the best point x_b using:

$$x_i = x_b + 0.5(x_i - x_b). \tag{2.7}$$

2.2. Classic bat algorithm (BA)

The bat algorithm emulates the echolocation trait and hunting behavior of microbats. [5] designed the classic BA from the following idealized rules:

(1) Different bats utilize echolocation to sense the distance between them and the target with a magical ability to distinguish between food and background barriers.
(2) Bats fly randomly with velocity v_i at position x_i with a fixed frequency f_m and a varied wavelength λ_i and loudness A_i to seek out prey. They can adjust automatically their wavelength (or frequency) according to the proximity of their target.
(3) Loudness is assumed to vary from a large value A_0 to a minimum value A_{min}.

According to these rules, the BA is designed as follows:

(1) Initialize randomly a population of n solutions $x_i \in R^d$.
(2) Update the frequency f_i and the velocity v_i using:

$$f_i = f_{min} + rand \quad (f_{max} - f_{min}), \tag{2.8}$$

$$v_i^t = v_i^{t-1} + (x_i^{t-1} - x_*)f_i, \tag{2.9}$$

where f_{min} and f_{max} are prespecified minimum and maximum values for f, $rand$ is a random number, t is the current iteration, and x_* is the current global solution. Then a new solution x_i is generated as follows:

$$x_i^t = x_i^{t-1} + v_i^t. \tag{2.10}$$

(3) If $rand < r_i$, evaluate a new local solution around the current global one using:

$$x_{new} = x_i + \epsilon A^t, \tag{2.11}$$

where $\epsilon \in [-1, 1]$, and A^t is the average loudness of all solutions at iteration t.
(4) Update the best individual solutions and both the loudness and rate of pulse emissions using the following formulae if an improvement is achieved:

$$A_i^t = \alpha A_i^{t-1}, \tag{2.12}$$

$$r_i^t = r_i^0(1 - exp(-\gamma t)), \tag{2.13}$$

where $\alpha \in (0, 1)$, $\gamma > 0$, $A_i^t \to 0$, and $r_i^t \to r_i^0$, as $t \to \infty$.
(5) Select the current global solution from the best individual solutions.

2.3. Proposed HBNMA

A novel hybridization strategy between the BA and NMA is introduced in this section called the HBNMA. The standard BA guides particles toward the best solution whereas the standard NM guides the worst particle (vertex) toward the centroid, or the center of gravity, of the other particles. The HBNMA is based on the concept that a solution at the beginning of the iterations should experience a different direction besides the direction toward the best solution in order to avoid falling into local minimum. Accordingly, the HBNMA developed a new updating formula for the velocity of a particle composed of the usual direction of the BA and the direction of the NM in order to enhance the global search ability of the proposed HBNMA.

The proposed algorithm starts by initializing n solutions x_i randomly using:

$$x_i = x_{min} + rand \quad (x_{max} - x_{min}), \tag{2.14}$$

where $[x_{min}, x_{max}]$ is the range of the search space, $i = 1, 2, ..., n$, and $rand \in (0, 1)$ is a random number. Further, the velocity, frequency, loudness, and pulse emission rate are initialized. Utilizing the solution's initial information, the HBNMA evaluates the fitness value. The solution with the best fitness is stored as a current global solution x_*.

Each particle has two choices from which its step can be determined. The first is a combination between the classic BA step v_{bat} (2.9) and the reflection step v_r of the NM:

$$v_r = x_c + (x_c - x_i), \tag{2.15}$$

where x_c is defined in (2.2). Figure 1 shows this combination geometrically in green-colored lines or more detailed in blue-colored lines that generate the step from v^t to v^{t+1}. Hence the improved updating formula of each particle's velocity v_i becomes:

$$v_i = v_{bat} + v_r. \tag{2.16}$$

If progress is achieved in the fitness value for x_i, that is, $f(x_i^{t+1}) < f(x_i^t)$, sequential expansion steps are carried out as long as progress is made to exploit the area deeply. The expansion steps are performed by replacing v_r in (2.16) by v_e which is calculated as:

$$v_e = x_c + \mu(x_c - x_i), \tag{2.17}$$

with $\mu = 2 * \mu_{pr}$, where μ_{pr} is the value of μ in the previous expansion step. The expansion steps are of the form:

$$v_i = v_{bat} + v_e. \tag{2.18}$$

The term v_{bat} solely guides particles toward the current global solution as in the classic BA, as shown in Figure 1 with red-colored lines, which may make particles fall easily in a local optimum. Combining the terms v_r or v_e with v_{bat} in (2.16–2.18), shown graphically in Figure 1, distracts the particle's concentration from the exclusive dependence on the direction toward the best one, especially at the beginning of the iterations. This helps to target new areas in the search space which may be promising. Accordingly, it allows one to effectively explore the region. This choice of the particle's step is accepted if progress in the fitness value is achieved. If not, the second choice is to keep the same updating process as in the standard BA (2.9). The proposed mechanism takes place at the beginning of the iterations. While in the later iterations, as is usually done in the standard BA, if $rand < r_i$, solutions are updated locally around the current global solution using (2.11). This local intensification occurs in the area where the global optimal is expected.

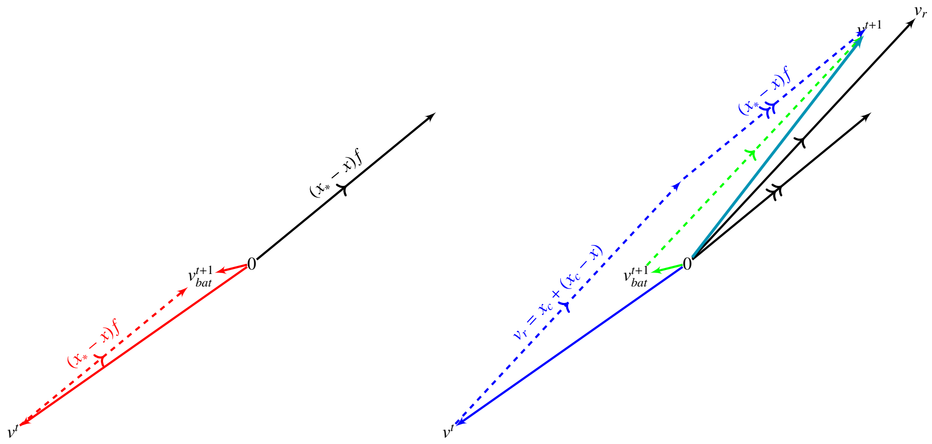

Figure 1. Clarification of the direction v_{bat}^{t+1} and the proposed direction v^{t+1} .

The full description of the HBNMA is shown in the following algorithm and flowchart (Figure 2):

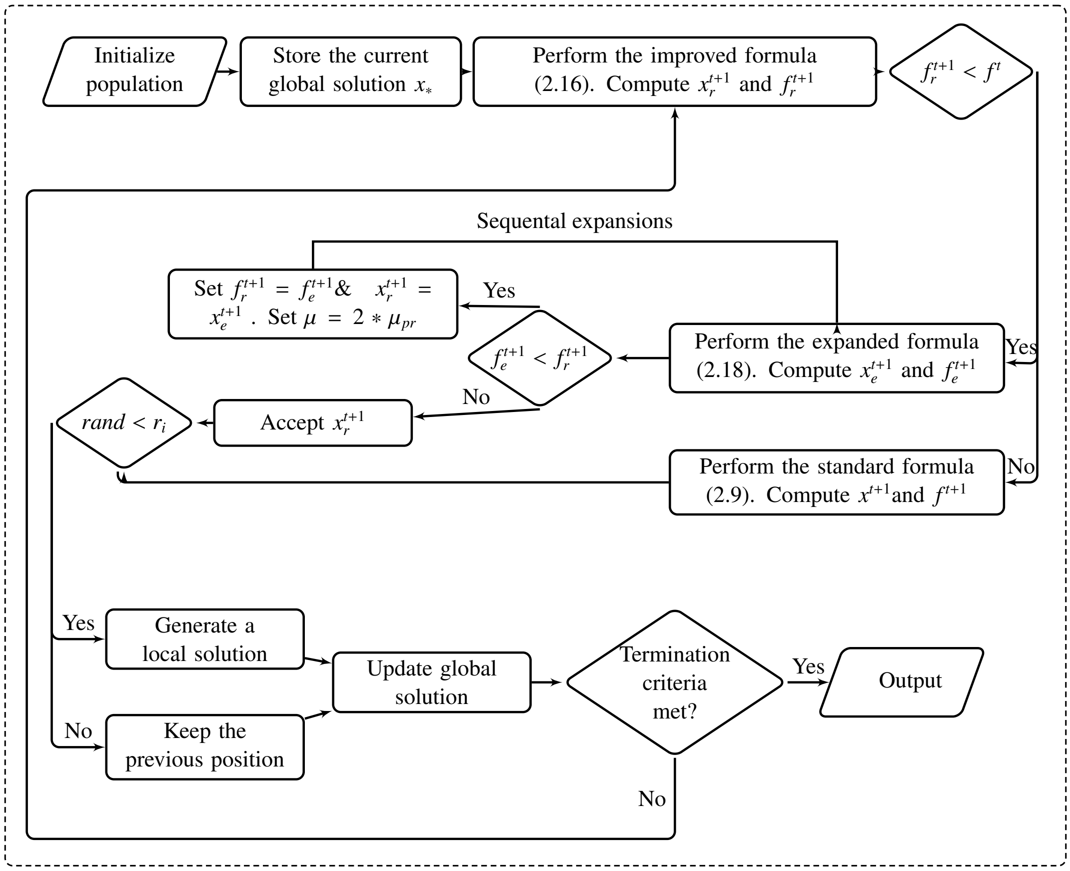

Figure 2. Flowchart of the proposed HBNMA.

Algorithm 1: HBNMA

Input: Initial population of solutions x_i;
The prespecified parameters
Output: The global optimal solution x_*
Evaluate the fitness value f of each solution;
Store the current global solution x_*^t
while *stopping criterion is not met* **do**
    for *each bat individual i* **do**
        Update the frequency;
        Update the velocity using (2.16);
        Calculate the fitness $(f_i^{t+1})_r$;
        if $(f_i^{t+1})_r < f_i^t$ **then**
            Expansion step is performed using (2.18);
            Calculate the fitness after expansion $(f_i^{t+1})_e$;
            if $(f_i^{t+1})_e < (f_i^{t+1})_r$ **then**
                Set $f_i^{t+1} = (f_i^{t+1})_r$;
                while $(f_i^{t+1})_e < f_i^{t+1}$ **do**
                    Set $f_i^{t+1} = (f_i^{t+1})_e$;
                    Perform another expansion step with $\mu = 2 * \mu_{pr}$
                The expansion step is accepted
            else
                The reflection step is accepted
        else
            Update the velocity using (2.9)
        if *rand* $< r_i$ **then**
            Generate a local solution around the global solution using equation (2.11)
        Evaluate the fitness value;
        if *rand* $> A_i$ & $f(x_i^{t+1}) < f(x_*^t)$ **then**
            Update the best individual solution;
            Update the loudness A_i and the rate of pulse emission r_i using equations (2.12) and (2.13)
    Update the global solution.

3. Experiments and results

In this section, we validate the efficiency of the proposed HBNMA and its counterparts using classical and complex benchmark test functions along with the Wilcoxon signed rank test. Furthermore, the working mechanism of the HBNMA is analyzed numerically.

3.1. Benchmark test functions

The performance of the HBNMA is examined utilizing two sets of test functions which provide a good representation of real-world optimization problems and show the merit of the proposed algorithm. The first set is the classical test functions whose definitions are found in [22]. The selected functions vary in their characteristics. Some are unimodal to test the intensification ability of the proposed algorithm while the others are multimodal functions to test the ability to jump out of the local optimum. Moreover, the selected test functions vary in separability that measures the difficulty of the problem. The dimensions of the selected functions, the range used, their optimum values, and their characteristics are described in the table found in the Supplementary Materials. The second set is the IEEE CEC2014 special session and competition on a single-objective real-parameter, numerical optimization benchmarking suit [23]. This set of complex test functions contains 3 rotated unimodal functions F1–F3, 13 rotated and shifted multimodal functions F4–F16, 6 hybrid functions F17–F23, and 8 composition functions F24–F30 that validate the performance of the proposed algorithm.

3.2. Statistical analysis

The significance of the proposed HBNMA is measured using some evaluation metrics and the Wilcoxon signed-rank test on the results of the experiments in Section 3.3.

3.2.1. Evaluation metrics

The performance of the proposed HBNMA in optimizing an objective function is evaluated using three metrics which are the best value (BV), the mean value (MV), and the standard deviation (SD) over several runs.

3.2.2. Wilcoxon signed-rank test

This test is a non-parametric statistical test that aims to detect whether there is a significant difference between two sample means or not.

In this research, the Wilcoxon signed-rank test is applied at a significance level of $\alpha < 0.05$. The sum of the ranks for the functions in which the HBNMA outperformed its counterpart is denoted by R^+. While the sum of the ranks for the opposite is denoted by R^-. The two criteria R^+ and R^- are calculated as described in [24] in which the average ranks are used in dealing with ties. The Wilcoxon test produces a test statistic value (Z) which is converted into a probability (p-value). A low p-value ($p < 0.05$) reflects a significant difference between the HBNMA and its counterpart. IBM SPSS software is used in this work to calculate the p-value.

3.3. *Experiments and discussions*

In this section, three experiments are performed to show the merit of the proposed HBNMA on different dimensional problems. The first experiment is a comparison against the standard BA using problems with $d = 5$ up to $d = 1000$ for each to show the positive impact of the hybridization of the BA with the NMA in the performance of the proposed algorithm. Furthermore, Experiment 1 provides an analysis of the positive impact of the hybridization numerically. The second experiment is a comparison against different variants of the BA that is described in Section 3.3.2 using standard benchmark test functions with different properties to perfectly validate the proposed algorithm. The third experiment is against state-of-the-art heuristics utilizing CEC2014. The results obtained from these experiments are evaluated utilizing three evaluation metrics and the Wilcoxon signed-rank test as described in Section 3.2. The code of the HNMBA was written and run in MATLAB 2013, on Windows 10 with an Intel ® Core i5-1035G CPU processor and 8.00 GB RAM.

3.3.1. Experiment 1 (Analysis of the HBNMA working mechanism with a comparison utilizing high-dimensional problems)

This experiment is divided into two parts. The first part is a comparison between the proposed variant of the BA which is the HBNMA and the standard BA on low- and high-dimensional problems to show the effect of the hybridization of the BA with the NM in the performance of the HBNMA. The second part is an analysis of the algorithm's working mechanism which gives an explanation of the good results obtained for the HBNMA.

Tables 2 and 3 show the results of the comparison utilizing several test functions of dimensions $d = 5, 10, 100$ and 1000 for each. The experimental parameter values are the same in this comparision for both the BA and HBNMA to ensure fairness and they are shown in Table 1. The results are evaluated using the error between the global optimal and the obtained solution, the mean number of function evaluations NFE of 40 independent runs needed to obtain the accuracy indicated, as well as the average time taken to perform one run. Each algorithm terminates if either the error equals zero or the maximum number of function evaluations is reached ($MaxFE = 2 * 10^4$). The results showed the ability of the proposed HBNMA in convergence to the global optimal with zero error and a very low number of function evaluations in most functions with low or high dimension. In contrast, the BA failed to obtain good results in most indicated functions, even though it took the prespecified maximum number of function evaluations, at which the results got worse with an increasing function dimension as illustrated in Tables 2 and 3 and clarified in the example shown in Figure 3 using the Schwefel function. The figure demonstrated that the error value was getting worse in the BA with the increase in dimension, up to 90 in the left-hand figure and from 100 up to 800 in the right-hand figure, because of its weakness to handle the high dimensionality of the problems. Meanwhile, the performance of the HBNMA with the increase in dimension was stable with an error equal to zero. This comparison indicates that the hybridization in the HBNMA at first enhances the exploration ability in the proposed algorithm than that in the standard BA. This enhancement enables the HBNMA to search deeply for the global optimum and jump out from any local optimum no matter the dimension of the test function, as illustrated in the results of the error values of the HBNMA and BA. Second, the hybridization decreases the time consumed to reach the global optimum in the HBNMA from that in the BA. This proves the merit of this work.

Table 1. Conditions and common parameters of all experiments.

	Definition	Experiment 1	Experiment 2	Experiment 3
Condition	Population size	$n = 40$	$n = 30$	$n = 50$
	Number of runs	$r = 40$	$r = 30$	$r = 51$
	Maximum number of function evaluations	$MaxFE = 2 * 10^4$	-	$MaxFE = d * 10^5$
	Maximum number of iterations	-	$MaxIt = 200$	-
Common Parameter	Minimum frequency	$f_{min} = -1$	$f_{min} = -1$	$f_{min} = -1$
	Maximum frequency	$f_{max} = 1$	$f_{max} = 1$	$f_{max} = 1$
	Decay coefficient of loudness	$\alpha = 0.5$	$\alpha = 0.9$	$\alpha = 0.5$
	Enhancement coefficient of pulse emission rate	$\gamma = 0.5$	$\gamma = 0.6$	$\gamma = 0.5$

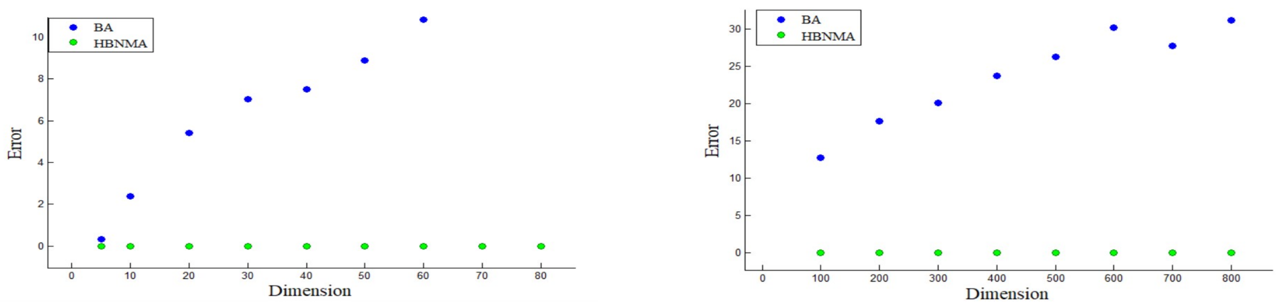

Figure 3. Effect of increasing dimension on the performance of the BA and HBNMA.

Further in this experiment, the behavior of the particles in the HBNMA is demonstrated using four metrics to give a clarification to the superiority of the algorithm. Table 4 illustrates the percentage of the mean number of particles that follow the improved updating formulae (2.16) and (2.18) (M_{im}) in several iterations in one random run and those that follow the standard updating formula of the BA (2.9) (M_{ba}). Also, the success in any iteration ($t + 1$) to achieve an improvement in the global solution ($f(x_*^{t+1}) < f(x_*^t)$) is studied whether referring to a particle that follows the improved formulae or the standard formula. Then the percentages of the successes are calculated and denoted in Table 4 as (Suc_{im}) or (Suc_{ba}), respectively. The results indicated that the majority, around two-thirds, of the particles follow the standard updating process of the BA. The reason behind this is that the particles follow the improved updating process if progress in their previous fitness value is achieved which is not continuously done. However, the high percentage of success in improving the global solution is due to particles that follow the improved updating process in all multimodal and some unimodal functions. The high percentage of success in few unimodal functions is in favour of particles that follow the classic updating formula as there is no need in these functions for high diversification. This proves the positive impact of the improved version to search precisely for better solutions as clarified in the following flowchart.

Table 2. Results of comparison in Experiment 1.

Function	Dim.	Algorithm	Error	NFE	AVT	Function	Dim.	Algorithm	Error	NFE	AVT
Sphere	5	HBNMA	0	400	0.04	Sumsquares	5	HBNMA	0	320	0.02
		BA	1.38×10^{-2}	2×10^4	0.56			BA	1.89×10^{-4}	2×10^4	0.48
	10	HBNMA	0	400	0.02		10	HBNMA	0	400	0.02
		BA	1.45×10^1	2×10^4	0.44			BA	4.26×10^{-1}	2×10^4	0.46
	100	HBNMA	0	560	0.04		100	HBNMA	0	480	0.05
		BA	5.97×10^3	2×10^4	0.70			BA	2.80×10^3	2×10^4	0.53
	1000	HBNMA	0	560	0.11		1000	HBNMA	0	640	0.11
		BA	1.39×10^5	2×10^4	0.67			BA	8.54×10^5	2×10^4	0.65
Schwefel 2.21	5	HBNMA	0	400	0.03	Schwefel 2.22	5	HBNMA	0	320	0.02
		BA	3.35×10^{-1}	2×10^4	0.43			BA	5.48×10^{-2}	2×10^4	0.45
	10	HBNMA	0	480	0.04		10	HBNMA	0	400	0.02
		BA	2.37×10^0	2×10^4	0.42			BA	2.00×10^0	2×10^4	0.76
	100	HBNMA	0	1360	0.12		100	HBNMA	0	480	0.03
		BA	1.27×10^1	2×10^4	0.49			BA	1.85×10^2	2×10^4	0.97
	1000	HBNMA	0	9760	1.74		1000	HBNMA	0	720	0.12
		BA	3.34×10^1	2×10^4	0.59			BA	2.12×10^3	2×10^4	0.76
Step	5	HBNMA	0	320	0.02	Dixon-Price	5	HBNMA	2.15×10^{-1}	2×10^4	0.71
		BA	8.50×10^{-1}	2×10^4	0.57			BA	2.14×10^{-1}	2×10^4	1.97
	10	HBNMA	0	400	0.03		10	HBNMA	6.67×10^{-1}	2×10^4	0.77
		BA	8.57×10^0	2×10^4	0.54			BA	3.61×10^1	2×10^4	0.51
	100	HBNMA	0	480	0.04		100	HBNMA	6.67×10^{-1}	2×10^4	0.94
		BA	4.73×10^2	2×10^4	0.80			BA	2.52×10^4	2×10^4	0.50

Table 3. Results of comparison in Experiment 1.

Function	Dim.	Algorithm	Error	NIT	AVT	Function	Dim.	Algorithm	Error	NIT	AVT
	1000	HBNMA	0	560	0.14		1000	HBNMA	9.85×10^{-1}	2×10^4	3.72
		BA	6.27×10^3	2×10^4	0.76			BA	4.70×10^7	2×10^4	0.79
Sum of Different Powers	5	HBNMA	0	320	0.02	Griewank	5	HBNMA	0	320	0.02
		BA	2.71×10^{-8}	2×10^4	0.53			BA	3.08×10^{-1}	2×10^4	0.54
	10	HBNMA	0	320	0.02		10	HBNMA	0	480	0.03
		BA	2.00×10^{-7}	2×10^4	0.56			BA	8.78×10^{-1}	2×10^4	0.53
	100	HBNMA	0	480	0.05		100	HBNMA	0	560	0.05
		BA	1.70×10^{-5}	2×10^4	0.64			BA	5.15×10^1	2×10^4	0.60
	1000	HBNMA	0	560	0.19		1000	HBNMA	0	640	0.18
		BA	3.91×10^{-4}	2×10^4	1.82			BA	1.47×10^3	2×10^4	1.29
Ackley	5	HBNMA	8.8818×10^{-16}	2×10^4	0.79	Alpine	5	HBNMA	0	400	0.02
		BA	8.14×10^{-1}	2×10^4	0.47			BA	2.43×10^{-1}	2×10^4	0.43
	10	HBNMA	8.8818×10^{-16}	2×10^4	0.77		10	HBNMA	0	400	0.02
		BA	1.50×10^0	2×10^4	0.45			BA	2.02×10^0	2×10^4	0.39
	100	HBNMA	8.8818×10^{-16}	2×10^4	1.05		100	HBNMA	0	560	0.04
		BA	2.36×10^0	2×10^4	0.58			BA	3.95×10^1	2×10^4	0.52
	1000	HBNMA	8.8818×10^{-16}	2×10^4	5.45		1000	HBNMA	0	560	0.15
		BA	2.87×10^0	2×10^4	1.09			BA	3.41×10^2	2×10^4	0.88
Rastrigin	5	HBNMA	0	320	0.04	Xin-She Yang N.1	5	HBNMA	0	320	0.02
		BA	5.88×10^0	2×10^4	0.71			BA	2.77×10^{-2}	2×10^4	0.46
	10	HBNMA	0	400	0.04		10	HBNMA	0	400	0.02
		BA	2.01×10^1	2×10^4	0.42			BA	4.02×10^{-1}	2×10^4	0.57
	100	HBNMA	0	480	0.05		100	HBNMA	0	480	0.05
		BA	5.13×10^2	2×10^4	0.52			BA	1.43×10^{30}	2×10^4	0.55
	1000	HBNMA	0	480	0.14		1000	HBNMA	0	3440	1.08
		BA	4.57×10^3	2×10^4	0.96			BA	Inf	2×10^4	2.02
Zakharov	5	HBNMA	0	400	0.03	Salomon	5	HBNMA	0	320	0.02
		BA	3.80×10^{-3}	2×10^4	0.41			BA	2.53×10^{-1}	2×10^4	0.44
	10	HBNMA	0	400	0.03		10	HBNMA	0	400	0.03
		BA	2.10×10^1	2×10^4	0.42			BA	8.29×10^{-1}	2×10^4	0.48
	100	HBNMA	0	560	0.03		100	HBNMA	0	560	0.04
		BA	1.81×10^{10}	2×10^4	0.54			BA	9.88×10^0	2×10^4	0.54
	1000	HBNMA	0	560	0.10		1000	HBNMA	0	560	0.12
		BA	1.79×10^{22}	2×10^4	0.78			BA	3.62×10^1	2×10^4	0.67

Table 4. Performance analysis of the HBNMA.

Function	$M_{im}(\%)$	$M_{ba}(\%)$	$Suc_{im}(\%)$	$Suc_{ba}(\%)$
Sphere	34.8649%	65.1351%	38.4615%	61.5385%
Sumsquares	33.1383%	66.8617%	39.1304%	60.8696%
Schwefel 2.21	43.2065%	56.7935%	69.5652%	30.4348%
Schwefel 2.22	33.2622%	66.7378%	100%	0%
Step	34.2692%	65.7308%	62.5000%	37.5000%
Dixon-Price	32.6144%	67.3856%	50%	50%
Sum of Different Powers	34.1429%	65.8571%	60.6061%	39.3939%
Griewank	35.2727%	64.7273%	64.2857%	35.7143%
Ackley	37.5000%	62.5000%	88.8889%	11.1111%
Alpine	40.7317%	59.2683%	65.3846%	34.6154%
Rastrigin	41.5714%	58.4286%	78.9474%	21.0526%
Xin-She Yang N.1	42.6250%	57.3750%	66.6667%	33.3333%
Xin-She Yang N.2	39.4565%	60.5435%	83.3333%	16.6667%
Salomon	38.9063%	61.0938%	72.7273%	27.2727%

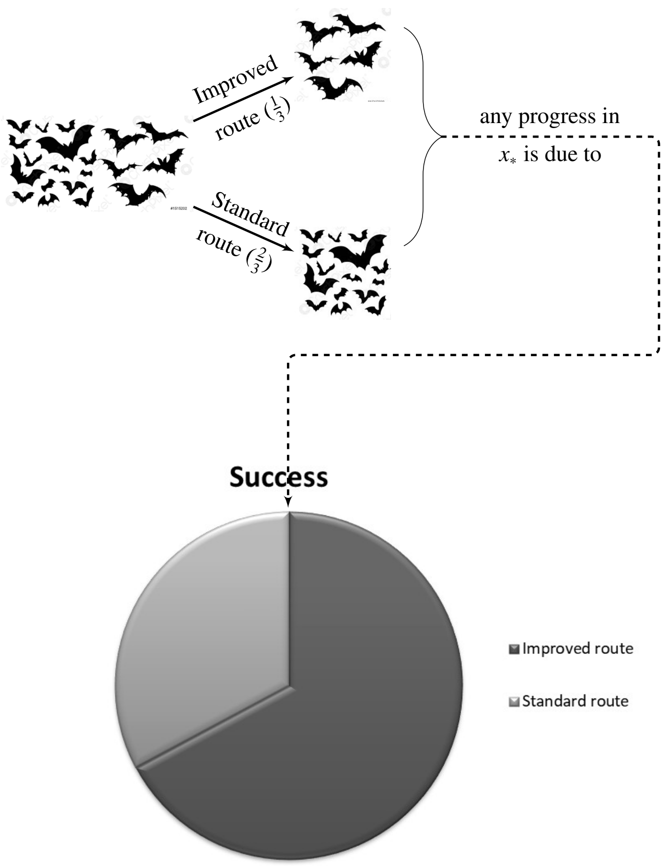

Figure 4. Clarification of the behavior of bats in the HBNMA.

Table 5. Results of Experiment 2.

Function		HBNMA	QABA	IQBA	LMBA	BADE	BA
Sphere	BV	**0.000**	**0.000**	2.658×10^{-137}	7.060×10^{-29}	4.777×10^{-35}	2.912×10^{-1}
	MV	**0.000**	**0.000**	2.743×10^{-31}	4.584×10^{-10}	8.720×10^{-11}	9.552×10^{-1}
	SD	**0.000**	**0.000**	1.480×10^{-30}	1.570×10^{-9}	4.154×10^{-10}	4.532×10^{-1}
	Rank (MV)	1	1	2	4	3	5
Sumsquares	BV	**0.000**	**0.000**	9.882×10^{-148}	8.600×10^{-29}	2.089×10^{-38}	1.617×10^{0}
	MV	**0.000**	**0.000**	1.977×10^{-41}	4.704×10^{-9}	7.812×10^{-11}	5.836×10^{0}
	SD	**0.000**	**0.000**	1.060×10^{-40}	1.950×10^{-8}	2.978×10^{-10}	2.121×10^{0}
	Rank (MV)	1	1	2	4	3	5
Schwefel's 2.20	BV	**0.000**	**0.000**	1.843×10^{-78}	1.743×10^{-15}	4.767×10^{-21}	1.116×10^{0}
	MV	**0.000**	1.301×10^{-309}	5.869×10^{-28}	2.406×10^{-5}	1.226×10^{-7}	2.393×10^{0}
	SD	**0.000**	**0.000**	3.160×10^{-27}	5.680×10^{-5}	3.977×10^{-7}	5.264×10^{-1}
	Rank (MV)	1	2	3	5	4	6
Step	BV	**0.000**	2.982×10^{-19}	3.813×10^{-3}	1.046×10^{-6}	3.920×10^{-2}	4.111×10^{-1}
	MV	**0.000**	2.819×10^{-9}	4.323×10^{-1}	8.421×10^{-3}	4.551×10^{-1}	1.340×10^{0}
	SD	**0.000**	4.534×10^{-9}	4.601×10^{-1}	1.396×10^{-2}	2.892×10^{-1}	6.187×10^{-1}
	Rank (MV)	1	2	4	3	5	6
Dixon-Price	BV	6.667×10^{-1}	$\mathbf{6.322 \times 10^{-2}}$	2.426×10^{-1}	2.263×10^{-1}	6.667×10^{-1}	2.342×10^{0}
	MV	6.668×10^{-1}	$\mathbf{1.791 \times 10^{-1}}$	3.626×10^{-1}	2.748×10^{-1}	9.245×10^{-1}	9.981×10^{0}
	SD	$\mathbf{2.3 \times 10^{-4}}$	5.924×10^{-2}	1.604×10^{-1}	4.146×10^{-2}	9.194×10^{-2}	6.783×10^{0}
	Rank (MV)	4	1	3	2	5	6
Sum of Different Powers	BV	**0.000**	**0.000**	6.107×10^{-157}	3.394×10^{-21}	2.748×10^{-40}	1.142×10^{-2}
	MV	**0.000**	**0.000**	1.992×10^{-54}	2.817×10^{-9}	3.252×10^{-18}	2.295×10^{-1}
	SD	**0.000**	**0.000**	1.070×10^{-53}	1.172×10^{-8}	1.746×10^{-17}	2.039×10^{-1}
	Rank (MV)	1	1	2	4	3	5
Griewank	BV	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	3.485×10^{-2}
	MV	**0.000**	**0.000**	**0.000**	6.655×10^{-11}	5.361×10^{-13}	1.207×10^{-1}
	SD	**0.000**	**0.000**	**0.000**	1.719×10^{-10}	2.592×10^{-12}	4.767×10^{-2}
	Rank (MV)	1	1	1	3	2	4

3.3.2. Experiment 2 (Comparisons against variants of the BA)

In this experiment, The operational results of the HBNMA using several standard benchmark test functions with dimension $d = 10$ for all test functions except those of certain dimensions are compared with the standard BA and four variants of the BA to examine the ability of the proposed variant HBNMA in competing efficient different variants. The HBNMA in this comparison competes with the quantum annealing bat algorithm (QABA) [14], improved quantum annealing bat algorithm (IQBA) [25], group evolution hybrid bat algorithm (LMBA) [26], bat differential hybrid algorithm (BADE) [27], and the BA [5]. All algorithms involved in this experiment undergo the same conditions and common parameter values as stated in [14] and shown in Table 1 for fair comparison. The values of the other parameters related to a certain algorithm used in this comparison are found in their corresponding original literature. Following the same conditions that are stated by other established literature and yielding better results shows the adaptability and the efficiency of the HBNMA under any circumstances. Tables 5 and 6 show superior mean values (MV) compared to the BA and BADE in all test functions. In comparing with the LMBA, the HBNMA achieved the best MV in 13 of 15 functions, equal value in 1 function, and the worst value in 1 function. The competition was increased when comparing the HBNMA against the IQBA and QABA, but the superiority was in favor of the HBNMA. The HBNMA, with respect to the mean value, achieved the best values in 6 functions, the same in 8 functions, and the worst in only one function against the IQBA. In comparing with the QABA, the HBNMA recorded the best in 4, the same in 9, and the worst in 2. To statistically compare the mean results of the HBNMA against its counterparts, the Wilcoxon signed-rank test was used and is illustrated in Table 7. The p-value indicates whether or not there is a significant difference between the competing algorithms. Table 7 indicates that the HBNMA provided significantly better results than the IQBA, LMBA, BADE, and BA with $p < 0.05$. However, no significant difference was seen between the HBNMA and IQBA as $p > 0.05$. In terms of standard deviation (SD), the HBNMA recorded the best in 4, the same in 8, and the worst in 3 when comparing with the QABA. In comparing with the IQBA, the HBNMA had the best values in 11 functions, the same in 3, and the worst in 1. According to the LMBA and BADE, the HBNMA outperformed the SD results of them in most test functions. When comparing the HBNMA against the BA in SD, the superiority was in favor of the HBNMA. The similarity in some results in this comparison between the HBNMA and any other algorithm means that both algorithms achieved the optimum value. The low SD results reflected the stability of the HBNMA.

Table 6. Rest of Experiment 2.

Function		HBNMA	QABA	IQBA	LMBA	BADE	BA
Ackley	BV	**8.882×10^{-16}**	**8.882×10^{-16}**	**8.882×10^{-16}**	**8.882×10^{-16}**	**8.882×10^{-16}**	1.835×10^{0}
	MV	**8.882×10^{-16}**	**8.882×10^{-16}**	**8.882×10^{-16}**	3.260×10^{-5}	2.810×10^{-1}	2.727×10^{0}
	SD	**0.000**	**0.000**	**0.000**	9.776×10^{-5}	7.249×10^{-1}	4.061×10^{-1}
	Rank (MV)	1	1	1	2	3	4
Alpine	BV	**0.000**	**0.000**	5.506×10^{-70}	7.231×10^{-14}	1.271×10^{-19}	2.632×10^{-1}
	MV	**0.000**	1.785×10^{-306}	6.224×10^{-19}	5.990×10^{-5}	3.797×10^{-3}	9.019×10^{-1}
	SD	**0.000**	**0.000**	2.650×10^{-18}	2.306×10^{-4}	4.400×10^{-3}	4.235×10^{-1}
	Rank (MV)	1	2	3	4	5	6
Rastrigin	BV	**0.000**	**0.000**	**0.000**	**0.000**	3.330×10^{0}	2.700×10^{1}
	MV	**0.000**	**0.000**	**0.000**	1.047×10^{-5}	3.051×10^{1}	5.209×10^{1}
	SD	**0.000**	**0.000**	**0.000**	5.512×10^{-5}	1.320×10^{1}	1.025×10^{1}
	Rank (MV)	1	1	1	2	3	4
Michalewicz	BV	**−8.299**	−7.768	−5.736	−4.441	−5.566	−5.202
	MV	**−6.000**	−5.782	−2.529	−3.694	−3.695	−3.575
	SD	1.298×10^{0}	1.101×10^{0}	7.607×10^{-1}	**5.159×10^{-1}**	9.096×10^{-1}	6.495×10^{-1}
	Rank (MV)	1	2	6	4	3	5
Goldstein-Price	BV	**3.000**	**3.000**	**3.000**	**3.000**	**3.000**	3.003
	MV	**3.000**	3.009	22.450	11.605	12.005	5.236
	SD	**5.888×10^{-5}**	3.135×10^{-2}	2.025×10^{1}	1.236×10^{1}	2.124×10^{1}	6.072×10^{0}
	Rank (MV)	1	2	6	4	5	3
Schubert	BV	**−186.7309**	**−186.7309**	−186.3490	**−186.7309**	**−186.7309**	−186.7272
	MV	**−186.7308**	−186.7307	−142.4349	−186.2162	−164.2457	−179.5836
	SD	**1.2217×10^{-4}**	5.618×10^{-4}	4.594×10^{1}	9.798×10^{-1}	3.054×10^{1}	1.088×10^{1}
	Rank (MV)	1	2	6	3	5	4
Hartmann 3-D	BV	**−3.8628**	−3.8627	−3.8039	−3.8540	**−3.8628**	−3.8566
	MV	−3.7339	**−3.8615**	−2.5718	−3.4375	−3.5619	−3.4144
	SD	7.052×10^{-1}	**1.097×10^{-3}**	8.506×10^{-1}	3.661×10^{-1}	4.712×10^{-1}	4.684×10^{-1}
	Rank (MV)	2	1	6	4	3	5
Six-Hump Camel	BV	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**	**−1.0316**
	MV	**−1.0316**	**−1.0316**	−0.9460	**−1.0316**	−1.0068	−1.0260
	SD	5.820×10^{-5}	**1.970×10^{-7}**	1.001×10^{-1}	1.384×10^{-4}	9.946×10^{-2}	1.215×10^{-2}
	Rank (MV)	1	1	4	1	3	2

Table 7. Wilcoxon signed-ranks test of the results of Experiment 2.

Comparison	R^+	R^-	Z	p-value	Better	Equal	Worse
HBNMA versus QABA	69.5	50.5	−0.105	0.917	4	9	2
HBNMA versus IQBA	92	28	−2.028	0.043	6	8	1
HBNMA versus LMBA	107.5	12.5	−2.605	0.009	13	1	1
HBNMA versus BADE	120	0	−3.296	< 0.001	15	0	0
HBNMA versus BA	120	0	−3.408	< 0.001	15	0	0

3.3.3. Experiment 3 (comparisons against well-known heuristics utilizing IEEE CEC2014)

This experiment aims to show the efficiency of the proposed HBNMA using the 30 functions of the IEEE CEC2014 benchmark suite against state-of-the-art heuristics which are the genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], differential evolution (DE) [28], and bat algorithm (BA) [5]. Each function for all competing algorithms is of dimension $d = 10$ that runs for 51 times independently with $d{*}10^5$ function evaluations for each run as a stopping criterion and a population size equal to 50 as shown in Table 1. Also, the values of the common parameters between the HBNMA and BA are stated in Table 1. The crossover and mutation probabilities of the GA are assigned as 0.9 and 0.05, respectively. In PSO, the inertia weight ranges from 0.9 to 0.4 and the coefficients of acceleration are of value 1.2. The parameters, that are used in DE, are the scale factor which belonging $[0.5, 0.9]$ and the crossover rate belonging to $[0.1, 0.9]$. Tables 8 and 9 illustrate the findings of the experiment in terms of MV and SD. The HBNMA succeeded to obtain the best mean error in 29 of 30 test functions and the same in 1 function when comparing with the GA and BA. In comparing with the PSO, the HBNMA obtained the best mean value in 21 of 30 test functions, the same value in 1 functions, and the worst in 8 functions. Furthermore, the HBNMA achieved the best mean value in 23 of 30 test functions, the same in 2 functions, and the worst in 5 functions when comparing against DE. The results showed the superiority of the HBNMA compared with all other algorithms in this experiment. This superiority is also translated in the p-value in Table 10 which demonstrated a significant difference between the HBNMA and the indicated algorithms.

The experimental results illustrated the superior behavior of the HBNMA in different test functions, which mimics real world problems. This is indicated in the following:

(1) In classical unimodal functions, there exists a single global optimum, but there are no local optima. So, these functions need fair intensification in the algorithm to be solved. The HBNMA succeeded easily in obtaining the global optimum for these functions because it has a good intensification ability as indicated in Experiments 1 and 2.
(2) The rotated unimodal functions, specifically functions F1–F3 from the CEC2014 benchmarking suit, are harder than the classical ones. So this suit of problems is perfect in evaluating the local searchability of the algorithm. The HBNMA obtained the best mean value in these functions relative to the state-of-the-art heuristics that are illustrated in Experiment 3 due to its strong local search.
(3) In classical multimodal functions, there are many local minima but only one global optimal. So, these functions are good to test the global searchability of any algorithm. The HBNMA behaved superiorly in these functions due to the strong diversification ability that is enhanced in this work

as shown in Experiments 1 and 2.

(4) In shifted and rotated multimodal functions, specifically functions F4–F16 from the CEC2014 benchmarking suit, the HBNMA succeeded in obtaining the global optimum in the separable functions F8 and F10, but failed to obtain the global optimum in the other non-separable functions as shown in Experiment 3.

(5) In hybrid functions, F17–F22, which more closely approximate real-world benchmarks, the HBNMA obtained the best mean value in 4 of 6 when comparing against other algorithms. Furthermore, HBNMA has superior results with all composition functions, F23–F30, when comparing with the others.

Table 8. Results of Experiment 3.

Function		HBNMA	GA	PSO	DE	BA
F1	MV	$\mathbf{1.50 \times 10^{3}}$	1.93×10^{7}	3.71×10^{4}	7.68×10^{4}	1.51×10^{8}
	SD	$\mathbf{9.54 \times 10^{2}}$	5.20×10^{6}	1.99×10^{4}	2.50×10^{4}	9.11×10^{7}
F2	MV	$\mathbf{2.02 \times 10^{3}}$	6.01×10^{3}	2.10×10^{3}	2.99×10^{6}	5.76×10^{9}
	SD	4.04×10^{3}	5.01×10^{3}	$\mathbf{2.37 \times 10^{3}}$	1.50×10^{6}	4.22×10^{10}
F3	MV	$\mathbf{4.08 \times 10^{-1}}$	9.53×10^{2}	2.20×10^{2}	9.08×10^{1}	4.01×10^{5}
	SD	$\mathbf{6.44 \times 10^{-2}}$	1.56×10^{4}	2.54×10^{3}	1.70×10^{1}	2.98×10^{6}
F4	MV	2.29×10^{1}	3.80×10^{1}	3.40×10^{1}	$\mathbf{1.51 \times 10^{1}}$	7.35×10^{2}
	SD	$\mathbf{1.61 \times 10^{1}}$	1.62×10^{1}	2.40×10^{1}	2.66×10^{1}	7.89×10^{2}
F5	MV	$\mathbf{1.80 \times 10^{1}}$	2.00×10^{1}	2.02×10^{1}	2.05×10^{1}	2.00×10^{1}
	SD	5.91×10^{0}	4.52×10^{-3}	7.89×10^{-2}	9.30×10^{-2}	$\mathbf{3.17 \times 10^{-6}}$
F6	MV	2.82×10^{0}	4.79×10^{0}	$\mathbf{1.32 \times 10^{0}}$	3.13×10^{0}	2.21×10^{2}
	SD	1.27×10^{0}	1.73×10^{0}	$\mathbf{1.11 \times 10^{0}}$	3.21×10^{0}	1.33×10^{0}
F7	MV	3.30×10^{-1}	5.19×10^{-1}	$\mathbf{1.20 \times 10^{-1}}$	1.25×10^{0}	9.15×10^{1}
	SD	1.34×10^{-1}	1.80×10^{-1}	$\mathbf{6.15 \times 10^{-2}}$	1.20×10^{0}	3.51×10^{2}
F8	MV	$\mathbf{0.00 \times 10^{0}}$	1.78×10^{0}	1.12×10^{0}	2.30×10^{1}	6.51×10^{1}
	SD	$\mathbf{0.00 \times 10^{0}}$	1.30×10^{0}	1.32×10^{0}	0.29×10^{1}	2.12×10^{1}
F9	MV	1.52×10^{1}	1.13×10^{1}	$\mathbf{9.02 \times 10^{0}}$	3.27×10^{1}	7.11×10^{1}
	SD	5.75×10^{0}	9.09×10^{0}	$\mathbf{2.92 \times 10^{0}}$	1.09×10^{1}	2.67×10^{1}
F10	MV	$\mathbf{0.00 \times 10^{0}}$	1.08×10^{2}	9.86×10^{1}	1.31×10^{3}	1.20×10^{3}
	SD	$\mathbf{0.00 \times 10^{0}}$	1.48×10^{3}	7.31×10^{2}	2.14×10^{3}	4.18×10^{3}
F11	MV	$\mathbf{1.94 \times 10^{2}}$	6.40×10^{2}	2.11×10^{2}	1.41×10^{3}	1.23×10^{3}
	SD	1.65×10^{2}	3.13×10^{2}	$\mathbf{1.40 \times 10^{2}}$	2.35×10^{3}	3.40×10^{2}
F12	MV	$\mathbf{8.57 \times 10^{-2}}$	2.70×10^{-1}	2.81×10^{-1}	2.31×10^{0}	0.74×10^{0}
	SD	$\mathbf{6.42 \times 10^{-3}}$	9.50×10^{-3}	2.30×10^{-2}	2.60×10^{-2}	5.69×10^{-2}
F13	MV	3.25×10^{-1}	4.95×10^{-1}	$\mathbf{0.56 \times 10^{-1}}$	2.50×10^{-1}	3.34×10^{0}
	SD	1.36×10^{-1}	$\mathbf{1.33 \times 10^{-1}}$	8.22×10^{-2}	1.50×10^{-1}	1.30×10^{0}

Table 9. Rest of Experiment 3.

Function		HBNMA	GA	PSO	DE	BA
F14	MV	1.71×10^{-2}	2.60×10^{-2}	$\mathbf{1.40 \times 10^{-2}}$	2.10×10^{-2}	3.42×10^{0}
	SD	1.65×10^{-1}	1.13×10^{-1}	$\mathbf{4.34 \times 10^{-2}}$	1.86×10^{-1}	1.36×10^{1}
F15	MV	1.46×10^{0}	2.60×10^{0}	$\mathbf{9.58 \times 10^{-1}}$	3.74×10^{0}	1.13×10^{4}
	SD	5.79×10^{-1}	1.24×10^{0}	$\mathbf{3.72 \times 10^{-1}}$	1.21×10^{0}	1.86×10^{4}
F16	MV	2.32×10^{0}	2.49×10^{0}	$\mathbf{0.94 \times 10^{0}}$	4.36×10^{0}	5.45×10^{0}
	SD	4.10×10^{-1}	3.88×10^{-1}	3.98×10^{-1}	4.63×10^{-1}	$\mathbf{1.60 \times 10^{-1}}$
F17	MV	6.05×10^{3}	3.81×10^{6}	7.06×10^{3}	$\mathbf{1.03 \times 10^{2}}$	1.36×10^{6}
	SD	8.23×10^{3}	2.05×10^{6}	3.30×10^{3}	$\mathbf{1.90 \times 10^{2}}$	5.01×10^{6}
F18	MV	$\mathbf{7.95 \times 10^{2}}$	7.45×10^{3}	6.06×10^{3}	1.30×10^{3}	7.94×10^{6}
	SD	2.79×10^{3}	9.20×10^{3}	9.81×10^{3}	$\mathbf{1.57 \times 10^{3}}$	1.26×10^{5}
F19	MV	$\mathbf{5.96 \times 10^{-1}}$	1.49×10^{1}	1.51×10^{1}	1.41×10^{1}	5.44×10^{1}
	SD	$\mathbf{4.32 \times 10^{-1}}$	8.75×10^{1}	8.15×10^{0}	1.10×10^{1}	4.24×10^{1}
F20	MV	$\mathbf{1.93 \times 10^{1}}$	7.52×10^{4}	1.03×10^{4}	6.43×10^{3}	8.58×10^{6}
	SD	$\mathbf{1.63 \times 10^{0}}$	8.60×10^{3}	2.71×10^{3}	5.11×10^{2}	3.21×10^{5}
F21	MV	7.93×10^{2}	1.70×10^{6}	2.68×10^{2}	$\mathbf{1.30 \times 10^{2}}$	3.61×10^{5}
	SD	4.13×10^{3}	2.81×10^{7}	2.65×10^{2}	$\mathbf{1.39 \times 10^{2}}$	9.68×10^{4}
F22	MV	$\mathbf{8.04 \times 10^{-1}}$	2.24×10^{2}	2.82×10^{2}	2.13×10^{2}	4.53×10^{2}
	SD	$\mathbf{8.89 \times 10^{-1}}$	5.17×10^{1}	1.30×10^{2}	3.70×10^{2}	2.24×10^{2}
F23	MV	$\mathbf{2.00 \times 10^{2}}$	3.31×10^{2}	3.29×10^{2}	3.34×10^{2}	5.18×10^{2}
	SD	$\mathbf{0.00 \times 10^{0}}$	1.15×10^{-1}	$\mathbf{0.00 \times 10^{0}}$	6.67×10^{2}	9.74×10^{1}
F24	MV	$\mathbf{1.33 \times 10^{2}}$	1.46×10^{2}	1.74×10^{2}	$\mathbf{1.33 \times 10^{2}}$	2.01×10^{2}
	SD	$\mathbf{9.45 \times 10^{0}}$	2.03×10^{1}	5.51×10^{1}	1.18×10^{1}	3.03×10^{1}
F25	MV	$\mathbf{1.39 \times 10^{2}}$	1.80×10^{2}	3.01×10^{2}	1.96×10^{2}	2.00×10^{2}
	SD	$\mathbf{1.07 \times 10^{1}}$	2.75×10^{1}	4.68×10^{1}	3.92×10^{1}	2.29×10^{1}
F26	MV	$\mathbf{1.00 \times 10^{2}}$	$\mathbf{1.00 \times 10^{2}}$	$\mathbf{1.00 \times 10^{2}}$	$\mathbf{1.00 \times 10^{2}}$	$\mathbf{1.00 \times 10^{2}}$
	SD	1.05×10^{-1}	1.21×10^{-1}	$\mathbf{5.60 \times 10^{-2}}$	2.49×10^{-1}	1.92×10^{0}
F27	MV	$\mathbf{4.64 \times 10^{1}}$	2.91×10^{2}	2.37×10^{2}	2.27×10^{2}	5.79×10^{2}
	SD	$\mathbf{9.68 \times 10^{1}}$	1.78×10^{2}	1.73×10^{2}	2.62×10^{2}	1.23×10^{2}
F28	MV	$\mathbf{2.03 \times 10^{2}}$	5.63×10^{2}	5.39×10^{2}	5.82×10^{2}	7.89×10^{2}
	SD	$\mathbf{2.50 \times 10^{1}}$	1.04×10^{2}	5.87×10^{1}	1.10×10^{2}	3.26×10^{2}
F29	MV	$\mathbf{4.64 \times 10^{1}}$	4.67×10^{5}	1.20×10^{5}	8.54×10^{2}	7.35×10^{4}
	SD	$\mathbf{9.68 \times 10^{1}}$	7.57×10^{6}	6.39×10^{6}	2.20×10^{3}	1.97×10^{5}
F30	MV	$\mathbf{2.00 \times 10^{2}}$	2.63×10^{3}	9.55×10^{2}	6.50×10^{2}	9.40×10^{3}
	SD	$\mathbf{0.00 \times 10^{0}}$	1.25×10^{3}	3.16×10^{2}	2.23×10^{2}	3.40×10^{3}

Table 10. Wilcoxon signed-ranks test of the results of Experiment 3.

Comparison	R^+	R^-	Z	p-value	Better	Equal	Worse
HBNMA versus GA	435	0	−4.703	< 0.001	29	1	0
HBNMA versus PSO	375	60	−3.406	< 0.001	21	1	8
HBNMA versus DE	346	60	−3.256	0.001	23	2	5
HBNMA versus BA	435	0	−4.703	< 0.001	29	1	0

4. Time complexity

In this section, the time complexity of the chosen algorithms is discussed using big O notation. In the standard BA, the time complexity is O(n* MaxIt) [29], where n is the population size and MaxIt is the maximum number of iterations. The maximum number of iterations is taking as the stopping criterion in all algorithms in computing time complexity instead of the number of function evaluations for simplicity. In the proposed HBNMA, the population initialization process has a computational time of O(n). In the main optimization of the proposed algorithm, the calculation of the centroid of the simplex that is needed to compute the reflection and expansion steps is of time complexity O(n*d), where d is the dimension of the problem. In addition, each particle undergoes one of two updating processes which are constant time operations, O(1). Also, continuous expansions may occur as long as refining in the best solution found so far occurs. These continuous expansions have a time complexity of O(k), where k is the maximum number of improved expansions found. The value of the parameter k in the HBNMA is small because if there is an improved expansion, and the probability of obtaining another improved expansion sequentially is small. As a consequence, this probability decreases during further sequential expansions. So, this parameter can be neglected in the time complexity of the HBNMA despite its importance in the robustness of the algorithm. Therefore, the overall time complexity of the HBNMA is O(n*d*MaxIt). The time complexity of the DE algorithm, PSO algorithm, and GA is O(n*d*MaxIt) [30, 31].

Hence, the proposed HBNMA has the same time complexity as in the state-of-the-art metaheuristics GA, PSO, and DE. In comparing with the BA in low-dimensional problems, the HBNMA has almost the same time complexity as in the BA. In high-dimensional problems, the HBNMA is more complex than the BA, but this does not affect the robustness of the HBNMA as it needs a lower number of iterations than that in the BA to reach the global optimal.

5. Conclusions

In this work, a hybrid bat-Nelder-Mead algorithm (HBNMA) is presented to solve multidimensional, unconstrained optimization problems. This algorithm is based on introducing two steps of the NM in the updating formula of the particle's velocity of the standard BA. The reflection step is first introduced besides the standard updating formula of the BA. If an improvement in the fitness value is achieved, sequential expansion steps are then introduced to explore the area deeply, which accelerates the performance of the algorithm. Otherwise, the classic updating process, as in the standard BA, is performed. This mechanism enables particles to globally discover the search space and exploit any progress for deeper research for even better results. Accordingly, a satisfactory balance is achieved between global and local searches in the proposed algorithm. Finally, several numerical experiments were established and the simulation results of the proposed HBNMA showed superiority of the algorithm.

6. Future work

In future research, multi-objective and constrained optimization are going to be addressed as most real-world continuous problems are of that form. New techniques are going to be innovated to deal with the selection of solutions in multiobjective problems rather than the common used approaches. These new techniques are going to be integrated with the HBNMA and any other single-objective, continuous optimization algorithm to be able to handle complex multiobjective problems. Furthermore, greater emphasis can be devoted to create new techniques that can handle constraints in constrained optimization problems.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Author contributions

Enas Suhail: Formal analysis, Original draft preparation, Visualization, Software; Mahmoud El-Alem: Investigation, Methodology, Writing—review and editing, Supervision; Omar Bazighifan: Resources, Writing—review and editing; Ahmed S. Zekri: Conceptualization, Validation, Writing—review and editing, Supervision. All authors have read and approved the final version of the manuscript for publication.

Conflict of interest

All authors declare that they have no conflicts of interest.

References

1. J. Holland, Genetic algorithms, *Sci. Am.,* **267** (1992), 66–73.
2. J. Kennedy, R. Eberhart, Particle swarm optimization, *Proceedings of ICNN'95-International conference on neural network*, 1995, 1942–1948. http://doi.org/10.1109/ICNN.1995.488968
3. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: Optimization by a colony of cooperating agents, *IEEE T. Syst. Man Cy. B* , **26** (1996), 29–41. http://doi.org/10.1109/3477.484436
4. X. L. Li, Z. J. Shao, J. X. Qian, An optimizing method based on autonomous animats: Fish-swarm algorithm, *Syst. Eng.-Theory Pract.*, **22** (2002), 32–38. https://doi.org/10.12011/1000-6788(2002)11-32
5. X-S. Yang, A new metaheuristic bat-inspired algorithm, In: *Nature inspired cooperative strategies for optimization (NICSO 2010)*, Berlin, Heidelberg: Springer, 2010, 65–74. https://doi.org/10.1007/978-3-642-12538-6_6
6. E. H. Houssein, D. Oliva, N. A. Samee, N. F. Mahmoud, M. M. Emam, liver cancer algorithm: a novel bio-inspired optimizer, *Comput. Biol. Med.*, **165** (2023), 107–389. https://doi.org/10.13140/RG.2.2.11139.27688
7. A. Qi, D. Zhao, A. A. Heidari, L. Liu, Yi. Chen, H. Chen, FATA: An efficient optimization method based on geophysics, *Neurocomputing*, **607** (2024), 128–289. https://doi.org/10.1016/j.neucom.2024.128289

8. A-Q. Tian, F-F. Liu, H-X. Lv, snow geese algorithm: A novel migration-inspired meta-heuristic algorithm for constrained engineering optimization problems, *Appl. Math. Modell.*, **126** (2024), 327–347. https://doi.org/10.1016/j.apm.2023.10.045

9. M. Chen, L. Yang, G. Zeng, K. Lu, Y. Huang, IFA-EO: An improved firefly algorithm hybridized with extremal optimization for continuous unconstrained optimization problems, 2021, Available from: https://doi.org/10.21203/rs.3.rs-190790/v1

10. B. Shi, J. Chen, H. Chen, W. Lin, prediction of recurrent spontaneous abortion using evolutionary machine learning with joint self-adaptive sime mould algorithm, *Comput. Bio. Med.*, **148** (2022), 105–885. https://doi.org/10.1016/j.compbiomed.2022.105885

11. M. Shehab, M. A. Abu-Hashem, M. K. Y. Shambour, A. I. Alsalibi, O. A. Alomari, A. Gupta, et al., A comprehensive review of bat inspired algorithm: variants, applications, and hybridization, *Arch. Computat. Methods Eng.*, **30**(2023), 765–797. https://doi.org/10.1007/s11831-022-09817-5

12. J-H. Lin, C-W. Chou, C-H. Yang, H-L. Tsai, A chaotic Levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems, *CIT*, **2** (2012), 56–63.

13. C. Gan, W. Cao, M. Wu, X. Chen, A new bat algorithm based on iterative local search and stochastic inertia weight, *Expert Syst. Appl.*, **104** (2018), 202–212. https://doi.org/10.1016/j.eswa.2018.03.015

14. S. Yu, J. Zhu, C. Lv, A Quantum Annealing Bat Algorithm for Node Localization in Wireless Sensor Network, *Sensors*, **23** (2023), 782. https://doi.org/10.3390/s23020782

15. G. Wang, L. Guo, A novel hybrid bat algorithm with harmony search for global numerical optimization, *J. Appl. Math.*, **2013** (2013). https://doi.org/10.1155/2013/696491

16. R. Gupta, N. Chaudhary, S. K. Pal, Hybrid model to improve Bat algorithm performance, *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2014, 1967–1970. https://doi.org/10.1109/ICACCI.2014.6968649

17. G. B. Mahanta, A. Rout, G. B. Muralia, B. Deepak, B. B. Biswal, Application of Hybrid Nelder-Mead Bat Algorithm to Improve the Grasp Quality during the Automated Robotic Grasping, *Proc. Comput. Sci.*, **133** (2018), 612–619. https://doi.org/10.1016/j.procs.2018.07.093

18. B. Kheireddine, B. Zoubida, H. Tarik, Improvements of bat algorithm using crossover technique and hybridization with Nelder-Mead simplex method, *COMPEL*, **38** (2019), 977–989.

19. A. F. Ali, Accelerated bat algorithm for solving integer programming problems, *Egypt. Comput. Sci. J.*, **39** (2015), 507–518.

20. A. F. Ali, M. A. Tawhid, Solving integer programming problems by hybrid bat algorithm and direct search method, *Trends Artif. Intell.*, **2** (2018), 46–59. http://doi.org/10.36959/643/303

21. J. A. Nelder, R. Mead, A simplex method for function minimization, *Comput. J.*, **7** (1965), 308–313. http://doi.org/10.1093/COMJNL/7.4.308

22. S. Surjanovic, D. Bingham, Virtual Library of Simulation Experiments: Test Functions and Datasets, accessed on 31 December 2022. Available from: `https://www.sfu.ca/~ssurjano`.

23. J. Liang, B. Qu, P. Suganthan, Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization, *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, 2013, Technical Report 201311.

24. J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, *Swarm Evol. Comput.*, **1** (2011), 3–18. https://doi.org/10.1016/j.swevo.2011.02.002

25. Z. Zhou, Z. Sun, Research and application of improved quantum-behaved bat algorithm, *Comp. Eng. D*, **40** (2019), 84–91.

26. Z. Li, Improved bat algorithm based on grouping evolution and hybrid optimization, *Math. Pr. Th*, **50** (2020), 141–149.

27. Z. Zhao, M. Zeng, H-M. Mo, Z. Li, T. Wen, Cooperatively intelligent hybrid bat and differential evolution algorithm, *Comp. Eng. D*, **41** (2020), 402–410.

28. R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, **11** (1997), 341–359. https://doi.org/10.1023/A:1008202821328

29. M-R. Chen, Y-Y. Huang, G-Q. Zeng, K-D. Lu, L-Q. Yang, An improved bat algorithm hybridized with extremal optimization and Boltzmann selection, *Expert Syst. Appl.*, **175** (2021), 114–812. https://doi.org/10.1016/j.eswa.2021.114812

30. M. Omran, A. Engelbrecht, Time complexity of population-based metaheuristics, *MENDEL*, **29** (2023), 255–260. https://doi.org/10.13164/mendel.2023.2.255

31. L. Zhou, K. Chen, H. Dong, S. Chi, Z. Chen, An improved beetle swarm optimization algorithm for the intelligent navigation control of autonomous sailing robots, *IEEE Access*, **9** (2020), 5296–5311. https://doi.org/10.1109/ACCESS.2020.3047816

Supplementary

The following table describes the benchmark test functions used in Section 3.3:

Function	Dimension	Range	Optimum	Description
Sphere	vary	$[-100, 100]$	0	Unimodal, separable
Sumsquares	vary	$[-10, 10]$	0	Unimodal, separable
Schwefel 2.20	vary	$[-100, 100]$	0	Unimodal, separable
Schwefel 2.21	vary	$[-100, 100]$	0	Unimodal, separable
Schwefel 2.22	vary	$[-10, 10]$	0	Unimodal, non-separable
Step	vary	$[-100, 100]$	0	Unimodal, separable
Dixon-Price	vary	$[-10, 10]$	0	Unimodal, non-separable
Sum of Different Powers	vary	$[-1, 1]$	0	Unimodal
Griewank	vary	$[-600, 600]$	0	Multimodal, non-separable
Ackley	vary	$[-30, 30]$	0	Multimodal, non-separable
Alpine	vary	$[-10, 10]$	0	Multimodal, separable
Rastrigin	vary	$[-5.12, 5.12]$	0	Multimodal
Michalewicz	vary	$[0, \pi]$	-9.6602	Multimodal
Goldstein-Price	2	$[-2, 2]$	3	Multimodal, non-separable
Shubert	2	$[-10, 10]$	-186.7309	Multimodal
Hartmann 3-D	3	$[0, 10]$	-3.8628	Multimodal, non-separable
Six-Hump Camel	2	$[-2, 2]$	-1.0316	Multimodal
Qing	vary	$[-500, 500]$	0	Multimodal, separable
Xin-She Yang N.1	vary	$[-5, 5]$	0	Multimodal, separable
Salomon	vary	$[-100, 100]$	0	Multimodal, non-separable
Xin-She Yang N.2	vary	$[-2\pi, 2\pi]$	0	Multimodal, non-separable
Penalized	vary	$[-50, 50]$	0	Multimodal
Zakharov	vary	$[-5, 10]$	0	Multimodal, non-separable

AIMS Press