Hindawi
Scientific Programming
Volume 2021, Article ID 5557259, 14 pages
https://doi.org/10.1155/2021/5557259

Research Article

A New Design of Metaheuristic Search Called Improved Monkey Algorithm Based on Random Perturbation for Optimization Problems

Mustafa Tunay

Department of Computer Engineering, Istanbul Gelisim University, Istanbul, Turkey

Correspondence should be addressed to Mustafa Tunay; mtunay@gelisim.edu.tr

Received 22 February 2021; Revised 28 March 2021; Accepted 29 April 2021; Published 7 May 2021

Academic Editor: Roberto Natella

The aim of this paper is to present a design of a metaheuristic search called improved monkey algorithm (MA+) that provides a suitable solution for the optimization problems. The proposed algorithm has been renewed by a new method using random perturbation (RP) into two control parameters ($p1$ and $p2$) to solve a wide variety of optimization problems. A novel RP is defined to improve the control parameters and is constructed off the proposed algorithm. The main advantage of the control parameters is that they more generally prevented the proposed algorithm from getting stuck in optimal solutions. Many optimization problems at the maximum allowable number of iterations can sometimes lead to an inferior local optimum. However, the search strategy in the proposed algorithm has proven to have a successful global optimal solution, convergence optimal solution, and much better performance on many optimization problems for the lowest number of iterations against the original monkey algorithm. All details in the improved monkey algorithm have been represented in this study. The performance of the proposed algorithm was first evaluated using 12 benchmark functions on different dimensions. These different dimensions can be classified into three different types: low-dimensional (30), medium-dimensional (60), and high-dimensional (90). In addition, the performance of the proposed algorithm was compared with the performance of several metaheuristic algorithms using these benchmark functions on many different types of dimensions. Experimental results show that the improved monkey algorithm is clearly superior to the original monkey algorithm, as well as to other well-known metaheuristic algorithms, in terms of obtaining the best optimal value and accelerating convergence solution.

1. Introduction

In this section, the background understanding of optimization in calculus, mathematical optimization, heuristics, and metaheuristic approaches is given. The research based on optimization [1–3] seeks out a solution iteratively for analytical solutions that have been analyzed. The design of an improved monkey algorithm for a multivariate system is noticed.

Fermat and Lagrange were the first to suggest formulas that are based on calculations for determining the optima. Newton and Gauss were the first to suggest iterative methods for finding the best solution. Actually, this means an approach to optimization in calculus in the case of a point on a function of one variable. It gives the best solution (the maximum or minimum of the function). Many optimization problems are primarily to find the best solution within certain boundaries. It refers to the best available functions that solve objective applied mathematics functions. Formally, "linear programming" was started by Kantorovich in 1939. It is also called linear optimization (LO). The LO is a technique to get the best solution in a calculus model whose elements are represented by linear relationships. Therefore, linear optimization is also a special case of mathematical optimization. The first well-known approach was the simplex method by using mathematical optimization. Danzig studied the simplex method in 1947 for solving linear programming problems. Since then, many optimization methods or techniques have been developed. These are, respectively, as follows: quasi-Newton method [4], steepest descent method [5], possible directions method [6], Newton method [7], penalty method [8, 9], and quadratic programming [10].

Karush–Kuhn–Tucker conditions are first derivative tests for a solution in nonlinear programming (nonlinear optimization) as an optimal expression in mathematical optimization. Kuhn and Tucker studied first derivative tests in 1951. Karush explained in his Master's thesis in 1939 the necessary conditions for a constrained optimum. The Karush–Kuhn–Tucker conditions of nonlinear programming generalize the method of Lagrange multipliers, which allows only equality constraints. Mathematical programming is briefly about the selection of a best element from some set of available alternatives. Quantitative disciplines are common optimization problems of sorts arising in all from computer science, engineering operations research, economics, and industry. Since then, many optimization methods or techniques have been developed as solutions of interest in mathematics for centuries. Thus, mathematical programming is a rising trend for many fields. These kinds are, respectively, as follows: linear programming [11–13], nonlinear programming [14, 15], objective programming [16, 17], and dynamic programming [18, 19]. With regard to nonlinear optimization, that is, having at least one goal or nonlinear constraint function, the known approaches have encountered a lot of difficulties. Unfortunately, all tasks in engineering design are almost nonlinear.

Heuristic methods were first used in philosophy and mathematics for finding solutions to complex problems. Heuristics are problem-dependent methods. Thus, they are usually adapted to a specific problem and try to make full use of its features. However, they are often too greedy, tend to fall into the local optimum trap, and generally cannot get a global optimal solution. The study of this method was developed in human decision-making in the 1970s–1980s by Tversky and Kahneman. In the 1980s, metaheuristic approaches attracted the attention of engineers and they studied all kinds of optimization. Metaheuristics are problem-independent methods and they are of a high level. A set of strategies are provided for developing heuristic optimization algorithms. In general, they are not greedy. In fact, they can even accept temporary deterioration of the solution which allows them to explore the solution space more deeply and thus get a better solution. One of the most well-known approaches was genetic algorithms (GAs). Holland studied the principle of "survival of the fittest" in the 1960s. Subsequently, Simulated Annealing was published in 1983. The optimization problems were solved by Simulated Annealing (SA).

The SA is currently formulated by an objective function for many variables; that is, it means several constraints. Therefore, with SA in practice, the constraint can be penalized as part of the objective function for the best solution. The following are five metaheuristic periods:

(1) In 1940: pretheoretical period

(2) From 1940 to 1980: the early period

(3) From 1980 to 2000: the method-centric period

(4) In 2000s: the framework-centric period

(5) Scientific period (future)

Nowadays, there are many optimization algorithms that are designed to find the global optimal solutions to optimization problems. One of them is metaheuristic algorithms that can be efficiently used to solve "local minima" problems and determine global solutions of the optimization problems. The set of metaheuristic algorithms include ant colony optimization (ACO) [20, 21], ant lion optimizer (ALO) [22], bat algorithm (BAT) [23], cuckoo search (CS) [24], elephant herding optimization (EHO) [25], particle swarm optimization (PSO) [26], krill herd (KH) [27], moth-flame optimization (MFO) [28], monarch butterfly optimization (MBO) [29, 30], mussels wandering optimization (MWO) [31], moth search algorithm (MSA) [32], and whale optimization algorithm (WOA) [33] for finding good solution to optimization problems. Even today, new methods are being developed as new metaheuristics are invented. Other metaheuristics research works have been done on the designing of the evolutionary theory such as biogeography-based optimization (BBO) [34], the differential evolution (DE) [35], evolution strategies (ES) [36], genetic algorithm (GA) [37, 38], harmony search (HS) [39], gravitational search algorithm (GSA) [40], sine cosine algorithm (SCA) [41], dragonfly algorithm (DA), and hybrid ABC/DA (HAD) [42].

What is more, the improved monkey algorithm (MA+), which finds the best solution and solves optimization problems, is designed in this study. In addition, the proposed algorithm is a new metaheuristic search for the optimization of multivariate systems. There exists much insufficiency for monkey algorithm about its solution search area which may bring about the premature convergence and the low search accuracy when solving complex optimization of multivariate systems. Then, considering that monkey algorithm converges very slowly, a random perturbation method can be used to ensure the diversity of monkey algorithm against premature convergence. The design of a random perturbation into two parameters in a convergence state helps the best monkey position to jump out of possible local optima to further increase the performance of the proposed algorithm (MA+). Thus, the search strategy in the proposed algorithm has proven to have a successful global optimal solution, convergence optimal solution, and much better performance on many complex optimization problems for the lowest number of iterations.

This paper is organized as follows: Section 2 describes the proposed algorithm and the design of a random perturbation into two parameters is explained clearly. Section 3 describes the experimental results and discussion. The information of twelve benchmark functions is given. Moreover, the performance of the proposed algorithm is evaluated and is compared with many comparative algorithms (many metaheuristic algorithms and modified comparative algorithms) on different dimensional functions. Finally, the conclusion is summarized in Section 4.

2. Related Work

The aim of this paper is to present the design of a new optimization method called improved monkey algorithm (MA+) to find a good solution for the optimization of multivariate systems. The design of the proposed algorithm (MA+) is a new

metaheuristic search method for optimization problems inspired by the behavior of the movement of a monkey. The original monkey algorithm (MA) mainly consists of four processes, namely, initialization process, climb process, watch-jump process, and somersault process. The improvement of the monkey algorithm is renewed by adding random perturbation (RP) in the original four processes. All processes in the proposed algorithm have been designed in Figure 1.

Step A. Initialization Process

The proposed algorithm begins with random generation of a position for each monkey. Each monkey position is set to M, where M represents population size (number of monkeys). Hence, ith is a position as $x_i = (i = 1, 2, \ldots, M)$ for each monkey as in the following equation:

$$x_i = [x_1, x_2, \ldots x_M]. \tag{1}$$

Each monkey's position is evaluated in objective function. It is also set to be present in the searching area (lower boundary-upper boundary). Lower and upper boundaries (lb and ub) are for all solutions (monkey's position). All solutions should be in the searching area between lower boundary (lb) and upper boundary (ub).

Step B. Climb Process

The climb process changes the monkeys' positions step by step from the initial positions to new ones that can evaluate an improvement in the objective function. Step length (a) parameter in the design of climb process updates the movement of monkeys' positions. The total number of monkeys is M. Hence, ith is a position as $x_i = ((x_{i1}, x_{i2}, \ldots x_{in}), i = 1, 2, \ldots, M)$ for each monkey. The step length in changing (updating) position of monkey is in the following equation:

$$\Delta x_{ij} = \begin{cases} a, & \text{with probability } \dfrac{1}{2}, \\ -a, & \text{with probability } \dfrac{1}{2}, \end{cases} \tag{2}$$

where Δx_{ij} is updating the position of monkeys ($j = 1, 2, \ldots, n$); a (positive number) is step length in climb process with $a = 10^{-3}$. Each ith monkey position evaluates an improvement in the objective function (F') in the climb number of iterations (N_c). This function is called the pseudogradient of the objective function and is expressed as follows:

$$y_j = x_{ij} + a \cdot \operatorname{sign}\left(F'_{ij} x_i\right), \quad j = 1, 2, \ldots, n. \tag{3}$$

The step length in the climb process has a crucial role in the precision of the approximation of the local solution, so the climb process supports a feasible solution. y is a feasible position for each monkey. x_i is updating with y till reaching a feasible solution. Otherwise, x_i does not change.

Step C. Watch-Jump Process

This process checks each monkey position after the climb process. In other words, it is checked whether their position has reached the top or not. Moreover, each monkey looks around to see whether there is a position that is higher than the current position. If they have it, they will jump from their current position. Otherwise, it means that their position has not reached the top. Of course this will be realized for the monkeys who have the best positions (close to the top or at the top). Therefore, each monkey takes a maximal distance from its current position. The maximal distance in the watch-jump process is parameter b as the eyesight. This process is expressed as follows:

$$y_j = \left(x_{ij} - b, x_{ij} + b\right). \tag{4}$$

The parameter b is the eyesight with $b = 0.5$. This process updates x_i with y. Both are evaluated: $f(y) \geq f(x_i)$. Otherwise, it checks equation (4) till reaching an appropriate point (y). Then, in this process, the climb process is repeated by employing y. Thus, each monkey takes a maximal distance from its current position.

Step D. Somersault Process

This process enables finding out new positions (searching domains). This process enables finding new positions (searching domains) to the barycentre of all monkeys' current positions defined as a pivot. Monkeys will somersault along the direction pointing to pivot. All solutions should be in somersault interval $[c, d]$ with $[-1, 1]$. The search space of monkeys for this problem has large feasible spaces till increasing values of $|c|$ and d, respectively. In this process, a real number is generated randomly as s between somersault intervals $[-1, 1]$ and is expressed as follows:

$$y_j = x_{ij} + s\left(p_j - x_{ij}\right), \tag{5}$$

$$p_j = \sum_{i=1}^{M} x_{ij}, \tag{6}$$

where p is the somersault pivot, $j = 1, 2, \ldots, n$, and then $x_i = y$ if $y = y_1, y_2, \ldots y_n$ till feasible solution. Otherwise, this process repeats equations (5) and (6) until a feasible solution (y) is found.

Step E. Random Perturbation (RP) Process

This process controls monkeys current position, which can be stuck at some optimal solution. After somersault process, a novel random perturbation (RP) process is constructed of the proposed algorithm into two control parameters with $p1 = 0.5$ and $p2 = 0.2$, respectively. $p1$ improves monkeys' current position if they are stuck at a local minimum. The same or worse value is found in searching space consecutively; they have a tolerance number (tolX) in the number of iterations for improving monkeys' position (one at a time). $p2$ improves along the direction pointing to their current position with different perturbation to out from some possible local minimum or to search for other (and better) minima. The details regarding the pseudocode of the proposed algorithm are shown in Algorithm 1.

The design of the proposed algorithm for solution to global numerical optimization problems begins with population (M), boundaries (lb, ub), eyesight (b), climb number

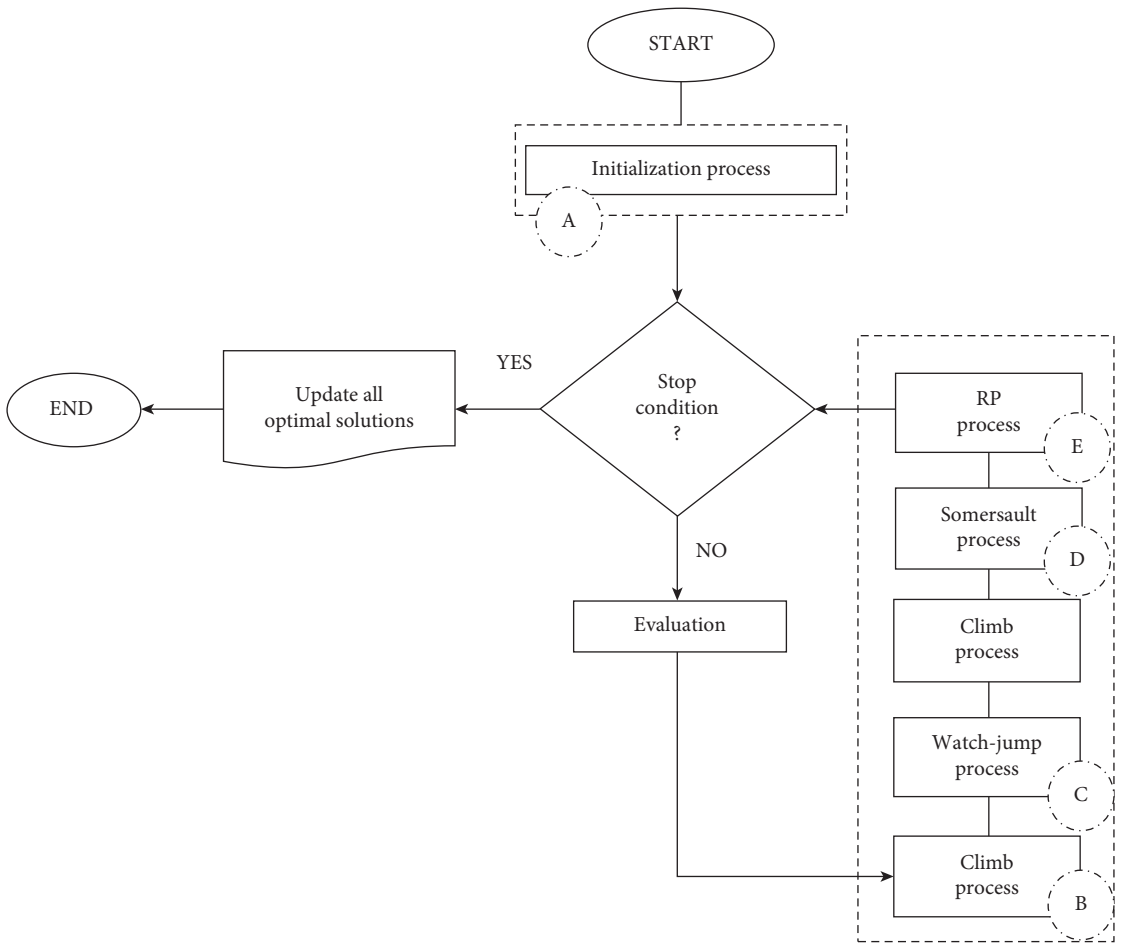

Figure 1: Flowchart of the improved monkey algorithm: (a) is initialization process, (b) is climb process, (c) is watch-jump process, (d) is somersault process, and (e) is random perturbation process.

(N_c), somersault interval c, $d \in [-1, 1]$, and control parameters ($p1$, $p2$). All input parameters are designed for the proposed algorithm. In addition, a random dimension is calculated as ceil (rand x D). This calculation is evaluated as a random scalar that is drawn from the standard normal distribution multiplying with the parameter $p1$. Thus, it prevents the proposed algorithm from getting stuck in some optimal solutions while controlling the monkeys' position. The second one is about uniformly distributed random numbers (rand [1 x D]) multiplying with the parameter $p2$. This scalar point is combined with each element of the vector x. Thus, improvement of monkeys' positions is found along the direction pointing with different perturbation to out from some possible local minimum or to search for other (and better) minima. The special steps of the improved control parameters with RP are described in Algorithm 1.

3. Results and Discussion

3.1. Benchmark Functions. The performance of the improved monkey algorithm (MA+) is implemented in Matlab (2017). The features of equipment of the computer used are given as follows:

(1) CPU: i5–6200 U

(2) CPU speed: 2.30 GHz–2401 MHz

(3) RAM: 4.00 GB

(4) OS: Microsoft Windows 10

The information for 12 benchmark functions is listed in Table 1 as the name of each function, its equation, and range. The improved monkey algorithm (MA+) performed on various benchmark functions. They are 12 benchmark functions, namely, sphere function ($F1$), Schwefel 2.22 function ($F2$), Schwefel 1.2 function ($F3$), Rosenbrock function ($F4$), Ackley function ($F5$), Griewank function ($F6$), sum squares function ($F7$), Dixon-Price function ($F8$), Bent Cigar function ($F9$), sum of different powers function ($F10$), Holzman function ($F11$), and hyperellipsoid function ($F12$). The performance of the improved monkey algorithm and the performance of comparative algorithms (metaheuristic) are evaluated against 12 benchmark functions in the next section.

3.2. The Performance of Improved Monkey Algorithm on Different Dimensional Functions. The performance of the improved monkey algorithm and the performance of original monkey algorithm (MA) were evaluated against 12 benchmark functions on different dimensions that can be

```
Step A–D: Inputs ⟶ Step E
global_min = −1
for i = 1 to M do
  for j = 1 to tolX do
    d ← ceil (rand x D)
    y_i ← x_id ∗ (1 + p1 ∗ randn)
    if lb < y_i < ub then
      Continue
    end if
    if global_min > 0 then
    |if f (y_i) > f (x_ij) then
      |x_ij ← y_i
    |end if
    Else
    |if f (y_i) < f (x_ij) then
      |x_ij ← y_i
    |end if
    end if
  end for
end for
for i = 1 to M do
  for j = 1 to n do
    y_i ← x_ij (·) (1 + p2 ∗ rand [1, D])
    if lb < y_i < ub then
      Continue
    end if
    if global_min > 0 then
    |if f (y_i) > f (x_ij) then
      |x_ij ← y_i
    |end if
    Else
    |if f (y_i) < f (x_ij) then
      |x_ij ← y_i
    |end if
    end if
  end for
end for
Output
```

Algorithm 1: A novel RP is constructed of the proposed algorithm.

classified into three different types: low-dimensional (30D), medium-dimensional (60D), and high-dimensional (90D). Both algorithms have the same size of population and number of iterations (max) and their maximum function evaluation times are equal. Each algorithm is run 100 times independently for all conditions. The parameters and the same conditions for both algorithms are set as follows: size of population (M) = 50, number of iterations (Ite.) = 50, and dimension (D) = 30, 60, and 90. The best mean and the best standard deviation of experimental results are marked in bold for each function.

The first part of this experiment was conducted on the 30, 60, and 90 dimensions. The experimental results were illustrated in Table 2 that shows 12 benchmark optimization functions ($F1$–$F12$), and the improved monkey algorithm (MA+) achieves the best optimization results on the best, mean, and standard deviation values. Therefore, the experimental results on the all-dimensional benchmark functions showed that the performance of the improved monkey algorithm is much better than that of the original monkey algorithm (MA) for the all-dimensional functions. The experimental results are more intuitively demonstrated by the convergence plots and global search ability of the two algorithms (MA and MA+), and the convergence plots of the both algorithms on the 30-dimensional functions are in Figures 2(a)–2(l) for all benchmark functions.

Table 2 shows the best mean optimization results for all functions ($F1$–$F12$) on the 30 dimensions, and the performance of the proposed improved algorithm was obtained: $1.03E-40$, $1.09E-23$, $1.22E-36$, $2.69E+01$, $9.55E-15$, $0.00E+00$, $4.68E-38$, $6.67E-01$, $1.40E-31$, $1.9E-196$, $3.49E-54$, and $1.15E-37$ respectively. Additionally, Figures 2(a) to 2(l) reveal that the original monkey algorithm is much poorer for the 30-dimensional functions, while the proposed improved algorithm still shows a distinguished searching ability, global optimal solution, and its convergence speed in all functions.

Table 1: Benchmark functions.

Name of function	Equation	Range
Sphere	$F1(x) = \sum_{i=1}^{D} x_i^2$	$-5.12 \le x_i \le 5.12$
Schwefel 2.22	$F2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$-10 \le x_i \le 10$
Schwefel 1.2	$F3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	$-100 \le x_i \le 100$
Rosenbrock	$F4(x) = \sum_{i=1}^{D-1} [100\sqrt{\lvert x_i - x_i^2 \rvert} + (1 - x_i)^2]$	$-30 \le x_i \le 30$
Ackley	$F5(x) = -20\exp(-0.2\sqrt{(1/D)\sum_{i=1}^{D} x_i^2}) - \exp((1/D)\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e$	$-32 \le x_i \le 32$
Griewank	$F6(x) = (1/4000)\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos(x_i/\sqrt{i}) + 1$	$-600 \le x_i \le 600$
Sum squares	$F7(x) = \sum_{i=1}^{D} i x_i^2$	$-10 \le x_i \le 10$
Dixon-Price	$F8(x) = (x_1 - 1)^2 + \sum_{i=1}^{D} i(2x_i^2 - x_{i-1})^2$	$-10 \le x_i \le 10$
Bent Cigar	$F9(x) = x_i^2 + 10^6 \sum_{i=1}^{D} x_i^2$	$-100 \le x_i \le 100$
Sum of different powers	$F10(x) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	$-1 \le x_i \le 1$
Holzman	$F11(x) = \sum_{i=1}^{D} i x_i^4$	$-10 \le x_i \le 10$
Hyperellipsoid	$F12(x) = \sum_{i=1}^{D} i^2 x_i^2$	$-5.12 \le x_i \le 5.12$

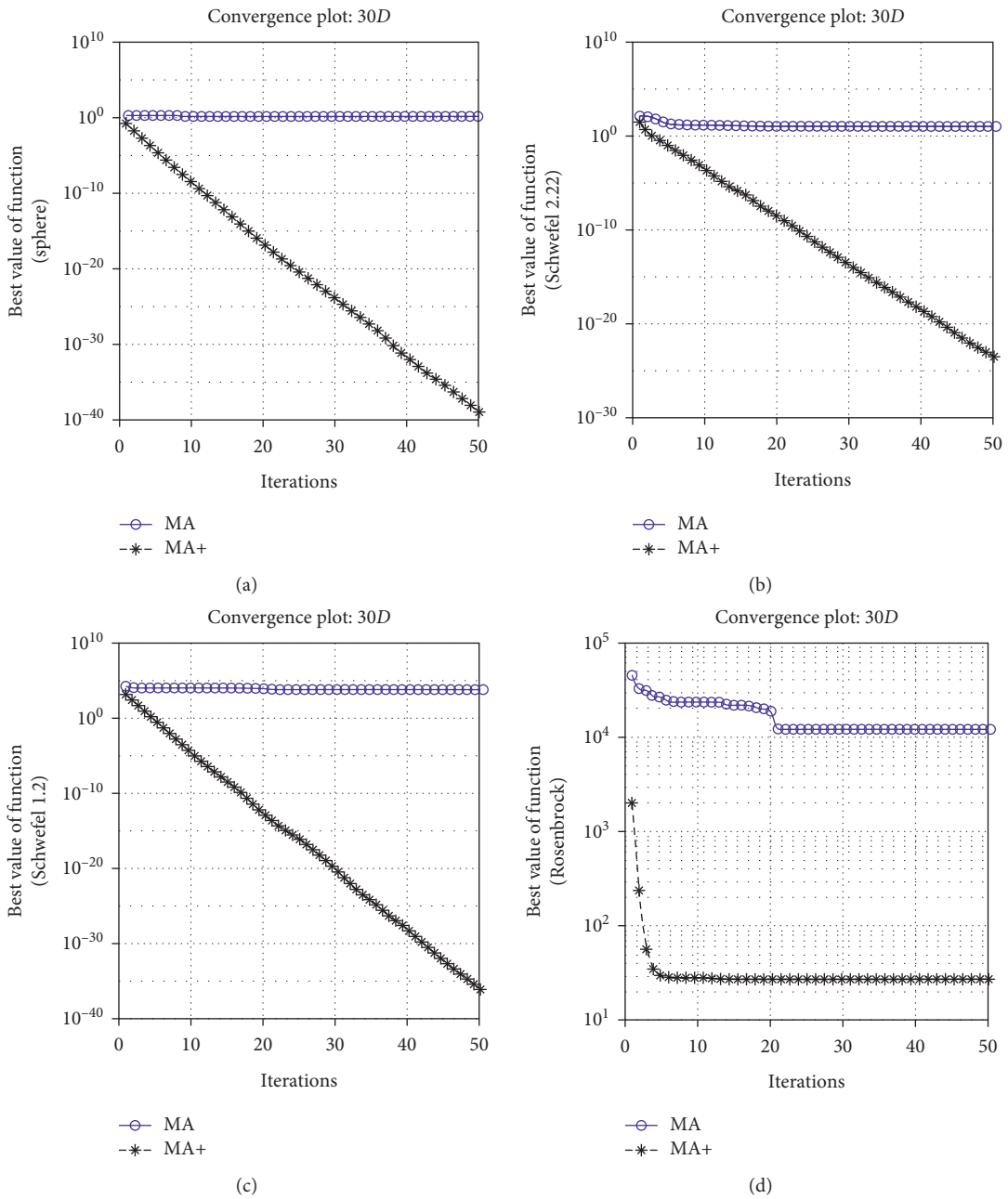

Figure 2: Continued.

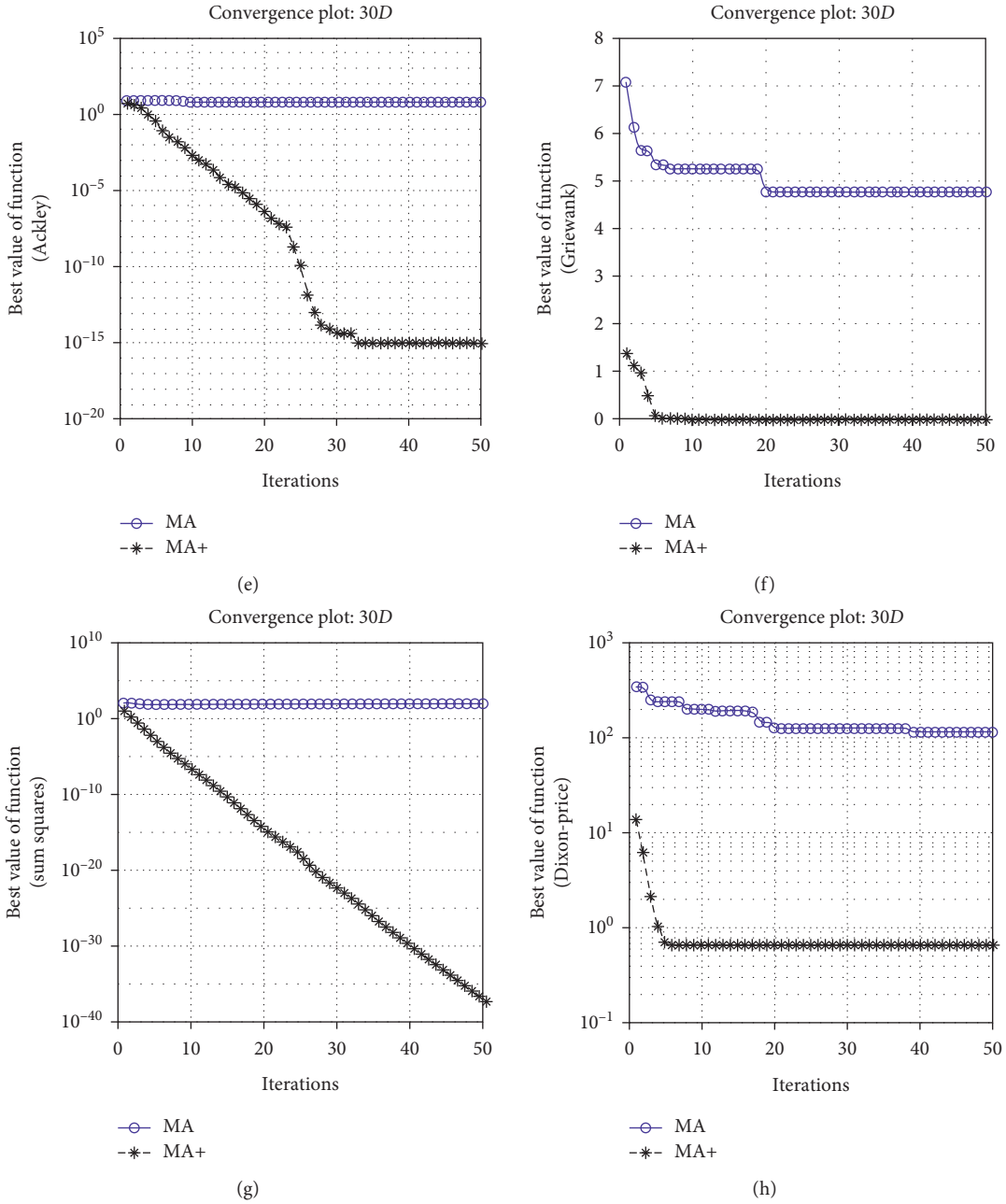

Figure 2: Continued.

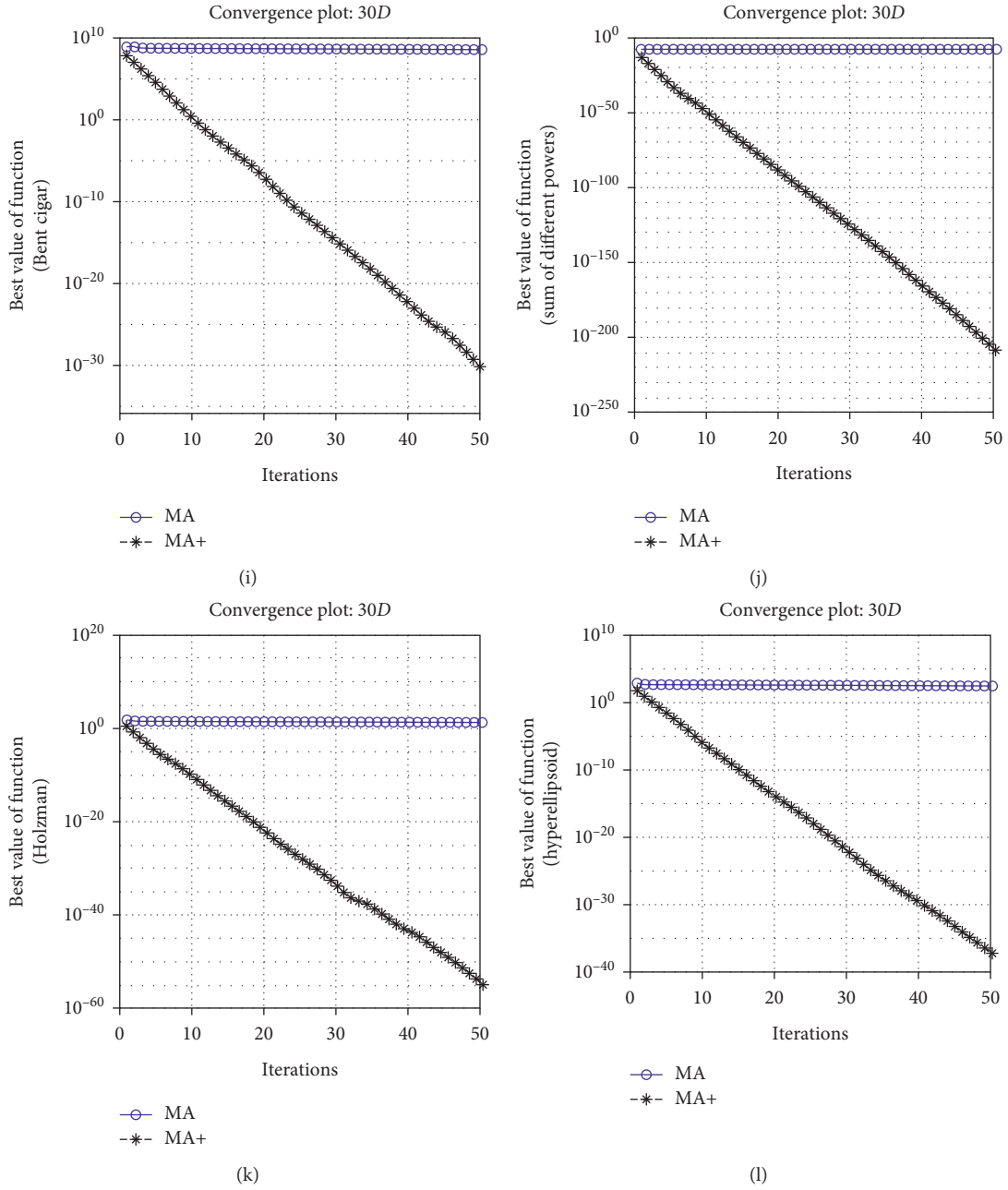

Figure 2: Convergence plot of both algorithms (MA and MA+) against 12 30-dimensional functions. (a) $F1$. (b) $F2$. (c) $F3$. (d) $F4$. (e) $F5$. (f) $F6$. (g) $F7$. (h) $F8$. (i) $F9$. (j) $F10$. (k) $F11$. (l) $F12$.

In addition, the performance of the proposed improved algorithm was obtained, $8.20E-29$, $8.35E-18$, $1.14E-24$, $5.71E+01$, $3.53E-14$, $0.00E+00$, $5.54E-28$, $6.67E-01$, $5.59E-21$, $9.3E-190$, $4.49E-31$, and $1.35E-26$, respectively, against 12 60-dimensional functions. Finally, the performance of the proposed algorithm was obtained, $1.16E-23$, $2.52E-14$, $1.45E-19$, $8.71E+01$, $1.33E-11$, $0.00E+00$, $2.20E-21$, $6.67E-01$, $6.76E-15$, $1.3E-165$, $2.90E-23$, and $1.97E-20$, respectively, against 12 90-dimensional functions.

3.3. Comparison of MA+ with Metaheuristic Algorithms on Different Dimensions. The improved monkey algorithm (MA+) was compared with many metaheuristic optimization algorithms against 12 benchmark optimization functions on different dimensions. The information of benchmark functions is listed in Table 1. All algorithms have the same information and use the same initial parameters, the same dimensions, and the same number of iterations, and their maximum function evaluation times are equal

Table 2: The performances of both algorithms on 30-, 60-, and 90-dimensional benchmark functions.

F	D	MA			MA+		
		Best	Mean	STD	Best	Mean	STD
$F1$	30	$8.61E-01$	$9.90E-01$	$6.35E-02$	$\mathbf{9.95E-112}$	$\mathbf{1.03E-40}$	$\mathbf{1.69E-40}$
	60	$8.40E+00$	$9.30E+00$	$5.40E-01$	$\mathbf{7.79E-79}$	$\mathbf{8.20E-29}$	$\mathbf{1.13E-28}$
	90	$3.10E+01$	$3.49E+01$	$1.65E+00$	$\mathbf{1.70E-72}$	$\mathbf{1.16E-23}$	$\mathbf{1.51E-23}$
$F2$	30	$7.85E+00$	$9.25E+00$	$7.70E-01$	$\mathbf{2.05E-89}$	$\mathbf{1.09E-23}$	$\mathbf{1.18E-23}$
	60	$4.35E+01$	$6.35E+01$	$3.61E+01$	$\mathbf{6.60E-65}$	$\mathbf{8.35E-18}$	$\mathbf{7.35E-18}$
	90	$2.44E+02$	$7.32E+09$	$1.39E+10$	$\mathbf{1.40E-49}$	$\mathbf{2.52E-14}$	$\mathbf{3.71E-14}$
$F3$	30	$5.15E+03$	$5.75E+03$	$4.75E+02$	$\mathbf{1.10E-122}$	$\mathbf{1.22E-36}$	$\mathbf{1.00E-36}$
	60	$1.03E+05$	$1.10E+05$	$5.65E+03$	$\mathbf{3.09E-90}$	$\mathbf{1.14E-24}$	$\mathbf{1.07E-24}$
	90	$5.20E+05$	$5.75E+05$	$3.09E+04$	$\mathbf{3.16E-62}$	$\mathbf{1.45E-19}$	$\mathbf{1.80E-19}$
$F4$	30	$9.82E+03$	$1.23E+04$	$2.00E+03$	$\mathbf{2.69E+01}$	$\mathbf{2.69E+01}$	$\mathbf{2.04E-01}$
	60	$3.99E+05$	$5.29E+05$	$7.25E+04$	$\mathbf{5.70E+01}$	$\mathbf{5.71E+01}$	$\mathbf{5.74E-02}$
	90	$4.10E+06$	$4.79E+06$	$5.59E+05$	$\mathbf{8.70E+01}$	$\mathbf{8.71E+01}$	$\mathbf{5.40E-02}$
$F5$	30	$5.20E+00$	$5.75E+00$	$3.02E-01$	$\mathbf{8.88E-16}$	$\mathbf{9.55E-15}$	$\mathbf{9.70E-15}$
	60	$9.19E+00$	$9.40E+00$	$1.81E-01$	$\mathbf{8.88E-16}$	$\mathbf{3.53E-14}$	$\mathbf{5.40E-14}$
	90	$1.21E+01$	$1.26E+01$	$2.09E-01$	$\mathbf{4.44E-15}$	$\mathbf{1.33E-11}$	$\mathbf{2.69E-11}$
$F6$	30	$3.20E+00$	$3.75E+00$	$6.29E-01$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$
	60	$3.19E+01$	$3.36E+01$	$1.75E+00$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$
	90	$1.09E+02$	$1.16E+02$	$5.35E+00$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$
$F7$	30	$5.0E+01$	$5.6E+01$	$4.19E+00$	$\mathbf{1.2E-141}$	$\mathbf{4.68E-38}$	$\mathbf{4.70E-38}$
	60	$1.02E+03$	$1.09E+03$	$4.88E+01$	$\mathbf{1.15E-92}$	$\mathbf{5.54E-28}$	$\mathbf{8.75E-28}$
	90	$5.32E+03$	$5.86E+03$	$3.03E+02$	$\mathbf{4.30E-55}$	$\mathbf{2.20E-21}$	$\mathbf{1.53E-21}$
$F8$	30	$8.09E+01$	$1.09E+02$	$2.09E+01$	$\mathbf{6.67E-01}$	$\mathbf{6.67E-01}$	$\mathbf{2.55E-05}$
	60	$5.59E+03$	$7.65E+03$	$9.22E+02$	$\mathbf{6.67E-01}$	$\mathbf{6.67E-01}$	$\mathbf{4.70E-05}$
	90	$1.03E+05$	$1.13E+05$	$5.76E+03$	$\mathbf{6.67E-01}$	$\mathbf{6.67E-01}$	$\mathbf{1.25E-04}$
$F9$	30	$3.30E+08$	$3.70E+08$	$2.97E+07$	$\mathbf{1.25E-144}$	$\mathbf{1.40E-31}$	$\mathbf{1.75E-31}$
	60	$2.70E+09$	$3.34E+09$	$3.92E+08$	$\mathbf{9.20E-115}$	$\mathbf{5.59E-21}$	$\mathbf{5.40E-21}$
	90	$1.18E+10$	$1.32E+10$	$8.63E+08$	$\mathbf{3.29E-65}$	$\mathbf{6.76E-15}$	$\mathbf{7.40E-15}$
$F10$	30	$3.05E-10$	$9.75E-09$	$7.80E-09$	$\mathbf{0.00E+00}$	$\mathbf{1.9E-196}$	$\mathbf{0.00E+00}$
	60	$2.12E-08$	$5.73E-08$	$3.68E-08$	$\mathbf{0.00E+00}$	$\mathbf{9.3E-190}$	$\mathbf{0.00E+00}$
	90	$1.01E-08$	$1.35E-07$	$7.80E-08$	$\mathbf{0.00E+00}$	$\mathbf{1.3E-165}$	$\mathbf{0.00E+00}$
$F11$	30	$1.35E+01$	$1.75E+01$	$2.39E+00$	$\mathbf{5.25E-165}$	$\mathbf{3.49E-54}$	$\mathbf{5.20E-54}$
	60	$1.59E+03$	$1.65E+03$	$6.02E+01$	$\mathbf{2.49E-82}$	$\mathbf{4.49E-31}$	$\mathbf{1.07E-30}$
	90	$2.19E+04$	$2.50E+04$	$1.69E+03$	$\mathbf{2.59E-65}$	$\mathbf{2.90E-23}$	$\mathbf{3.65E-23}$
$F12$	30	$2.10E+02$	$2.56E+02$	$3.50E+01$	$\mathbf{1.00E-124}$	$\mathbf{1.15E-37}$	$\mathbf{1.67E-37}$
	60	$8.50E+03$	$1.06E+04$	$1.05E+03$	$\mathbf{5.33E-97}$	$\mathbf{1.35E-26}$	$\mathbf{1.53E-26}$
	90	$7.92E+04$	$8.61E+04$	$3.69E+03$	$\mathbf{2.88E-65}$	$\mathbf{1.97E-20}$	$\mathbf{2.91E-20}$

[30, 42]. The best experimental comparative results are marked in bold for each function and all details are shown in Tables 3–6.

Tables 3 to 5 show the experimental comparative results for all functions ($F1$–$F12$) on the 30, 60, and 90 dimensions. Each algorithm is run 100 times independently for all 12 benchmark optimization functions on all these dimensions in these tables. However, Table 6 shows the experimental comparative results for some functions ($F1$–$F5$) on 20, 50, and 100 dimensions. Each algorithm is run 30 times independently for each of 5 benchmark optimization functions on all these dimensions in Table 6.

The initial parameters and the same conditions for all algorithms were set as follows: size of population is set to 50 and each algorithm ran till it reached a number of iterations (50).

In the first stage, the performance of the proposed algorithm is compared with the performances of a selected collection of comparative algorithms that have been evaluated. The included algorithms are ABC, DA, and HAD. The best mean and the best standard deviation of experimental results are shown in Table 3 for each function. Table 3 shows that the proposed algorithm has an outstanding performance in the majority of the evaluation cases for $F1$–$F7$ and $F9$–$F12$ benchmark functions, respectively. However, the performance of HAD algorithm is equal to the result in the case of all dimensions on only Dixon-Price function ($F8$) with the proposed algorithm. The best standard deviation on Dixon-Price function was obtained by HAD algorithm. All details are shown in Table 3.

In the second stage, the performance of the proposed algorithm is compared with those of some metaheuristic optimization algorithms that have been evaluated. The included algorithms are ACO, BAT, BBO, DE, GA, and PSO. The best mean of experimental results is listed in Table 4 for each function.

Table 3: The performance of MA + compared with ABC, DA, and HAD algorithms on 30-, 60-, and 90-dimensional benchmark functions.

The best mean and the best standard deviation for comparing the performances of some metaheuristic algorithms (ABC, DA, and HAD) after 100 runs									
F	D	ABC		DA		HAD		MA+	
		Mean	STD	Mean	STD	Mean	STD	Mean	STD
$F1$	30	$2.98E+01$	$1.02E+01$	$1.76E+00$	$4.34E+00$	$2.61E-14$	$6.53E-14$	$\mathbf{1.03E-40}$	$\mathbf{1.69E-40}$
	60	$1.97E+02$	$2.30E+01$	$2.29E+00$	$4.68E+00$	$3.46E-10$	$8.08E-10$	$\mathbf{8.20E-29}$	$\mathbf{1.13E-28}$
	90	$4.07E+02$	$3.72E+01$	$8.53E+00$	$1.27E+01$	$2.05E-09$	$3.08E-09$	$\mathbf{1.16E-23}$	$\mathbf{1.51E-23}$
$F2$	30	$7.54E+00$	$1.19E+00$	$7.46E+00$	$5.38E+00$	$3.40E-08$	$6.41E-08$	$\mathbf{1.09E-23}$	$\mathbf{1.18E-23}$
	60	$6.19E+01$	$1.17E+01$	$2.80E+01$	$2.77E+01$	$4.66E-06$	$4.04E-06$	$\mathbf{8.35E-18}$	$\mathbf{7.35E-18}$
	90	$1.59E+02$	$2.97E+01$	$3.10E+01$	$2.50E+01$	$4.89E-05$	$4.85E-05$	$\mathbf{2.52E-14}$	$\mathbf{3.71E-14}$
$F3$	30	$3.82E+04$	$8.56E+03$	$1.24E+03$	$2.23E+03$	$4.31E-05$	$7.84E-05$	$\mathbf{1.22E-36}$	$\mathbf{1.00E-36}$
	60	$1.57E+05$	$3.43E+04$	$1.09E+04$	$9.04E+03$	$3.32E-05$	$5.79E-05$	$\mathbf{1.14E-24}$	$\mathbf{1.07E-24}$
	90	$3.31E+05$	$3.78E+04$	$4.48E+04$	$4.85E+04$	$3.92E-05$	$5.64E-05$	$\mathbf{1.45E-19}$	$\mathbf{1.80E-19}$
$F4$	30	$1.36E+07$	$1.14E+07$	$1.15E+05$	$3.47E+05$	$2.75E+01$	$3.29E-01$	$\mathbf{2.69E+01}$	$\mathbf{2.04E-01}$
	60	$2.09E+08$	$5.78E+07$	$5.71E+05$	$1.69E+06$	$5.77E+01$	$9.94E-02$	$\mathbf{5.70E+01}$	$\mathbf{5.74E-02}$
	90	$5.48E+08$	$8.12E+07$	$7.18E+05$	$1.59E+06$	$8.77E+01$	$1.52E-01$	$\mathbf{8.71E+01}$	$\mathbf{5.40E-02}$
$F5$	30	$1.70E+01$	$5.08E-01$	$1.76E+00$	$3.83E+00$	$5.89E-08$	$8.64E-08$	$\mathbf{9.55E-15}$	$\mathbf{9.70E-15}$
	60	$1.95E+01$	$1.62E-01$	$7.41E+00$	$3.91E+00$	$7.18E-06$	$9.71E-06$	$\mathbf{3.53E-14}$	$\mathbf{5.40E-14}$
	90	$2.00E+01$	$1.66E-01$	$7.00E+00$	$2.22E+00$	$3.08E-05$	$4.34E-05$	$\mathbf{1.33E-11}$	$\mathbf{2.69E-11}$
$F6$	30	$9.22E+01$	$4.22E+01$	$7.64E+00$	$1.79E+01$	$4.79E-12$	$1.20E-11$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$
	60	$6.93E+02$	$1.11E+02$	$1.74E+01$	$2.90E+01$	$8.68E-09$	$1.48E-08$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$
	90	$1.43E+03$	$1.07E+02$	$2.27E+01$	$5.99E+01$	$2.72E-07$	$4.38E-07$	$\mathbf{0.00E+00}$	$\mathbf{0.00E+00}$
$F7$	30	$1.32E+03$	$4.45E+02$	$5.26E+01$	$1.03E+02$	$2.01E-15$	$2.67E-15$	$\mathbf{4.68E-38}$	$\mathbf{4.70E-38}$
	60	$2.04E+04$	$5.32E+03$	$8.19E+02$	$2.27E+03$	$1.18E-09$	$3.50E-09$	$\mathbf{5.54E-28}$	$\mathbf{8.75E-28}$
	90	$7.00E+04$	$6.98E+03$	$2.23E+03$	$4.78E+03$	$8.01E-10$	$1.35E-09$	$\mathbf{2.20E-21}$	$\mathbf{1.53E-21}$
$F8$	30	$7.68E+04$	$7.07E+04$	$1.42E+03$	$3.91E+03$	$6.67E-01$	$\mathbf{8.29E-08}$	$6.67E-01$	$2.55E-05$
	60	$2.84E+06$	$9.97E+05$	$2.14E+04$	$5.99E+04$	$6.67E-01$	$\mathbf{1.22E-05}$	$6.67E-01$	$4.70E-05$
	90	$1.25E+07$	$1.73E+06$	$5.42E+04$	$1.30E+05$	$6.67E-01$	$\mathbf{1.23E-04}$	$6.67E-01$	$1.25E-04$
$F9$	30	$7.47E+09$	$2.77E+09$	$1.51E+08$	$3.15E+08$	$3.72E-12$	$7.07E-12$	$\mathbf{1.40E-31}$	$\mathbf{1.75E-31}$
	60	$7.19E+10$	$1.07E+10$	$1.23E+09$	$2.31E+09$	$3.11E-05$	$4.24E-05$	$\mathbf{5.59E-21}$	$\mathbf{5.40E-21}$
	90	$1.55E+11$	$1.30E+10$	$2.36E+09$	$4.40E+09$	$1.03E-02$	$2.23E-02$	$\mathbf{6.76E-15}$	$\mathbf{7.40E-15}$
$F10$	30	$7.62E-03$	$5.74E-03$	$1.35E-05$	$4.26E-05$	$4.40E-14$	$1.25E-13$	$\mathbf{1.9E-196}$	$\mathbf{0.00E+00}$
	60	$1.35E-01$	$6.36E-02$	$9.39E-06$	$2.57E-05$	$2.99E-11$	$9.21E-11$	$\mathbf{9.3E-190}$	$\mathbf{0.00E+00}$
	90	$3.98E-01$	$2.20E-01$	$1.80E-06$	$4.81E-06$	$5.91E-11$	$1.50E-10$	$\mathbf{1.3E-165}$	$\mathbf{0.00E+00}$
$F11$	30	$1.19E+04$	$1.18E+04$	$6.30E+01$	$1.97E+02$	$8.72E-18$	$1.12E-17$	$\mathbf{3.49E-54}$	$\mathbf{5.20E-54}$
	60	$8.35E+05$	$1.29E+05$	$4.42E+03$	$1.24E+04$	$5.34E-15$	$8.87E-15$	$\mathbf{4.49E-31}$	$\mathbf{1.07E-30}$
	90	$2.77E+06$	$4.33E+05$	$2.58E+04$	$7.14E+04$	$1.61E-14$	$3.13E-14$	$\mathbf{2.90E-23}$	$\mathbf{3.65E-23}$
$F12$	30	$2.36E+02$	$1.14E+02$	$4.98E+00$	$1.07E+01$	$5.16E-14$	$1.54E-13$	$\mathbf{1.15E-37}$	$\mathbf{1.67E-37}$
	60	$7.72E+03$	$1.09E+03$	$1.41E+02$	$1.73E+02$	$1.16E-10$	$1.38E-10$	$\mathbf{1.35E-26}$	$\mathbf{1.53E-26}$
	90	$3.85E+04$	$2.49E+03$	$6.56E+02$	$1.01E+03$	$2.09E-08$	$4.14E-08$	$\mathbf{1.97E-20}$	$\mathbf{2.91E-20}$

In the third stage, the performance of the proposed algorithm is compared with those of other metaheuristic optimization algorithms that have been evaluated. The included algorithms are EHO, KH, MFO, MSA, SCA, and WOA. The best mean of experimental results is listed in Table 5 for each function.

Finally, the performance of the proposed algorithm was also compared with the performances of three algorithms, namely, the monarch butterfly optimization (MBO) algorithm, MBO with greedy strategy and self-adaptive crossover operator (GCMBO), and MBO with opposition-based learning and random local perturbation (OPMBO) using five benchmark functions, and all details are listed in Table 6.

To sum up, the experimental comparative results showed reaching the much better solution and the best convergence performance of escaping local optimum for the proposed algorithm when it is compared with ACO, BAT, BBO, DE, GA, PSO, EHO, KH, MFO, MSA, SCA, WOA, MBO, GCMBO, and OPMBO. All those comparative results showed an outstanding performance of the proposed algorithm in the majority of the evaluation cases. All details are listed in Tables 4–6.

Table 4: The performance of MA + compared with ACO, BAT, BBO, DE, GA, and PSO algorithms on 30-, 60-, and 90-dimensional benchmark functions.

The best mean for comparing the performances of some metaheuristic algorithms on 30, 60, and 90 dimensions after 100 runs								
F	D	ACO	BAT	BBO	DE	GA	PSO	MA+
	30	$1.63E+02$	$1.67E+02$	$5.73E+00$	$2.79E+01$	$9.58E+01$	$5.12E+01$	**$1{,}03E-40$**
$F1$	60	$3.76E+02$	$3.91E+02$	$3.09E+01$	$1.74E+02$	$2.86E+02$	$2.13E+02$	**$8.20E-29$**
	90	$6.02E+02$	$6.19E+02$	$7.44E+01$	$3.80E+02$	$4.65E+02$	$4.29E+02$	**$1.16E-23$**
	30	$1.13E+02$	$2.95E+12$	$1.19E+01$	$5.38E+01$	$8.60E+01$	$1.14E+02$	**$1.09E-23$**
$F2$	60	$2.48E+02$	$2.29E+28$	$4.65E+01$	$1.71E+02$	$2.03E+02$	$2.49E+02$	**$8.35E-18$**
	90	$3.88E+02$	$6.75E+43$	$9.55E+01$	$2.97E+02$	$3.23E+02$	$3.89E+02$	**$2.52E-14$**
	30	$6.01E+04$	$1.28E+05$	$2.72E+04$	$6.13E+04$	$4.83E+04$	$1.90E+05$	**$1.22E-36$**
$F3$	60	$2.52E+05$	$4.84E+05$	$1.11E+05$	$2.38E+05$	$1.85E+05$	$8.30E+05$	**$1.14E-24$**
	90	$5.61E+05$	$1.10E+06$	$2.41E+05$	$5.40E+05$	$4.13E+05$	$2.18E+06$	**$1.45E-19$**
	30	$1.06E+08$	$2.32E+08$	$8.62E+05$	$1.59E+07$	$4.09E+07$	$2.25E+07$	**$2.69E+01$**
$F4$	60	$5.87E+08$	$5.97E+08$	$1.11E+07$	$2.12E+08$	$3.24E+08$	$2.11E+08$	**$5.70E+01$**
	90	$1.00E+09$	$9.99E+08$	$3.96E+07$	$5.81E+08$	$6.66E+08$	$7.39E+08$	**$8.71E+01$**
	30	$1.85E+01$	$1.99E+01$	$8.82E+00$	$1.87E+01$	$1.77E+01$	$1.87E+01$	**$9.55E-15$**
$F5$	60	$1.90E+01$	$1.99E+01$	$1.18E+01$	$1.90E+01$	$1.86E+01$	$1.90E+01$	**$3.53E-14$**
	90	$1.91E+01$	$1.99E+01$	$1.37E+01$	$1.91E+01$	$1.88E+01$	$1.91E+01$	**$1.33E-11$**
	30	$8.57E+01$	$5.77E+02$	$2.12E+01$	$9.38E+01$	$1.27E+02$	$1.69E+02$	**$0.00E+00$**
$F6$	60	$4.32E+02$	$1.33E+03$	$1.09E+02$	$6.02E+02$	$4.64E+02$	$7.27E+02$	**$0.00E+00$**
	90	$7.13E+02$	$2.14E+03$	$2.51E+02$	$1.31E+03$	$8.87E+02$	$1.62E+03$	**$0.00E+00$**
	30	$9.37E+03$	$9.27E+03$	$3.12E+02$	$1.29E+03$	$5.03E+03$	$2.30E+03$	**$4.68E-38$**
$F7$	60	$4.39E+04$	$4.29E+04$	$3.09E+03$	$1.65E+04$	$3.05E+04$	$1.62E+04$	**$5.54E-28$**
	90	$1.03E+05$	$1.03E+05$	$1.15E+04$	$5.51E+04$	$8.04E+04$	$5.06E+04$	**$2.20E-21$**
	30	$1.65E+06$	$1.66E+06$	$7.07E+03$	$9.65E+04$	$3.79E+05$	$2.28E+05$	**$6.67E-01$**
$F8$	60	$8.84E+06$	$8.63E+06$	$1.54E+05$	$2.54E+06$	$4.67E+06$	$2.58E+06$	**$6.67E-01$**
	90	$2.19E+07$	$2.15E+07$	$7.95E+05$	$1.03E+07$	$1.34E+07$	$9.36E+06$	**$6.67E-01$**
	30	$3.26E+10$	$6.09E+10$	$2.14E+09$	$9.29E+09$	$1.82E+10$	$1.55E+10$	**$1.40E-31$**
$F9$	60	$8.60E+10$	$1.46E+11$	$1.11E+10$	$6.46E+10$	$8.06E+10$	$5.24E+10$	**$5.59E-21$**
	90	$1.32E+11$	$2.37E+11$	$2.60E+10$	$1.42E+11$	$1.56E+11$	$1.02E+11$	**$6.76E-15$**
	30	$8.84E+00$	$7.17E-01$	$6.15E-34$	$1.81E-02$	$8.20E-01$	$7.49E-01$	**$1.9E-196$**
$F10$	60	$2.11E+01$	$1.00E+00$	$1.35E-06$	$4.62E-01$	$9.82E+00$	$1.93E+00$	**$9.3E-190$**
	90	$3.42E+01$	$1.26E+00$	$2.44E-05$	$2.25E+00$	$2.16E+01$	$2.97E+00$	**$1.3E-165$**
	30	$4.19E+05$	$4.19E+05$	$1.60E+03$	$2.51E+04$	$1.02E+05$	$6.73E+04$	**$3.49E-54$**
$F11$	60	$2.24E+06$	$2.13E+06$	$3.98E+04$	$6.43E+05$	$1.16E+06$	$9.31E+05$	**$4.49E-31$**
	90	$5.50E+06$	$5.24E+06$	$1.85E+05$	$2.65E+06$	$3.39E+06$	$3.19E+06$	**$2.90E-23$**
	30	$2.34E+03$	$1.76E+03$	$1.61E-02$	$1.75E+02$	$1.20E+02$	$2.79E+02$	**$1.15E-37$**
F12	60	$2.28E+04$	$1.66E+04$	$2.11E+01$	$4.88E+03$	$7.93E+03$	$3.44E+03$	**$1.35E-26$**
	90	$8.42E+04$	$5.92E+04$	$3.99E+02$	$2.61E+04$	$4.28E+04$	$1.48E+04$	**$1.97E-20$**

Table 5: The performance of MA + compared with EHO, KH, MFO, MSA, SCA, and WOA algorithms on 30-, 60-, and 90-dimensional benchmark functions.

The best mean optimization results for comparing the performances of other metaheuristic algorithms with $D = 30$, 60, and 90 after 100 runs								
F	D	EHO	KH	MFO	MSA	SCA	WOA	MA+
	30	$2.49E-07$	$4.63E-01$	$6.57E+01$	$2.30E-08$	$2.32E+01$	$2.42E-09$	**$1.03E-40$**
$F1$	60	$6.44E-07$	$4.75E+00$	$2.70E+02$	$3.67E-07$	$1.15E+02$	$4.67E-09$	**$8.20E-29$**
	90	$1.06E-06$	$9.04E+00$	$4.97E+02$	$1.17E-06$	$2.26E+02$	$7.45E-09$	**$1.16E-23$**
	30	$4.12E-03$	$1.14E+01$	$4.66E+02$	$1.78E-04$	$1.52E+01$	$3.76E-05$	**$1.09E-23$**
$F2$	60	$9.34E-03$	$2.45E+14$	$1.13E+17$	$9.50E-04$	$4.20E+01$	$9.19E-05$	**$8.35E-18$**
	90	$1.46E02$	$3.56E+27$	$1.37E+32$	$1.63E-03$	$6.81E+01$	$1.52E-04$	**$2.52E-14$**
	30	$2.30E-04$	$6.71E+03$	$4.61E+04$	$1.59E-07$	$4.07E+04$	$2.25E+02$	**$1.22E-36$**
$F3$	60	$9.56E-04$	$1.05E+05$	$1.79E+05$	$7.08E-06$	$1.74E+05$	$1.33E+03$	**$1.14E-24$**
	90	$2.08E-03$	$2.46E+05$	$3.84E+05$	$4.86E-05$	$4.14E+05$	$2.25\,E+03$	**$1.45E-19$**
	30	$2.89E+01$	$8.69E+03$	$4.74E+07$	$2.86E+01$	$3.30E+07$	$2.87E+01$	**$2.69E+01$**
$F4$	60	$5.89E+01$	$2.28E+04$	$3.61E+08$	$5.89E+01$	$2.25E+08$	$5.85E+01$	**$5.70E+01$**
	90	$8.89E+01$	$3.85E+04$	$7.55E+08$	$8.89E+01$	$5.21E+08$	$8.83E+01$	**$8.71E+01$**

Table 5: Continued.

The best mean optimization results for comparing the performances of other metaheuristic algorithms with $D = 30$, 60, and 90 after 100 runs								
F	D	EHO	KH	MFO	MSA	SCA	WOA	MA+
	30	$1.94E-03$	$4.84E+00$	$1.85E+01$	$6.84E-05$	$1.59E+01$	$1.21E-04$	$\mathbf{9.55E-15}$
$F5$	60	$2.22E-03$	$7.16E+00$	$2.01E+01$	$1.88E-04$	$1.89E+01$	$1.19E-04$	$\mathbf{3.53E-14}$
	90	$2.33E-03$	$7.80E+00$	$2.04E+01$	$3.57E-04$	$1.83E+01$	$1.41E-04$	$\mathbf{1.33E-11}$
	30	$1.44E-04$	$3.03E+00$	$2.28E+02$	$1.09E-09$	$8.46E+01$	$6.10E-02$	$\mathbf{0.00E+00}$
$F6$	60	$2.14E-04$	$6.83E+00$	$9.28E+02$	$7.15E-09$	$4.15E+02$	$3.88E-02$	$\mathbf{0.00E+00}$
	90	$2.58E-04$	$7.77E+00$	$1.70E+03$	$3.90E-08$	$8.37E+02$	$3.47E-02$	$\mathbf{0.00E+00}$
	30	$1.16E-05$	$4.21E+01$	$3.09E+03$	$1.95E-07$	$1.13E+03$	$1.30E-07$	$\mathbf{4.68E-38}$
$F7$	60	$6.21E-05$	$5.33E+02$	$2.63E+04$	$1.24E-05$	$1.10E+04$	$7.04E-07$	$\mathbf{5.54E-28}$
	90	$1.55E-04$	$1.55E+03$	$7.70E+04$	$4.37E-05$	$3.34E+04$	$9.83E-07$	$\mathbf{2.20E-21}$
	30	$9.51E-01$	$4.69E+01$	$3.21E+05$	$6.71E-01$	$2.10E+05$	$7.59E-01$	$\mathbf{6.67E-01}$
$F8$	60	$9.89E-01$	$6.05E+02$	$4.55E+06$	$7.88E-01$	$2.98E+06$	$9.29E-01$	$\mathbf{6.67E-01}$
	90	$9.95E-01$	$1.86E+03$	$1.45E+07$	$9.95E-01$	$1.03E+07$	$9.73E-01$	$\mathbf{6.67E-01}$
	30	$7.35E+01$	$2.16E+06$	$2.23E+10$	$1.74E-02$	$8.21E+09$	$2.21E+00$	$\mathbf{1.40E-31}$
$F9$	60	$1.94E+02$	$7.53E+08$	$1.01E+11$	$3.61E-01$	$4.19E+10$	$1.52E+00$	$\mathbf{5.59E-21}$
	90	$3.19E+02$	$3.25E+09$	$1.88E+11$	$1.52E+00$	$8.77E+10$	$2.76E+00$	$\mathbf{6.76E-15}$
	30	$4.29E-12$	$-7.30E+02$	$4.82E-02$	$9.39E-15$	$8.48E-02$	$7.48E-17$	$\mathbf{1.9E-196}$
$F10$	60	$5.48E-12$	$-1.34E+03$	$2.30E-01$	$2.12E-14$	$4.85E-01$	$5.51E-16$	$\mathbf{9.3E-190}$
	90	$5.40E-12$	$-1.85E+03$	$4.53E-01$	$1.13E-14$	$8.17E-01$	$7.91E-17$	$\mathbf{1.3E-165}$
	30	$8.13E-13$	$-2.21E+08$	$8.48E+04$	$2.46E-14$	$4.50E+04$	$5.38E-11$	$\mathbf{3.49E-54}$
$F11$	60	$5.90E-12$	$-6.49E+12$	$1.14E+06$	$6.19E-12$	$7.49E+05$	$2.28E-12$	$\mathbf{4.49E-31}$
	90	$1.59E-11$	$-1.65E+17$	$3.76E+06$	$7.59E-11$	$2.36E+06$	$9.29E-10$	$\mathbf{2.90E-23}$
	30	$4.21E-06$	$4.27E+00$	$4.60E+02$	$3.33E-06$	$1.55E+02$	$2.85E-08$	$\mathbf{1.15E-37}$
$F12$	60	$4.57E-05$	$1.96E+02$	$8.62E+03$	$1.44E-04$	$3.33E+03$	$3.31E-07$	$\mathbf{1.35E-26}$
	90	$1.82E-04$	$9.54E+02$	$3.97E+04$	$9.07E-04$	$1.59E+04$	$1.89E-06$	$\mathbf{1.97E-20}$

Table 6: The performance of MA + compared with MBO, GCMBO, and OPMBO algorithms on 20-, 50-, and 100-dimensional benchmark functions.

The mean and standard deviations obtained by the MBO, GCMBO, OPMBO, and MA + on the test optimization functions after 30 runs									
F	D	MBO		GCMBO		OPMBO		MA+	
		Mean	STD	Mean	STD	Mean	STD	Mean	STD
	20	$1.02E+01$	$2.41E+01$	$4.03E-09$	$6.14E-09$	$1.99E-10$	$5.68E-10$	$\mathbf{6.67E-52}$	$\mathbf{8.17E-52}$
$F1$	50	$2.09E+02$	$1.39E+02$	$1.11E-09$	$2.15E-09$	$8.85E-09$	$7.50E-09$	$\mathbf{2.40E-32}$	$\mathbf{3.00E-32}$
	100	$4.83E+02$	$3.27E+02$	1.00E+01	$4.01E+01$	$4.87E+00$	$2.52E+01$	$\mathbf{1.70E-22}$	$\mathbf{1.94E-22}$
	20	$1.95E+01$	$2.47E+01$	$2.27E+00$	$5.83E+00$	$1.72E-06$	$1.80E-06$	$\mathbf{8.80E-29}$	$\mathbf{1.10E-28}$
F2	50	$1.30E+02$	$7.90E+01$	$2.36E+01$	$3.56E+01$	$2.73E-05$	$1.65E-05$	$\mathbf{1.99E-20}$	$\mathbf{3.92E-20}$
	100	$2.66E+02$	$1.75E+02$	$5.61E+01$	$7.16E+01$	$7.60E-02$	$4.13E-01$	$\mathbf{1.90E-13}$	$\mathbf{1.65E-13}$
	20	$9.91E+03$	$7.05E+03$	$5.47E+03$	$3.63E+03$	$1.29E+01$	$2.54E+01$	$\mathbf{4.04E-45}$	$\mathbf{4.67E-45}$
$F3$	50	$7.17E+04$	$5.12E+04$	$2.81E+04$	$1.75E+04$	$1.98E+03$	$8.64E+03$	$\mathbf{1.39E-27}$	$\mathbf{1.45E-27}$
	100	$3.63E+05$	$1.78E+05$	$1.12E+05$	$6.38E+04$	$9.42E+03$	$2.84E+04$	$\mathbf{9.10E-18}$	$\mathbf{1.12E-17}$
	20	$6.36E+02$	$1.01E+03$	$7.10E+01$	$9.60E+01$	$1.41E+01$	$7.96E+00$	$\mathbf{1.65E+01}$	$\mathbf{2.90E-01}$
$F4$	50	$5.93E+03$	$6.24E+03$	$2.43E+02$	$3.53E+02$	$4.60E+01$	$5.35E+01$	$\mathbf{4.72E+01}$	$\mathbf{1.19E-01}$
	100	$1.35E+04$	$1.43E+04$	$1.00E+03$	$2.06E+03$	$4.32E+02$	$1.25E+03$	$\mathbf{9.75E+01}$	$\mathbf{4.20E-01}$
	20	$8.92E+00$	$7.96E+00$	$2.61E-01$	$1.08E+00$	$6.94E-06$	$5.00E-06$	$\mathbf{8.59E-15}$	$\mathbf{8.69E-15}$
$F5$	50	$1.59E+01$	$6.11E+00$	$1.96E+00$	$4.66E+00$	$4.03E-05$	$2.24E-05$	$\mathbf{2.80E-14}$	$\mathbf{1.95E-14}$
	100	$1.86E+01$	$3.99E+00$	$8.63E+00$	$9.13E+00$	$3.93E-02$	$1.89E-01$	$\mathbf{4.03E-10}$	$\mathbf{7.08E-11}$

4. Conclusions

This paper presented a novel metaheuristic search and cognitively inspired algorithm, based on the monkey algorithm. The proposed algorithm has been widely employed for solving various kinds of optimization problems and was evaluated extensively against 12 benchmark optimization functions on different types of dimensions for each function. A new random perturbation was defined to improve the control parameters and was constructed of the proposed algorithm. The main advantage of the control parameters was that they efficiently prevented the improved monkey algorithm from getting stuck in optimal solutions and found global optimal solution for 8 benchmark functions, namely,

sphere function ($F1$), Schwefel 2.22 function ($F2$), Schwefel 1.2 function ($F3$), sum squares function ($F7$), Bent Cigar function ($F9$), sum of different powers function ($F10$), Holzman function ($F11$), and hyperellipsoid function ($F12$) in Figures 2(a)–2(c), 2(g), and 2(i)–2(l), respectively. Moreover, Ackley function ($F5$) in Figure 2(e) and Griewank function ($F6$) in Figure (2f) early obtained $8.88E-16$ and $0.00E+00$, respectively, for the lowest iterations. These experimental results are the best solution for these functions, and they also reached global optimum solutions early without getting stuck in local optimum solutions. However, Rosenbrock function ($F4$) in Figure (2d) and Dixon-Price function ($F8$) in Figure (2h) caused getting stuck in optimal solutions and obtained poor performance for only the lowest iterations, but of course the proposed algorithm obtained the best solution for these functions at the maximum allowable number of iterations. Briefly, the search strategy in the proposed algorithm has more generally proven to have a successful global optimal solution, convergence optimal solution, and much better performance on many optimization problems for the lowest number of iterations against the original monkey algorithm.

The performance of the improved monkey algorithm was compared with many metaheuristic optimization algorithms, including a collection of 18 optimizer algorithms. The comparative results included simple statistics for the best, mean, and convergence plots. All those comparative results showed that the proposed algorithm had an outstanding performance in majority of the evaluation cases.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The author declares that he has no conflicts of interest.

Acknowledgments

The author would like to acknowledge Faculty of Engineering and Architecture, Department of Computer Engineering, Istanbul Gelisim University, Avcılar-Istanbul, Turkey.

References

[1] R. H. Abiyev and M. Tunay, "Optimization of high dimensional functions through hypercube evaluation," *Computational Intelligence and Neuroscience*, vol. 2015, Article ID 967320, 11 pages, 2015.

[2] M. Tunay, "Evolutionary search algorithm based on hypercube optimization for high-dimensional functions," *International Journal of Computational and Experimental Science and Engineering (IJCESEN)*, vol. 6, no. 1, pp. 42–62, 2020.

[3] R. H. Abiyev and M. Tunay, "Optimization search using hypercubes," in *Proceedings of the 2020 4th International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, pp. 1–8, Istanbul, Turkey, October 2020.

[4] L. Marini, B. Morini, and M. Porcelli, "Quasi-Newton methods for constrained nonlinear systems: complexity analysis and applications," *Computational Optimization and Applications*, vol. 71, no. 1, pp. 147–170, 2018.

[5] S. F. Husin, M. Mamat, and M. A. H. Ibrahim, "A modification of steepest descent method for solving large-scaled unconstrained optimization problems," *International Journal of Engineering & Technology*, vol. 7, no. 3, pp. 72–75, 2018.

[6] O. O. Kryazhych, O. M. Trofymchuk, and O. V. Kovalenko, "The algorithm for determining the starting point in the simulation by the method of possible directions," *Radio Electronics, Computer Science, Control*, vol. 3, pp. 40–46, 2019.

[7] S. Wang, E. Roosta-Khorasani, P. Xu, and M. W. Mahoney, "Giant: globally improved approximate Newton method for distributed optimization," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 2332–2342, Montreal, Canada, December 2018.

[8] M. Tunay, "A new intense stochastic search method based on hypercube evaluation for examination timetabling problems," in *Proceedings of the 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, pp. 1–5, Istanbul, Turkey, June 2020.

[9] C. Han, L. Ming, and Z. Dinghua, "Optimization of varying-parameter drilling for multi-hole parts using metaheuristic algorithm coupled with self-adaptive penalty method," *Applied Soft Computing*, vol. 95, p. 106489, 2020.

[10] M. A. Akbay, C. B. Kalayci, and O. Polat, "A parallel variable neighborhood search algorithm with quadratic programming for cardinality constrained portfolio optimization," *Knowledge-Based Systems*, vol. 198, p. 105944, 2020.

[11] U. M. Diwekar, *Introduction to Applied Optimization*, Vol. 22, Springer Nature, Basingstoke, UK, 2020.

[12] P. Kuendee and U. Janjarassuk, "A comparative study of mixed-integer linear programming and genetic algorithms for solving binary problems," in *Proceedings of the 2018 5th International Conference on Industrial Engineering and Applications (ICIEA)*, pp. 284–288, Singapore, April 2018.

[13] Z. Wang and H. Li, "A novel multi-objective evolutionary algorithm based on linear programming," in *Proceedings of the 2018 14th International Conference on Computational Intelligence and Security (CIS)*, pp. 345–348, Hangzhou, China, November 2018.

[14] S. Agarwal, A. P. Singh, and N. Anand, "Evaluation performance study of Firefly algorithm, particle swarm optimization and artificial bee colony algorithm for non-linear mathematical optimization functions," in *Proceedings of the 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pp. 1–8, Tiruchengode, India, July 2013.

[15] K. Schittkowski and C. Zillober, "Nonlinear programming: algorithms, software, and applications," *IFIP Advances in Information and Communication Technology*, vol. 166, pp. 73–107, 2006.

[16] N. Andreasson, M. Patriksson, and A. Evgrafov, *An Introduction to Continuous Optimization: Foundations and Fundamental Algorithms*, Courier Dover Publications, Garden City, NY, USA, 2020.

[17] R. W. Sebesta, *Concepts of Programming Languages*, Pearson, Boston, MA, USA, 2012.

[18] E. V. Denardo, *Dynamic Programming: Models and Applications*, Courier Corporation, Chelmsford, MA, USA, 2012.

[19] D. P. Bertsekas, *Abstract Dynamic Programming*, Athena Scientific, Nashua, NH, USA, 2018.

[20] M. Dorigo and T. Stützle, "Ant colony optimization: overview and recent advances," *Handbook of Metaheuristics*, Springer, Boston, MA, USA, .
[21] D. Ke-Lin and M. N. S. Swamy, "Ant colony optimization," *Search and Optimization by Metaheuristics*, pp. 191–199, Birkhäuser, Basel, Switzerland, 2016.
[22] S. Mirjalili, "The ant lion optimizer," *Advances in Engineering Software*, vol. 83, pp. 80–98, 2015.
[23] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Springer, Berlin, Germany, pp. 65–74, 2010.
[24] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, pp. 210–214, Coimbatore, India, December 2009.
[25] G.-G. Wang, S. Deb, and L. S. Coelho, "Elephant herding optimization," in *Proceedings of the 3rd International Symposium on Computational and Business Intelligence (ISCBI)*, pp. 1–5, Bali, Indonesia, December 2015.
[26] Q. Zhao and C. Li, "Two-stage multi-swarm particle swarm optimizer for unconstrained and constrained global optimization," *IEEE Access*, vol. 8, pp. 124905–124927, 2020.
[27] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.
[28] S. Mirjalili, "Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228–249, 2015.
[29] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Computing and Applications*, vol. 31, pp. 1995–2014, 2019.
[30] L. Sun, S. Chen, J. Xu, and Y. Tian, "Improved monarch butterfly optimization algorithm based on opposition-based learning and random local perturbation," *Complexity*, vol. 2019, Article ID 4182148, 20 pages, 2019.
[31] J. An, Q. Kang, L. Wang, and Q. Wu, "Mussels wandering optimization: an ecologically inspired algorithm for global optimization," *Cognitive Computation*, vol. 5, no. 2, pp. 188–199, 2013.
[32] G.-G. Wang, "Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems," *Memetic Computing*, vol. 10, pp. 151–164, 2018.
[33] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.
[34] L. Wee Loon, W. Antoni, D. Mohammad, and H. Habibollah, "A biogeography-based optimization algorithm hybridized with tabu search for the quadratic assignment problem," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 5803893, 12 pages, 2016.
[35] R. Storn and K. Price, ""Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[36] M. Emmerich, O. M. Shir, and H. Wang, "Evolution Strategies," in *Handbook of Heuristics*, R. Martí, P. Panos, and M. Resende, Eds., Springer, Berlin, Germany, pp. 1–31, 2018.
[37] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm-a literature review," in *Proceedings of the 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, pp. 380–384, Faridabad, India, February 2019.
[38] M. Tunay and R. H. Abiyev, "Hybrid local search based genetic algorithm and its practical application," *International Journal of Soft Computing and Engineering*, vol. 5, no. 2, pp. 21–27, 2015.
[39] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.
[40] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
[41] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.
[42] W. A. H. M. Ghanem and A. Jantan, "A cognitively inspired hybridization of artificial bee colony and dragonfly algorithms for training multi-layer perceptrons," *Cognitive Computation*, vol. 10, pp. 1096–1134, 2018.