Soft Comput
DOI 10.1007/s00500-017-2894-y

METHODOLOGIES AND APPLICATION

An efficient hybrid algorithm based on Water Cycle and Moth-Flame Optimization algorithms for solving numerical and constrained engineering optimization problems

Soheyl Khalilpourazari[1] · **Saman Khalilpourazary**[2]

© Springer-Verlag GmbH Germany 2017

Abstract This paper proposes a hybrid algorithm based on Water Cycle and Moth-Flame Optimization algorithms for solving numerical and constrained engineering optimization problems. The spiral movement of moths in Moth-Flame Optimization algorithm is introduced into the Water Cycle Algorithm to enhance its exploitation ability. In addition, to increase randomization in the new hybrid method, the streams in the Water Cycle Algorithm are allowed to update their position using a random walk (Levy flight). The random walk significantly improves the exploration ability of the Water Cycle Algorithm. The performance of the new hybrid Water Cycle–Moth-Flame Optimization algorithm (WCMFO) is investigated in 23 benchmark functions such as unimodal, multimodal and fixed-dimension multimodal benchmark functions. The results of the WCMFO are compared to the other state-of-the-art metaheuristic algorithms. The results show that the hybrid method is able to outperform the other state-of-the-art metaheuristic algorithms in majority of the benchmark functions. To evaluate the efficiency of the WCMFO in solving complex constrained engineering and real-life problems, three well-known structural engineering problems are solved using WCMFO and the results are compared with the ones of the other metaheuristics in the literature. The results of the simulations revealed that the WCMFO is able to provide very competitive and promising results comparing to the other hybrid and metaheuristic algorithms.

Keywords Numerical optimization · Water cycle algorithm · Moth-flame optimization · Constrained engineering optimization · Hybrid metaheuristic algorithm

Communicated by V. Loia.

✉ Soheyl Khalilpourazari
std_khalilpourazari@khu.ac.ir

Saman Khalilpourazary
s.khalilpour@mee.uut.ac.ir

[1] Department of Industrial Engineering, Faculty of Engineering, Kharazmi University, Tehran, Iran

[2] Department of Mechanical Engineering, Faculty of Engineering, Urmia University of Technology, Urmia, Iran

1 Introduction

Optimization is the procedure to find the best solution of a given problem among all feasible solutions (Khalilpourazari and Khalilpourazary 2016). Most of the traditional optimization methodologies are inefficient in solving today's complicated problems. Therefore, many researchers started to propose new solution techniques named metaheuristic algorithms to solve complex optimization problems in reasonable computation time and cost. In last decades, metaheuristics become more popular between researchers because they have several advantages over the traditional techniques. The first and most important advantage of the metaheuristic algorithm over the traditional optimization methods is that they don't need gradient information to solve optimization problems. The second one is that they are simple and easy to implement in different fields of study. Third, using their exploration ability, they can decrease the probability of trapping in local optima.

The basic concept behind most of the metaheuristic algorithms is inspired by the nature, animal behaviour or physical phenomena (Khalilpourazari and Khalilpourazary 2017a, b). Mirjalili and Lewis (2016) classify the metaheuristic algorithms in three main classes: evolutionary-based, physics-based and swarm-based techniques. Evolutionary-based methods mimic the evolution process in the nature to

Published online: 28 October 2017

Table 1 Classification of the Metaheuristic Algorithms

Metaheuristic Algorithms			
Evolutionary Algorithms	Physics-Based Algorithms	Swarm-Based Algorithms	Other Population-Based Algorithms
Genetic Algorithm (GA) (Holland 1992)	Simulated Annealing (SA) (Kirkpatrick et al. 1983; Cerný 1985)	Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995)	Stochastic Fractal Search (SFS) (Salimi 2015)
Evolution Strategy (ES) (Rechenberg 1978)	Gravitational Search Algorithm (GSA) (Rashedi et al. 2009)	Grey Wolf Optimizer (GWO) (Mirjalili et al. 2014)	Sine Cosine Algorithm (SCA) (Mirjalili et al. 2016)
Genetic Programming (GP) (Koza 1992)	Charged System Search (CSS) (Kaveh and Talatahari 2010)	Dragonfly Algorithm (DA) (Mirjalili 2016a)	Water Cycle Algorithm (WCA) (Eskandar et al. 2012)]
Biogeography-Based Optimizer (BBO) (Simon 2008)	Central Force Optimization (CFO) (Formato 2007)	Moth-Flame Optimization (MFO) (Mirjalili 2015a, b)	Thermal Exchange Optimization (Kaveh and Dadras 2017)
Evolutionary Programming (EP) (Fogel et al. 1966)	Black Hole (BH) algorithm (Hatamlou 2013)	Whale Optimization Algorithm (WOA) (Mirjalili and Lewis 2016)	Virus Colony Search (Li et al. 2016)
	Galaxy-Based Search Algorithm (GBSA) (Kaveh and Khayatazad 2012)	Artificial Bee Colony (ABC) (Karaboga and Basturk 2007)	
	Small-World Optimization Algorithm (SWOA) (Du et al. 2006)]	Ant Lion Optimization Algorithm (ALO) (Mirjalili 2015a, b)	
	Curved Space Optimization (CSO) (Moghaddam et al. 2012).	Cuckoo Search (CS) (Yang and Deb 2009)	
	Multiverse Optimization (MVO) Algorithm (Mirjalili et al. 2016)	Crow Search Algorithm (CSA) (Askarzadeh 2016)	
		Grasshopper Optimization Algorithm (Saremi et al. 2017)	
		Dynamic Virtual Bats Algorithm (Topal and Altun 2016)	

perform optimization (Khalilpourazari and Khamseh 2017), while physics-based algorithms perform optimization using the rules of physics in the universe. The third class of the metaheuristic algorithms is swarm-based techniques, which mimic the behaviour of animals in a group. Table 1 presents a schematic view of the classification of the metaheuristic algorithms.

As illustrated earlier, many metaheuristic algorithms are proposed to solve today's complex problems. Although they can obtain promising solution for the optimization problems, they often get trapped in local optima when the problem is complex and has several local optima (Ali et al. 2016; Khalilpourazari and Pasandideh 2016; Khalilpourazari and Khalilpourazary 2016). Developing hybrid metaheuristic algorithms can significantly improve this problem. Most of the times, the hybrid metaheuristic algorithms are robust and efficient comparing to the basic versions of the algorithms which are hybridized because they can benefit from advantages of different metaheuristic algorithms (Ali et al. 2016; Gao et al. 2012; Khalilpourazari and Pasandideh 2016; Khalilpourazari and Khalilpourazary 2017a, b). Many researchers aimed to develop hybrid metaheuristic algorithms to create more efficient algorithm for global optimization. The most popular hybrid methods are hybrid Cultural-trajectory-based search (Ali et al. 2016), Big Bang-Big Crunch (BB-BC) algorithm (Erol and Eksin 2006), hybrid harmony search–cuckoo search (HS/CS) algorithm (Wang et al. 2016), hybrid krill herd-biogeography-based optimization (KHBBO) algorithm (Wang et al. 2014a, b, c), hybrid Tissue Membrane Systems (TMS) and CMA-ES (Liu and Linan 2016), krill herd-differential evolution (KHDE) (Wang et al. 2014a, b, c), Hybrid Bat Algorithm with Harmony Search (BHS) (Wang and Guo 2013).

Water Cycle Algorithm (WCA) is a novel nature inspired metaheuristic algorithm which mimics the water cycle in the nature to perform optimization. The WCA performs well in exploration phase, and it suffers from lack of an efficient exploitation operator. Conversely, the Moth-Flame Optimization (MFO) algorithm by mimicking the spiral fly of moths around artificial lights performs very well in exploiting the solution space, but occasionally traps in local optima. Therefore, the spiral movement behaviour of the MFO is introduced into the basic WCA to increase its ability in exploiting the solution space. Also, the streams in the WCA are allowed to update their position using random walk (Levy flight) to increase randomization in the WCA. Putting all the advantages of the WCA and MFO together and adding Levy flight results in a new hybrid algorithm which can perform significantly robust and efficient in solving optimization problems.

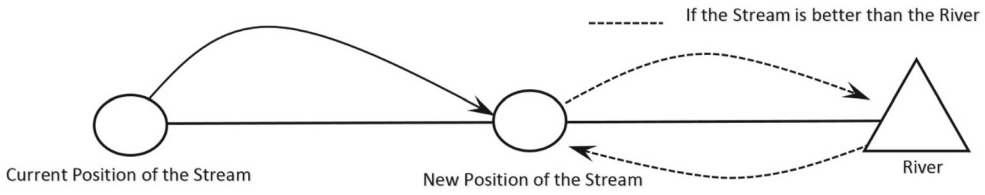

Fig. 1 The position updating procedure in the WCA

In order to evaluate the efficiency and robustness of the WCMFO, the performance of the proposed hybrid method is investigated in 23 benchmark functions. In addition, three engineering design optimization problems are solved using the WCMFO. The results in all test problems are compared to the other hybrid and state-of-the-art metaheuristic algorithms.

The remainder of this paper is organized as follows. In Sect. 2, the basic versions of the WCA and MFO are illustrated clearly. In Sect. 3, the proposed hybrid method is elaborated. In Sect. 4, the performance of the WCMFO investigated on unimodal, multimodal and fixed-dimension multimodal benchmark functions, and the simulation results are compared with state-of-the-art algorithms. Section 5 aims to modify the WCMFO to handle complex constraints in real-life optimization problems. Three well-known structural engineering design problems are optimized using WCMFO and the simulation results are compared with other algorithms. At the end, Sect. 6 concludes the paper.

2 Related works

2.1 Water Cycle Algorithm

Water Cycle Algorithm (WCA) is first proposed by Eskandar et al. (2012). WCA a novel Metaheuristic algorithm mimics the water cycle in nature to solve optimization problems. The WCA is based on strong conceptual bases which enables the algorithms to solve the optimization problems, efficiently.

2.1.1 Streams

As in most of the metaheuristic algorithms, the WCA uses an array called stream to show the decision variables of an optimization problem as follows.

$$\text{Raindrop} = [x_1, x_2, \ldots, x_n] \tag{1}$$

where $x_1, x_2, \ldots, x_n$ are the decision variables of the problem and n is the number of decision variables. WCA is a population-based metaheuristic algorithm; therefore, a set of streams should be created at the beginning as follows.

$$\text{Raindrops} = \begin{bmatrix} x_1, & x_2, & \ldots, & x_n \\ x_1, & x_2, & \ldots, & x_n \\ \vdots & & & \vdots \\ x_1, & x_2, & \ldots, & x_n \end{bmatrix} \tag{2}$$

where the number of rows is equal to the number of streams in the initial population, which is an input parameter of the algorithm.

2.1.2 Creation of the streams, rivers and sea

In the second step, the WCA calculates the objective function value of each stream in the initial population, then sorts the streams with respect to their corresponding objective function value from the best to the worst and sets the best stream (1st in the sorted population) as the sea. The second to $N_{sr} - 1$th streams are considered as rivers. The remaining $(N_{pop} - N_{sr})$ are considered as streams.

2.1.3 Flow of streams to rivers and sea

In order to mimic the flow of the streams and rivers to the sea in nature, WCA uses the following equation to update the position of the streams towards the rivers (Khalilpourazari and Mohammadi b; Sadollah et al. 2015a, b, c, d)

$$x_{\text{stream}}^{i+1} = x_{\text{stream}}^{i} + R \times C \times \left(x_{\text{river}}^{i} - x_{\text{stream}}^{i}\right) \tag{3}$$

where R is a random number between 0 and 1 generated using uniform distribution, and C in a continuous number between 1 and 2. When each stream updates its position towards its corresponding river, if the objective function of the stream in its new position is better than its corresponding river, the WCA changes the position of the stream and river. In order to illustrate the updating procedure of the WCA, Fig. 1 is presented.

Note that the WCA uses the same procedure to update the position of the rivers towards the sea. It is also worth mentioning that the WCA assigns streams to the rivers and sea depending on the intensity of the flow which is determined using below formula.

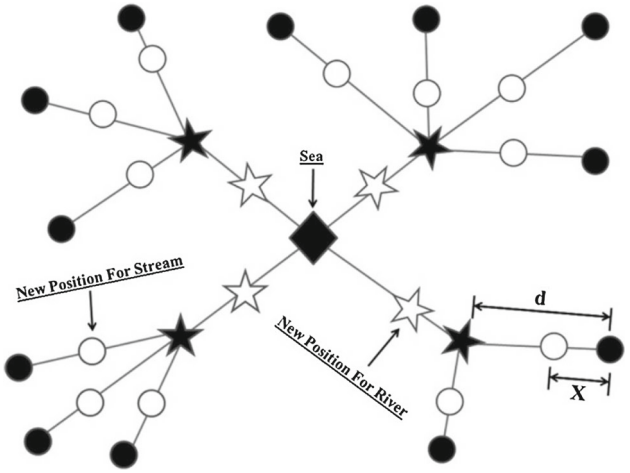

Fig. 2 A schematic view of the WCA, *Source*: Eskandar et al. (2012)

$$\mathrm{NS}_n = \mathrm{round}\left\{\left|\frac{\mathrm{Cost}_n}{\sum_{i=1}^{N_{\mathrm{sr}}}\mathrm{Cost}_i}\right| \times N_{\mathrm{Raindrops}}\right\}, \quad n = 1, 2, \ldots, N_{\mathrm{sr}} \tag{4}$$

2.1.4 Evaporation condition

One of the most important characteristics of the metaheuristic algorithms is randomization. In WCA, to avoid trapping in local optima and to increase randomization, the evaporation and raining conditions are considered. Evaporation and raining occur when the distance between a river and the sea is less than d_{max}; this procedure also occurs when the distance between any stream and the sea is less than $d_{\max}$ (Sadollah et al. 2015a, b, c, d).

To focus more on exploration, a large number for $d_{\max}$ should be considered, while, for exploitation, a small value for $d_{\max}$ is preferable. Therefore, since in the first iterations, the value of the $d_{\max}$ should be large to focus more on exploration and in the last iterations should be small to exploit the solution space, the value of the $d_{\max}$ considered to change over the course of iterations (Khalilpourazari and Mohammadi b). For this purpose, the following equation is used to decrease the value of the $d_{\max}$ linearly over the course of iterations.

$$d_{\max}^{i+1} = d_{\max}^{i} - \frac{d_{\max}^{i}}{Max\ It} \tag{5}$$

where $Max\ It$ shows the maximum number of iterations.

Raining process occurs when the distance between a river or stream and the sea is less than $d_{\max}$ to create new streams as streams to flow to rivers and sea. This process occurs iteratively to meet stopping criteria. Figure 2 presents a schematic view of the WCA.

The pseudo-code of the Water Cycle Algorithm is presented as follows.

```
Water Cycle Algorithm Pseudo Code
set the parameters of WCA such as Npop, Nsr and maximum number of iterations
for i=1:Npop
    Create a stream
    Calculate the objective function value of the stream
end for
sort the streams from best to worst based on their objective function value
Sea ← the first stream
Rivers ← n_sr - 1
Stream ← n_pop - n_sr
Determine the intensity of flow for rivers and sea
i=0;
While    i < maximum number of iterations
i=i+1;
        Update the position of streams
        Stream_objective=objective function value of the new stream
    for    each stream
        if    stream_objective < river_objective
            River_position= the new stream
            if    stream_objective < sea_objective
                Sea_position= the new stream
            end if
        end if
        if river_objective < sea_objective
            Sea_position =River_position
        end if
    end for
    for    each river
Update the position of rivers
river_objective =objective function value of the new river
        if river_objective < sea_objective
            Sea_position =River_position
        end if
    end for
    for    Rivers and streams
d=calculate the distance between each river or stream and the sea
        if    d < d_max
            raining process (for both rivers and streams)
        end if
    end for
    Decrease the  d_max
end while
```

2.2 Moth-Flame Optimization algorithm

Moths as a group of insects are very similar to butterflies. One of the most interesting behaviour of moths is their unique navigation approach. To travel long distances in straight path, they fly by keeping a fixed angle with respect to the moon. This effective approach is called transverse orientation (Frank et al. 2006; Gaston et al. 2013; Mirjalili 2015a, b). The effectiveness of the transverse orientation strongly depends on the distance of the light source. For example, when the light source is close to the moth, the moth starts flying in a spiral path around the light. This spiral fly path eventually converges the moth to the light (Khalilpourazari and Pasandideh 2017). Using this behaviour and mathematical modelling, the Moth-Flame Optimization algorithm is proposed by Mirjalili (2015a, b).

In the MFO algorithm, the moths are considered as the candidate solutions and their position is considered as a vector of decision variables. Therefore, each moth can fly in the

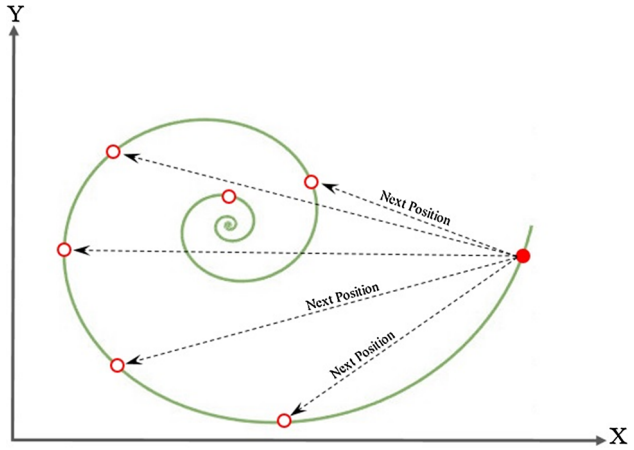

Fig. 3 Spiral fly path of the moths around their corresponding flame

solution space of the problem freely.

$$MO = \begin{bmatrix} mo_{1,1} & mo_{1,2} & \dots & mo_{1,n-1} & mo_{1,n} \\ v_{2,1} & \ddots & \dots & \cdots & mo_{2,n} \\ \vdots & \dots & \ddots & \dots & \vdots \\ mo_{npop-1,1} & \cdots & \cdots & \ddots & mo_{npop-1,n} \\ mo_{npop,1} & mo_{npop,2} & \dots & mo_{npop,n-1} & mo_{npop,n} \end{bmatrix} \tag{6}$$

where *npop* is the number of moths in initial population and *n* presents the number of decision variables. Another basic concept of the MFO algorithm is the flame matrix. Since each moth flies around its corresponding flame, therefore, the flame matrix is in the same size as the moth's matrix.

$$Fx = \begin{bmatrix} Fl_{1,1} & Fl_{1,2} & \dots & Fl_{1,n-1} & Fl_{1,n} \\ Fl_{2,1} & \ddots & \dots & \cdots & Fl_{2,n} \\ \vdots & \dots & \ddots & \dots & \vdots \\ Fl_{npop-1,1} & \cdots & \cdots & \ddots & Fl_{npop-1,n} \\ Fl_{npop,1} & Fl_{npop,2} & \dots & Fl_{npop,n-1} & Fl_{npop,n} \end{bmatrix} \tag{7}$$

where *npop* and *n* are the number of moths and number of decision variables, respectively. The difference between moth and flame is that the moth flies around its corresponding flame to find better solutions, while the flame is the best solution obtained so far by the moth. To show the spiral fly path of the moths around their corresponding flame, Fig. 3 is presented.

Since, the flying path of the moths is spiral around their corresponding flame, therefore, a logarithmic spiral function is defined to set a spiral fly path for the moths (Mirjalili 2015a, b; Khalilpourazari and Pasandideh 2017).

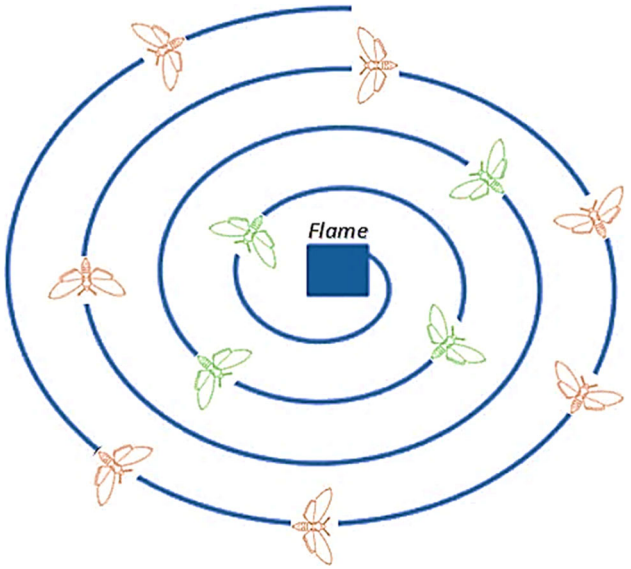

Fig. 4 The positions of the moths in the first and last iterations of the MFO algorithm

$$Mo_i^{X+1} = \left| Mo_i^X - Fl_i \right| \cdot e^{bt} \cdot \cos(2\pi t) + Fl_i \tag{8}$$

The parameter t is a random uniform number between -1 and 1 which defines the closeness of the next position of the moth to its corresponding flame. To explore the solution space more effectively in the first iterations and exploitation of the solution space in last iterations, an adaptive procedure is proposed to reduce the values of the parameter t over the iterations (Khalilpourazari and Pasandideh 2017).

$$a = -1 + Current\,it \left(\frac{-1}{\max it} \right) \tag{9}$$

$$t = (a-1) \times rand + 1 \tag{10}$$

where *Maxit* is the maximum number of iterations. Mirjalili (2015a, b) defined a as convergence constant which decreases linearly from -1 to -2 over the course of iterations. This guarantees both exploration and exploitation in the MFO algorithm. Figure 4 presents the positions of the moths in the first and last iterations of the algorithm.

Here green moths present the position of the moths around their corresponding flame in last iterations, and orange one presents the location of the moths in the first iterations of the MFO. To obtain the final solution, the number of flames is reduced over the iteration of the MFO algorithm using the following formula.

$$Fl = round\left(N_{fl} - l \frac{N_{fl} - 1}{maxit} \right) \tag{11}$$

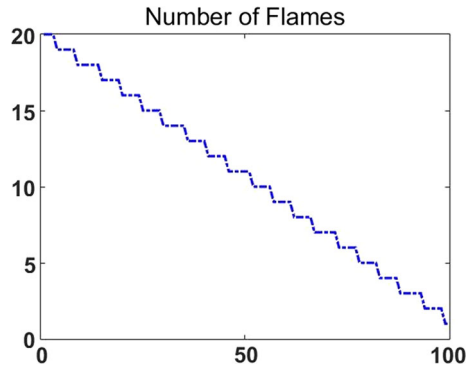

Fig. 5 The number of flames over iterations

where Fl is the number of flames in the current iteration, N_{fl} is the number of flames in the last iteration, l is the current iteration and $Maxit$ is the maximum number of iterations. Figure 5 shows how the number of flames decreases linearly over the iterations.

The pseudo-code of the MFO algorithm is presented as follows.

```
Input the values of the main parameters of the algorithm
Enter the lower and upper bound of the decision variables

for i= 1:number of moths
      for j= 1:number of the decision variables

         MO (i,j)=Lower bound(i)+(Upper bound (i)-Lower bound(i))*rand

      end for
end for

While Current iteration<Maximum number of iterations

if Iteration==1
         Enter the number of flames equal to the number of moths in initial population
else
         Linearly decrease the number of flames
end
FL=Fitness Function (MO);
if Iteration==1
         Sort the moths based on FL
         Update the Flames
         Iteration=0;
else
         Sort the moths based on FL from last iteration
         Update the Flames
end

Decrease the convergence constant

for j = 1 : npop
         for k= 1 : Number of variables
                  Calculate e r and t
                  Update the position of moths with respect to their corresponding
                  flame
         end
end
Current iteration=Current iteration+1;
end
```

3 Hybrid Water Cycle Moth-Flame Optimization (WCMFO) algorithm

As described earlier, the WCA has great ability in exploring the solution space of the problem. In the WCA, streams and rivers update their position towards the sea, and this procedure helps the search agents to update their position with respect to the best solution. Conversely, the WCA suffers from lack of an efficient operator to perform exploitation. On the other hand, MFO using its spiral movement ability performs very well in exploitation but cannot explore the solution space efficiently. This is because each moth updates its position towards its corresponding flame. Therefore, the information of the best solution obtained so far by the MFO is not shared between the search agents. The motivation behind this research is to develop an efficient hybridization of the WCA and MFO which can benefit from advantages of the both algorithms. In the proposed WCMFO algorithm, the WCA is considered as the basic algorithm. The first improvement in the WCA is using the spiral movement of the moths to update the position of the streams and the rivers. The updating procedure of the basic WCA just considers the space between the stream and a river when its updating the position of a stream. In other words, the next position of the stream would be in the space between the stream and its corresponding river. Conversely, the updating procedure of the MFO algorithm allows the moths to update their position anywhere around their corresponding flame. Allowing the streams and rivers to update their position using the spiral movement of the moths, significantly increases the exploitation ability of the hybrid WCMFO.

The second improvement in the basic WCA is improving the raining process. As mentioned earlier, randomization plays a major role in the metaheuristic algorithms. To increase the randomization in the WCMFO algorithm, two processes are considered. The first one is raining process as in the basic WCA. When the distance between a river or stream and the sea is less than $d_{\max}$, the WCMFO performs raining process to create new solutions. The second one is letting the streams to flow randomly in the solution space using a random walk (Levy flight). Consider an iteration of the WCA, if the streams update their positions and cannot find better solution, then the position of the rivers and the sea would not be changed till the next iteration. In the WCMFO, to increase the randomness in the algorithm, streams are allowed to update their position using Levy flight using the following equation.

$$x_{i+1} = x_i + Levy(\dim) \otimes x_i \tag{12}$$

where x_{i+1} is the next position of the stream, x_i is the current position of the stream and dim is the dimension of the problem or number of the decision variables. The Levy flight is calculated using below formula.

$$Levy(x) = \frac{0.01 \times \sigma \times r_1}{|r_2|^{\frac{1}{\beta}}} \tag{13}$$

where r_1 and r_2 are randomly generated numbers between 0 and 1. In the above formulation, the parameter σ is calculated

as follows (Yang 2010a, b).

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \tag{14}$$

The pseudo-code of the WCMFO is presented as follows.

```
WCMFO Pseudo Code
set the parameters of WCMFO such as Npop, Nsr, a, and maximum number of iterations
for i=1:Npop
    Create a random stream
    Calculate the objective function value of the stream
end for
sort the streams from best to worst based on their objective function value
Sea ← the first stream
Rivers ← n_sr - 1
Stream ← n_pop - n_sr
Determine the intensity of flow for rivers and sea
i=0;
While    i < maximum number of iterations
i=i+1;
    for    streams
Update the position of stream using spiral movement
Stream_objective=objective function value of the new stream
            if    stream_objective < river_objective
                River_position= the new stream
                if    stream_objective < sea_objective
                    Sea_position= the new stream
                end if
            end if
            if river_objective < sea_objective
                Sea_position =River_position
            end if
    end for
    for    rivers
Update the position of rivers using spiral movement
river_objective =objective function value of the new river
            if river_objective < sea_objective
                Sea_position =River_position
            end if
    end for
    for    streams
        Update the position of the streams using Levy flight
    end for
    for    Rivers and streams
d=calculate the distance between each river or stream and the sea
            if    d < d_max
                raining process (for both rivers and streams)
            end if
    end for
    Linearly decrease the parameter  d_max
    Linearly decrease the parameter a
end while
```

4 Results and discussion

Due to stochastic nature of the metaheuristic algorithms, the performance of the algorithms may vary on a specific test problem within different runs (Khalilpourazari and Khalilpourazary 2017a, b). Sometimes the algorithms perform well and sometimes not. Therefore, in order to evaluate the performance of the metaheuristic algorithms, several benchmark test problems should be solved within different runs to make a correct conclusion. In this paper, the performance of the proposed WCMFO is evaluated on 23 benchmark functions including unimodal, multimodal and fixed-dimension multimodal functions which are commonly used by researchers (Digalakis and Margaritis 2001; Yao et al. 1999; Molga and Smutnicki 2005; Yang 2010a, b). The first class of functions has no local optima, and they have a unique single global optimum. These benchmark functions are designed to evaluate the exploitation ability of a metaheuristic algorithm. Conversely, the second and third class benchmark functions are designed to evaluate the exploration ability of the metaheuristic algorithms. These functions have a lot of local optima as well as unique global optima. In addition to the aforementioned benchmark functions, three challenging engineering problems are solved using WCMFO as well to show its applicability in engineering and constrained optimization.

To verify the efficiency of the proposed hybrid algorithm, the WCMFO is compared to Artificial Bee Colony (ABC) algorithm (Karaboga and Basturk 2007), Cuckoo Search (CS) (Yang and Deb 2009), Genetic Algorithm (GA) (Holland 1992), Particle Swarm Optimization (PSO) (Kennedy and Eberhart 1995), Hybrid Particle Swarm Optimization and Gravitational Search Algorithm (PSOGSA) (Mirjalili and Hashim 2010), Gravitational Search Algorithm (GSA) (Rashedi et al. 2009), Moth-Flame Optimization (MFO) algorithm (Mirjalili 2015a, b), Water Cycle Algorithm (WCA) (Eskandar et al. 2012) and Dragonfly Algorithm (DA) (Mirjalili 2016b).

First, the performance of the WCMFO is investigated in solving unimodal benchmark functions. Table 2 presents the seven unimodal functions which are used in this paper. Also, Fig. 6 shows a 2D vision of the unimodal benchmark functions.

To solve the unimodal and multimodal benchmark functions, 100 search agents are considered for each algorithm over 1000 iterations. To make an unbiased comparison, each algorithm is run 30 times on each benchmark function. Table 3 presents the statistical results of the 10 algorithms. For each algorithm, the average and standard deviation of each benchmark function are reported.

From the results in Table 3, it is clear that the WCMFO algorithm can obtain very competitive solutions comparing to the other algorithms. Table 3 shows that the proposed hybrid WCMFO algorithm outperforms all the other metaheuristics in four of the seven unimodal benchmark functions (F1, F2, F3, F4). In addition, in F6 the WCMFO algorithm is ranked third among the algorithms. These show the superiority of the hybrid WCMFO algorithm over the other algorithms. Convergence plot of the algorithms on some of the unimodal benchmark functions is presented in Fig. 7. Figure 7 shows that the WCMFO algorithm is able to find significantly better solutions comparing to the other algorithms. This is because of the efficient hybrid operators of the WCMFO algorithm.

Detailed information of the performance of the hybrid WCMFO algorithm on unimodal functions is presented in Fig. 8. In the first column of Fig. 8, a two-dimensional view of the benchmark function is presented. The second column

Table 2 Unimodal benchmark functions

Function	Range	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^{10}$	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]^{10}$	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$	$[-100, 100]^{10}$	0
$f_4(x) = \max_i \{ \lvert x_i \rvert , 1 \leq i \leq n \}$	$[-100, 100]^{10}$	0
$f_5(x) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^{10}$	0
$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	$[-100, 100]^{10}$	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]^{10}$	0

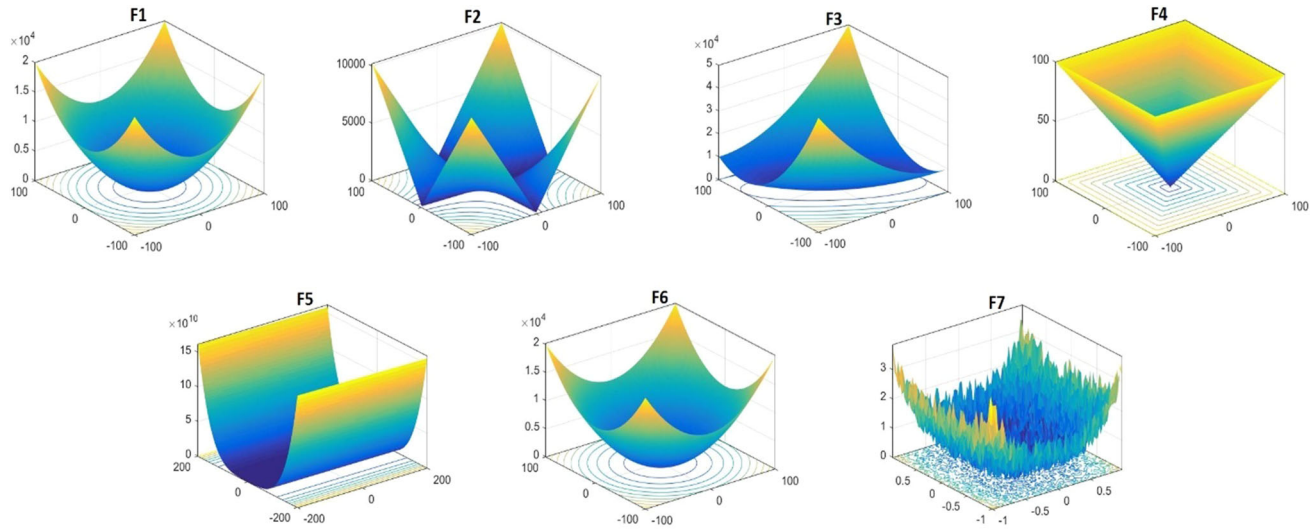

Fig. 6 Two-dimensional view of the unimodal functions

in the Fig. 8 presents the average fitness of the search agents in the WCMFO algorithm. The third column shows the best solution obtained so far by the WCMFO algorithm. This column shows that the WCMFO is able to converge to the best global optima very fast. The fourth column presents the value of the parameter a over the course of iterations. The value of this parameter decreases adaptively over the iterations.

Second, the WCMFO algorithm is benchmarked on six multimodal benchmark functions. As mentioned earlier, these functions have many local optima and challenges the exploration ability of metaheuristic algorithms. Tables 4 and 5 present the formulation of the multimodal and fixed-dimension multimodal benchmark functions. A schematic view of the multimodal and fixed-dimension multimodal functions is presented in Figs. 9 and 10.

Table 6 presents the statistical results of the algorithms in solving multimodal benchmark functions in 30 times run. For each algorithm, the average and standard deviation in each benchmark function are reported.

From the results in Table 6, the WCMFO algorithm is able to obtain very competitive results. In F8 benchmark function, the WCMFO outperforms all the other algorithms except ABC algorithm. In F9, the WCMFO ranked third after GSA and PSO and it outperforms all the other algorithms in this benchmark function. In F10, the WCMFO, PSO, GA and MFO obtain the global optima; therefore, the WCMFO algorithm outperforms all the other solution methods. In F11, the WCMFO outperforms all the other metaheuristic algorithms by obtaining significantly better solutions. In F12 and F13 benchmark functions, the WCMFO algorithm is ranked second after PSO and outperforms all the other solution methods. A graphical presentation of the convergence of the algorithms is presented in Fig. 11.

Table 3 Statistical results of the algorithms in 30 runs on each benchmark functions

F	ABC		CS		GSA		PSOGSA		WCMFO	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_1	1.93E−2	1.46E−2	9.84E−13	4.69E−13	1.02E−18	3.3E−19	1.24E−20	3E−21	1.10E−95	6.05E−95
F_2	3.42E−2	1.3E−2	1.67E−05	4.3E−06	2.33E−09	4.39E−10	2.58E−10	4.57E−11	3.52E−33	1.35E−32
F_3	848.34	227.45	2.17E−06	9.13E−07	1.00E−05	5.50E−05	2.48E−20	9.34E−21	1.62E−32	8.88E−32
F_4	7.939	1.938	4.9E−3	0.00125	4.76E−10	8.44E−11	6.35E−11	1.26E−11	3.29E−24	1.31E−23
F_5	44.513	15.196	0.689467	0.571426	5.423	0.1238	1.1607	2.0941	2.4259	3.467
F_6	1.13E−2	7.3E−3	1.48E−12	8.39E−13	6.40E−19	2.30E−19	1.380E−20	3.37E−21	6.97E−29	3.29E−28
F_7	3.93E−2	1.09E−2	4.735E−3	1.754E−3	1.86E−3	6.7E−4	2.3E−3	1.2E−3	0.4987	0.305106

F	MFO		WCA		DA		PSO		GA	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_1	1.65E−31	4.91E−31	3.1E−14	5.84E−14	5.303E−1	1.3180	7.38E−54	2.17E−53	8E−4	8.7E−4
F_2	2.69E−19	6.22E−19	2.11E−07	3.96E−07	2.392	3.912	5.68E−31	1.38E−30	3E−3	1.8E−3
F_3	2.05E−11	4.21E−11	3.56E−12	9.56E−12	215.45	935.17	3.15E−18	9.67E−18	13.213	8.042
F_4	5.79E−06	3.17E−05	1.08E−11	5.73E−11	1.153	2.702	4.30E−16	9.76E−16	0.209	5.8E−2
F_5	133.11	555.57	1.252	1.831	6784.5	21974.5	3.311	1.647	16.913	22.375
F_6	4.78E−32	1.27E−31	4.60E−18	2.26E−17	2.2023	5.528	0	0	7.5E−4	7.2E−4
F_7	1.2E−3	7.2E−4	0.5155	0.2552	6.9E−3	7.6E−3	1.4E−3	7E−4	8.1E−4	5.5E−4

From Fig. 11, it is clear that the WCMFO algorithm can obtain the global optima faster than the other algorithms without trapping in local optima. The two main processes of the WCMFO, Levy flight and raining process help the WCMFO to explore the solution space of the problem more efficiently and find the global optima as fast as possible. Figure 12 presents detailed information of the performance of the hybrid WCMFO algorithm on multimodal benchmark functions.

The third class of the benchmark functions was fixed-dimension multimodal functions. In these benchmark functions, the 10 metaheuristic algorithms are involved and the statistical results of the algorithms in 30 time runs are reported in Table 7.

From the results in Table 7, the hybrid WCMFO algorithm is able to find the global optima in 9 of 10 benchmark functions, and with significantly less standard deviation it is ranked first in 9 of 10 benchmark functions. The small value of the standard deviation for the WCMFO shows its robustness in finding global optima. Therefore, the WCMFO algorithm outperforms the other metaheuristic algorithms in this class of benchmark functions. Figure 13 presents the convergence curve of the algorithms in two of the fixed-dimension multimodal benchmark functions.

From Fig. 13, it is clear that the WCMFO algorithm outperforms the other metaheuristic algorithms in finding better solution in fewer iterations.

To sum it up, the efficiency of the hybrid WCMFO algorithm was proved in 23 benchmark functions, experimentally. The results show that the WCMFO algorithm is able to find the global optima in majority of the benchmark functions.

4.1 Statistical analyses

To make a reliable conclusion and to show the superiority of the proposed hybrid algorithm over state-of-the-art algorithms, in this section, statistical tests are carried out to show the significance of the results. In addition to above-mentioned measures (worst, mean, best and SD), a multiple comparison test (Friedman's test) is performed to determine significant differences between the properties of the algorithms. The average rankings obtained by Friedman test are used to show that how successful is the proposed hybrid WCMFO algorithm. Based on the Friedman's test rankings, the lower the rank, the more efficient the algorithm is. Table 8 presents the mean ranks obtained by this test at 95% confidence level.

Based on the average and standard deviations, the WCMFO algorithm is the best solution method which outperforms other state-of-the-art algorithms in majority of the test problems. Figure 14 presents a schematic view of the results of the Friedman's test.

Based on Fig. 14, it becomes obvious that the WCMFO not only is the best solution method among other state-of-the-art algorithms, but also is the most robust solution methodology comparing to the other solution methods. For this purpose, comparing the worst ranking of the WCMFO to other algorithms, it has the lowest worst rank among the algorithms. Therefore, it can be inferred that the WCMFO is the most

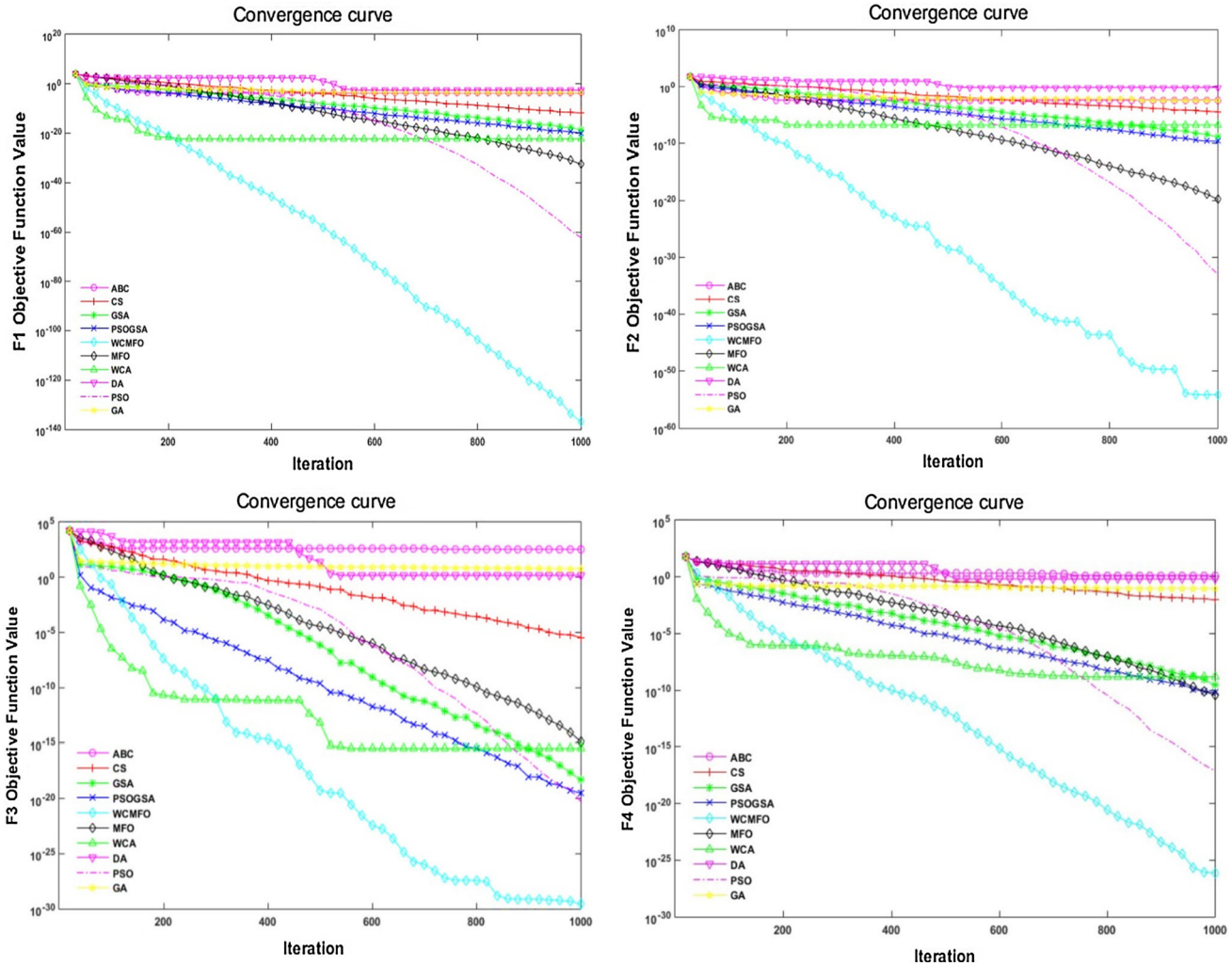

Fig. 7 Convergence Plot of the algorithms on unimodal benchmark functions

robust solution methodology which performs well in all types of the benchmark functions.

5 Engineering optimization problems

Most of the real-world problems are constrained problems with various equality and inequality constraints (Zareh-Desari et al. 2015; Khalilpourazary et al. 2014a, b). Therefore, the efficiency of the hybrid WCMFO should be investigated in solving challenging real-world constrained optimization problems. Thus, in this research three classical engineering problems are solved using WCMFO algorithm and the results are compared to the other state-of-the-art algorithms.

5.1 Constraint handling

Constraint handling is one of the biggest challenges in using metaheuristic algorithms in solving real-world problems. According to (Coello 2002), there are five constraint handling techniques: penalty functions, hybrid methods, repair algorithms, separation of objective functions and constraints and special operators. Among the constraint handling techniques, the penalty functions are simple to implement. There are different types of the penalty functions: static penalty, annealing penalty, adaptive co-evolutionary penalty, dynamic penalty and death penalty. These approaches turn constrained optimization problems to an unconstraint problem by adding penalty values for violation of each constraint. A constrained optimization problem can be presented as follows.

$$\begin{aligned} &Min\, z = f(x) \\ &s.t \\ &k_i(x) \le 0 \\ &L_j(x) = 0 \end{aligned} \tag{15}$$

In this paper, a static penalty approach is used to handle complex constraints in optimization problems. The static penalty approach modifies the objective function as follows to create an unconstraint optimization problem.

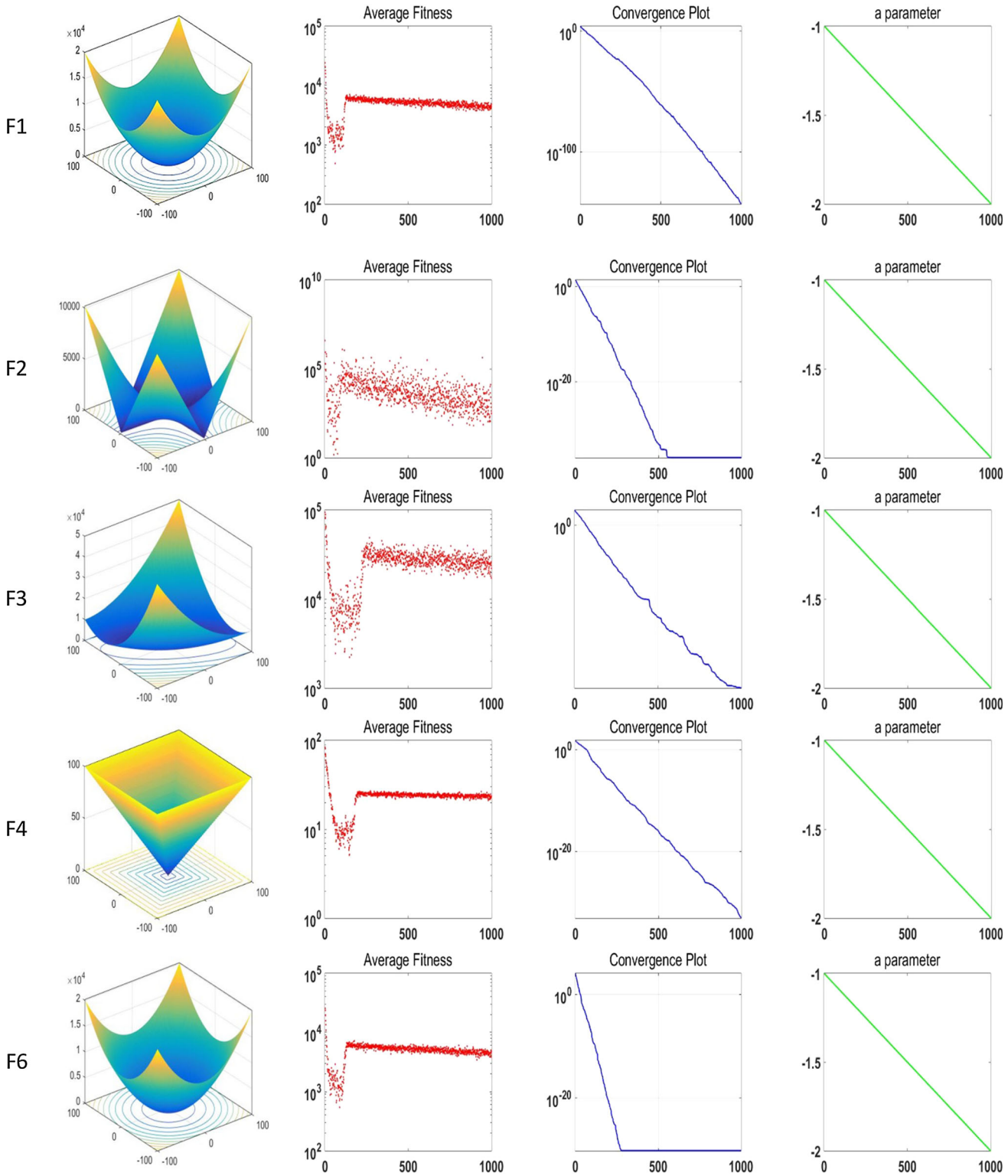

Fig. 8 A schematic view of the performance of the WCMFO algorithm on unimodal benchmark functions

Table 4 Multimodal benchmark functions

Function	Range	$f_{\min}$
$F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]^{10}$	-418.9829×5
$F_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]^{10}$	0
$F_{10}(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	$[-32, 32]^{10}$	0
$F_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]^{10}$	0
$F_{12}(x) = \frac{\pi}{n}\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}\{(y_i^- 1)^2[1 + 10\sin^2(\pi y_{i+1})]$ $+(y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4}$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m x_i < a \end{cases}$	$[-50, 50]^{10}$	0
$F_{13}(x) = 0.1\{\sin^2(3\pi x_i) + \sum_{i=1}^{n}(x_i - 1)^2[1 + \sin^2(3\pi x_i + 1)]$ $+(x_n - 1)^2[1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]^{10}$	0

Table 5 Fixed-dimension multimodal benchmark functions

Fixed-dimension multimodal functions	Range	$f_{\min}$
$F_{14}(x) = (\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6})^{-1}$	$[-65, 65]^2$	1
$F_{15}(x) = \sum_{i=1}^{11}[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]^2$	$[-5, 5]^4$	0.00030
$F_{16}(x) = 4x_1^2 - 2.1x_1^4 = \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	-1.0316
$F_{17}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos x_1 + 10$	$[-5, 5]^2$	0.398
$F_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $\times[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$[-2, 2]^2$	3
$F_{19}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	$[1, 3]^3$	-3.86
$F_{20}(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	$[0, 1]^6$	-3.32
$F_{21}(x) = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	-10.1532
$F_{22}(x) = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	-10.4028
$F_{23}(x) = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	$[0, 10]^4$	-10.5363

$$\xi(x) = f(x) \pm \left[\sum_{i=1}^{n} r_i \times \max[0, k_i(x)]^{\beta} + \sum_{j=1}^{p} c_j \times \left|L_j(x)\right|\right] \tag{16}$$

where $\xi(x)$ is the modified objective function and r_i, c_j are positive penalty values (Coello 2002). This penalty approach has some advantages over the other penalty approaches. For instance, this approach assigns a penalty value for each infeasible solution, where the value of the penalty factor depends on the violation of the constraints. Therefore, the objective

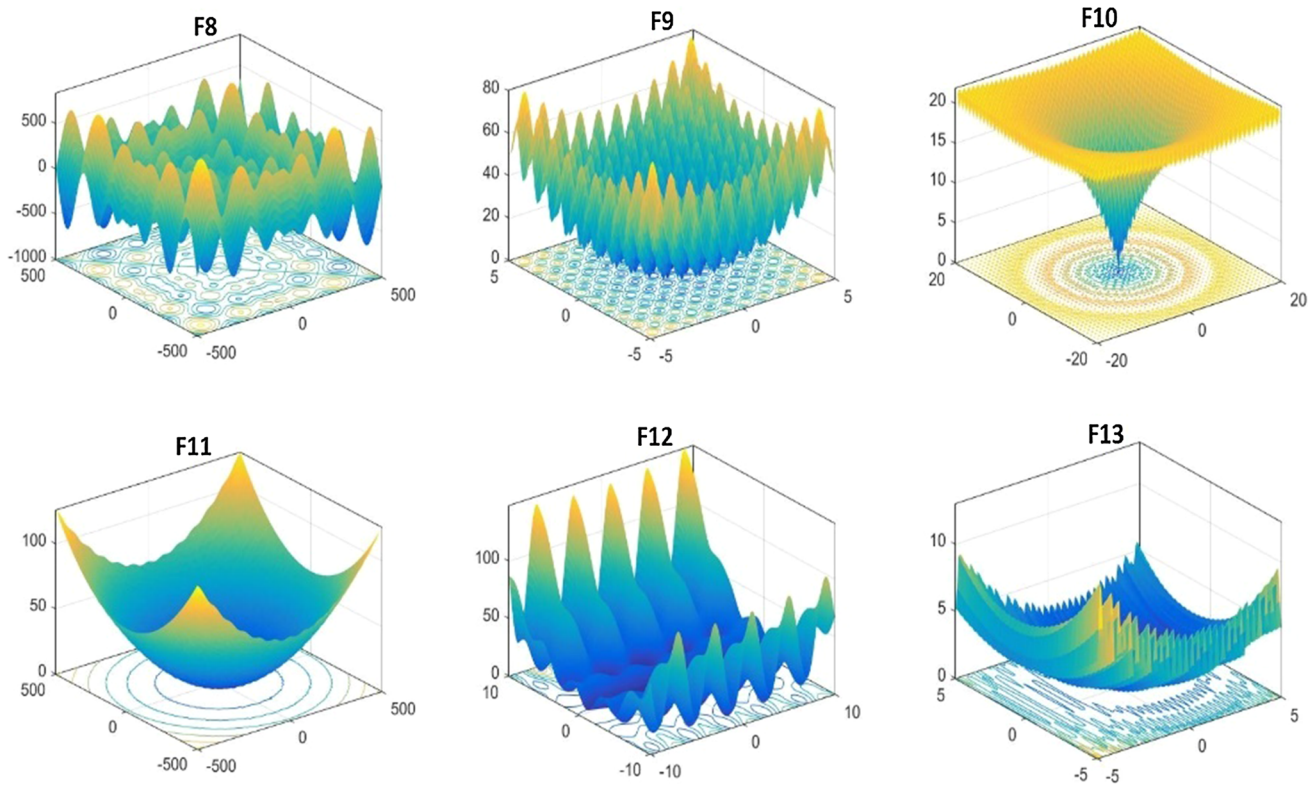

Fig. 9 Two-dimensional view of the multimodal functions

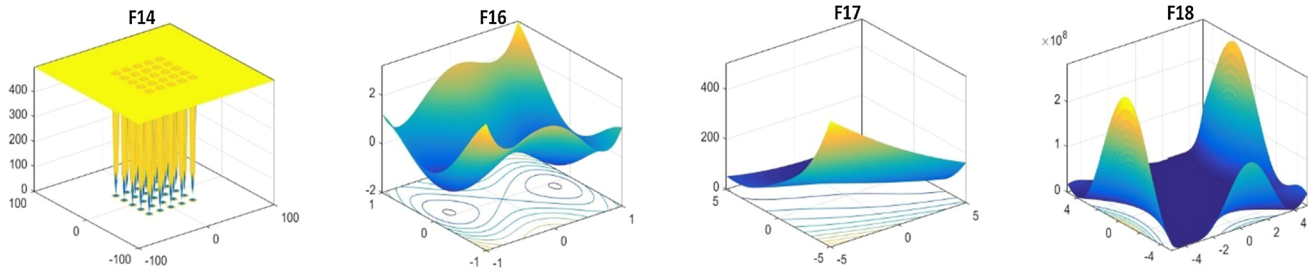

Fig. 10 Two-dimensional view of the fixed -dimension multimodal functions

function value of different infeasible solutions is different. But, in some penalty function approaches such as death penalty, a large number is assigned to objective function value of any infeasible solution. In other words, the amount of the violation of constraint has no effect on penalty value. Thus, we can infer that the static penalty function uses the information of infeasible solution which helps the search agents to move faster towards the feasible solution space of the problem. The following three well-known engineering problems are solved using WCMFO.

5.2 Pressure vessel design

In pressure vessel design problem, the goal is to minimize the total cost including material cost, forming cost and welding cost of the vessel. There are four variables in this problem which should be optimized (Gandomi et al. 2013): thickness of the head T_h (x_2), thickness of the shell T_s (x_1), inner radius R (x_3) and length of the cylinder section of the vessel without semi-spherical head L (x_4). A schematic view of this problem and the decision variables are given in Fig. 15.

Table 6 Statistical results of the algorithms in 30 runs on each multimodal benchmark functions

F	ABC		CS		GSA		PSOGSA		WCMFO	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_8	−3922.73	88.61857	−3712.01	167.4447	−1694.53	190.6721	−3271.76	278.508	−3729.7	96.325
F_9	3.677	1.0365	6.574	1.367	1.392	1.214	23.281	12.968	2.089	1.508
F_{10}	1.21E−06	9.37E−07	1.24E−15	1.08E−15	1.28E−10	6.71E−11	4.94E−12	2.26E−12	8.88E−16	1.00E−31
F_{11}	0.281	0.1086	3.96E−2	8.8E−3	1.67E−2	2.79E−2	0.2004	0.1141	9.91E−02	5.31E−2
F_{12}	1.9E−3	1.3E−3	9.77E−05	1.3E−4	7.95E−21	3.23E−21	0.2491	0.581	2.00E−29	6.44E−29
F_{13}	8.3E−3	5.1E−3	1.31E−09	1.39E−09	5.67E−20	1.88E−20	3.11E−21	1.06E−21	4.49E−22	2.06E−21
F	MFO		WCA		DA		PSO		GA	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_8	−3329.13	288.317	−3422.55	304.572	−3213.66	431.748	−2742.78	274.7175	−3692.39	182.42
F_9	12.8372	7.352	20.993	10.524	11.561	10.177	1.757	1.1592	3.8E−4	3.2E−4
F_{10}	8.88E−16	1.00E−31	2.42E−15	1.79E−15	3.14E−05	1.7E−4	8.88E−16	1.00E−31	8.88E−16	1.00E−31
F_{11}	1.78E−01	8.43E−2	0.1502	9.44E−2	0.3846	0.3826	0.1244	8.04E−2	5.610E−2	3E−2
F_{12}	3.11E−02	9.487E−2	1.036E−2	5.67E−2	0.5296	0.6912	4.71E−32	1.67E−47	5.73E−05	1.4E−4
F_{13}	1.10E−03	3.33E−3	7.3E−4	2.7E−3	0.5292	0.7173	1.34E−32	5.56E−48	6.21E−05	1.1E−4

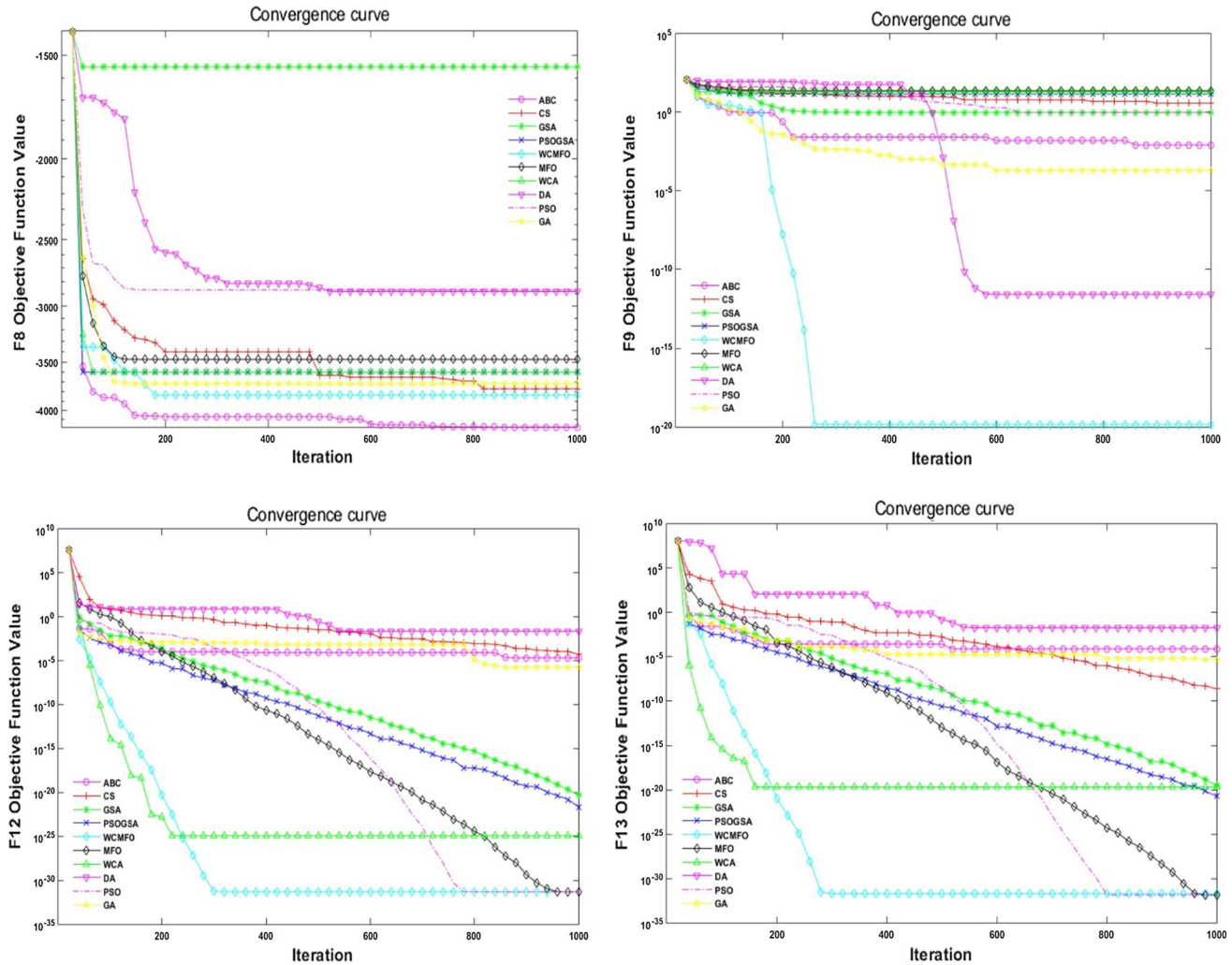

Fig. 11 Convergence Plot of the algorithms on multimodal benchmark functions

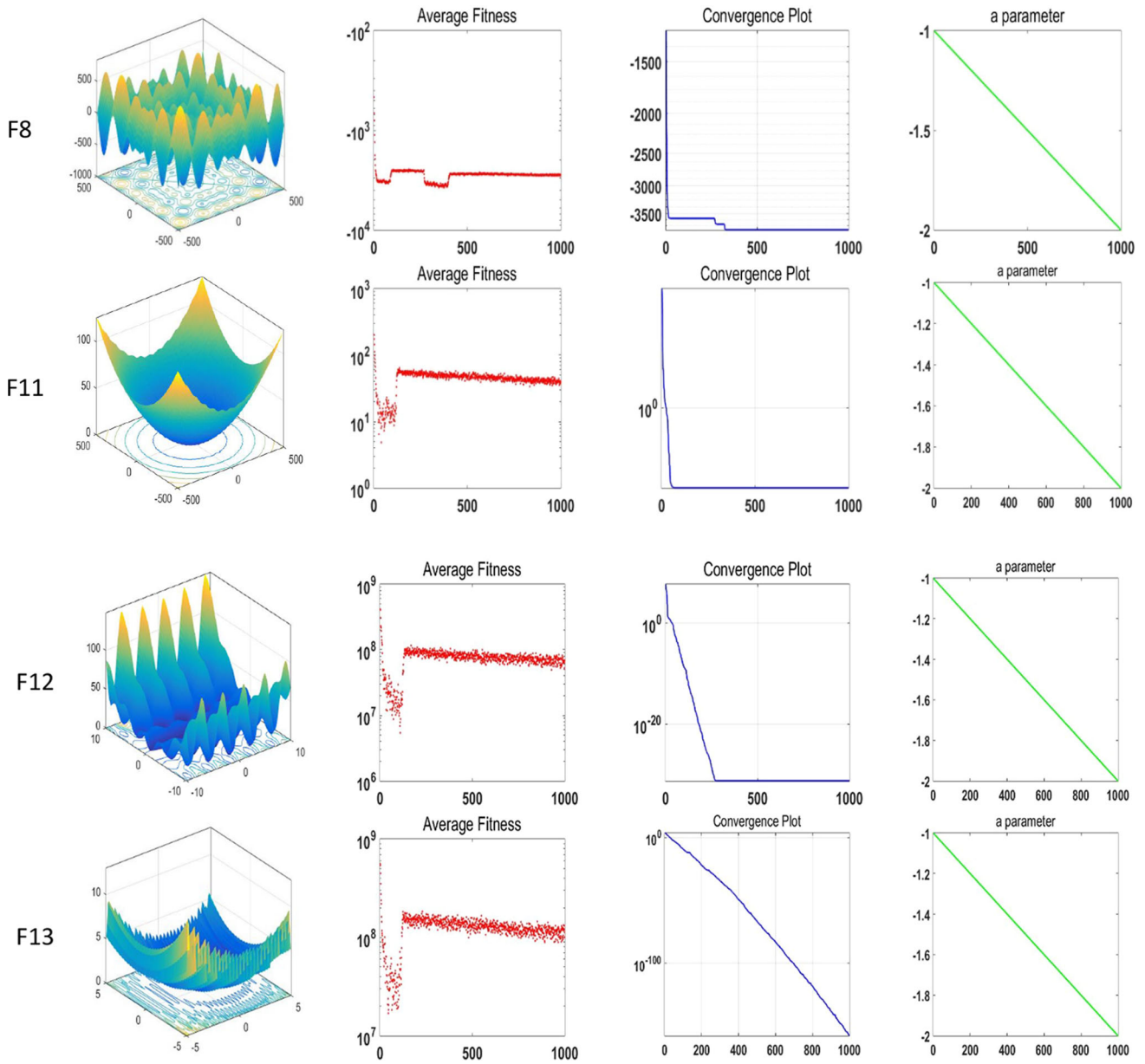

Fig. 12 A schematic view of the performance of the WCMFO algorithm on multimodal benchmark functions

The mathematical model of this problem is as follows.

$$
\begin{aligned}
&Min\ f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 \\
&\qquad +19.84x_1^2x_3, \\
&\text{S.t} \\
&g_1(x) = -x_1 + 0.0193x_3 \le 0, \\
&g_2(x) = -x_3 + 0.00954x_3 \le 0, \\
&g_3(x) = -\pi x_3^2x_4 - \tfrac{4}{3}\pi x_3^3 + 1296000 \le 0, \\
&g_4(x) = x_4 - 240 \le 0, \\
&0 \le x_1 \le 99, \\
&0 \le x_2 \le 99, \\
&10 \le x_3 \le 200, \\
&10 \le x_4 \le 200,
\end{aligned}
\tag{17}
$$

Table 9 presents the statistical results of the hybrid WCMFO algorithm over 50 independent runs. The results of the hybrid WCMFO are compared to the other state-of-the-art algorithms as well. The results show that the WCMFO algorithm is able to find competitive solution for the problem comparing to the other state-of-the-art methods.

From Table 9, it is clear that the WCMFO algorithm outperforms the other solution methods as well as novel meta-heuristic algorithms such as CSA (Askarzadeh 2016). The small value of the standard deviation for the WCMFO shows its robustness in optimizing the problem. Table 10 presents the value of the decision variables and constraints in the best solution obtained by the WCMFO algorithm.

Table 7 Statistical results of the algorithms in 30 runs on each fixed- dimension multimodal benchmark functions

F	ABC		CS		GSA		PSOGSA		WCMFO	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_{14}	0.998	1.02E−13	0.998	3.39E−16	3.4	2.578637	1.06	0.252193	0.998	5.36E−16
F_{15}	7E−4	1.3E−4	3E−4	4.23E−09	1.8E−3	4.9E−4	3.791E−3	7.5E−3	3.0E−4	1.07E−15
F_{16}	−1.03	7.36E−11	−1.03	0	−1.03	0	−1.03	0	−1.03	0
F_{17}	3.98E−1	5.68E−09	3.98E−1	1.13E−16	3.98E−1	1.13E−16	3.98E−1	1.13E−16	3.98E−1	1.13E−16
F_{18}	3	8.64E−05	3	4.52E−16	3	4.02E−15	3	4.52E−16	3	9.57E−15
F_{19}	−3.86	7.89E−11	−3.86	2.71E−15	−3.86	2.71E−15	−3.86	2.71E−15	−3.86	2.71E−15
F_{20}	−3.32	4.82E−06	−3.32	1.26E−13	−3.32	1.36E−15	−3.26	6.032E−2	−3.25	6.027E−2
F_{21}	−10.1	6.784E−3	−10.2	1.81E−15	−7.45	3.381188	−5.9	3.421068	−8.89	2.361515
F_{22}	−10.4	2.886E−3	−10.4	6.18E−14	−10.4	0	−5.76	3.454976	−10.4	1.59E−12
F_{23}	−10.5	4.074E−3	−10.5	2.15E−12	−10.5	9.03E−15	−6.99	3.890193	−10.5	4.09E−14

F	MFO		WCA		DA		PSO		GA	
	Ave	SD	Ave	SD	Ave	SD	Ave	SD	Ave	SD
F_{14}	1.03	0.181483682	0.998	3.39E−16	1.1	0.303306	1.56	0.959452	0.998	8.83E−14
F_{15}	8.37E−4	2.54E−4	3.69E−4	2.32E−4	1.343E−3	5.11E−4	7E−4	3.2E−4	8.4E−4	2.9E−4
F_{16}	−1.03	0	−1.03	0	−1.03	2.55E−11	−1.03	0	−1.03	5.02E−10
F_{17}	3.98E−1	1.13E−16	3.98E−1	3.79E−16	3.98E−1	7.60E−13	3.98E−1	1.13E−16	3.98E−1	4.73E−07
F_{18}	3	1.95E−15	3	1.79E−14	3	1.38E−06	3	4.52E−16	3	1.21E−08
F_{19}	−3.86	2.71E−15	−3.86	2.71E−15	−3.86	1.587E−3	−3.86	2.71E−15	−3.86	2.203E−3
F_{20}	−3.22	4.5066E−2	−3.26	6.046E−2	−3.25	6.720E−2	−3.26	6.046E−2	−3.32	2.170E−2
F_{21}	−7.56	3.323037	−8.31	2.718152	−9.81	1.280913	−9.31	1.925505	−10.2	0.000484
F_{22}	−9.35	2.423664	−9.52	2.00228	−10.4	0.192434	−9.52	2.00228	−9.93	1.822252
F_{23}	−10.3	1.39948	−9.82	2.235265	−10.3	1.060781	−10	1.635722	−9.61	2.405191

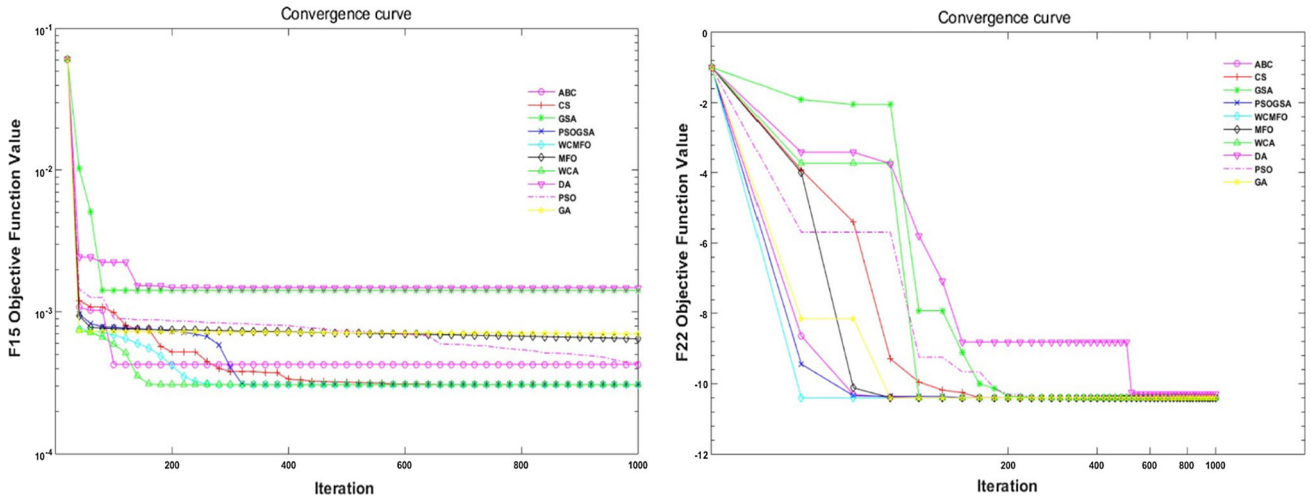

Fig. 13 Convergence plot of the algorithms on fixed- dimension multimodal benchmark functions

5.3 Tension/compression spring design problem

In tension/compression spring design problem, the aim is to minimize the weight of the tension/compression spring (Belegundu 1983; Arora 2004). There are three decision variables in this problem which should be optimized: wire diameter d (x_1), mean coil diameter D (x_2) and number of active coils N (x_3). A schematic view of this is presented in Fig. 16.

The mathematical model of this problem is as follows.

$$f(\vec{x}) = (x_3 + 2)x_2x_1^2,$$

Table 8 Results of the Friedman's test

Benchmark function	Friedman's mean rank										*P* value
	ABC	CS	GSA	PSOGSA	WCMFO	MFO	WCA	DA	PSO	GA	
F_1	9.6	7.267	5.3	4.267	1.267	3.267	6.233	7.1	2.267	8.433	5.99E−42
F_2	9.167	7.1	5.233	4.133	1.1	3.1	5.933	9.033	2.1	8.1	4.077E−47
F_3	9.933	7.033	3.933	2.267	1.067	5.7	5.367	8.033	3.1	8.567	1.328E−48
F_4	9.967	7.133	5.7	4.267	1.167	5.4	2.6	7.8	2.667	8.3	4.564E−46
F_5	9.1	3.033	6.533	3.133	3.4	6.6	2.367	9.367	4.433	7.033	3.652E−37
F_6	9.333	7	5.933	4.933	3.133	1.9	3.833	9.367	1.267	8.3	4.642E−51
F_7	8.1	6.367	3.8	4.367	9.367	2.767	9.533	5.733	3.2	1.767	4.400E−44
F_8	1.433	3.417	10	6.567	3.167	6.1	5.617	6.333	8.6	3.767	2.001E−38
F_9	5	6.567	2.783	9.083	3.517	7.7	8.75	6.8	3.267	1.533	5.985E−40
F_{10}	9.967	3.85	8.933	7.9	3.517	3.517	4.967	5.317	3.517	3.517	1.413E−44
F_{11}	8.7	2.8	1.583	7.433	5.233	7.067	6.033	6.683	5.7	3.767	1.828E−26
F_{12}	8.7	7.4	5.633	6.033	3.3	2.533	3.55	9.633	1.317	6.9	2.916E−43
F_{13}	9	6.833	5.5	4.267	3	2.35	5.05	9.833	1.333	7.833	5.251E−47
F_{14}	8.2	4.016	9.9	4.416	4.333	4.183	4.016	5.3	6.333	4.3	6.974E−37
F_{15}	6.3	3.067	9.5	4.267	2.067	6.4	2.517	8.4	6.117	6.367	1.180E−33
F_{16}	9.933	4.95	4.95	4.95	4.95	4.95	4.95	5.283	4.95	5.133	1.909E−47
F_{17}	9.933	4.916	4.916	4.916	4.916	4.916	5.066	5.216	4.916	5.283	3.731E−45
F_{18}	10.9	4.9666	4.966	4.966	4.9666	4.9666	4.9666	5.18333	4.966	5.116	8.066E−41
F_{19}	8.4	4.183	4.183	4.183	4.183	4.183	4.183	8.916	4.183	8.4	3.998E−43
F_{20}	6.133	2.567	2.567	5.683	6.067	7.433	5.483	8.2	5.6	5.267	3.123E−17
F_{21}	7.567	3.4	5.95	7.45	4.7	5.917	5.167	5.55	4.35	4.95	2.257E−11
F_{22}	8.633	3.75	3.75	7.95	3.75	4.717	4.7	6.833	4.733	6.183	1.439E−22
F_{23}	8.9	4.017	4.017	6.817	4.6	4.183	4.617	7.217	4.567	6.067	2.746E−22
Average	8.39	5.03	5.46	5.40	3.77	4.78	5.0	7.27	4.06	5.86	
SD	2.055	1.713	2.272	1.713	1.872	1.680	1.692	1.609	1.817	2.119	

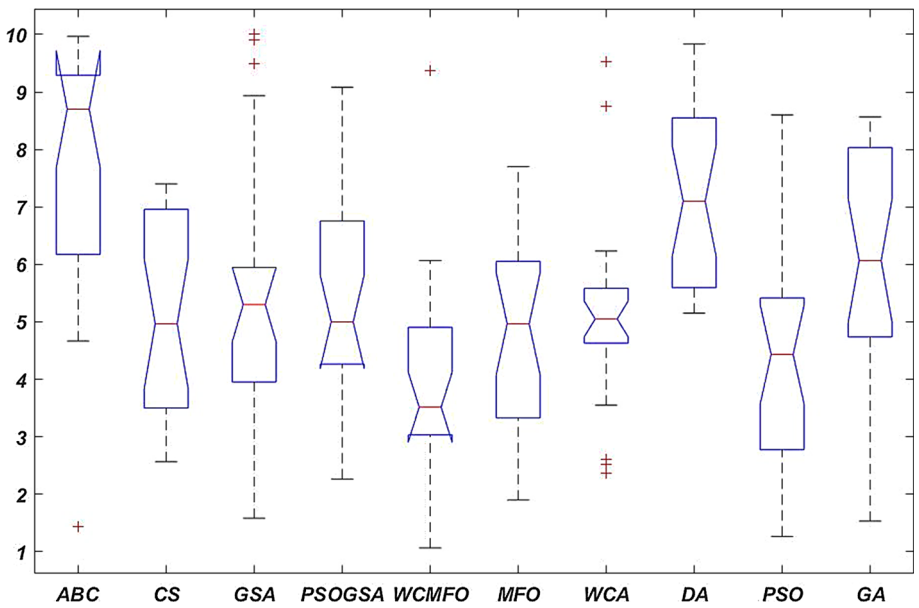

Fig. 14 A schematic view of the results of the Friedman's test

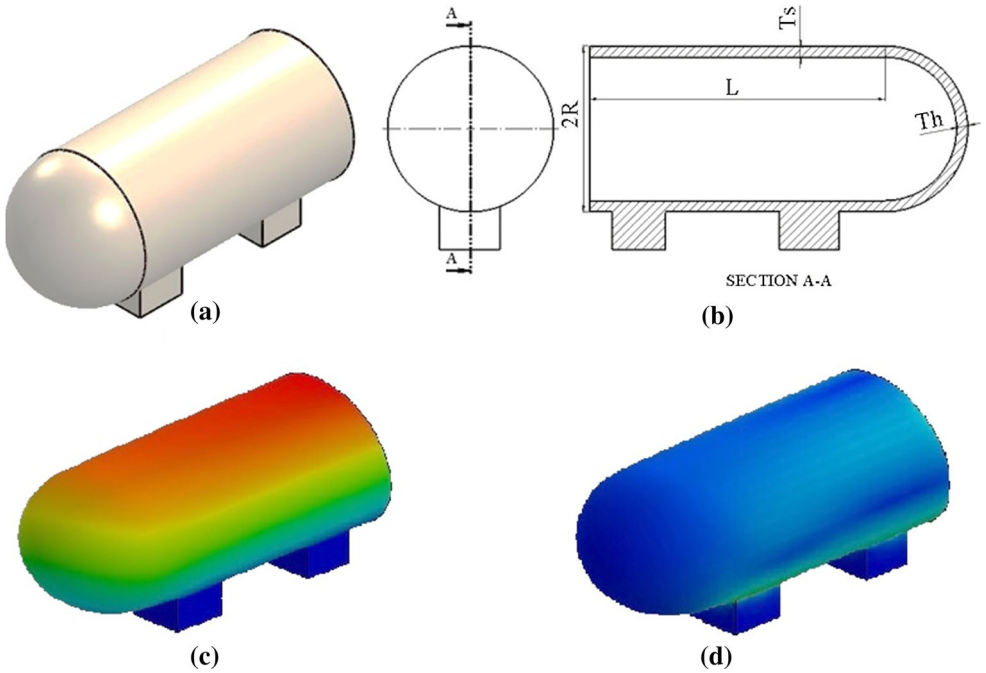

Fig. 15 **a** 3D view of the pressure vessel, **b** 2D view of the pressure vessel, **c** displacement heat map, **d** stress heat map

Table 9 Comparison of statistical results of the WCMFO algorithm and other algorithms for pressure vessel design problem

Method	Worst	Mean	Best	SD
GA3 (Coello 2000)	6308.4970	6293.8432	6288.7445	7.4133
GA4 (Coello and Montes 2002)	6469.3220	6177.2533	6059.9463	130.9297
HPSO (He and Wang 2007a)	6288.6770	6099.9323	6059.7143	86.20
G-QPSO (Coelho 2010)	7544.4925	6440.3786	6059.7208	448.4711
QPSO (Coelho 2010)	8017.2816	6440.3786	6059.7209	479.2671
PSO (Coelho 2010)	14076.3240	8756.6803	6693.7212	1492.5670
CDE (Huang et al. 2007)	6371.0455	6085.2303	6059.7340	43.0130
ABC (Akay and Karaboga 2012; Fogel et al. 1966)	N.A.	6245.3081	6059.7147	205
CPSO (He and Wang 2007b)	6363.8041	6147.1332	6061.0777	86.45
UPSO (Parsopoulos and Vrahatis 2005)	9387.77	8016.37	6154.70	745.869
PSO-DE (Liu et al. 2010)	N.A.	6059.714	6059.714	N.A.
TLBO (Rao et al. 2011)	N.A.	6059.7143	6059.7143	N.A.
CSA (Askarzadeh 2016)	7332.841	6342.4991	6059.7143	384.9454
WCMFO	**6059.7145**	**6059.71435**	**6059.714**	**4.47233E−05**

Bold values indicate the best results

Table 10 The value of the decision variables and constraints in the best solution obtained by the WCMFO algorithm for pressure vessel design problem

Parameter	x_1	x_2	x_3	x_4	g_1	g_2	g_3	g_4	f
Value	0.8125	0.4375	42.098	176.636	−2.134E−10	−0.03588	−8.386E−07	−63.363	6059.714

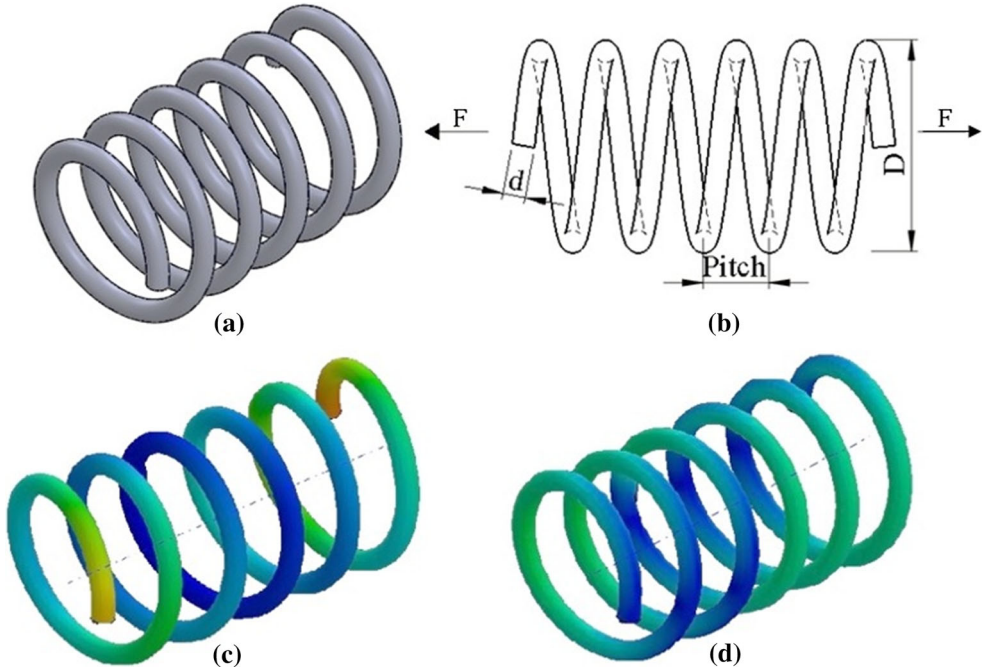

Fig. 16 **a** 3D view of the spring, **b** 2D view of the spring, **c** displacement heat map, **d** stress heat map

S.t

$$g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0,$$

$$g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} \leq 0,$$

$$g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} \leq 0,$$

$$g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0,$$

$$g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0,$$

$$0.05 \leq x_1 \leq 2.00,$$

$$0.25 \leq x_2 \leq 1.30,$$

$$2.00 \leq x_3 \leq 15.0, \tag{18}$$

The performance of the WCMFO algorithm is compared to the other solution methods in the literature. Table 11 presents the statistical results of the WCMFO algorithm and the other state-of-the-art solution methods.

From Table 11, it is clear that the WCMFO algorithm outperforms the other solution methods in terms of mean and best solution obtained in 50 independent runs. Table 12 presents the value of the decision variables in the best solution obtained by the WCMFO algorithm.

5.4 Welded beam design problem

The welded beam problem is a well-known problem which is widely studied by the researchers. The goal of this problem is to minimize the total cost of a welded beam. Figure 17 presents a schematic view of the welded beam.

There are four decision variables in this problem which should be optimized. The formulation of the mathematical model of the welded beam problem is as follows (Wang et al. 2014a, b, c).

$$Min\ f(x) = 1.10471 x_1^2 x_2 + 0.04811 x_3 x_4 (x_2 + 14.0)$$

$$g_1(x) = \tau(x) - \tau_{\max} \leq 0$$

$$g_2(x) = \sigma(x) - \sigma_{\max} \leq 0$$

$$g_3(x) = \delta(x) - \delta_{\max} \leq 0$$

$$g_4(x) = x_1 - x_4 \leq 0$$

$$g_5(x) = P - P_c(x) \leq 0$$

$$g_6(x) = 0.125 - x_1 \leq 0$$

$$g_7(x) = 1.1047 x_1^2 + 0.04811 x_3 x_4 (14.0 + x_2) - 5.0 \leq 0$$

$$0.1 \leq x_1 \leq 2,$$

$$0.1 \leq x_2 \leq 10$$

$$0.1 \leq x_3 \leq 10$$

$$0.1 \leq x_4 \leq 2$$

$$\tau(x) = \sqrt{(\tau')^2 + \frac{x_2}{2R} 2\tau''\tau' + (\tau'')^2}$$

Table 11 Comparison of statistical results of the WCMFO algorithm and other algorithms for spring design problem

Method	Worst	Mean	Best	SD
GA3 (Coello 2000)	0.0128220	0.0127690	0.0127048	3.94E−5
GA4 (Coello and Montes 2002)	0.0129730	0.0127420	0.0126810	5.90E−5
CPSO (He and Wang 2007b)	0.0129240	0.0127300	0.0126747	5.20E−4
HPSO (He and Wang 2007a)	0.0127190	0.0127072	0.0126652	1.58E−5
G-QPSO (Coelho 2010)	0.017759	0.013524	0.012665	0.001268
QPSO (Coelho 2010)	0.018127	0.013854	0.012669	0.001341
PSO (Coelho 2010)	0.071802	0.019555	0.012857	0.011662
SC (Sadollah et al. 2013)	0.016717272	0.012922669	0.012669249	5.9E−4
PSO-DE (Liu et al. 2010)	0.012665304	0.012665244	0.012665233	1.2E−8
ABC (Akay and Karaboga 2012; Fogel et al. 1966)	N.A.	0.012709	0.0126652	0.012813
TLBO (Rao et al. 2011)	N.A.	0.01266576	0.0126652	N.A.
MBA (Sadollah et al. 2013)	0.012900	0.012713	0.0126652	6.3E−5
CSA (Askarzadeh 2016)	0.0126701816	0.0126659984	0.0126652	1.357079E−6
WCMFO	**0.012670681**	**0.012666297**	**0.0126652**	**1.26293E−06**

Bold values indicate the best results

Table 12 The value of the decision variables and constraints in the best solution obtained by the WCMFO algorithm for spring design problem

Parameter	x_1	x_2	x_3	g_1	g_2	g_3	g_4	f
Value	0.051687	0.356669	11.29181	8.714E−09	−1.14E−08	−4.053	−0.727	0.0126652

$$
\begin{aligned}
\tau' &= \frac{P}{\sqrt{2x_1x_2}}, \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right)\\
R &= \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}\\
J &= 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}\\
\sigma(x) &= \frac{6PL}{x_4x_3^2}\\
\delta(x) &= \frac{6PL^3}{Ex_4x_3^2}\\
P_c(x) &= \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)\\
P &= 6000\ \text{lb}\\
L &= 14\ \text{in.}\\
\delta_{\max} &= 0.25\ \text{in.}\\
\tau_{\max} &= 13600\ \text{psi}\\
\sigma_{\max} &= 30000\ \text{psi}\\
E &= 30 \times 10^6\ \text{psi}\\
G &= 12 \times 10^6\ \text{psi}
\end{aligned}
\tag{19}
$$

The welded beam problem is optimized using the WCMFO algorithm, and the results are compared to different solution methods in the literature.

From Table 13, it is clear that the WCMFO algorithm outperforms the other solution methods in terms of obtaining better solution. Table 14 presents the value of the decision variables and constraints in the best solution obtained by the WCMFO algorithm.

As a summary, in this section the performance of the WCMFO was investigated in three well-known engineering problems. The results show that the WCMFO outperforms the other solution methods in all of these three engineering problems. These problems have various operational constraints; therefore, they show the efficiency of the proposed methodology in solving constraint problems.

The main reason which makes the WCMFO superior to the other solution methods is its efficient exploration and exploitation ability. In WCMFO, the streams and rivers update their position with respect to the sea. This ensures that all the search agents update their position towards the best solution. In addition, the spiral movement of streams and rivers around rivers and the sea help the search agents to explore and exploit the solution space of the problem more efficiently.

6 Conclusions

In this paper, a new hybrid metaheuristic algorithm named WCMFO is proposed for solving constrained and unconstrained problems. The WCMFO uses advantages of WCA

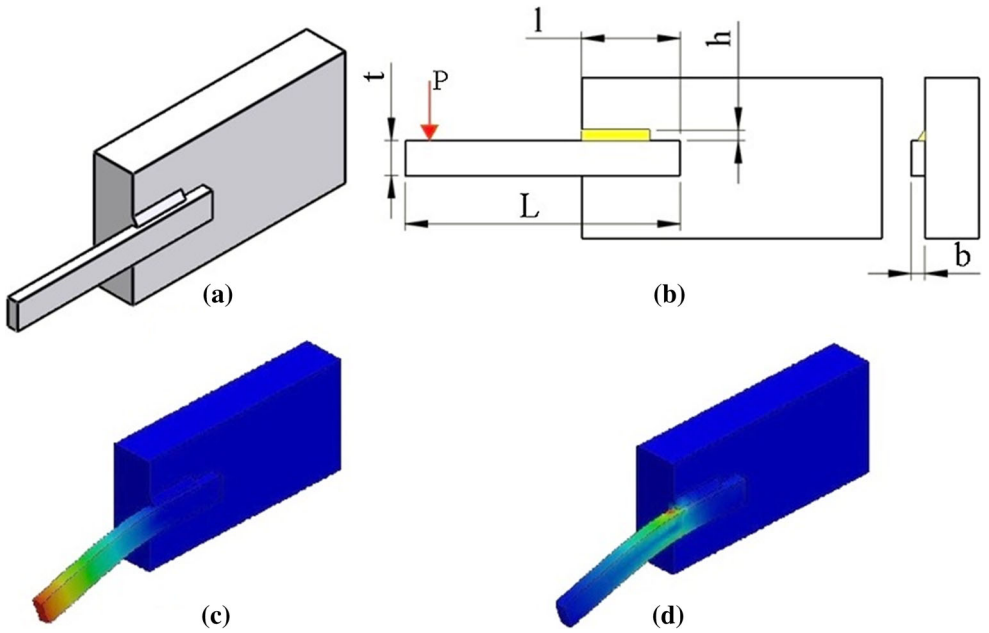

Fig. 17 **a** 3D view of the welded beam, **b** 2D view of the welded beam, **c** displacement heat map, **d** stress heat map

Table 13 Comparison of statistical results of the WCMFO algorithm and other algorithms for welded beam design problem

Method	Worst	Mean	Best	SD
GA3 (Coello 2000)	1.785835	1.771973	1.748309	1.12 E−2
GA4 (Coello and Montes 2002)	1.993408	1.792654	1.728226	7.47 E−2
CPSO (He and Wang 2007b)	1.782143	1.748831	1.728024	1.29E−2
HPSO (He and Wang 2007a)	1.814295	1.749040	1.724852	4.01E−2
CDE (Huang et al. 2007)	N.A.	1.76815	1.73346	N.A.
UPSO (Parsopoulos and Vrahatis 2005)	N.A.	2.83721	1.92199	0.683
PSO-DE (Liu et al. 2010)	1.724852	1.724852	1.724852	6.7 E−16
ABC (Akay and Karaboga 2012; Fogel et al. 1966)	N.A.	1.741913	1.724852	3.1e_2
TLBO (Rao et al. 2011)	N.A.	1.728446	1.724852	N.A.
MBA (Sadollah et al. 2013)	1.724853	1.724853	1.724853	6.94E−19
CSA (Askarzadeh 2016)	1.724852	1.724852	1.724852	1.19450E−15
SFS (Salimi 2015)	1.724852	1.724852	1.72485	7.7087E−16
WCMFO	**1.732704**	**1.724836**	**1.723583**	**0.001656**

Bold values indicate the best results

Table 14 The value of the decision variables and constraints in the best solution obtained by the WCMFO algorithm for welded beam design problem

Parameter	x_1	x_2	x_3	x_4	g_1	g_2
Value	0.206711	3.449553	9.03679	0.205731	−0.369	−1.325
Parameter	g_3	g_4	g_5	g_6	g_7	f
Value	0.0009	−3.434	−0.0811	−0.235	−0.2053	1.723583

and MFO in exploring and exploiting the solution space. To develop the WCMFO, first, the updating procedure of the WCA is modified by using the spiral movements of moths in the MFO. The spiral movement of streams around the rivers and rivers around the sea enables the WCMFO to exploit the solution space of the problem more efficiently. Second, to improve the exploration ability of the WCMFO, the streams are allowed to update their position using random walk (Levy flight) which increases the randomization in the algorithm. The performance of the WCMFO investigated

on unimodal, multimodal and fixed-dimension multimodal benchmark functions to show its efficiency in avoidance of trapping in local optima and exploration and exploitation abilities. The results show that the WCMFO is able to provide very competitive results comparing to the other state-of-the-art metaheuristic algorithms and outperform them in majority of the benchmark functions.

Since metaheuristic algorithms are commonly used by engineers to solve complex constrained engineering problems, in this paper, three well-known structural engineering problems (i.e. spring design, welded beam design and pressure vessel design) were solved to show the efficiency of the proposed WCMFO. The WCMFO was able to find very competitive solutions and outperform the other state-of-the-art metaheuristic algorithms.

This paper can lead to several research directions for future research. First, the binary and multiobjective versions of the WCMFO can be developed. Second, investigating the applicability of the proposed hybrid method in different fields would be worthwhile. Third, proposing discrete version of the WCMFO for discrete optimization (i.e. scheduling problems) is another direction for future research.

Compliance with ethical standards

Conflict of interest The authors declare that there is no conflict of interests regarding the publication of this paper.

References

Akay B, Karaboga D (2012) Artificial bee colony algorithm for large-scale problems and engineering design optimization. J Intel Manuf 23:1001–14. https://doi.org/10.1007/s10845-010-0393-4

Ali MZ, Awad NH, Suganthan PN, Duwairi RM, Reynolds RG (2016) A novel hybrid Cultural Algorithms framework with trajectory-based search for global numerical optimization. Inf Sci 334:219–249. https://doi.org/10.1016/j.ins.2015.11.032

Arora JS (2004) Introduction to optimum design. Academic Press, London

Askarzadeh A (2016) A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. Comput Struct 169:1–12. https://doi.org/10.1016/j.compstruc.2016.03.001

Belegundu AD (1983) Study of mathematical programming methods for structural optimization. Diss Abstr Int Part B Sci Eng 43

Cerný V (1985) Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm. J Optim Theory Appl 45:41–51. https://doi.org/10.1007/BF00940812

Coelho LDS (2010) Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. Expert Syst Appl 37:1676–1683. https://doi.org/10.1016/j.eswa.2009.06.044

Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Ind 41:113–127. https://doi.org/10.1016/S0166-3615(99)00046-9

Coello CAC (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191:1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1

Coello CA, Montes EM (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Inf 16:193–203. https://doi.org/10.1016/S1474-0346(02)00011-3

Digalakis J, Margaritis K (2001) On benchmarking functions for genetic algorithms. Int J Comput Math 77:481–506. https://doi.org/10.1080/00207160108805080

Du H, Wu X, Zhuang J (2006) Small-world optimization algorithm for function optimization. In: Jiao L, Wang L, Gao X, Liu J, Wu F (eds) Advances in natural computation. ICNC 2006. Lecture Notes in Computer Science, vol 4222. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11881223_33

Erol OK, Eksin I (2006) New optimization method: big bang–big crunch. Adv Eng Soft 37:106–111. https://doi.org/10.1016/j.advengsoft.2005.04.005

Eskandar H, Sadollah A, Bahreininejad A, Hamdi M (2012) Water cycle algorithm—a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 110:151–166. https://doi.org/10.1016/j.compstruc.2012.07.010

Fogel LJ, Owens AJ, Walsh MJ (1966) Artificial intelligence through simulated evolution. Wiley, London

Formato RA (2007) Central force optimization: a new metaheuristic with applications in applied electromagnetics. Prog Electromagn Res 77:425–91. https://doi.org/10.2528/PIER07082403

Frank KD, Rich C, Longcore T (2006) Effects of artificial night lighting on moths. In: Ecological consequences of artificial night lighting, Island Press, USA, pp 305–344

Gandomi AH, Yang XS, Alavi AH (2013) Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. Eng Comput 29:17–35. https://doi.org/10.1007/s00366-011-0241-y

Gao XZ, Wang X, Jokinen T, Ovaska SJ, Arkkio A, Zenger K (2012) A hybrid optimization method for wind generator design. Int J Innov Comput Inf Control 8:4347–4373

Gaston KJ, Bennie J, Davies TW, Hopkins J (2013) The ecological impacts of nighttime light pollution: a mechanistic appraisal. Biol Rev 88(2013):912–927. https://doi.org/10.1111/brv.12036

Hatamlou A (2013) Black hole: a new heuristic optimization approach for data clustering. Inf Sci 222:175–184. https://doi.org/10.1016/j.ins.2012.08.023

He Q, Wang L (2007a) A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. Appl Math Comput 186:1407–1422. https://doi.org/10.1016/j.amc.2006.07.134

He Q, Wang L (2007b) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. Eng Appl Artif Intell 20:89–99. https://doi.org/10.1016/j.engappai.2006.03.003

Holland JH (1992) Genetic algorithms. Sci Am 267:66–72

Huang FZ, Wang L, He Q (2007) An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 186:340–356. https://doi.org/10.1016/j.amc.2006.07.105

Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Opt 39:459–471. https://doi.org/10.1007/s10898-007-9149-x

Kaveh A, Dadras A (2017) A novel meta-heuristic optimization algorithm: thermal exchange optimization. Adv Eng Soft 110:69–84. https://doi.org/10.1016/j.advengsoft.2017.03.014

Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–94. https://doi.org/10.1016/j.compstruc.2012.09.003

Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213:267–289. https://doi.org/10.1007/s00707-009-0270-4

Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, pp 1942–1948

Khalilpourazari S, Khalilpourazary S (2016) Optimization of production time in the multi-pass milling process via a Robust Grey Wolf Optimizer. Neural Comput Appl. https://doi.org/10.1007/s00521-016-2644-6

Khalilpourazari S, Khalilpourazary S (2017a) A lexicographic weighted Tchebycheff approach for multi-constrained multi-objective optimization of the surface grinding process. Eng Optim 49:878–895. https://doi.org/10.1080/0305215X.2016.1214437

Khalilpourazari S, Khamseh AA (2017) Bi-objective emergency blood supply chain network design in earthquake considering earthquake magnitude: a comprehensive study with real world application. Ann Oper Res. https://doi.org/10.1007/s10479-017-2588-y

Khalilpourazari S, Pasandideh SHR (2017) Multi-item EOQ model with nonlinear unit holding cost and partial backordering: moth-flame optimization algorithm. J Ind Prod Eng 34:42–51. https://doi.org/10.1080/21681015.2016.1192068

Khalilpourazari S, Khalilpourazary S (2017b) A Robust Stochastic Fractal Search approach for optimization of the surface grinding process. Swarm Evol Comput. https://doi.org/10.1016/j.swevo.2017.07.008

Khalilpourazari S, Mohammadi M (2016) Optimization of closed-loop Supply chain network design: a water cycle algorithm approach. In: 2th international conference on industrial engineering. IEEE, pp 41–45. https://doi.org/10.1109/INDUSENG.2016.7519347

Khalilpourazari S, Pasandideh SHR (2016) Bi-objective optimization of multi-product EPQ model with backorders, rework process and random defective rate. In: 2th international conference on industrial engineering. IEEE, pp 36–40. https://doi.org/10.1109/INDUSENG.2016.7519346

Khalilpourazary S, Kashtiban PM, Payam N (2014a) Optimization of turning operation of St37 steel using grey relational analysis. J Comput Appl Res Mech Eng 3:135–144. https://doi.org/10.22061/JCARME.2014.67

Khalilpourazary S, Abdi Behnagh R, Mahdavinejad RA, Payam N (2014b) Dissimilar friction stir lap welding of Al–Mg to CuZn34: application of grey relational analysis for optimization of process parameters. J Comput Appl Res Mech Eng 4:81–88. https://doi.org/10.22061/JCARME.2014.74

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simmulated annealing. Science 220:671–680

Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge

Li MD, Zhao H, Weng XW, Han T (2016) A novel nature-inspired algorithm for optimization: virus colony search. Adv Eng Soft 92:65–88. https://doi.org/10.1016/j.advengsoft.2015.11.004

Liu C, Linan F (2016) A hybrid evolutionary algorithm based on tissue membrane systems and CMA-ES for solving numerical optimization problems. Knowl Based Sys 105:38–47. https://doi.org/10.1016/j.knosys.2016.04.025

Liu H, Cai Z, Wang Y (2010) Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. Appl Soft Comput 10:629–640. https://doi.org/10.1016/j.asoc.2009.08.031

Mirjalili S (2015a) Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm. Knowl Based Sys 89:228–249. https://doi.org/10.1016/j.knosys.2015.07.006

Mirjalili S (2015b) The ant lion optimizer. Adv Eng Soft 83:80–98. https://doi.org/10.1016/j.advengsoft.2015.01.010

Mirjalili S (2016a) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete and multi-objective problems. Neural Comput Appl 27:1053–1073. https://doi.org/10.1007/s00521-015-1920-1

Mirjalili S (2016b) SCA: a sine cosine algorithm for solving optimization problems. Knowled Based Sys 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022

Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Soft 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008

Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Soft 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007

Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27:495–513. https://doi.org/10.1007/s00521-015-1870-7

Mirjalili S, Hashim SZM (2010) A new hybrid PSOGSA algorithm for function optimization. In: Computer and information application (ICCIA). IEEE, pp 374–377. https://doi.org/10.1109/ICCIA.2010.6141614

Moghaddam FF, Moghaddam RF, Cheriet M (2012) Curved space optimization: a random search based on general relativity theory. arXiv preprint arXiv:1208.2214

Molga M, Smutnicki C (2005) Test functions for optimization needs, p 101

Parsopoulos K, Vrahatis M (2005) Unified particle swarm optimization for solving constrained engineering optimization problems. In: Advances in natural computation, pp 582–591. https://doi.org/10.1007/11539902_71

Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43:303–315. https://doi.org/10.1016/j.cad.2010.12.015

Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179:2232–2248. https://doi.org/10.1016/j.ins.2009.03.004

Rechenberg I (1978) Evolutionsstrategien. Springer, Berlin

Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: a new population based algorithm for solving constrained engineering optimization problems. Appl Soft Comput 13:2592–2612. https://doi.org/10.1016/j.asoc.2012.11.026

Sadollah A, Eskandar H, Bahreininejad A, Kim JH (2015a) Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures. Comput Struct 149:1–16. https://doi.org/10.1016/j.compstruc.2014.12.003

Sadollah A, Eskandar H, Bahreininejad A, Kim JH (2015b) Water cycle algorithm for solving multi-objective optimization problems. Soft Comput 19:2587–2603. https://doi.org/10.1007/s00500-014-1424-4

Sadollah A, Eskandar H, Bahreininejad A, Kim JH (2015c) Water cycle algorithm with evaporation rate for solving constrained and unconstrained optimization problems. Appl Soft Comput 30:58–71

Sadollah A, Eskandar H, Kim JH (2015d) Water cycle algorithm for solving constrained multi-objective optimization problems. Appl Soft Comput 27:279–298. https://doi.org/10.1016/j.asoc.2014.10.042

Salimi H (2015) Stochastic fractal search: a powerful metaheuristic algorithm. Knowl Based Sys 75:1–18. https://doi.org/10.1016/j.knosys.2014.07.025

Saremi S, Mirjalili S, Lewis A (2017) Grasshopper optimization algorithm: theory and application. Adv Eng Soft 105:30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004

Simon D (2008) Biogeography-based optimization. IEEE Trans Evol Comput 12:702–713. https://doi.org/10.1109/TEVC.2008.919004

Topal AO, Altun O (2016) A novel meta-heuristic algorithm: dynamic virtual bats algorithm. Inf Sci 354:222–235. https://doi.org/10.1016/j.ins.2016.03.025

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

Wang GG, Guo L, Gandomi AH, Hao GS, Wang H (2014) Chaotic krill herd algorithm. Inf Sci 274:17–34. https://doi.org/10.1016/j.ins.2014.02.123

Wang GG, Gandomi AH, Alavi AH (2014b) An effective krill herd algorithm with migration operator in biogeography-based optimization. Appl Math Model 38:2454–2462. https://doi.org/10.1016/j.apm.2013.10.052

Wang GG, Gandomi AH, Alavi AH, Hao GS (2014c) Hybrid krill herd algorithm with differential evolution for global numerical optimization. Neural Comput Appl 25:297–308. https://doi.org/10.1007/s00521-013-1485-9

Wang GG, Gandomi AH, Zhao X, Chu HC (2016) Hybridizing harmony search algorithm with cuckoo search for global numerical optimization. Soft Comput 20:273–285. https://doi.org/10.1007/s00500-014-1502-7

Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: World congress on IEEE nature & biologically inspired computing. https://doi.org/10.1109/NABIC.2009.5393690

Yang X-S (2010a) Appendix A: test problems in optimization. In: Engineering optimization, Wiley, Hoboken, NJ, USA. https://doi.org/10.1002/9780470640425.app1

Yang XS (2010b) Nature-inspired metaheuristic algorithms. Luniver Press, London

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3:82–102. https://doi.org/10.1109/4235.771163

Zareh-Desari B, Abaszadeh-Yakhforvazani M, Khalilpourazary S (2015) The effect of nanoparticle additives on lubrication performance in deep drawing process: evaluation of forming load, friction coefficient and surface quality. Int J Precis Eng Manuf 16:929–936. https://doi.org/10.1007/s12541-015-0121-2