Knowledge-Based Systems 233 (2021) 107555

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Enhanced Jaya algorithm: A simple but efficient optimization method for constrained engineering design problems

Yiying Zhang [a,*], Aining Chi [b], Seyedali Mirjalili [c,d]

[a] School of Electrical and Information Engineering, Jiangsu University, Zhenjiang 212013, China
[b] College of Computer Science and Technology, Taizhou University, Taizhou 225300, China
[c] Centre for Artificial Intelligence Research and Optimization, Torrens University Australia, Fortitude Valley, Brisbane, 4006 QLD, Australia
[d] Yonsei Frontier lab, Yonsei University, Seoul, South Korea

ARTICLE INFO

Article history:
Received 2 July 2020
Received in revised form 29 September 2021
Accepted 30 September 2021
Available online 2 October 2021

Keywords:
Jaya algorithm
Swarm intelligence
Evolutionary computing
Global optimization
Metaheuristics

ABSTRACT

Jaya algorithm (JAYA) is a new metaheuristic algorithm, which has a very simple structure and only requires population size and terminal condition for optimization. Given the two features, JAYA has been widely used to solve various types of optimization problems. However, JAYA may easily get trapped in local optima for solving complex optimization problems due to its single learning strategy with little population information. To improve the global search ability of JAYA, this work proposes an enhanced Jaya algorithm (EJAYA) for global optimization. In EJAYA, the local exploitation is based on defined upper and lower local attractors and global exploration is guided by historical population. Like JAYA, EJAYA does not need any effort for fine tuning initial parameters. To check the performance of the proposed EJAYA, EJAYA is first used to solve 45 test functions extracted from the well-known CEC 2014 and CEC 2015 test suites. Then EJAYA is employed to solve seven challenging real-world engineering design optimization problems. Experimental results support the strong ability of EJAYA to escape from the local optimum for solving complex optimization problems and the effectively of the introduced improved strategies to JAYA. Note that, the source codes of the proposed EJAYA are publicly available at https://ww2.mathworks.cn/matlabcentral/fileexchange/88877-enhanced-jaya-algorithm-for-global-optimization.

1. Introduction

Metaheuristic algorithms commonly operate by the combination between the defined simple rules and randomness to simulate natural phenomena [1]. Note that, metaheuristic algorithms have remarkable advantages over traditional optimization algorithms in the following two aspects:

- Simplicity. Traditional optimization algorithms usually first obtain some basic information of the given problems (e.g. gradient matrix), and then perform the search process according to some strict mathematical theories [2]. Unlike traditional optimization algorithms, metaheuristic algorithms rely on the cooperation of simple rules and randomness to execute the search task.
- Efficiency. Given the sensibility of traditional optimization algorithms to the initial solutions, traditional optimization algorithms are easily trapped in local optima for complex optimization problems with more than one local optimal solutions [2–5]. Here, it should be pointed out that randomness is a significant feature of metaheuristic algorithms, which is very effective in reducing the impact of the initial solutions on the final solutions obtained by metaheuristic algorithms. Therefore, it has been proven that metaheuristic algorithms can achieve far better solutions than traditional optimization algorithms for multimodal optimization problems [1,6].

Given the two advantages, in the last twenty years, many novel metaheuristic algorithms have been developed and applied to a lot of engineering optimization problems in the real life, such as team orienteering problem with time windows and partial scores [7], high-order graph matching [8], parameter estimation of photovoltaic systems [9], travelling salesman optimization [10], uncertain integrated process planning and scheduling with interval processing time [11], and economic load dispatch of power system [12]. Significantly, when one metaheuristic algorithm is used to solve an optimization problem, the optimization process can be seen as a black box. The input information consisting of basic information of the problem and parameters of the algorithm is at one end of the black box and the output information (i.e. the optimal solution) is at the other end of

* Corresponding author.
E-mail addresses: n1906817e@e.ntu.edu.sg, zhangyiying@ujs.edu.cn (Y. Zhang), b02016@tzu.edu.cn (A. Chi), ali.mirjalili@gmail.com (S. Mirjalili).

https://doi.org/10.1016/j.knosys.2021.107555
0950-7051/

the black box. The basic information of one problem usually includes the number of variables, the objective function, the boundaries of the lower and upper of the variables, and some constrained conditions about variables. The parameters of one metaheuristic algorithm can be divided into the following two broad categories [13]:

- Common parameters. These parameters are required for every metaheuristic algorithm, which usually include population size and terminal condition (e.g. the number of iterations, the number of function evaluations or some given precision).
- Special parameters. Special parameters reflect the features of algorithms, such as crossover rate and mutation rate in differential evolution [14,15], harmony memory consideration rate in harmony search [16,17], light absorption coefficient in firefly algorithm [18], discovery rate in cuckoo search [19,20], and cluster number in henry gas solubility optimization [21].

Our previous works [13,22] pointed out that most existing metaheuristic algorithms need special parameters and the greatest challenge for metaheuristic algorithms with special parameters: how to set their special parameters for an unknown optimization problem to find the optimal solution ? The reason emerging this challenge is that every optimization problem has its distinct features. Thus, setting different values for the special parameters in solving different optimization problems is a common phenomenon. Take the classical particle swarm optimization (PSO) as an example, inertia weight is a special parameter of PSO and its variants. Inertia weight was from 0.9 to 0.7 and from 0.99 to 0.2 for comprehensive learning particle swarm optimization [23] and heterogeneous comprehensive learning particle swarm [24], respectively. Inertia weight was set to 0.01 and 0.72984 for density-based particle swarm optimization algorithm [25] and enhancing particle swarm optimization [26], respectively. Moreover, the hybrid algorithms based on several different algorithms have been proven to be effective [18,27,28]. When metaheuristic algorithms with special parameters are hybridized with other algorithms, the produced hybrid algorithms still face the mentioned challenge. Obviously, it is clear that the applications of some metaheuristic algorithms will be restricted due to their special parameters.

Jaya algorithm (JAYA) [29] is one of few metaheuristic algorithms without special parameters. JAYA is inspired by the concept that the solution obtained for a given problem should move towards the best solution and should avoid the worst solution. In addition, JAYA has a very simple structure and its effectiveness for optimization problems has been proven [30–32]. In view of the advantages of JAYA, many variants of JAYA have been presented to solve different types of optimization problems. Kishor et al. [33] designed an efficient JAYA algorithm with Lévy flight (JAYALF) for non-linear channel equalization. To solve the online load frequency control in wind integrated power systems, a modified JAYA optimization algorithm was reported (MJOA) [34]. Rao and Hameer [35] used a multi-team perturbation-guiding Jaya algorithm (MTPG-JAYA) for optimization of wind farm layout. An opposition-based JAYA with population reduction (PRJAYA) was proposed for parameter estimation of photovoltaic solar cells and modules in [36]. A Jaya algorithm with mutation and extreme learning machine (JAYA-ELM) for sensorineural hearing loss detection was presented in [37]. Note that, these reported variants usually introduce some special parameters to JAYA, such as step size of Lévy flight in JAYALF, inertia weight in MJOA, jumping rate in PRJAYA, and mutation variable in JAYA-ELM. As mentioned above, the practical applications of these variants with special parameters will be restricted.

Motivated by the above discussion, this paper presents an enhanced Jaya algorithm (EJAYA) for global optimization. Guiding the search direction of the population only by the current best solution and the current worst solution is the main reason of basic JAYA suffering from trapping in local minima. Unlike basic JAYA, EJAYA can use the population information more efficiently to balance its local exploitation and global exploration. The used population information in EJAYA includes the current best solution, the current worst solution, the current mean solution, and the historical solutions. In addition, like JAYA, EJAYA only needs the essential population size and terminal condition for optimization, which can distinguish EJAYA over most reported variants of JAYA. To verify the performance of EJAYA, EJAYA is first investigated by the well-known CEC 2014 test suite [38] and CEC 2015 test suite [39]. Then the performance of EJAYA is checked by seven challenging real-world engineering design problems. Experimental results have proven the superiority of the proposed EJAYA for complex optimization problems by comparing with several state-of-the-art metaheuristic algorithms.

The rest of this paper is organized as follows. Basic JAYA is introduced in Section 2. Section 3 describes the detailed implementation of EJAYA. EJAYA is evaluated on numerical optimization problems in Section 4. Section 5 presents the applications of EJAYA on real-world engineering optimization problems. Lastly, the conclusion is made in Section 6.

2. Basic JAYA

As mentioned previously, JAYA uses a simple learning strategy to complete its search process. This strategy can be written as [29]:

$$\boldsymbol{v}_i = \boldsymbol{x}_i + \lambda_1 \times (\boldsymbol{x}_{\text{Best}} - |\boldsymbol{x}_i|) - \lambda_2 \times (\boldsymbol{x}_{\text{Worst}} - |\boldsymbol{x}_i|),\ i = 1, 2, 3, \ldots, N \quad (1)$$

where λ_1 and λ_2 are two random numbers between 0 and 1, N is the population size, $\boldsymbol{x}_{\text{Best}}$ is the current best solution, $\boldsymbol{x}_{\text{Worst}}$ is the current worst solution, $\boldsymbol{x}_i$ is the solution of the ith individual and $\boldsymbol{v}_i$ is the trail vector of the ith individual. According to the authors of JAYA, the second term on the right of Eq. (1) means the tendency of the solution $\boldsymbol{x}_i$ to move closer the current best solution $\boldsymbol{x}_{\text{Best}}$ and the third term on the right of Eq. (1) indicates the tendency of the solution $\boldsymbol{x}_i$ to move away from the current worst solution $\boldsymbol{x}_{\text{Worst}}$. Moreover, in order to accelerate the convergence speed, the better solutions are selected into the next generation, which can be expressed as

$$\boldsymbol{x}_i = \begin{cases} \boldsymbol{v}_i, & \text{if } f(\boldsymbol{v}_i) \le f(\boldsymbol{x}_i) \\ \boldsymbol{x}_i, & \text{otherwise} \end{cases} \quad (2)$$

where $f(\cdot)$ is the objective function of the given problem. Like other population-based metaheuristic algorithms, population $\boldsymbol{x}_i$ in JAYA is initialized by

$$\boldsymbol{x}_i = \boldsymbol{l} + (\boldsymbol{u} - \boldsymbol{l}) \times \lambda_3,\ i = 1, 2, 3, \ldots, N \quad (3)$$

where λ_3 is a random number with uniform distribution, $\boldsymbol{u}$ and $\boldsymbol{l}$ are the upper limits of variables the lower limits of variables, respectively. The detailed implementation of basic JAYA has been shown in Fig. 1.

3. The proposed EJAYA

3.1. Motivation

Although the reported metaheuristic algorithms have different search strategies, they usually complete the search process for the given problem by balancing their global exploration and local exploitation. Local exploitation is to search better solutions around the current search space consisting of the current population

Input: population size N, the upper limits of variables $\boldsymbol{u}$, the lower limits of variables $\boldsymbol{l}$, the current number of function evaluations $T_{\text{current}}=0$ and the maximum number of function evaluations $T_{\max}$.

/ Initialization */*
Initialize individual $\boldsymbol{x}_i$ by Eq. (3)
Calculate the fitness value of every individual and achieve the optimal solution $\boldsymbol{x}_{\text{Best}}$
Update the current number of function evaluations T_{current} by $T_{\text{current}}=T_{\text{current}}+N$
/ Main loop */*
While $T_{\text{current}}<T_{\max}$ **do**
  For $i=1:N$
    Compute the trail vector $\boldsymbol{v}_i$ of the ith individual by Eq. (1).
    Select the better solution from $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$ by Eq. (2).
  End for
  Update the current number of function evaluations T_{current} by $T_{\text{current}}=T_{\text{current}}+N$
End while

Output: The optimal solution $\boldsymbol{x}_{\text{Best}}$

Fig. 1. The detailed implementation of basic JAYA.

information. Global exploration is to search better solutions in the whole feasible solution space. If an algorithm pays more attention on local exploitation, this algorithm may be trapped in local optima. Otherwise, if an algorithm is more focused on global exploration, its convergence speed will be significantly reduced. Thus how to balance local exploitation and global exploration is critical for an algorithm to find the optimal solution of the given problem.

For JAYA, local exploitation and global exploration are performed by Eq. (1). However, the following disadvantages need to be considered in Eq. (1):

- JAYA only uses the current best individual and the current worst individual to determine the search direction of the population. Once the current best individual gets trapped in local optima, the whole population has more chance to find a local optimal solution.
- The absolute value symbol plays a very important role in keeping the population diversity of JAYA. Note that, the upper and lower limits of the design variables for engineering problems are more than zero. That is, when JAYA is employed to solve these optimization problems, absolute symbol is invalidated and the risk of premature convergence for JAYA will increase.

In order to overcome the two disadvantages, EJAYA is proposed to enhance the global search ability of basic JAYA by making full use of population information.

3.2. The designed search mechanism

Fig. 2 shows the framework of EJAYA. Obviously, EJAYA has a very simple structure and the core of EJAYA is the search mechanism consisting of local exploitation and global exploration strategies by making full use of population information. The designed search mechanism includes local exploitation strategy and global exploration strategy. Next, the detailed description for the two strategies is given.

3.2.1. Local exploitation strategy

In order to avoid the potential risk caused by absolute value symbol, local exploitation strategy in EJAYA is designed by the defined lower and upper local attractors.

● The upper local attract point. This parameter is to describe the solution between the current best solution and the current mean solution, which can be denoted as

$$\boldsymbol{P}_{\text{u}} = \lambda_3 \times \boldsymbol{x}_{\text{Best}} + (1-\lambda_3) \times \boldsymbol{M} \tag{4}$$

where λ_3 is a random number with uniform distribution between 0 and 1, $\boldsymbol{P}_{\text{u}}$ is the upper local attract point, $\boldsymbol{x}_{\text{Best}}$ is the current best solution, and $\boldsymbol{M}$ is the current mean solution. Assume there is a population $\boldsymbol{X}$ including N individuals, i.e. $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ and $\boldsymbol{M}$ is computed by

$$\boldsymbol{M} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i, i = 1, 2, \ldots, N \tag{5}$$

● The lower local attract point. This parameter is to represent the solution between the current worst solution and the current mean solution, which can be written as

$$\boldsymbol{P}_{\text{l}} = \lambda_4 \times \boldsymbol{x}_{\text{Worst}} + (1-\lambda_4) \times \boldsymbol{M} \tag{6}$$

where λ_4 is a random number with uniform distribution between 0 and 1, $\boldsymbol{P}_{\text{l}}$ is the lower local attract point, and $\boldsymbol{x}_{\text{Worst}}$ is the current worst solution.

Based on the defined upper and lower attract points, the local exploitation strategy of EJAYA can be expressed as

$$\boldsymbol{v}_i = \boldsymbol{x}_i + \lambda_5 \times (\boldsymbol{P}_{\text{u}} - \boldsymbol{x}_i) - \lambda_6 \times (\boldsymbol{P}_{\text{l}} - \boldsymbol{x}_i), i = 1, 2, \ldots, N \tag{7}$$

where λ_5 and λ_6 are two random numbers with uniform distribution between 0 and 1, $\boldsymbol{v}_i$ is the trail vector of the ith individual. The second term on the right side of Eq. (7) indicates that the solution $\boldsymbol{x}_i$ is pulled in the direction of the current optimal solution and the third term on the right side of Eq. (7) means that the solution $\boldsymbol{x}_i$ is pulled out from the direction of the current worst solution.

3.2.2. Global exploration strategy

Along with the increasing of iteration times, most individuals gradually move closer to the current best individual in the later evolution period for metaheuristic algorithms. At this moment, once the obtained solutions get trapped in a local minimum, they are unable to escape. Motivated by backtracking search algorithm [40], to enhance global exploration ability of JAYA, the core idea of the designed global exploration strategy in EJAYA is that the differential vectors between historical population and current population have more search space than differential vectors among the same generation population. To better achieve this idea, as done in [40], historical population is first generated by a random selection method and then a random shuffling function is used to re-order the individuals of historical population. This strategy can be described as follows:

● Generate historical population. The historical population $\boldsymbol{X}_{\text{old}} (\boldsymbol{X}_{\text{old}} = \{\boldsymbol{x}_{\text{old},1}, \boldsymbol{x}_{\text{old},2}, \ldots \boldsymbol{x}_{\text{old},N}\})$ is first generated by

$$\boldsymbol{X}_{\text{old}} = \begin{cases} \boldsymbol{X}, & \text{if } P_{\text{switch}} <= 0.5 \\ \boldsymbol{X}_{\text{old}}, & \text{otherwise} \end{cases} \tag{8}$$

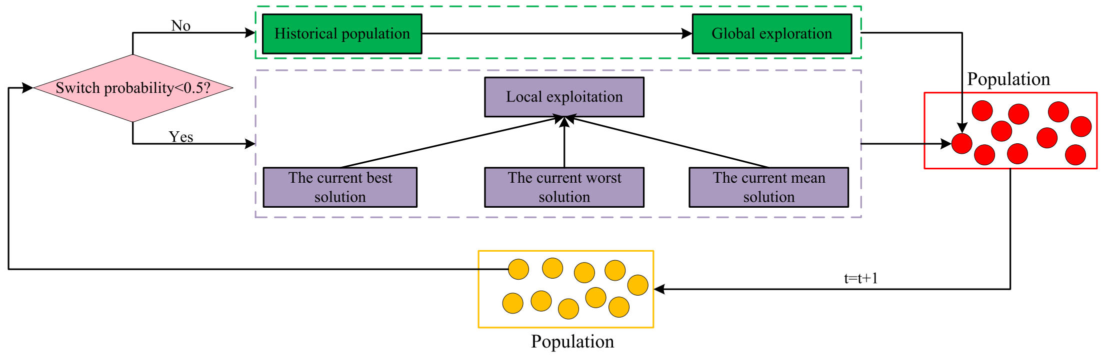

Fig. 2. The framework of the proposed EJAYA (t is the current number of iterations).

where $\boldsymbol{X}_{\text{old}}$ is the historical population and P_{switch} is called switch probability (a random number with uniform distribution between 0 and 1). Then $\boldsymbol{X}_{\text{old}}$ is processed by

$$\boldsymbol{X}_{\text{old}} = \text{permuting}\left(\boldsymbol{X}_{\text{old}}\right) \tag{9}$$

where permuting is a random shuffling function. Further, permuting is to sort all individuals of $\boldsymbol{X}_{\text{old}}$ in random order.

● Global exploration strategy. The global exploration strategy of EJAYA can be expressed as

$$\boldsymbol{v}_i = \boldsymbol{x}_i + \kappa \times (\boldsymbol{x}_{\text{old},i} - \boldsymbol{x}_i), i = 1, 2, \ldots, N \tag{10}$$

where κ is a random number subject to standard normal distribution. Metaheuristic algorithms usually work based on some simple rules and randomness to simulate natural phenomena [1]. Thus, randomness is a very remarkable feature of metaheuristic methods, which is embodied by the introduced random numbers. It is most common that random numbers between 0 and 1 obey uniform distribution. When generating random numbers, standard normal distribution has a stronger volatility than uniform distribution. Thus, random numbers with standard normal distribution outperform random numbers with uniform distribution in terms of the ability of escaping from the local optimal solutions. In addition, random numbers with standard normal distribution have been used in many reported metaheuristic methods and get promising results, such as backtracking search algorithm [40], generalized normal distribution optimization [41], and neural network algorithm with reinforcement learning [42].

3.3. The implementation of EJAYA for optimization

In the proposed EJAYA, local exploitation strategy and global exploration strategy have the same importance. Let a random number P_{select} with uniform distribution the between 0 and 1 indicates the selected probability. If P_{select} is more than 0.5, the local exploitation strategy is executed; otherwise, the global exploration strategy is performed. Moreover, the population $\boldsymbol{X}$ is initialized by Eq. (2). Note that the historical population $\boldsymbol{X}_{\text{old}}$ is initialized by the same method with the population $\boldsymbol{X}$ in EJAYA. The detailed implementation of EJAYA has been presented in Fig. 3. Based on Fig. 3, the implementation of EJAYA for optimization can be described as follows.

Step 1: Initialization

Initialization information includes population size N, **t**he upper limits of variables $\boldsymbol{u}$, **t**he lower limits of variables $\boldsymbol{l}$, the number of variables D the current number of function evaluations T_{current} and the maximum number of function evaluations $T_{\max}$. Moreover, the population $\boldsymbol{X}$ and historical population $\boldsymbol{X}_{\text{old}}$ are initialized by Eq. (2).

Step 2: Population evaluation

The fitness value of every individual is computed and the optimal solution $\boldsymbol{x}_{\text{Best}}$ is selected.

Step 3: Update the current number of function evaluations

The current number of function evaluations T_{current} is updated by $T_{\text{current}} = T_{\text{current}} + N$.

Step 4: Stop condition

If the current number of function evaluations T_{current} is more than $T_{\max}$, the algorithm stops and the optimal solution $\boldsymbol{x}_{\text{Best}}$ is output; otherwise Step 5 is executed.

Step 5: Local exploitation and Global exploration strategies

The selected probability P_{select} is generated. If P_{select} is more than 0.5, local exploitation strategy is performed by Eqs. (4)–(7) and Eq. (2); otherwise, global exploration strategy is performed by Eqs. (8)–(10) and Eq. (2).

Step 6: Go to Step 3.

3.4. Comparisons between JAYA and EJAYA

As an improved version of JAYA, EJAYA keeps the advantages of basic JAYA, which can be described as follows: (1) EJAYA only needs the essential population size and stopping criterion, which does not need extra control parameters; (2) EJAYA does not involve tedious calculations and complex logic. In addition, EJAYA has the following two remarkable advantages over JAYA:

- Mean position of the whole population. In the basic JAYA, as shown in Eq. (1), generating the next generation population is based on the guidance of the current best individual and the current worst individual. In other words, the designed search mechanism in the original JAYA does not make the best of the obtained population information. Thus, once the current best individual is trapped into a local optimum, JAYA is easy to premature convergence. Unlike the basic JAYA, mean position of the whole population is introduced to the proposed EJAYA by upper local attract point and lower local attract point defined in Eqs. (4) and (6). Further, the designed local exploitation strategy in EJAYA can generate the next generation population by considering the positions of all individuals in the population, which is very helpful for avoiding premature convergence.
- Historical population. To further enhance global search ability of EJAYA, global exploration strategy based on historical population is proposed. As mentioned in Section 3.2.2, in the later evolution period, most individuals gradually move closer to the current best individual for a metaheuristic algorithm. That is, at this moment, if the obtained current optimal solution is a local optimal solution and the algorithm has weak global search ability, the algorithm has more chance to find a local optimal solution. The basic JAYA only uses single search strategy to complete the search task, which does not take steps to improve its global search ability in the later evolution period. Given this drawback of the original JAYA, EJAYA employs historical population

Input: population size N, the upper limits of variables $\boldsymbol{u}$, the lower limits of variables $\boldsymbol{l}$, the current number of function evaluations $T_{\text{current}} = 0$ and the maximum number of function evaluations $T_{\max}$.

/* Initialization */

Initialize population $\boldsymbol{X}$ and historical population $\boldsymbol{X}_{\text{old}}$ by Eq. (3)

Calculate the fitness value of every individual and achieve the optimal solution $\boldsymbol{x}_{\text{Best}}$

Update the current number of function evaluations T_{current} by $T_{\text{current}} = T_{\text{current}} + N$

/* Main loop */

While $T_{\text{current}} < T_{\max}$ **do**

  For $i = 1 : N$

    Generate the selected probability P_{select}

   If $P_{\text{select}} > 0.5$

/* Local exploitation strategy */

    Select the current optimal solution $\boldsymbol{x}_{\text{Best}}$ and the current worst solution $\boldsymbol{x}_{\text{Worst}}$

    Compute the current mean solution by Eq. (5)

    Perform the local exploitation strategy by Eq. (4), Eq. (6), Eq. (7) and Eq. (2)

   Else

/* Global exploration strategy */

    Perform the global exploration strategy by Eq. (8), Eq. (9), Eq. (10) and Eq. (2)

   End if

  End for

  Update the current number of function evaluations T_{current} by $T_{\text{current}} = T_{\text{current}} + N$

End while

Output: the optimal solution $\boldsymbol{x}_{\text{Best}}$

Fig. 3. The detailed implementation of the proposed EJAYA.

Table 1
The description for CEC 2014 test suite.

No.	Types	Name	Dimension	Range	Optimum
F1	Unimodal functions	Rotated high conditioned elliptic function	30	[−100,100]	100
F2		Rotated bent cigar function	30	[−100,100]	200
F3		Rotated discus function	30	[−100,100]	300
F4	Simple multimodal functions	Shifted and rotated Rosenbrock's function	30	[−100,100]	400
F5		Shifted and rotated Ackley's function	30	[−100,100]	500
F6		Shifted and rotated Weierstrass function	30	[−100,100]	600
F7		Shifted and rotated Griewank's function	30	[−100,100]	700
F8		Shifted Rastrigin's function	30	[−100,100]	800
F9		Six Hump Camel Back	30	[−100,100]	900
F10		Shifted and rotated Rastrigin's function	30	[−100,100]	1000
F11		Shifted and rotated Schwefel's function	30	[−100,100]	1100
F12		Shifted and rotated Katsuura function	30	[−100,100]	1200
F13		Shifted and rotated HappyCat function	30	[−100,100]	1300
F14		Shifted and rotated HGBat function	30	[−100,100]	1400
F15		Shifted and rotated Expanded Griewank's plus Rosenbrock's function	30	[−100,100]	1500
F16		Shifted and rotated Expanded Schaffer's F6 function	30	[−100,100]	1600
F17	Hybrid functions	Hybrid function 1 (m= 3)	30	[−100,100]	1700
F18		Hybrid function 2 (m = 3)	30	[−100,100]	1800
F19		Hybrid function 3 (m = 4)	30	[−100,100]	1900
F20		Hybrid function 4 (m = 4)	30	[−100,100]	2000
F21		Hybrid function 5 (m= 5)	30	[−100,100]	2100
F22		Hybrid function 6 (m = 5)	30	[−100,100]	2200
F23	Composition functions	Composition function 1 (m=5)	30	[−100,100]	2300
F24		Composition function 2 (m=3)	30	[−100,100]	2400
F25		Composition function 3 (m=3)	30	[−100,100]	2500
F26		Composition function 4 (m=5)	30	[−100,100]	2600
F27		Composition function 5 (m=5)	30	[−100,100]	2700
F28		Composition function 6 (m=5)	30	[−100,100]	2800
F29		Composition function 7 (m=3)	30	[−100,100]	2900
F30		Composition function 8 (m=3)	30	[−100,100]	3000

processed by the random shuffling function to enhance its global search ability, which can increase the chance of EJAYA to escape from the local optimal solution.

4. EJAYA for numerical optimization problems

This section is to evaluate the performance of EJAYA for 45 numerical optimization problems that are extracted from the well-known CEC 2014 and CEC 2015 test suites. This section consists of the following three subsections. The first subsection describes the used numerical optimization problems. The compared algorithms and the evaluation metrics are given in the second subsection. The last subsection discusses the experimental results.

4.1. Numerical optimization problems

CEC 2014 and CEC 2015 test suites as listed in Tables 1 and 2 are often employed for checking the performance of the different algorithms for complex optimization problems [43–46]. There are four types of optimization problems (i.e. unimodal functions, simple multimodal functions, hybrid functions and composition functions) in the CEC 2014 and CEC 2015 test suites. More specially, CEC 2014 test suite has 30 functions and consists of three

Table 2
The description for CEC 2015 test suite.

No.	Types	Name	Dimension	Range	Optimum
F31	Unimodal functions	Rotated Bent Cigar Function	30	[−100,100]	100
F32		Rotated Discus Function	30	[−100,100]	200
F33	Simple multimodal functions	Shifted and Rotated Weierstrass Function	30	[−100,100]	300
F34		Shifted and Rotated Schwefel's Function	30	[−100,100]	400
F35		Shifted and Rotated Katsuura Function	30	[−100,100]	500
F36		Shifted and Rotated HappyCat Function	30	[−100,100]	600
F37		Shifted and Rotated HGBat Function	30	[−100,100]	700
F38		Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	30	[−100,100]	800
F39		Shifted and Rotated Expanded Schaffer's F6 Function	30	[−100,100]	900
F40	Hybrid functions	Hybrid Function 1 (m=3)	30	[−100,100]	1000
F41		Hybrid Function 2 (m=4)	30	[−100,100]	1100
F42		Hybrid Function 3 (m=5)	30	[−100,100]	1200
F43	Composition functions	Composition Function 1 (m=5)	30	[−100,100]	1300
F44		Composition Function 2 (m=3)	30	[−100,100]	1400
F45		Composition Function 3 (m=5)	30	[−100,100]	1500

unimodal functions (F1–F3), 13 simple multimodal functions (F4–F16), six hybrid functions (F17–F22) and eight composition functions (F23–F30). There are 15 functions in the CEC 2015 test suite, which are two unimodal functions (F31–F32), seven simple multimodal functions (F33–F39), three hybrid functions (F40–F42) and three composition functions (F43–F45). Compared with unimodal functions, multimodal functions with more than one local optima are more complex. Note that 27 of 30 functions in the CEC 2014 test suite and 13 of 15 functions in the CEC 2015 test suite are multimodal functions. Thus the two test suites are very suitable for checking performance of EJAYA in solving complex optimization problems. The used two test suites can be found from https://github.com/P-N-Suganthan.

4.2. Experiment setup

4.2.1. The compared algorithms

EJAYA has also been compared with eight powerful metaheuristic algorithms, which include particle swarm optimization (PSO) [47], sine cosine algorithm (SCA) [4], grey wolf optimizer (GWO) [48], whale optimization algorithm (WOA) [49], Jaya algorithm (JAYA) [29], improved Jaya algorithm (IJAYA) [32], performance-guided JAYA algorithm (PGJAYA) [50], and modified Jaya algorithm (MJAYA) [51]. To make a fair comparison, population size and the maximum number of function evaluations for all applied algorithms were set to 50 and 300,000, respectively. The other parameters of the compared algorithms were from the corresponding references. All the applied algorithms have been coded and implemented in MATLAB programming software. In addition, each algorithm for every test function was executed in 30 independent runs and then the results were recorded.

4.2.2. Evaluation metrics

To better show the performance differences between EJAYA and the compared algorithms, three evaluation metrics are considered, which can be described as follows:

- Mean value and standard deviance

  Mean value (MEAN) and standard deviance (STD) are often used to measure the solution quality [21,52–54]. Take the minimum problem as an example, the smaller the obtained MEAN, the closer to the global optimal solution the obtained solution; the smaller the obtained STD, the higher the stable of the algorithm. The obtained MEAN and STD by the applied algorithms are presented in Table 3. The best results were in bold type in Table 3.

- Friedman Mean Rank Test

  As a common statistical tool in the field of optimization, Friedman Mean Rank Test has been widely used for comparing the performance differences among optimization algorithms [55–58]. Here, the optimal solutions obtained by EJAYA from 30 independent runs and the compared algorithms are tested by Friedman Mean Rank Test. The test results can be found in Table 4 and Fig. 4.

- Wilcoxon sign-rank test

  Wilcoxon sign-rank test is used widely to compare the performance among different optimization algorithms [59–62]. Tables 5 and 6 give the statistical results produced by Wilcoxon sign-rank test with a significant level $\alpha = 0.05$. In Tables 5 and 6, R^+ means the sum of ranks for the problems in which EJAYA outperformed the compared algorithm and R^- indicates the sum of ranks for the opposite. As can be seen from Tables 5 and 6, there are the following three cases:

  (1) Case 1: If R^+ is more than R^- and the obtained p-value is less than the set significant level α, the symbol is marked with '+' and EJAYA outperforms the compared algorithm on the considered case.
  (2) Case 2: If R^+ is less than R^- and the obtained p-value is less than the set significant level α, the symbol is marked with '-' and the compared algorithm beats EJAYA on the considered case.
  (3) Case 3: If R^+, R^- and p-value do not meet Case 1 and Case 2, the symbol is marked with '=' and there is no significant difference between EJAYA and the compared algorithm on the considered case.

4.3. Experimental results and discussion

This section is to compare the performance between EJAYA and the other eight metaheuristic algorithms on CEC 2014 and CEC 2015 test suites from the following four aspects, i.e. solution accuracy, algorithm rank, statistical test, and convergence performance.

4.3.1. Comparison on solution accuracy

Table 3 shows the experimental results of EJAYA and the other eight compared algorithms on CEC 2014 and CEC 2015 test suites. From Table 3, all algorithms can get the same MEAN on F12, F16, and F43. In terms of MEAN, PGJAYA and WOA are superior to the other seven algorithms on F5. For F7, F15 and F45, PGJAYA and

Table 3
The experimental results obtained by EJAYA and the compared algorithms on 45 test functions.

No.	Metric	MJAYA	IJAYA	PGJAYA	PSO	GWO	WOA	SCA	JAYA	EJAYA
F1	MEAN	1.61E+09	2.63E+07	2.58E+05	3.72E+07	6.08E+07	3.05E+07	2.24E+08	7.23E+07	**1.69E+05**
	STD	4.79E+08	1.09E+07	**1.26E+05**	9.49E+06	3.82E+07	1.20E+07	6.11E+07	2.75E+07	3.19E+05
F2	MEAN	1.10E+11	2.93E+06	4.12E+03	1.88E+09	1.29E+09	5.18E+06	1.65E+10	5.93E+09	**2.00E+02**
	STD	2.05E+10	9.43E+05	5.45E+03	2.10E+08	1.57E+09	6.92E+06	3.36E+09	1.12E+09	**4.26E−08**
F3	MEAN	3.17E+05	4.28E+04	3.62E+02	7.22E+03	3.06E+04	4.31E+04	3.70E+04	5.16E+04	**3.00E+02**
	STD	5.50E+04	6.77E+03	9.91E+01	1.26E+03	8.70E+03	2.68E+04	5.19E+03	1.20E+04	**2.59E−02**
F4	MEAN	2.38E+04	5.70E+02	**4.05E+02**	6.33E+02	6.11E+02	6.00E+02	1.40E+03	7.86E+02	4.06E+02
	STD	5.36E+03	3.78E+01	1.60E+01	1.98E+01	6.54E+01	5.77E+01	2.29E+02	1.12E+02	**1.43E+01**
F5	MEAN	5.21E+02	5.21E+02	**5.20E+02**	5.21E+02	5.21E+02	**5.20E+02**	5.21E+02	5.21E+02	5.21E+02
	STD	5.15E−02	1.45E−01	6.05E−02	4.88E−02	**4.56E−02**	1.91E−01	4.65E−02	6.39E−02	1.14E−01
F6	MEAN	6.42E+02	6.30E+02	6.23E+02	6.21E+02	**6.12E+02**	6.36E+02	6.34E+02	6.34E+02	6.18E+02
	STD	1.70E+00	2.08E+00	3.10E+00	2.07E+00	2.23E+00	3.28E+00	2.77E+00	**1.47E+00**	4.54E+00
F7	MEAN	1.76E+03	7.01E+02	**7.00E+02**	7.17E+02	7.12E+02	7.01E+02	8.43E+02	7.15E+02	**7.00E+02**
	STD	2.01E+02	7.22E−02	**1.00E−02**	1.38E+00	1.08E+01	5.85E−02	2.41E+01	3.50E+00	3.00E−02
F8	MEAN	1.27E+03	9.22E+02	9.22E+02	9.88E+02	8.73E+02	9.98E+02	1.04E+03	1.01E+03	**8.70E+02**
	STD	3.94E+01	2.80E+01	3.95E+01	2.08E+01	1.71E+01	4.11E+01	1.79E+01	**1.40E+01**	2.02E+01
F9	MEAN	1.46E+03	1.06E+03	1.05E+03	1.09E+03	9.94E+02	1.13E+03	1.17E+03	1.15E+03	**9.84E+02**
	STD	5.36E+01	2.86E+01	3.21E+01	1.47E+01	1.83E+01	4.60E+01	2.18E+01	**1.43E+01**	1.94E+01
F10	MEAN	8.57E+03	5.34E+03	4.37E+03	6.74E+03	**3.16E+03**	4.97E+03	6.87E+03	6.47E+03	4.22E+03
	STD	**2.71E+02**	6.16E+02	8.67E+02	4.04E+02	5.04E+02	8.13E+02	3.93E+02	5.32E+02	7.00E+02
F11	MEAN	8.62E+03	5.83E+03	5.46E+03	7.46E+03	**4.15E+03**	5.82E+03	8.10E+03	8.09E+03	4.89E+03
	STD	3.12E+02	8.15E+02	6.49E+02	3.99E+02	7.00E+02	6.55E+02	2.96E+02	**2.72E+02**	7.35E+02
F12	MEAN	**1.20E+03**	**1.20E+03**	**1.20E+03**	**1.20E+03**	**1.20E+03**	**1.20E+03**	**1.20E+03**	**1.20E+03**	**1.20E+03**
	STD	2.75E−01	4.09E−01	**9.65E−02**	2.81E−01	1.04E+00	4.84E−01	2.43E−01	2.53E−01	3.66E−01
F13	MEAN	1.31E+03	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**
	STD	1.01E+00	9.09E−02	9.98E−02	**5.35E−02**	8.01E−02	1.20E−01	3.71E−01	3.21E−01	9.38E−02
F14	MEAN	1.77E+03	**1.40E+03**	**1.40E+03**	**1.40E+03**	**1.40E+03**	**1.40E+03**	1.44E+03	1.41E+03	**1.40E+03**
	STD	5.74E+01	6.00E−02	2.66E−01	3.62E−01	3.01E+00	**4.36E−02**	6.99E+00	3.52E+00	4.39E−02
F15	MEAN	1.91E+07	1.54E+03	**1.51E+03**	1.54E+03	1.54E+03	1.58E+03	4.24E+03	1.54E+03	**1.51E+03**
	STD	1.12E+07	1.22E+01	**2.49E+00**	3.41E+00	4.95E+01	2.14E+01	2.47E+03	1.34E+01	2.50E+00
F16	MEAN	**1.61E+03**	**1.61E+03**	**1.61E+03**	**1.61E+03**	**1.61E+03**	**1.61E+03**	**1.61E+03**	**1.61E+03**	**1.61E+03**
	STD	1.98E−01	4.00E−01	4.78E−01	3.79E−01	6.12E−01	4.60E−01	2.77E−01	**2.25E−01**	4.99E−01
F17	MEAN	6.83E+07	7.77E+05	2.02E+04	1.06E+06	1.22E+06	3.90E+06	6.11E+06	3.82E+06	**1.14E+04**
	STD	8.04E+07	4.19E+05	**9.41E+03**	4.10E+05	1.68E+06	2.15E+06	2.28E+06	1.64E+06	1.49E+04
F18	MEAN	3.31E+09	9.64E+03	1.30E+04	2.88E+07	6.89E+06	1.17E+04	1.67E+08	2.20E+07	**6.58E+03**
	STD	2.43E+09	**5.80E+03**	8.40E+03	9.78E+06	1.85E+07	1.97E+04	8.10E+07	3.13E+07	6.50E+03
F19	MEAN	2.65E+03	**1.91E+03**	**1.91E+03**	1.92E+03	1.94E+03	1.95E+03	1.99E+03	1.93E+03	**1.91E+03**
	STD	2.79E+02	**1.08E+00**	2.26E+00	2.38E+00	2.62E+01	3.83E+01	2.20E+01	1.95E+01	1.85E+01
F20	MEAN	1.28E+06	9.84E+03	2.45E+03	2.78E+03	1.63E+04	2.92E+04	1.68E+04	8.68E+03	**2.27E+03**
	STD	1.22E+06	5.30E+03	1.82E+02	2.08E+02	6.79E+03	1.65E+04	4.75E+03	2.33E+03	**1.08E+02**
F21	MEAN	2.11E+07	1.84E+05	1.70E+04	3.11E+05	1.09E+06	8.22E+05	1.39E+06	8.21E+05	**4.61E+03**
	STD	1.33E+07	9.07E+04	1.13E+04	1.21E+05	1.69E+06	5.31E+05	6.80E+05	4.18E+05	**1.43E+03**
F22	MEAN	4.68E+03	2.54E+03	2.80E+03	2.73E+03	2.56E+03	2.96E+03	2.94E+03	2.84E+03	**2.51E+03**
	STD	4.12E+03	**9.55E+01**	2.02E+02	1.25E+02	1.34E+02	2.93E+02	1.65E+02	1.46E+02	1.67E+02
F23	MEAN	3.82E+03	**2.62E+03**	**2.62E+03**	2.63E+03	2.63E+03	2.64E+03	2.67E+03	2.64E+03	**2.62E+03**
	STD	3.53E+02	2.65E−01	9.75E−09	5.09E+00	1.11E+01	1.15E+01	1.48E+01	5.26E+00	**1.75E−12**
F24	MEAN	2.94E+03	2.64E+03	2.64E+03	2.65E+03	**2.60E+03**	2.61E+03	**2.60E+03**	2.63E+03	2.63E+03
	STD	3.25E+01	4.70E+00	7.78E+00	2.19E+00	9.71E−04	5.74E+00	**4.10E−02**	2.43E+01	7.44E+00
F25	MEAN	2.78E+03	**2.71E+03**	**2.71E+03**	**2.71E+03**	**2.71E+03**	2.72E+03	2.73E+03	2.72E+03	**2.71E+03**
	STD	1.64E+01	**2.22E+00**	3.24E+00	2.38E+00	4.18E+00	1.62E+01	5.89E+00	4.94E+00	2.76E+00
F26	MEAN	2.71E+03	**2.70E+03**	**2.70E+03**	2.78E+03	2.74E+03	**2.70E+03**	**2.70E+03**	**2.70E+03**	**2.70E+03**
	STD	1.08E+00	**7.00E−02**	1.82E+01	4.41E+01	4.96E+01	1.40E−01	6.13E−01	1.89E+01	7.30E−02
F27	MEAN	4.13E+03	3.45E+03	3.60E+03	3.39E+03	**3.34E+03**	3.81E+03	3.51E+03	3.75E+03	3.42E+03
	STD	**4.29E+01**	2.91E+02	1.92E+02	2.32E+02	1.29E+02	3.07E+02	3.42E+02	2.09E+02	1.95E+02
F28	MEAN	5.32E+03	**3.81E+03**	4.10E+03	4.45E+03	3.92E+03	4.95E+03	4.89E+03	4.02E+03	3.94E+03
	STD	3.22E+02	**1.44E+02**	3.33E+02	7.38E+02	3.20E+02	7.02E+02	3.83E+02	1.51E+02	2.13E+02
F29	MEAN	3.83E+07	1.41E+06	1.91E+06	1.72E+06	**6.00E+05**	4.95E+06	1.29E+07	2.80E+06	3.20E+06
	STD	2.73E+07	3.79E+06	3.93E+06	2.78E+06	**2.22E+06**	5.05E+06	7.22E+06	4.10E+06	4.65E+06
F30	MEAN	7.94E+05	**6.24E+03**	6.60E+03	3.43E+04	5.23E+04	8.20E+04	2.19E+05	1.63E+04	**6.24E+03**
	STD	6.37E+05	**8.88E+02**	1.58E+03	9.67E+03	3.02E+04	4.85E+04	9.10E+04	1.45E+04	1.06E+03
F31	MEAN	1.38E+09	2.12E+07	9.51E+04	2.90E+07	2.16E+07	4.03E+07	1.47E+08	1.06E+08	**6.77E+04**
	STD	3.84E+08	8.82E+06	**4.96E+04**	6.44E+06	1.78E+07	2.09E+07	5.62E+07	4.42E+07	1.01E+05
F32	MEAN	1.17E+11	8.60E+06	7.32E+03	1.85E+09	2.00E+09	4.09E+06	1.68E+10	9.56E+09	**5.02E+03**
	STD	1.94E+10	1.28E+07	6.50E+03	2.71E+08	1.22E+09	2.40E+06	3.42E+09	1.34E+09	**3.82E+03**
F33	MEAN	3.21E+02	3.21E+02	**3.20E+02**	3.21E+02	3.21E+02	**3.20E+02**	3.21E+02	3.21E+02	3.21E+02
	STD	5.36E−02	1.38E−01	7.41E−02	**4.33E−02**	4.93E−02	1.45E−01	5.07E−02	4.63E−02	1.55E−01
F34	MEAN	1.01E+03	5.48E+02	5.53E+02	5.87E+02	4.91E+02	6.79E+02	6.73E+02	6.41E+02	**4.90E+02**
	STD	5.85E+01	3.78E+01	3.15E+01	1.40E+01	4.13E+01	5.13E+01	1.51E+01	**1.28E+01**	2.09E+01
F35	MEAN	7.91E+03	5.48E+03	4.62E+03	6.97E+03	**3.42E+03**	5.27E+03	7.51E+03	7.44E+03	4.39E+03
	STD	3.30E+02	6.96E+02	6.75E+02	2.86E+02	8.02E+02	7.81E+02	3.79E+02	**2.68E+02**	9.28E+02
F36	MEAN	5.19E+07	3.99E+05	2.04E+04	7.85E+05	1.46E+06	1.81E+06	4.66E+06	4.28E+06	**1.24E+04**
	STD	3.49E+07	2.67E+05	**1.29E+04**	2.53E+05	1.00E+06	1.12E+06	1.95E+06	2.04E+06	1.97E+04

(*continued on next page*)

Table 3 (*continued*).

No.	Metric	MJAYA	IJAYA	PGJAYA	PSO	GWO	WOA	SCA	JAYA	EJAYA
F37	MEAN	1.47E+03	7.15E+02	7.09E+02	7.26E+02	7.19E+02	7.37E+02	7.41E+02	7.25E+02	**7.07E+02**
	STD	2.63E+02	**1.16E+00**	4.00E+00	2.30E+00	2.30E+00	2.65E+01	4.44E+00	2.68E+00	2.57E+00
F38	MEAN	2.45E+07	1.52E+05	1.53E+04	2.28E+05	2.99E+05	2.81E+05	1.21E+06	7.62E+05	**7.01E+03**
	STD	1.41E+07	6.39E+04	1.02E+04	6.49E+04	2.93E+05	1.84E+05	5.92E+05	3.52E+05	**4.20E+03**
F39	MEAN	1.39E+03	1.00E+03	1.01E+03	1.06E+03	1.02E+03	1.04E+03	1.07E+03	1.04E+03	**1.00E+03**
	STD	5.67E+01	3.87E−01	3.86E+01	1.05E+02	4.37E+01	1.06E+02	1.08E+01	7.01E+00	**3.71E−01**
F40	MEAN	7.92E+07	3.17E+05	1.58E+04	5.89E+05	1.31E+06	2.56E+06	4.33E+06	2.75E+06	**7.06E+03**
	STD	3.81E+07	1.68E+05	8.54E+03	2.71E+05	1.09E+06	1.96E+06	1.39E+06	1.18E+06	**7.47E+03**
F41	MEAN	2.64E+03	2.16E+03	2.12E+03	**1.73E+03**	1.85E+03	2.29E+03	1.99E+03	2.30E+03	1.94E+03
	STD	**5.80E+01**	2.38E+02	2.24E+02	3.26E+02	1.33E+02	4.09E+02	4.24E+02	7.22E+01	2.41E+02
F42	MEAN	1.40E+03	1.34E+03	1.34E+03	**1.32E+03**	**1.32E+03**	1.34E+03	1.37E+03	1.33E+03	1.34E+03
	STD	4.23E+00	4.25E+01	4.32E+01	3.47E+01	**2.89E+01**	4.19E+01	3.66E+01	3.52E+01	4.60E+01
F43	MEAN	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**	**1.30E+03**
	STD	2.54E+00	**6.34E−04**	4.47E−03	6.29E−03	2.67E−02	1.03E−02	1.19E−01	1.93E−02	3.53E−03
F44	MEAN	6.28E+04	**3.47E+04**	3.61E+04	3.54E+04	3.63E+04	3.78E+04	4.73E+04	3.89E+04	3.56E+04
	STD	1.07E+04	**5.59E+02**	1.92E+03	1.16E+03	1.39E+03	2.66E+03	1.67E+03	1.66E+03	1.76E+03
F45	MEAN	3.46E+05	1.61E+03	**1.60E+03**	1.62E+03	1.64E+03	1.61E+03	2.07E+03	1.63E+03	**1.60E+03**
	STD	1.84E+05	1.38E+00	3.57E−05	6.25E−01	4.28E+01	4.00E+00	3.59E+02	3.55E+00	**8.65E−13**

Table 4
The ranking results produced by Friedman Mean Rank Test on 45 test functions.

No.	MJAYA	IJAYA	PGJAYA	PSO	GWO	WOA	SCA	JAYA	EJAYA
F1	9.00	3.83	1.83	4.90	5.67	4.20	8.00	6.40	1.17
F2	9.00	3.53	2.00	5.73	5.30	3.47	8.00	6.97	1.00
F3	9.00	6.57	2.00	3.00	4.77	5.97	5.57	7.13	1.00
F4	9.00	3.63	1.30	5.47	4.60	4.37	7.97	6.97	1.70
F5	8.27	3.27	1.07	6.93	7.07	2.17	6.83	5.87	3.53
F6	9.00	5.20	3.60	3.07	1.17	7.33	6.57	6.87	2.20
F7	9.00	3.17	1.40	6.60	5.33	3.83	8.00	6.07	1.60
F8	9.00	3.50	3.33	5.63	1.70	5.93	7.67	6.57	1.67
F9	9.00	3.93	3.43	4.93	1.67	6.07	7.30	7.13	1.53
F10	9.00	4.60	2.80	6.97	1.27	3.97	7.37	6.40	2.63
F11	8.80	3.83	3.37	6.13	1.33	3.97	7.40	7.63	2.53
F12	8.70	3.77	1.23	6.63	4.57	4.27	6.40	6.97	2.47
F13	9.00	4.63	4.63	4.20	2.03	3.87	8.00	7.00	1.63
F14	9.00	2.83	3.90	5.40	3.93	2.70	8.00	6.90	2.33
F15	9.00	4.80	2.00	4.90	3.73	6.63	7.97	4.43	1.53
F16	8.90	4.73	4.97	3.30	1.33	5.60	6.57	7.23	2.37
F17	9.00	3.70	1.87	4.50	4.23	6.60	7.33	6.63	1.13
F18	9.00	3.27	3.40	6.67	4.00	2.77	7.97	6.10	1.83
F19	9.00	2.70	2.00	5.20	5.20	5.97	7.73	5.53	1.67
F20	8.97	4.87	1.90	2.90	6.47	7.43	6.57	4.70	1.20
F21	9.00	3.60	1.97	4.53	5.10	5.97	7.53	6.27	1.03
F22	9.00	2.33	5.20	4.77	2.57	6.27	6.67	5.63	2.57
F23	9.00	3.00	2.00	5.40	4.80	5.40	7.93	6.47	1.00
F24	9.00	5.63	6.40	7.73	1.00	3.37	2.00	5.03	4.83
F25	9.00	3.20	1.90	3.60	4.13	4.90	7.47	6.53	4.27
F26	7.77	3.50	3.57	7.67	4.87	3.33	6.77	5.93	1.60
F27	9.00	4.10	4.83	3.17	2.87	6.97	4.40	6.50	3.17
F28	8.33	2.00	4.23	5.77	2.60	7.43	7.27	4.10	3.27
F29	8.87	2.87	2.73	5.43	3.93	5.47	7.60	5.23	2.87
F30	8.93	1.80	2.27	5.50	5.90	6.47	8.03	4.07	2.03
F31	9.00	3.97	1.73	4.93	3.73	5.57	7.67	7.13	1.27
F32	9.00	3.53	1.57	5.60	5.40	3.47	8.00	7.00	1.43
F33	7.10	3.10	1.07	7.37	6.97	2.13	6.87	6.67	3.73
F34	9.00	3.43	3.73	4.60	1.60	7.17	7.50	6.20	1.77
F35	8.57	4.23	3.00	6.13	1.43	3.90	7.70	7.53	2.50
F36	9.00	3.40	1.83	4.43	4.90	5.83	7.33	7.10	1.17
F37	9.00	3.03	1.90	6.10	4.07	6.03	7.80	5.80	1.27
F38	9.00	3.93	1.70	4.70	4.60	5.17	7.57	7.03	1.30
F39	9.00	3.37	1.87	5.87	4.27	4.90	7.60	6.63	1.50
F40	9.00	3.40	1.87	4.23	4.90	6.37	7.53	6.57	1.13
F41	8.97	5.40	4.73	2.40	2.60	6.80	4.40	6.37	3.33
F42	8.67	4.37	3.17	4.47	1.97	5.02	8.13	6.10	3.12
F43	8.40	1.37	3.33	6.83	5.53	3.63	8.47	5.10	2.33
F44	8.97	2.30	3.67	3.13	4.13	5.00	8.03	6.30	3.47
F45	9.00	3.77	2.00	4.97	6.27	3.43	8.00	6.57	1.00

EJAYA can achieve better MEAN than the other seven algorithms. EJAYA, IJAYA, JAYA, PGJAYA, PSO, GWO, WOA, and SCA have the sane MEAN on F13. IJAYA, PGJAYA, PSO, GWO, WOA, and EJAYA can obtain the best MEAN on F14. IJAYA, PGJAYA and EJAYA can offer better MEAN on F19. IJAYA, EJAYA, and PGJAYA can beat the other algorithms on F23 in terms of MEAN. GWO and SCA outperform the other seven algorithms on F24 in terms of MEAN. For F25, IJAYA, PGJAYA, PSO, GWO, and EJAYA can win MJAYA, SCA, JAYA, and WOA on MEAN. IJAYA, PGJAYA, EJAYA, SCA, JAYA and WOA share the best MEAN on F26. For F30, IJAYA and EJAYA beats the other seven algorithms on MEAN. PGJAYA and WOA show better performance than the other seven algorithms on F33

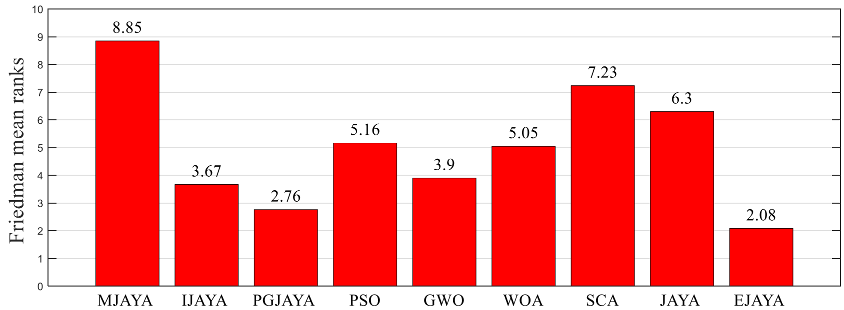

Fig. 4. Friedman mean ranks obtained by the applied algorithms on 45 test functions.

Table 5
The results between EJAYA and the compared algorithms (MJAYA, IJAYA, PGJAYA and PSO) produced by Wilcoxon sign-rank test with a significant level $\alpha = 0.05$ on 45 test functions.

No.	EJAYA vs. MJAYA				EJAYA vs. IJAYA				EJAYA vs. PGJAYA				EJAYA vs. PSO			
	R^+	R^-	p-value	S	R^+	R^-	p-value	S	R^+	R^-	p-value	S	R^+	R^-	p-value	S
F1	465	0	1.73E−06	+	465	0	1.73E−06	+	370	95	4.68E−03	+	465	0	1.73E−06	+
F2	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F3	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F4	465	0	1.73E−06	+	465	0	1.73E−06	+	122	343	2.30E−02	–	465	0	1.73E−06	+
F5	465	0	1.73E−06	+	125	340	2.70E−02	–	0	465	1.73E−06	–	465	0	1.73E−06	+
F6	465	0	1.73E−06	+	465	0	1.73E−06	+	430	35	4.86E−05	+	389	76	1.29E−03	+
F7	465	0	1.73E−06	+	465	0	1.73E−06	+	129	336	3.33E−02	–	465	0	1.73E−06	+
F8	465	0	1.73E−06	+	460	5	2.88E−06	+	432	33	4.07E−05	+	465	0	1.73E−06	+
F9	465	0	1.73E−06	+	465	0	1.73E−06	+	457	8	3.88E−06	+	465	0	1.73E−06	+
F10	465	0	1.73E−06	+	455	10	4.73E−06	+	263	202	5.30E−01	=	465	0	1.73E−06	+
F11	465	0	1.73E−06	+	402	63	4.90E−04	+	364	101	6.84E−03	+	465	0	1.73E−06	+
F12	465	0	1.73E−06	+	442	23	1.64E−05	+	0	465	1.73E−06	–	465	0	1.73E−06	+
F13	465	0	1.73E−06	+	461	4	2.60E−06	+	456	9	4.29E−06	+	461	4	2.60E−06	+
F14	465	0	1.73E−06	+	288	177	2.54E−01	=	413	52	2.05E−04	+	465	0	1.73E−06	+
F15	465	0	1.73E−06	+	465	0	1.73E−06	+	351	114	1.48E−02	+	465	0	1.73E−06	+
F16	465	0	1.73E−06	+	436	29	2.84E−05	+	446	19	1.13E−05	+	368	97	5.32E−03	+
F17	465	0	1.73E−06	+	465	0	1.73E−06	+	398	67	6.64E−04	+	465	0	1.73E−06	+
F18	465	0	1.73E−06	+	326	139	5.45E−02	=	386	79	1.59E−03	+	465	0	1.73E−06	+
F19	465	0	1.73E−06	+	378	87	2.77E−03	+	366	99	6.04E−03	+	378	87	2.77E−03	+
F20	465	0	1.73E−06	+	465	0	1.73E−06	+	419	46	1.25E−04	+	465	0	1.73E−06	+
F21	465	0	1.73E−06	+	465	0	1.73E−06	+	464	1	1.92E−06	+	465	0	1.73E−06	+
F22	465	0	1.73E−06	+	281	184	3.18E−01	=	433	32	3.72E−05	+	440	25	1.97E−05	+
F23	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F24	465	0	1.73E−06	+	439	26	2.16E−05	+	427	38	6.32E−05	+	465	0	1.73E−06	+
F25	465	0	1.73E−06	+	97	368	5.32E−03	–	21	444	1.36E−05	–	157	308	1.20E−01	=
F26	465	0	1.73E−06	+	456	9	4.29E−06	+	421	44	1.06E−04	+	465	0	1.73E−06	+
F27	465	0	1.73E−06	+	259	206	5.86E−01	=	421	44	1.06E−04	+	238	227	9.10E−01	=
F28	465	0	1.73E−06	+	113	352	1.40E−02	+	336	129	3.33E−02	+	396	69	7.71E−04	+
F29	465	0	1.73E−06	+	226	239	8.94E−01	=	250	215	7.19E−01	=	220	245	7.97E−01	=
F30	465	0	1.73E−06	+	211	254	6.58E−01	=	275	190	3.82E−01	=	465	0	1.73E−06	+
F31	465	0	1.73E−06	+	465	0	1.73E−06	+	360	105	8.73E−03	+	465	0	1.73E−06	+
F32	465	0	1.73E−06	+	465	0	1.73E−06	+	304	161	1.41E−01	=	465	0	1.73E−06	+
F33	465	0	1.73E−06	+	100	365	6.42E−03	–	0	465	1.73E−06	–	465	0	1.73E−06	+
F34	465	0	1.73E−06	+	445	20	1.24E−05	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F35	465	0	1.73E−06	+	430	35	4.86E−05	+	289	176	2.45E−01	=	464	1	1.92E−06	+
F36	465	0	1.73E−06	+	465	0	1.73E−06	+	398	67	6.64E−04	+	465	0	1.73E−06	+
F37	465	0	1.73E−06	+	465	0	1.73E−06	+	375	90	3.38E−03	+	465	0	1.73E−06	+
F38	465	0	1.73E−06	+	465	0	1.73E−06	+	395	70	8.31E−04	+	465	0	1.73E−06	+
F39	465	0	1.73E−06	+	465	0	1.73E−06	+	302	163	1.53E−01	=	465	0	1.73E−06	+
F40	465	0	1.73E−06	+	465	0	1.73E−06	+	415	50	1.74E−04	+	465	0	1.73E−06	+
F41	465	0	1.73E−06	+	391	74	1.11E−03	+	379	86	2.58E−03	+	123	342	2.43E−02	–
F42	465	0	1.73E−06	+	334	131	3.68E−02	+	272.5	192.5	3.37E−01	=	300	165	1.65E−01	=
F43	465	0	1.73E−06	+	95	370	4.68E−03	+	358	107	9.84E−03	+	465	0	1.73E−06	+
F44	465	0	1.73E−06	+	111	354	1.25E−02	–	270	195	4.41E−01	=	207	258	6.00E−01	=
F45	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
+/=/-	45/0/0				35/6/4				31/8/6				39/5/1			

in terms of MEAN. MEAN of PSO and GWO excels that of the other seven algorithms on F42. In addition, IJAYA, PGJAYA, GWO, PSO, and EJAYA can get the optimal MEAN on two (i.e. F28 and F44), one (i.e. F4), six (i.e. F6, F10, F11, F27, F29, and F35), one (i.e. F41) and 19 (i.e. F1, F2, F3, F8, F9, F15, F17, F18, F20, F21, F22, F31, F32, F34, F36, F37, F38, F39, and F40) functions, respectively. In general, EJAYA can get or share the best MEAN on 32 functions. Clearly, EJAYA shows stronger global search ability than JAYA, IJAYA, MJAYA, PGJAYA, GWO, PSO, SCA, and WOA.

4.3.2. Comparison on algorithm rank

Table 4 presents the ranking results obtained by EJAYA and the compared algorithms for 45 test functions. According to the results of Friedman Mean Rank Test shown in Table 4, EJAYA

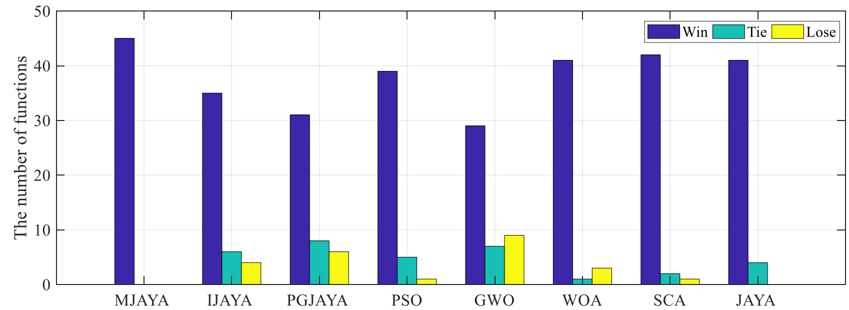

Fig. 5. The statistical results produced by Wilcoxon sign-rank test on 45 test functions. "Win" denotes EJAYA beat the compared algorithm; "Lose" indicates EJAYA is inferior to the compared algorithms; "Tie" denotes EJAYA has the same performance with the compared algorithms.

Table 6
The results between EJAYA and the compared algorithms (GWO, WOA, SCA and JAYA) produced by Wilcoxon sign-rank test with a significant level $\alpha = 0.05$ on 45 test functions.

No.	EJAYA vs. GWO				EJAYA vs. WOA				EJAYA vs. SCA				EJAYA vs. JAYA			
	R^+	R^-	*p*-value	S	R^+	R^-	*p*-value	S	R^+	R^-	*p*-value	S	R^+	R^-	*p*-value	S
F1	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F2	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F3	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F4	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F5	465	0	1.73E−06	+	12	453	5.75E−06	−	465	0	1.73E−06	+	464	1	1.92E−06	+
F6	19	446	1.13E−05	−	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F7	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F8	282	183	3.09E−01	=	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F9	319	146	7.52E−02	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F10	15	450	7.69E−06	−	388	77	1.38E−03	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F11	57	408	3.06E−04	−	437	28	2.60E−05	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F12	365	100	6.42E−03	+	440	25	1.97E−05	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F13	335	130	3.50E−02	+	443	22	1.49E−05	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F14	380	85	2.41E−03	+	288	177	2.54E−01	=	465	0	1.73E−06	+	465	0	1.73E−06	+
F15	424	41	8.19E−05	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F16	21	444	1.36E−05	−	457	8	3.88E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F17	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F18	395	70	8.31E−04	+	333	132	3.87E−02	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F19	423	42	8.92E−05	+	424	41	8.19E−05	+	465	0	1.73E−06	+	384	81	1.83E−03	+
F20	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F21	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F22	270	195	4.41E−01	=	448	17	9.32E−06	+	462	3	2.35E−06	+	448	17	9.32E−06	+
F23	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F24	0	465	1.73E−06	−	1	464	1.92E−06	−	0	465	1.73E−06	−	193	272	4.17E−01	=
F25	237	228	9.26E−01	=	340	125	2.70E−02	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F26	443	22	1.49E−05	+	419	46	1.25E−04	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F27	128	337	3.16E−02	−	440	25	1.97E−05	+	274	191	3.93E−01	=	445	20	1.24E−05	+
F28	165	300	1.65E−01	=	465	0	1.73E−06	+	465	0	1.73E−06	+	323	142	6.27E−02	=
F29	231	234	9.75E−01	=	343	122	2.30E−02	+	445	20	1.24E−05	+	279	186	3.39E−01	=
F30	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	464	1	1.92E−06	+
F31	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F32	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F33	465	0	1.73E−06	+	9	456	4.29E−06	−	464	1	1.92E−06	+	465	0	1.73E−06	+
F34	203	262	5.44E−01	=	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F35	71	394	8.94E−04	−	398	67	6.64E−04	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F36	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F37	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F38	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F39	460	5	2.88E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F40	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F41	101	364	6.84E−03	−	406	59	3.59E−04	+	270	195	4.41E−01	=	462	3	2.35E−06	+
F42	113	352	1.40E−02	−	345.5	119.5	1.01E−02	+	444	21	1.36E−05	+	306	159	1.31E−01	=
F43	460	5	2.88E−06	+	341	124	2.56E−02	+	465	0	1.73E−06	+	465	0	1.73E−06	+
F44	300	165	1.65E−01	=	375	90	3.38E−03	+	465	0	1.73E−06	+	450	15	7.69E−06	+
F45	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+	465	0	1.73E−06	+
+/=/-	29/7/9				41/1/3				42/2/1				41/4/0			

can obtain the best solutions on more than half of test functions, i.e. F1, F2, F3, F8, F9, F13, F14, F15, F17, F18, F19, F20, F21, F22, F23, F26, F31, F32, F36, F37, F38, F39, F40, and F45. PGJAYA, GWO and IAJYA also show strong global search ability, which can get the optimal solutions on seven (i.e. F4, F5, F7, F12, F25, F29, and F33), nine (i.e. F6, F10, F11, F16, F24, F27, F34, F35 and F42), and five (i.e. F22, F28, F30, F43, and F44) test functions, respectively. In addition, PSO is the best of all algorithms on F41.

Table 7
Statistical results obtained by EJAYA on welded beam design problem (NFEs=24,000).

WORST	MEAN	MEDIAN	BEST	STD
1.7248523105	1.7248523093	1.7248523091	1.7248523086	5.5631E−10

Table 9
Statistical results obtained by EJAYA on tension/compression spring design problem (NFEs=15,000).

WORST	MEAN	MEDIAN	BEST	STD
0.012687	0.012668	0.012666	0.012665	4.6331E−6

The rest algorithms including MJAYA, WOA, SCA, and JAYA cannot obtain the optimal solutions on any test functions. Moreover, Friedman mean ranks obtained by the applied algorithms on 45 test functions have been shown in Fig. 4. From Fig. 4, the applied algorithms can be sorted in the flowing order: EJAYA, PGJAYA, IJAYA, GWO, WOA, JAYA, SCA and MJAYA. That is, EJAYA is the best of all applied algorithms subject to the obtained Friedman mean ranks.

4.3.3. Comparison on statistical test

Table 5, Table 6 and Fig. 5 show the experimental results of Wilcoxon sign-rank test obtained by EJAYA and the compared algorithms. From Table 5, Table 6 and Fig. 5, EJAYA outperforms MJAYA on all test functions. In addition, EJAYA shows significant advantages over PSO, WOA, SCA and JAYA. EJAYA can obtain better performance than IJAYA, PSO, WOA, SCA and JAYA on 35, 39, 41, 42 and 41 test functions, respectively. But, JAYA cannot beat EJAYA on any test functions; PSO, IJAYA, WOA and SCA only achieve better solutions than EJAYA on one (i.e. F41) four (i.e. F5, F25, F33 and F44), three (i.e. F5, F24 and F33) and one (i.e. F24) test functions, respectively. PGJAYA and GWO show stronger competitiveness compared with IJAYA, MJAYA, PSO, SCA, WOA, and JAYA. PGJAYA and GWO are superior to EJAYA on six (i.e. F4, F5, F7, F12, F25 and F33) and nine (i.e. F6, F10, F11, F16, F24, F27, F35, F41 and F42) test functions, respectively. However, it should be noted that PGJAYA and GWO cannot compete with EJAYA on 31 and 29 functions, respectively.

Thus, according to the discussion for the results produced by Wilcoxon sign-rank test on CEC 2014 and CEC 2015 test suites, the proposed EJAYA shows better performance than the compared eight algorithms.

4.3.4. Comparison on convergence performance

EJAYA is a variant of JAYA. This section is to discuss the convergence performance of EJAYA by comparing with convergence curves obtained by JAYA, EJAYA, IJAYA, MJAYA and PGJAYA on the considered 45 test functions as shown in Fig. 6. From Fig. 6, EJAYA can find better solutions with faster speed than IJAYA, MJAYA and PGJAYA on more than half of test functions including F1, F2, F3, F6, F8, F9, F10, F11, F16, F17, F18, F20, F21, F24, F27, F31, F32, F34, F35, F36, F38, F40 and F41. MJAYA is the worst, which cannot compete with the other four algorithms on nearly all test functions. PGJAYA is superior to EJAYA on F5, F12, and F33. JAYA only shows better performance than EJAYA on F29 and F42. In addition, JAYA and EJAYA have the similar convergence performance on F15, F26, F37, F43 and F45. Note that, JAYA cannot compete with EJAYA on the other 38 test functions, which demonstrates the designed learning strategies in EJAYA are very helpful for improving the convergence performance of JAYA on complex problems.

5. EJAYA for real-world engineering design optimization problems

The performance of EJAYA is evaluated by solving seven challenging real-world engineering design optimization problems in this section. In order to show the competitiveness of EJAYA for these engineering optimization problems, the results obtained by EJAYA are compared with the recent reported results. In addition, population size for EJAYA was set to 50 for all test cases. The number of function evaluations consumed by EJAYA was given in each case. 30 independent runs were executed for every test case and then the best solution was selected. In addition, in Tables 7, 9, 11, 13, 15, 17 and 19, "BEST", "MEAN", "MEDIAN", "WORST" and "STD" stand for the optimal solution, the mean solution, the median solution, the worst solution and standard deviance, respectively. In Tables 7–20, "NFEs" means the consumed number of function evaluations.

Table 8
The optimal solutions obtained by EJAYA and the compared algorithms on welded beam design problem.

Algorithm	The optimal variable				The optimal cost	NFEs
	x_1	x_2	x_3	x_4	$f(\boldsymbol{x})$	
SBM	0.2407	6.4851	8.2399	0.2497	2.4426	19,259
SCA	0.2444382760	6.2379672340	8.2885761430	0.2445661820	2.3854347	33,095
FSA	0.24435257	6.1257922	8.2939046	0.24435258	2.38119	56,243
BA	0.2015	3.562	9.0414	0.2057	1.7312	50,000
IPSO	0.24436898	6.21751974	8.29147139	0.24436898	2.3809565827	30,000
HSA-GA	0.2231	1.5815	12.8468	0.2245	2.2500	26,466
CDE	0.203137	3.542998	9.033498	0.206179	1.733462	240,000
CPSO	0.202369	3.544214	9.048210	0.205700	1.728024	200,000
EO	0.2057	3.4705	9.03664	0.2057	1.7249	15,000
WCA	0.205728	3.47052	9.036620	0.205729	1.724856	46,450
WOA	0.205396	3.484293	9.037426	0.206276	1.730499	9,900
GSA	0.182129	3.856979	10.00000	0.202376	1.879952	10,750
RO	0.203687	3.528467	9.004233	0.207241	1.735344	NA
SSA	0.2057	3.4714	9.0366	0.2057	1.72491	NA
HGSO	0.2054	3.4476	9.0269	0.2060	1.7260	30,000
EHO	0.4834	2.4950	4.4538	0.8488	2.3234	30,000
GWO	0.2054	3.4778	9.0388	0.2067	1.7265	30,000
SA	0.2055	3.4751	9.0417	0.2063	1.7306	30,000
HHO	0.204039	3.531061	9.027463	0.206147	1.73199057	NA
CSA	0.2057296398	3.4704886656	9.0366239104	0.2057296398	**1.7248523086**	100,000
EJAYA	0.2057296398	3.4704886659	9.0366239103	0.2057296398	**1.7248523086**	24,000

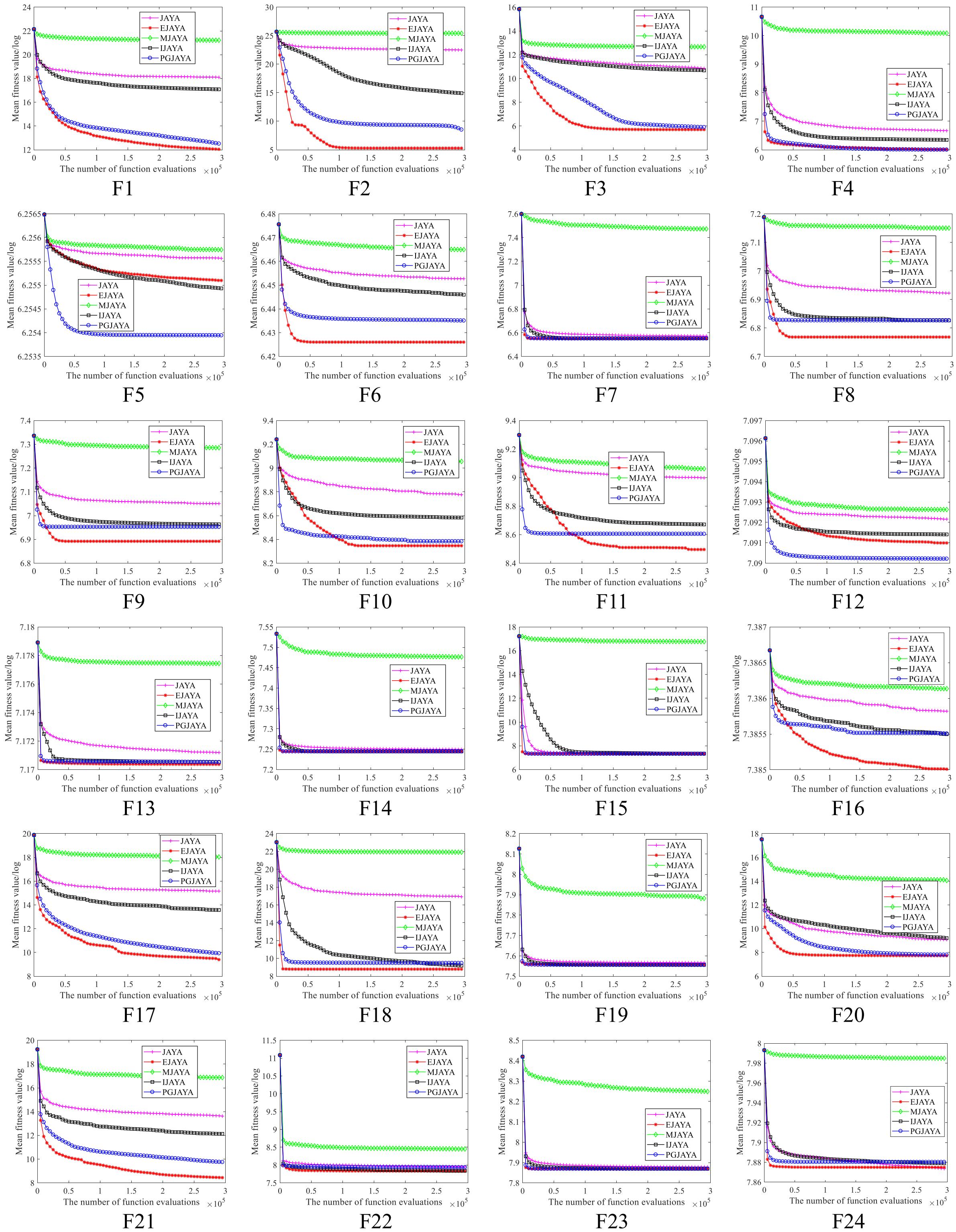

Fig. 6. Convergence curves obtained by JAYA and EJAYA for 45 test functions.

5.1. Case I: Welded beam design problem

The goal of this problem is to minimize the fabrication cost for a welded beam. This problem includes the following design variables: (1) the height of the bar $t(x_1)$; (2) the thickness of the weld $h(x_2)$; (3) the thickness of the bar $b(x_3)$; and (4) the length of the bar $l(x_4)$. The detailed description for this problem can be found in [63] and the mathematical model of this problem can be found in Appendix A.1

SBM [64], SCA [65], FSA [66], BA [67], IPSO [68], HSA-GA [69], CDE [70], CPSO [71], EO [56], WCA [1], WOA [49], GSA [49], RO [72], SSA [73], HGSO [21], EHO [21], GWO [21], SA [21],

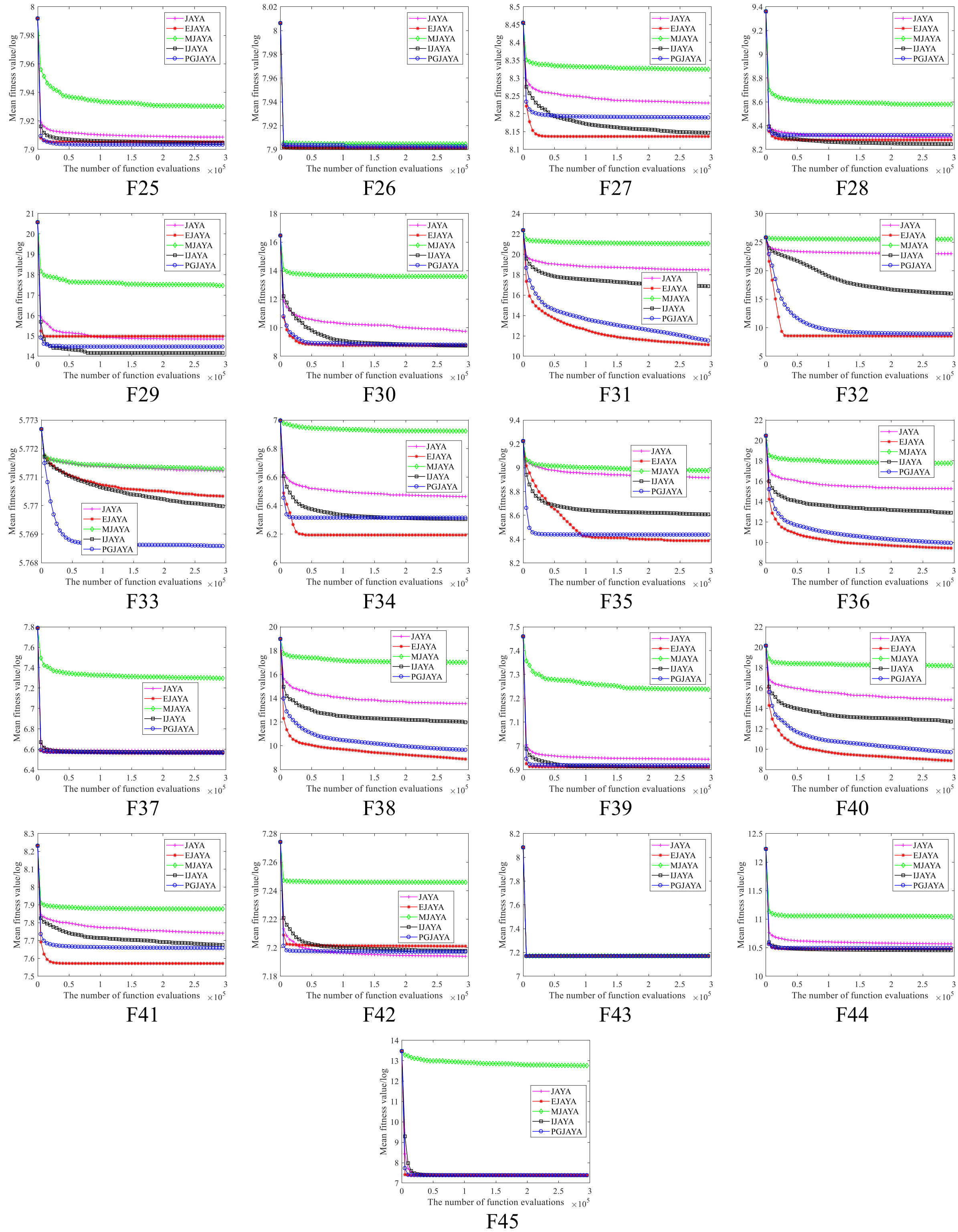

Fig. 6. (*continued*).

HHO [52], and CSA [74] have been used for solving this problem. Table 7 shows the statistical results obtained by EJAYA on the welded beam design problem. As can be seen from Table 7, the best solution of EJAYA is 1.7248523086. STD is almost zero, which means EJAYA is very stable. Table 8 presents the optimal solutions obtained by EJAYA and the compared algorithms for the welded beam design problem. From Table 8, EJAYA and CSA can achieve the best fitness value (i.e. 1.7248523086). Note that, the number of function evaluations consumed by CSA and EJAYA are 100,000 and 24,000, respectively. Obviously, EJAYA beats CSA in terms of computational efficiency.

Table 10
The optimal solutions obtained by EJAYA and the compared algorithms on tension/compression spring problem.

Algorithm	The optimal variable			The optimal weight	NFEs
	x_1	x_2	x_3	$f(\boldsymbol{x})$	
CPSO	0.051728	0.357644	11.244543	0.0126747	240,000
MFO	0.051994457	0.036410932	10.868421862	0.0126669	NA
DELC	0.051689	0.356717	11.288965	**0.012665**	20,000
SSA	0.051207	0.345215	12.004032	0.126763	NA
GWO	0.05169	0.356737	11.28885	0.012666	NA
WOA	0.051207	0.345215	12.004032	0.0126763	4,410
RO	0.051370	0.349096	11.76279	0.0126788	NA
HEAA	0.051689	0.356729	11.288293	**0.012665**	24,000
WEO	0.051685	0.356630	11.294103	**0.012665**	20,000
EO	0.0516199100	0.355054381	11.38796759	0.012666	15,000
IHS	0.051154	12.076432	0.349871	0.0126706	50,000
HGA	0.051622	0.355105	11.384534	**0.012665**	36,000
CA	0.050000	0.317395	14.031795	0.012721	50,000
CDE	0.051609	0.354714	11.410831	0.012670	240,000
MSCA	0.051668	0.356199	11.3207	0.0126670	NA
OLCGOA	0.051586809	0.354262809	11.4365114	0.012667456	30,000
CSA	0.051689	0.356717	11.289012	**0.012665**	50,000
ISOS	0.051689061903120	0.356717759535058	11.288964594575669	**0.012665**	40,000
GPEA	0.051860	0.360847	11.050894	**0.012665**	19,760
EJAYA	0.05174315969	0.35802045837	11.2130152685	**0.012665**	15,000

Table 11
Statistical results obtained by EJAYA on pressure vessel design problem (NFEs=16,000).

WORST	MEAN	MEDIAN	BEST	STD
5894.777	5885.886	5885.366	5885.333	1.734

5.2. Case II: Tension/compression spring design problem

The objective of this problem is to minimize the weight of a tension/compression spring design problem. Three design variables are considered in the problem: (1) the wire diameter $d(x_1)$; (2) the mean coil diameter $D(x_2)$; and (3) the number of active coils $P(x_3)$. The detailed description for this problem can be found in [75] and the mathematical model of this problem can be found in Appendix A.2.

This problem has been optimized previously by CPSO [76], MFO [77], DELC [78], SSA [73], GWO [48], WOA [49], RO [72], HEAA [79], WEO [80], EO [56], IHS [81], HGA [82], CA [83], CDE [70], MSCA [84], OLCGOA [85], CSA [74], ISOS [86], and GPEA [87]. Table 9 presents the statistical results of EJAYA on this problem. From Table 9, the best solution of EJAYA is 0.012665. The optimal solutions obtained by EJAYA and the compared algorithms for the problem are shown in Table 10. According to Table 10, DELC, HEAA, WEO, HGA, CSA, ISOS, GPEA and EJAYA are the best in terms of the obtained optimal weight. It should be noted that the number of function evaluations consumed by DELC, HEAA, WEO, HGA, CSA, ISOS, GPEA and EJAYA is 20,000, 24,000, 20,000, 36,000, 50,000, 40,000, 19,760 and 15,000, respectively. In other words, although eight algorithms can get the same solution, EJAYA outperforms the rest seven algorithms in terms of computational efficiency.

5.3. Case III: Pressure vessel design problem

Pressure vessel design problem is a very classical engineering design problem, whose objective is to minimize the total cost consisting of material, forming, and welding. There are four design variables in this problem: (1) the thickness of the shell $T_s(x_1)$; (2) the thickness of the head $T_h(x_2)$; (3) inner radius $R(x_3)$; and (4) the length of the cylindrical section of the vessel $L(x_4)$. The detailed description for this problem can be found in [88] and the mathematical model of this problem is given in Appendix A.3.

The optimization algorithms previously applied to this problem include CDE [70], CPSO [76], HPSO [89], NM-PSO [90], G-QPSO [91], HAIS-GA [92], EO [56], PO [93], GWO [48], WOA [49], GSA [94], MSCA [84], IHS [81], HS [95], PRO [96], MFO [77], ISOS [86], MBA [86], PSO [86], ICA-1 [97], ICA-4 [97], and ICA-PSO [97]. The statistical results of EJAYA on this problem are displayed in Table 11. From Table 11, the optimal solution obtained by EJAYA is 5885.333. The comparisons for the best solutions obtained by the applied algorithms are presented in Table 12. According to Table 12, EJAYA can offer the optimal solution. In addition, although the solutions obtained by PO and ICA-PSO are close to that of EJAYA, the two methods use more function evaluations than EJAYA.

5.4. Case IV: Speed reducer design problem

The objective of this problem is aimed at minimizing the weight of speed reducer, which has the following seven design variables: (1) the face width $b(x_1)$; (2) the module of teeth $m(x_2)$; (3) the number of teeth in the pinion $z(x_3)$; (4) the length of the first shaft between bearings $l_1(x_4)$; (5) the length of the second shaft between bearings $l_2(x_5)$; (6) the diameter of the first shaft $d_1(x_6)$; and (7) the diameter of the second shaft $d_2(x_7)$. The detailed description for this problem can be found in [98] and the mathematical model of this problem is given in Appendix A.4.

This problem has been optimized by MDE [99], PSO-DE [100], MBA [86], DELC [78], HEAA [79], DEDS [101], SBM [64], PSO-OPS [102], AFA [103], CS [104], ABC [105], GDA [106], GSA [107], MRFO [107], PSO [107], GA [107], and hHHO-SCA [108]. Table 13 gives the statistical results of EJAYA on this problem. From Table 13, the optimal solution offered by EJAYA is 2994.471066. Table 14 shows the optimal solutions obtained by EJAYA and the reported algorithms. Based on Table 14, DELC, DEDS, PSO-OPS and EJAYA can find the optimal weight (i.e. 2994.471066). By observing Table 10, the consumed number of function evaluations of DELC, DEDS, PSO-OPS and EJAYA is 30,000, 30,000, 25,000 and 17,000, respectively. Thus, EJAYA outperforms DELC, DEDS and PSO-OPS in terms of computational efficiency.

5.5. Case V: Car side impact design problem

The target of this problem is to minimize the weight. This problem is related to eleven parameters: thickness of B-Pillar

Table 12
The optimal solutions obtained by EJAYA and the compared algorithms for pressure vessel design problem.

Algorithm	The optimal variable				The optimal cost	NFEs
	x_1	x_2	x_3	x_4	$f(\boldsymbol{x})$	
CDE	0.8125	0.4375	42.0984	176.6376	6059.7340	204,800
CPSO	0.8125	0.4375	42.091266	176.7465	6061.0777	240,000
HPSO	0.8125	0.4375	42.0984	176.6366	6059.7143	81,000
NM-PSO	0.8036	0.3972	41.6392	182.4120	5930.3137	80,000
G-QPSO	0.8125	0.4375	42.0984	176.6372	6059.7208	8,000
HAIS-GA	0.8125	0.4375	42.0931	176.7031	6060.367	150,000
EO	0.8125	0.4375	42.098411	176.637690	6059.7340	150,00
PO	0.7782	0.3847	40.3125	199.9733	5885.3997	20,520
GWO	0.8125	0.4345	42.0892	176.7587	5930.3137	NA
WOA	0.8125	0.4375	42.0983	176.6390	6059.7410	6,300
GSA	1.1250	0.6250	55.9887	84.4542	8538.8359	7,110
MSCA	0.779256	0.399600	40.325450	199.9213	5935.7161	NA
IHS	1.125	0.625	58.29015	43.69268	7197.730	200,000
HS	1.125	0.625	58.2789	43.7549	7198.433	NA
PRO	0.7445	0.4424	38.489983	200.0000	6050.7134	NA
MFO	0.8125	0.4375	42.098445	176.636596	6059.7143	NA
ISOS	0.8125	0.4375	42.09844559585	176.63659584	6059.71433505	15,000
MBA	0.7802	0.3856	40.4292	198.4964	5889.3216	70,650
PSO	0.8125	0.4375	42.0984	176.6366	6059.7143	60,000
ICA-1	0.8125	0.4375	42.0759	176.9162	6062.468	20,000
ICA-4	0.8125	0.4375	42.9083	176.6379	6059.728	20,000
ICA-PSO	0.778258	0.384692	40.3242	199.9365	5885.484	20,000
EJAYA	0.778168665	0.38464918	40.319619559	199.99999545	**5885.333**	16,000

Table 13
Statistical results obtained by EJAYA on speed reducer design problem (NFEs=17,000).

WORST	MEAN	MEDIAN	BEST	STD
2994.471097	2994.471070	2994.471067	2994.471066	7.1926E−6

inner (x_1), thickness of B-Pillar reinforcement (x_2), thickness of floor side inner (x_3), thickness of cross members (x_4), thickness of door beam (x_5), thickness of door beltline reinforcement (x_6), thickness of roof rail (x_7), materials of B-Pillar inner (x_8), materials of floor side inner (x_9), barrier height (x_{10}) and hitting position (x_{11}). The detailed description for this problem can be found in [109] and the mathematical model of this problem can be found in Appendix A.5.

PSO [104], DE [104], GA [104], FA [110], TLBO [111] and TLCS [111] have been used to solve this problem. The statistical results produced by EJAYA on this problem are shown in Table 15. According to Table 15, the optimal solution of EJAYA is 22.8429707. The optimal solutions obtained by EJAYA and the compared algorithms are displayed in Table 16. From Table 16, EJAYA can offer the best solution. In addition, DE and FA have strong competitiveness, which can find the close solutions with that of EJAYA.

5.6. Case VI: Hydrostatic thrust bearing problem

This problem is to minimize the power loss of a hydrostatic thrust bearing. In this problem, Four design variables are needed to be considered, which are: (1) the bearing step radius $R(x_1)$; (2) the recess radius $R_o(x_2)$; (3) the oil viscosity $u(x_3)$; (4) the flow rate $Q(x_4)$. Seven different constraints are associated with this problem including load carrying capacity, inlet oil pressure, oil temperature rise, oil film thickness and physical constraints. The description for the problem can be found in [68] and the mathematical model of the problem is given in Appendix A.6.

Several algorithms have been attempted to solve this problem, such as TLBO [112], ABC [112], rank-iMDDE [113], NDE [114],

Table 14
The optimal solutions obtained by EJAYA and the compared algorithms on speed reducer design problem.

Algorithm	The optimal variable							The optimal weight	NFEs
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	$f(\boldsymbol{x})$	
MDE	3.500010	0.70000	17.0000	7.300156	7.800027	3.350221	5.286685	2996.356689	30,000
PSO-DE	3.500000	0.70000	17.0000	7.300000	7.800000	3.350214	5.2866832	2996.348167	54,350
MBA	3.500000	0.70000	17.0000	7.300033	7.715772	3.350218	5.286654	2994.482453	6,300
DELC	3.500000	0.70000	17.0000	7.300000	7.715319	3.350214	5.286654	**2994.471066**	30,000
HEAA	3.500022	0.70000	17.0000	7.300427	7.715377	3.350230	5.286663	2994.499107	40,000
DEDS	3.500000	0.70000	17.0000	7.300000	7.715319	3.350214	5.286654	**2994.471066**	30,000
SBM	3.506122	0.70001	17.0000	7.549126	7.859330	3.365576	5.289773	2008.080000	12,630
PSO-OPS	3.500000	0.70000	17.0000	7.300000	7.715320	3.350215	5.286654	**2994.471066**	25,000
AFA	3.500000	0.70000	17.0000	7.302489	7.800067	3.350219	5.286683	2996.669016	50,000
CS	3.501500	0.70000	17.0000	7.605000	7.818100	3.352000	5.2875000	3000.981000	5,000
ABC	3.499999	0.70000	17.0000	7.300000	7.800000	3.350215	5.287800	2997.058412	30,000
GDA	3.500000	0.70000	17.0000	7.300000	7.800000	3.350215	5.286683	2996.348072	20,000
GSA	3.534231	0.70087	17.8168	7.397778	8.245481	3.492062	5.4292329	3301.5843116	25,000
MRFO	3.500000	0.70000	17.0000	7.300000	7.715320	3.350215	5.2866545	2994.4710667	25,000
PSO	3.565812	0.702949	17.0905	7.588956	7.942941	3.440177	5.3141866	3098.7993343	25,000
GA	3.563840	0.700963	17.0972	7.589257	7.988106	3.399804	5.3871209	3127.7366085	25,000
hHHO-SCA	3.560612	0.70000	17.0000	7.300000	7.991410	3.452569	5.286749	3029.873076	NA
EJAYA	3.500000	0.70000	17.0000	7.300000	7.715320	3.350215	5.286654	**2994.471066**	17,000

Table 15
Statistical results obtained by EJAYA on car side impact design problem (NFEs=27,000).

WORST	MEAN	MEDIAN	BEST	STD
23.2619126	22.9439823	22.8430435	22.8429707	1.7098E−01

PVS [2], IPSO [68], and SDO [115]. The obtained statistical results by EJAYA on this problem are given in Table 17. As shown in Table 17, the optimal solution is 1625.442764498248. The optimal solutions offered by EJAYA and the compared algorithms are shown in Table 18. From Table 18, EJAYA can find the best solution. In addition, NDE and PVS can get very competitive solutions while they still cannot compete with EJAYA.

5.7. Case VII: Rolling element bearing design problem

The objective of this problem is to maximize the dynamic load carting capacity of a rolling element bearing. Ten design variables are considered for the optimization problem where $D_{\mathrm{m}}(x_1)$ is the pitch diameter, $D_{\mathrm{b}}(x_2)$ is the ball diameter, $Z(x_3)$ is the number of balls, $f_{\mathrm{i}}(x_4)$ is the inner raceway curvature coefficients, $f_{\mathrm{o}}(x_5)$ is the outer raceway curvature coefficients, $K_{\mathrm{Dmin}}(x_6)$ is the minimum ball diameter limiter, $K_{\mathrm{Dmax}}(x_7)$ is the maximum ball diameter limiter, $\varepsilon(x_8)$ is the parameter for outer ring strength consideration, $e(x_9)$ is the parameter for mobility condition, $\zeta(x_{10})$ is the bearing width limiter. Note that, Z is a discrete variable and the rest variables are continuous variables. The detailed description for the problem can be found in [52] and the mathematical model of the problem is given in Appendix A.7.

To solve this problem, PVS [2], HHO [52], TLBO [112], EPO [116], NDE [114], WCA [1] and MBA [117] have been used. The obtained statistical results by EJAYA on this problem are given in Table 19. From Table 20, the optimal solution is 85549.23914236. In addition, NDE can get the competitive solution (i.e. 85549.239142260223), which is very close to that of EJAYA.

6. Conclusions and further work

This paper proposes a simple but efficient optimization algorithm called enhanced Jaya algorithm (EJAYA). EJAYA is a variant of Jaya algorithm (JAYA). To enhance the global search ability of JAYA, local exploitation and global exploration of EJAYA are designed by making full use of population information. Further, local exploitation of EJAYA is achieved by using the current best solution, the current worst solution and the current mean solution. Global exploration of EJAYA is guided by the historical population information. The most remarkable features of EJAYA are that it has a very simple structure and only needs the essential parameters (i.e. population size and terminal condition) for solving optimization problems. The performance of EJAYA is tested by 45 complex test functions extracted from CEC 2014 and CEC 2015 test suites and seven real-world engineering design optimization problems. Experimental results demonstrate the improved strategies for JAYA are very effective and EJAYA can offer better solutions than the compared algorithms on most test cases.

Note that, EJAYA is a new variant of JAYA. This work only checks the performance of EJAYA on some classical test cases. Given the excellent global search ability and the feature of EJAYA without any effort for fine tuning initial parameters, EJAYA has great potential to be used to solve various types of optimization problems. Therefore, our future work will focus on the applications of EJAYA on practical engineering optimization problems, such as urban trip recommendation in smart city, permutation flow shop scheduling problem and smooth path planning of mobile robots.

CRediT authorship contribution statement

Yiying Zhang: Conceptualization, Methodology, Writing – original draft. **Aining Chi:** Writing – review & editing. **Seyedali Mirjalili:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Table 16
The optimal solutions obtained by EJAYA and the compared algorithms on car side impact design problem.

Variable	PSO	DE	GA	FA	TLBO	TLCS	EJAYA
x_1	0.50000	0.50000	0.50005	0.50000	0.50000	0.50000	0.50000000
x_2	1.11670	1.11670	1.28017	1.36000	1.11350	1.11630	1.11631315
x_3	0.50000	0.50000	0.50001	0.50000	0.50000	0.50000	0.50000000
x_4	1.30208	1.30208	1.03302	1.20200	1.30700	1.30230	1.30228464
x_5	0.50000	0.50000	0.50001	0.50000	0.50000	0.50000	0.50000022
x_6	1.50000	1.50000	0.50000	1.12000	1.50000	1.50000	1.49999999
x_7	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000	0.50000006
x_8	0.34500	0.34500	0.34994	0.34500	0.34500	0.34500	0.34499999
x_9	0.19200	0.19200	0.19200	0.19200	0.19200	0.19200	0.32679979
x_{10}	−19.54935	−19.54935	10.3119	8.87307	−20.0655	−19.5721	−19.570927
x_{11}	−0.00431	−0.00431	0.00167	−18.99808	0.11390	0.0157	0.00837595
$f(\boldsymbol{x})$	22.84474	22.84298	22.85653	22.84298	22.8436	22.8430	**22.8429707**
NFEs	20,000	20,000	20,000	20,000	20,000	8,000	27,000

Table 17
Statistical results obtained by EJAYA on hydrostatic thrust bearing problem (NFEs=150,000).

WORST	MEAN	MEDIAN	BEST	STD
1767.660483606390	1631.509586823626	1625.442764510401	1625.442764498248	26.27208859624

Table 18
The optimal solutions obtained by EJAYA and the compared algorithms on hydrostatic thrust bearing problem.

Algorithm	The optimal variable				The optimal power loss	NFEs
	x_1	x_2	x_3	x_4	$f(\boldsymbol{x})$	
TLBO	5.9557805026154158	5.3890130519416788	0.0000053586972670629	2.2696559728097379	1625.443	25,000
ABC	5.9557805026154158	5.3890130519416788	0.0000053586972670629	2.2696559728097379	1625.276	25,000
rank-iMDDE	5.955817	5.389051	0.000005358711	2.269693	1625.460142	25,000
NDE	5.9557804954072431	5.3890130457499099	0.0000053586972684	2.2696559656861695	1625.4427649676115	24,000
PVS	5.95578050261541	5.38901305194167	0.00000535869726706299	2.26965597280973	1625.4427649821	25,000
IPSO	5.956868685	5.389175395	0.00000540213310	2.30154678	1632.1249	90,000
SDO	5.957853282295	5.391201948055	0.000005361337511	2.273155235181	1626.2227	50,000
EJAYA	5.955780495321750	5.389013045775860	0.000005358697266	2.269655963392383	**1625.442764498248**	150,000

Table 19
Statistical results obtained by EJAYA on rolling element bearing problem (NFEs=20,000).

WORST	MEAN	MEDIAN	BEST	STD
84372.78285857	85324.22100686	85505.85946791	85549.23914614	401.2027

Table 20
The optimal solutions obtained by EJAYA and the compared algorithms for rolling element bearing design problem.

Algorithm	PVS	HHO	TLBO	EPO	NDE	WCA	MBA	EJAYA
x_1	125.719060	125	125.7191	125	125.7190556146683	125.721167	125.7153	125.7190556
x_2	21.42559	21	21.42559	21.41890	21.42559024077250	21.423300	21.423300	21.42559024
x_3	11	11.092073	11	10.94113	11	11.001030	11	11
x_4	0.515	0.515	0.515	0.515	0.515000000000388	0.515000	0.515	0.515
x_5	0.515	0.515	0.515	0.515	0.515000016599447	0.515000	0.515	0.515
x_6	0.400430	0.4	0.424266	0.4	0.459856414789225	0.401514	0.488805	0.400861666
x_7	0.680160	0.6	0.633948	0.7	0.619398026392338	0.659047	0.627829	0.622844252
x_8	0.3	0.3	0.3	0.3	0.300000000001946	0.300032	0.300149	0.3
x_9	0.0079990	0.050474	0.068858	0.02	0.044951766218101	0.040045	0.097305	0.1
x_{10}	0.7	0.6	0.799498	0.6	0.654902937732652	0.600000	0.646095	0.602262291
$f(\boldsymbol{x})$	81859.74120	83011.88329	81859.74	85067.983.	85549.239142260223	85538.48	85535.9611	**85549.23914236**
NFEs	20,000	NA	20,000	NA	15,000	10,000	15,100	20,000

Appendix

A.1. The mathematical model of the welded beam design problem

$$
\begin{aligned}
&\text{Minimize} f(\boldsymbol{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2)\\
&\text{Subject to:}\\
&g_1(\boldsymbol{x}) = \tau(\boldsymbol{x}) - \tau_{\max} \le 0\\
&g_2(\boldsymbol{x}) = \sigma(\boldsymbol{x}) - \sigma_{\max} \le 0\\
&g_3(\boldsymbol{x}) = x_1 - x_4 \le 0\\
&g_4(\boldsymbol{x}) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0\\
&g_5(\boldsymbol{x}) = 0.125 - x_1 \le 0\\
&g_6(\boldsymbol{x}) = \delta(\boldsymbol{x}) - \delta_{\max} \le 0\\
&g_7(\boldsymbol{x}) = P - P_c(\boldsymbol{x}) \le 0\\
&0.1 \le x_i \le 2 i = 1, 4\\
&0.1 \le x_i \le 10 i = 2, 3\\
&\text{where,}\\
&\tau(\boldsymbol{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2},\ \tau' = \frac{P}{\sqrt{2}x_1x_2},\ \tau'' = \frac{MR}{J},\\
&M = P\left(L + \frac{x_2}{2}\right),\ R = \sqrt{\left(\frac{x_2}{2}\right)^2 + \left(\frac{x_1+x_3}{2}\right)^2},\\
&J = 2\left(\sqrt{2}x_1x_2\left(\frac{x_2^2}{12} + \left(\frac{x_1+x_3}{2}\right)^2\right)\right),\ \sigma(\boldsymbol{x}) = \frac{6PL}{x_4x_3^2},\\
&\delta(\boldsymbol{x}) = \frac{4PL^3}{Ex_3^3x_4},\\
&P_c(\boldsymbol{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right)\\
&P = 6000\text{lb},\ L = 14\text{in},\ E = 30 \times 10^6\text{psi},\ G = 12 \times 10^6\text{psi},\\
&\tau_{\max} = 13,600\text{psi},\ \sigma_{\max} = 30,000\text{psi},\ \delta_{\max} = 0.25\text{in}
\end{aligned}
$$

A.2. The mathematical model of the tension/compression spring design problem

$$
\begin{aligned}
&\text{Minimize } f(\boldsymbol{x}) = (x_3 + 2)x_2x_1^2\\
&\text{Subject to:}\\
&g_1(\boldsymbol{x}) = 1 - \frac{x_2^3x_3}{71,785x_1^4} \le 0\\
&g_2(\boldsymbol{x}) = 4x_2^2 - \frac{x_1x_2}{12.566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} - 1 \le 0\\
&g_3(\boldsymbol{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0\\
&g_4(\boldsymbol{x}) = x_2 + \frac{x_1}{1.5} - 1 \le 0\\
&\text{where,}\\
&0.05 \le x_1 \le 2,\ 0.25 \le x_2 \le 1.30,\ 2.00 \le x_3 \le 15.00
\end{aligned}
$$

A.3. The mathematical model of the pressure vessel design problem

$$
\begin{aligned}
&\text{Minimize } f(\boldsymbol{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4\\
&+19.84x_1^2x_3\\
&\text{Subject to:}\\
&g_1(\boldsymbol{x}) = -x_1 + 0.0193x_3 \le 0\\
&g_2(\boldsymbol{x}) = -x_2 + 0.00954x_3 \le 0\\
&g_3(\boldsymbol{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296,000 \le 0\\
&g_4(\boldsymbol{x}) = x_4 - 240 \le 0\\
&\text{where,}\\
&0 \le x_i \le 100,\ i = 1, 2\\
&10 \le x_i \le 200,\ i = 3, 4
\end{aligned}
$$

A.4. The mathematical model of the speed reducer design problem

$$\text{Minimize } f(\boldsymbol{x}) = 0.7854x_1x_2^2\left(3.3333x_3^2+14.9334x_3-43.0934\right) -1.508x_1\left(x_6^2+x_7^2\right)+7.4777\left(x_6^3+x_7^3\right)+ 0.7854\left(x_4x_6^2+x_5x_7^2\right)$$

Subject to:

$$g_1(\boldsymbol{x}) = \frac{27}{x_1x_2^2x_3} - 1 \le 0$$
$$g_2(\boldsymbol{x}) = \frac{397.5}{x_1x_2^2x_3} - 1 \le 0$$
$$g_3(\boldsymbol{x}) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \le 0$$
$$g_4(\boldsymbol{x}) = \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \le 0$$
$$g_5(\boldsymbol{x}) = \frac{\left(\left(\frac{745x_4}{x_2x_3}\right)^2+16.9\times10^6\right)^{1/2}}{110x_6^3} - 1 \le 0$$
$$g_6(\boldsymbol{x}) = \frac{\left(\left(\frac{745x_5}{x_2x_3}\right)^2+157.5\times10^6\right)^{1/2}}{85x_7^3} - 1 \le 0$$
$$g_7(\boldsymbol{x}) = \frac{x_2x_3}{40} - 1 \le 0$$
$$g_8(\boldsymbol{x}) = \frac{5x_2}{x_1} - 1 \le 0$$
$$g_9(\boldsymbol{x}) = \frac{x_1}{12x_2} - 1 \le 0$$
$$g_{10}(\boldsymbol{x}) = \frac{1.5x_6+1.9}{x_4} - 1 \le 0$$
$$g_{11}(\boldsymbol{x}) = \frac{1.1x_7+1.9}{x_5} - 1 \le 0$$

where,

$$2.6 \le x_1 \le 3.6,\ 0.7 \le x_2 \le 0.8,\ 17 \le x_3 \le 28,\ 7.3 \le x_4 \le 8.3,\ 7.3 \le x_5 \le 8.3,\ 2.9 \le x_6 \le 3.9,\ 5.0 \le x_7 \le 5.5$$

A.5. The mathematical model of the car side impact design problem

$$\text{Minimize } f(\boldsymbol{x}) = 1.98+4.90x_1+6.67x_2+6.98x_3+4.01x_4 +1.78x_5+2.73x_7$$

Subject to

$$g_1(\boldsymbol{x}) = 1.16-0.3717x_2x_4-0.00931x_2x_{10}-0.484x_3x_9 +0.01343x_6x_{10} \le 1\text{KN}$$
$$g_2(\boldsymbol{x}) = 0.261-0.0159x_1x_2-0.0188x_1x_8-0.0191x_2x_7 +0.0144x_3x_5+0.0008757x_5x_{10}+0.08045x_6x_9 +0.00139x_8x_{11}+0.00001575x_{10}x_{11} \le 0.32\text{m/s}$$
$$g_3(\boldsymbol{x}) = 0.214+0.00817x_5-0.131x_1x_8-0.0704x_1x_9 +0.03099x_2x_6-0.018x_2x_7+0.0208x_3x_8+0.121x_3x_9 -0.00364x_5x_6+0.0007715x_5x_{10}-0.0005354x_6x_{10} +0.00121x_8x_{11} \le 0.32\text{m/s}$$
$$g_4(\boldsymbol{x}) = 0.74-0.61x_2-0.163x_3x_8+0.001232x_3x_{10} -0.166x_7x_9+0.227x_2^2 \le 0.32\text{m/s}$$
$$g_5(\boldsymbol{x}) = 28.98+3.818x_3-4.2x_1x_2+0.0207x_5x_{10}+6.63x_6x_9 -7.7x_7x_8+0.32x_9x_{10} \le 32\text{ mm}$$
$$g_6(\boldsymbol{x}) = 33.86+2.95x_3+0.1792x_{10}-5.057x_1x_2-11.0x_2x_8 -0.0215x_5x_{10}-9.98x_7x_8+22.0x_8x_9 \le 32\text{ mm}$$
$$g_7(\boldsymbol{x}) = 46.36-9.9x_2-12.9x_1x_8-5.057x_1x_2 +0.1107x_3x_{10} \le 32\text{ mm}$$
$$g_8(\boldsymbol{x}) = 4.72-0.5x_4-0.19x_2x_3-0.0122x_4x_{10} +0.009325x_6x_{10}+0.000191x_{11}^2 \le 4\text{KN}$$
$$g_9(\boldsymbol{x}) = 10.58-0.674x_1x_2-1.95x_2x_8+0.02054x_3x_{10} -0.0198x_4x_{10}+0.028x_6x_{10} \le 9.9\text{ mm/ms}$$
$$g_{10}(\boldsymbol{x}) = 16.45-0.489x_3x_7-0.843x_5x_6+0.0432x_9x_{10} -0.0556x_9x_{11}-0.000786x_{11}^2 \le 15.7\text{ mm/ms}$$

where $0.5 \le x_i \le 1.5,\ i = 1,2,3,4,5,6,7;\ 0.192 \le x_i \le 0.345,\ i = 8,9;\ -30 \le x_i \le 30,\ i = 10,11.$

A.6. The mathematical model of hydrostatic thrust bearing problem

$$\text{Minimize } f(\boldsymbol{x}) = \frac{QP_o}{0.7} + E_f$$

Subject to

$$g_1(\boldsymbol{x}) = W - W_s \ge 0$$
$$g_2(\boldsymbol{x}) = P_{\max} - P_o \ge 0$$
$$g_3(\boldsymbol{x}) = \Delta T_{\max} - \Delta T \ge 0$$
$$g_4(\boldsymbol{x}) = h - h_{\min} \ge 0$$
$$g_5(\boldsymbol{x}) = R - R_o \ge 0$$
$$g_6(\boldsymbol{x}) = 0.001 - \frac{\gamma}{gP_o}\left(\frac{Q}{2\pi Rh}\right) \ge 0$$
$$g_7(\boldsymbol{x}) = 5000 - \frac{W}{\pi\left(R^2-R_o^2\right)} \ge 0$$

where,

$$W = \frac{\pi P_o}{2}\frac{R^2-R_o^2}{\ln\frac{R}{R_o}},\ P_o = \frac{6uQ}{\pi h^3}\ln\frac{R}{R_o},\ p = \frac{\log_{10}\log_{10}(8.122e6u+0.8)-C_1}{n},$$
$$h = \left(\frac{2\pi N}{60}\right)^2\frac{2\pi u}{E_f}\left(\frac{R^4}{4}-\frac{R_o^4}{4}\right),\ E_f = 9336Q\gamma C\Delta_T,$$
$$\Delta_T = 2\left(10^p-560\right),\ \gamma = 0.0307,\ C = 0.5,\ n = -3.55,$$
$$C_1 = 10.04,\ W_s = 101000,\ P_{\max} = 1000,\ \Delta_{T\max} = 50,$$
$$h_{\min} = 0.001,\ g = 386.4,\ N = 750.1 \le R, R_o, Q \le 16,$$
$$1e-6 \le u \le 16e-6.$$

A.7. The mathematical model of rolling element bearing design problem

$$\text{Maximum } f(\boldsymbol{x}) = \begin{cases} f_c Z^{2/3} D_b^{1.8}, & \text{if } D \le 25.4\text{mm} \\ 3.647 f_c Z^{2/3} D_b^{1.4}, & \text{if } D > 25.4\text{mm} \end{cases}$$

Subject to

$$g_1(\boldsymbol{x}) = \frac{\phi_0}{2\sin^{-1}(D_b/D_m)} - Z + 1 \ge 0$$
$$g_2(\boldsymbol{x}) = 2D_b - K_{D\min}(D-d) \ge 0$$
$$g_3(\boldsymbol{x}) = K_{D\max}(D-d) - 2D_b \ge 0$$
$$g_4(\boldsymbol{x}) = \zeta B_w - D_b \le 0$$
$$g_5(\boldsymbol{x}) = D_m - 0.5(D+d) \ge 0$$
$$g_6(\boldsymbol{x}) = (0.5+e)(D+d) - D_m \ge 0$$
$$g_7(\boldsymbol{x}) = 0.5(D-D_m-D_b) - \varepsilon D_b \ge 0$$
$$g_8(\boldsymbol{x}) = f_i \ge 0.515$$
$$g_9(\boldsymbol{x}) = f_o \ge 0.515$$
$$\gamma = \frac{D_b}{D_m},\ f_i = \frac{r_i}{D_b},\ f_o = \frac{r_o}{D_b},\ T = D-d-2D_b,\ D = 160,\ d = 90,\ B_w = 30,\ D = 160,\ r_i = r_o = 11.033$$
$$0.5(D+d) \le D_m \le 0.6(D+d),\ 0.15(D-d) \le D_b \le 0.45(D-d),\ 4 \le Z \le 50,\ 0.515 \le f_i \le 0.6,\ 0.515 \le f_o \le 0.60.4 \le K_{D\min} \le 0.5,\ 0.6 \le K_{D\max} \le 0.7,\ 0.3 \le e \le 0.4,\ 0.02 \le \varepsilon \le 0.1,\ 0.6 \le \zeta \le 0.85$$
$$f_c = 37.91\left[1+\left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i(2f_o-1)}{f_o(2f_i-1)}\right)^{0.41}\right\}^{10/3}\right]^{-0.3} \times\left(\frac{\gamma^{0.3}(1-\gamma)^{1.39}}{(1+\gamma)^{1./3}}\right)\left[\frac{2f_i}{2f_i-1}\right]^{0.41}$$
$$\phi_0 = 2\pi - 2\cos^{-1}\frac{\left[\{(D-d)/2-3(T/4)\}^2+\{D/2-(T/4)-D_b\}^2-\{d/2+(T/4)\}^2\right]}{2\{(D-d)/2-3(T/4)\}\{D/2-(T/4)-D_b\}}$$

References

[1] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, Comput. Struct. 110–111 (2012) 151–166, http://dx.doi.org/10.1016/j.compstruc.2012.07.010.

[2] P. Savsani, V. Savsani, Passing vehicle search (PVS): A novel metaheuristic algorithm, Appl. Math. Model. 40 (2016) 3951–3978, http://dx.doi.org/10.1016/j.apm.2015.10.040.

[3] J. Zhang, M. Xiao, L. Gao, Q. Pan, Queuing search algorithm: A novel metaheuristic algorithm for solving engineering optimization problems, Appl. Math. Model. 63 (2018) 464–490, http://dx.doi.org/10.1016/j.apm.2018.06.036.

[4] S. Mirjalili, SCA: A Sine cosine algorithm for solving optimization problems, Knowl.-Based Syst. 96 (2016) 120–133, http://dx.doi.org/10.1016/j.knosys.2015.12.022.

[5] H. Garg, A hybrid GSA-GA algorithm for constrained optimization problems, Inform. Sci. 478 (2019) 499–523, http://dx.doi.org/10.1016/j.ins.2018.11.041.

[6] M.-Y. Cheng, D. Prayogo, Symbiotic Organisms Search: A new metaheuristic optimization algorithm, Comput. Struct. 139 (2014) 98–112, http://dx.doi.org/10.1016/j.compstruc.2014.03.007.

[7] V.F. Yu, A.A.N.P Redi, P. Jewpanya, A. Gunawan, Selective discrete particle swarm optimization for the team orienteering problem with time windows and partial scores, Comput. Ind. Eng. 138 (2019) 106084, http://dx.doi.org/10.1016/j.cie.2019.106084.

[8] M. Gong, Y. Wu, Q. Cai, W. Ma, A.K. Qin, Z. Wang, L. Jiao, Discrete particle swarm optimization for high-order graph matching, Inform. Sci. 328 (2016) 158–171, http://dx.doi.org/10.1016/j.ins.2015.08.038.

[9] R. Abbassi, A. Abbassi, A.A. Heidari, S. Mirjalili, An efficient salp swarm-inspired algorithm for parameters identification of photovoltaic cell models, Energy Convers. Manag. 179 (2019) 362–372, http://dx.doi.org/10.1016/j.enconman.2018.10.069.

[10] L.T. Kóczy, P. Földesi, B. Tüű-Szabó, Enhanced discrete bacterial memetic evolutionary algorithm - an efficacious metaheuristic for the traveling salesman optimization, Inform. Sci. 460–461 (2018) 389–400, http://dx.doi.org/10.1016/j.ins.2017.09.069.

[11] X. Li, L. Gao, W. Wang, C. Wang, L. Wen, Particle swarm optimization hybridized with genetic algorithm for uncertain integrated process planning and scheduling with interval processing time, Comput. Ind. Eng. 135 (2019) 1036–1046, http://dx.doi.org/10.1016/j.cie.2019.04.028.

[12] M. Gholamghasemi, E. Akbari, M.B. Asadpoor, M. Ghasemi, A new solution to the non-convex economic load dispatch problems using phasor particle swarm optimization, Appl. Soft Comput. 79 (2019) 111–124, http://dx.doi.org/10.1016/j.asoc.2019.03.038.

[13] Y. Zhang, Z. Jin, Group teaching optimization algorithm: A novel metaheuristic method for solving global optimization problems, Expert Syst. Appl. 148 (2020) 113246, http://dx.doi.org/10.1016/j.eswa.2020.113246.

[14] A.W. Mohamed, An improved differential evolution algorithm with triangular mutation for global numerical optimization, Comput. Ind. Eng. 85 (2015) 359–375, http://dx.doi.org/10.1016/j.cie.2015.04.012.

[15] S. Rahnamayan, H.R. Tizhoosh, M.M.A. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (2008) 64–79, http://dx.doi.org/10.1109/TEVC.2007.894200.

[16] R. Sarkhel, N. Das, A.K. Saha, M. Nasipuri, An improved harmony search algorithm embedded with a novel piecewise opposition based learning algorithm, Eng. Appl. Artif. Intell. 67 (2018) 317–330, http://dx.doi.org/10.1016/j.engappai.2017.09.020.

[17] A. Turky, S. Abdullah, A. Dawod, A dual-population multi operators harmony search algorithm for dynamic optimization problems, Comput. Ind. Eng. 117 (2018) 19–28, http://dx.doi.org/10.1016/j.cie.2018.01.003.

[18] I.B. Aydilek, A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems, Appl. Soft Comput. 66 (2018) 232–249, http://dx.doi.org/10.1016/j.asoc.2018.02.025.

[19] D. Laha, J.N.D. Gupta, An improved cuckoo search algorithm for scheduling jobs on identical parallel machines, Comput. Ind. Eng. 126 (2018) 348–360, http://dx.doi.org/10.1016/j.cie.2018.09.016.

[20] X.-S. Yang, S. Deb, Cuckoo search: recent advances and applications, Neural Comput. Appl. 24 (2014) 169–174, http://dx.doi.org/10.1007/s00521-013-1367-1.

[21] F.A. Hashim, E.H. Houssein, M.S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Gener. Comput. Syst. 101 (2019) 646–667, http://dx.doi.org/10.1016/j.future.2019.07.015.

[22] Y. Zhang, Z. Jin, Y. Chen, Hybrid teaching–learning-based optimization and neural network algorithm for engineering design optimization problems, Knowl.-Based Syst. 187 (2020) 104836, http://dx.doi.org/10.1016/j.knosys.2019.07.007.

[23] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (2006) 281–295, http://dx.doi.org/10.1109/TEVC.2005.857610.

[24] N. Lynn, P.N. Suganthan, Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation, Swarm Evol. Comput. 24 (2015) 11–24, http://dx.doi.org/10.1016/j.swevo.2015.05.002.

[25] M. Alswaitti, M. Albughdadi, N.A.M. Isa, Density-based particle swarm optimization algorithm for data clustering, Expert Syst. Appl. 91 (2018) 170–186, http://dx.doi.org/10.1016/j.eswa.2017.08.050.

[26] H. Wang, Z. Wu, S. Rahnamayan, Y. Liu, M. Ventresca, Enhancing particle swarm optimization using generalized opposition-based learning, Spec. Issue Interpret. Fuzzy Syst. 181 (2011) 4699–4714, http://dx.doi.org/10.1016/j.ins.2011.03.016.

[27] M. Abdel-Basset, G. Manogaran, D. El-Shahat, S. Mirjalili, A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem, Future Gener. Comput. Syst. 85 (2018) 129–145, http://dx.doi.org/10.1016/j.future.2018.03.020.

[28] Y. Lin, Y. Liu, W.-N. Chen, J. Zhang, A hybrid differential evolution algorithm for mixed-variable optimization problems, Inform. Sci. 466 (2018) 170–188, http://dx.doi.org/10.1016/j.ins.2018.07.035.

[29] R. Rao, Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems, Int. J. Ind. Eng. Comput. 7 (2016) 19–34.

[30] M. Aslan, M. Gunduz, M.S. Kiran, JayaX: Jaya algorithm with xor operator for binary optimization, Appl. Soft Comput. 82 (2019) 105576, http://dx.doi.org/10.1016/j.asoc.2019.105576.

[31] R.V. Rao, K.C. More, J. Taler, P. Ocłoń, Dimensional optimization of a micro-channel heat sink using Jaya algorithm, Appl. Therm. Eng. 103 (2016) 572–582, http://dx.doi.org/10.1016/j.applthermaleng.2016.04.135.

[32] K. Yu, J.J. Liang, B.Y. Qu, X. Chen, H. Wang, Parameters identification of photovoltaic models using an improved JAYA optimization algorithm, Energy Convers. Manag. 150 (2017) 742–753, http://dx.doi.org/10.1016/j.enconman.2017.08.063.

[33] K.K. Ingle, Dr.R.K. Jatoth, An efficient JAYA algorithm with Lévy flight for non-linear channel equalization, Expert Syst. Appl. 145 (2020) 112970, http://dx.doi.org/10.1016/j.eswa.2019.112970.

[34] C. Pradhan, C.N. Bhende, Online load frequency control in wind integrated power systems using modified jaya optimization, Eng. Appl. Artif. Intell. 77 (2019) 212–228, http://dx.doi.org/10.1016/j.engappai.2018.10.003.

[35] R.V. Rao, H.S. Keesari, Multi-team perturbation guiding jaya algorithm for optimization of wind farm layout, Appl. Soft Comput. 71 (2018) 800–815, http://dx.doi.org/10.1016/j.asoc.2018.07.036.

[36] X. Yang, W. Gong, Opposition-based JAYA with population reduction for parameter estimation of photovoltaic solar cells and modules, Appl. Soft Comput. 104 (2021) 107218, http://dx.doi.org/10.1016/j.asoc.2021.107218.

[37] D.R. Nayak, Y. Zhang, D.S. Das, S. Panda, Mjaya-ELM: A jaya algorithm with mutation and extreme learning machine based approach for sensorineural hearing loss detection, Appl. Soft Comput. 83 (2019) 105626, http://dx.doi.org/10.1016/j.asoc.2019.105626.

[38] J. Liang, B. Qu, P. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Vol. 635, Comput. Intell. Lab. Zhengzhou Univ. Zhengzhou China Tech. Rep. Nanyang Technol. Univ. Singap., 2013.

[39] J. Liang, B. Qu, P. Suganthan, Q. Chen, Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-Based Real-Parameter Single Objective Optimization, Tech. Report201411A, Comput. Intell. Lab. Zhengzhou Univ. Zhengzhou China Tech. Rep. Nanyang Technol. Univ. Singap., 2014.

[40] P. Civicioglu, Backtracking search optimization algorithm for numerical optimization problems, Appl. Math. Comput. 219 (2013) 8121–8144, http://dx.doi.org/10.1016/j.amc.2013.02.017.

[41] Y. Zhang, Z. Jin, S. Mirjalili, Generalized normal distribution optimization and its applications in parameter extraction of photovoltaic models, Energy Convers. Manag. 224 (2020) 113301, http://dx.doi.org/10.1016/j.enconman.2020.113301.

[42] Y. Zhang, Neural network algorithm with reinforcement learning for parameters extraction of photovoltaic models, IEEE Trans. Neural Netw. Learn. Syst. (2021) 1–11, http://dx.doi.org/10.1109/TNNLS.2021.3109565.

[43] F. Xu, H. Li, C.-M. Pun, H. Hu, Y. Li, Y. Song, H. Gao, A new global best guided artificial bee colony algorithm with application in robot path planning, Appl. Soft Comput. 88 (2020) 106037, http://dx.doi.org/10.1016/j.asoc.2019.106037.

[44] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, A. Zamuda, Distance based parameter adaptation for success-history based differential evolution, Swarm Evol. Comput. 50 (2019) 100462, http://dx.doi.org/10.1016/j.swevo.2018.10.013.

[45] U. Škvorc, T. Eftimov, P. Korošec, Understanding the problem space in single-objective numerical optimization using exploratory landscape analysis, Appl. Soft Comput. 90 (2020) 106138, http://dx.doi.org/10.1016/j.asoc.2020.106138.

[46] K. Chen, B. Xue, M. Zhang, F. Zhou, Novel chaotic grouping particle swarm optimization with a dynamic regrouping strategy for solving numerical optimization tasks, Knowl.-Based Syst. (2020) 105568, http://dx.doi.org/10.1016/j.knosys.2020.105568.

[47] K.R. Harrison, A.P. Engelbrecht, B.M. Ombuki-Berman, Self-adaptive particle swarm optimization: a review and analysis of convergence, Swarm Intell. 12 (2018) 187–226, http://dx.doi.org/10.1007/s11721-017-0150-9.

[48] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61, http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

[49] S. Mirjalili, A. Lewis, The whale optimization algorithm, Adv. Eng. Softw. 95 (2016) 51–67, http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

[50] K. Yu, B. Qu, C. Yue, S. Ge, X. Chen, J. Liang, A performance-guided JAYA algorithm for parameters identification of photovoltaic cell and module, Appl. Energy. 237 (2019) 241–257, http://dx.doi.org/10.1016/j.apenergy.2019.01.008.

[51] E.E. Elattar, S.K. ElSayed, Modified JAYA algorithm for optimal power flow incorporating renewable energy sources considering the cost, emission, power loss and voltage profile improvement, Energy 178 (2019) 598–609, http://dx.doi.org/10.1016/j.energy.2019.04.159.

[52] A.A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, H. Chen, Harris hawks optimization: Algorithm and applications, Future Gener. Comput. Syst. 97 (2019) 849–872, http://dx.doi.org/10.1016/j.future.2019.02.028.

[53] X. Zhang, Q. Kang, X. Wang, Hybrid biogeography-based optimization with shuffled frog leaping algorithm and its application to minimum spanning tree problems, Swarm Evol. Comput. 49 (2019) 245–265, http://dx.doi.org/10.1016/j.swevo.2019.07.001.

[54] W. Zang, L. Ren, W. Zhang, X. Liu, A cloud model based DNA genetic algorithm for numerical optimization problems, Future Gener. Comput. Syst. 81 (2018) 465–477, http://dx.doi.org/10.1016/j.future.2017.07.036.

[55] Q. Askari, M. Saeed, I. Younas, Heap-based optimizer inspired by corporate rank hierarchy for global optimization, Expert Syst. Appl. 161 (2020) 113702, http://dx.doi.org/10.1016/j.eswa.2020.113702.

[56] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, Knowl.-Based Syst. 191 (2020) 105190, http://dx.doi.org/10.1016/j.knosys.2019.105190.

[57] A. Wunnava, M.K. Naik, R. Panda, B. Jena, A. Abraham, An adaptive Harris hawks optimization technique for two dimensional grey gradient based multilevel image thresholding, Appl. Soft Comput. 95 (2020) 106526, http://dx.doi.org/10.1016/j.asoc.2020.106526.

[58] S. Kumar, G.G. Tejani, N. Pholdee, S. Bureerat, Multiobjecitve structural optimization using improved heat transfer search, Knowl.-Based Syst. 219 (2021) 106811, http://dx.doi.org/10.1016/j.knosys.2021.106811.

[59] H. Rakhshani, A. Rahati, Snap-drift cuckoo search: A novel cuckoo search optimization algorithm, Appl. Soft Comput. 52 (2017) 771–794, http://dx.doi.org/10.1016/j.asoc.2016.09.048.

[60] C. Lu, L. Gao, J. Yi, Grey wolf optimizer with cellular topological structure, Expert Syst. Appl. 107 (2018) 89–114, http://dx.doi.org/10.1016/j.eswa.2018.04.012.

[61] H. Chen, A.A. Heidari, X. Zhao, L. Zhang, H. Chen, Advanced orthogonal learning-driven multi-swarm sine cosine optimization: Framework and case studies, Expert Syst. Appl. 144 (2020) 113113, http://dx.doi.org/10.1016/j.eswa.2019.113113.

[62] V. Kansal, J.S. Dhillon, Emended salp swarm algorithm for multiobjective electric power dispatch problem, Appl. Soft Comput. 90 (2020) 106172, http://dx.doi.org/10.1016/j.asoc.2020.106172.

[63] C.A. Coello Coello, Use of a self-adaptive penalty approach for engineering optimization problems, Comput. Ind. 41 (2000) 113–127, http://dx.doi.org/10.1016/S0166-3615(99)00046-9.

[64] S. Akhtar, K. Tai, T. Ray, A socio-behavioural simulation model for engineering design optimization, Eng. Optim. 34 (2002) 341–354, http://dx.doi.org/10.1080/03052150212723.

[65] T. Ray, K.M. Liew, Society and civilization: An optimization algorithm based on the simulation of social behavior, IEEE Trans. Evol. Comput. 7 (2003) 386–396, http://dx.doi.org/10.1109/TEVC.2003.814902.

[66] A.-R. Hedar, M. Fukushima, Derivative-free filter simulated annealing method for constrained continuous global optimization, J. Global Optim. 35 (2006) 521–549, http://dx.doi.org/10.1007/s10898-005-3693-z.

[67] A.H. Gandomi, X.-S. Yang, A.H. Alavi, S. Talatahari, Bat algorithm for constrained optimization tasks, Neural Comput. Appl. 22 (2013) 1239–1255, http://dx.doi.org/10.1007/s00521-012-1028-9.

[68] S. He, E. Prempain, Q.H. Wu, An improved particle swarm optimizer for mechanical design optimization problems, Eng. Optim. 36 (2004) 585–605, http://dx.doi.org/10.1080/03052150410001704854.

[69] S.-F. Hwang, R.-S. He, A hybrid real-parameter genetic algorithm for function optimization, Adv. Eng. Inf. 20 (2006) 7–21, http://dx.doi.org/10.1016/j.aei.2005.09.001.

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

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

[72] A. Kaveh, M. Khayatazad, A new meta-heuristic method: Ray optimization, Comput. Struct. 112–113 (2012) 283–294, http://dx.doi.org/10.1016/j.compstruc.2012.09.003.

[73] S. Mirjalili, A.H. Gandomi, S.Z. Mirjalili, S. Saremi, H. Faris, S.M. Mirjalili, Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems, Adv. Eng. Softw. 114 (2017) 163–191, http://dx.doi.org/10.1016/j.advengsoft.2017.07.002.

[74] A. Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm, Comput. Struct. 169 (2016) 1–12, http://dx.doi.org/10.1016/j.compstruc.2016.03.001.

[75] J.S. Arora, Introduction To Optimum Design, McGraw-Hill, New York, 1989.

[76] R.A. Krohling, L. dos S. Coelho, Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems, IEEE Trans. Syst. Man Cybern. B 36 (2006) 1407–1416, http://dx.doi.org/10.1109/TSMCB.2006.873185.

[77] S. Mirjalili, Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm, Knowl.-Based Syst. 89 (2015) 228–249, http://dx.doi.org/10.1016/j.knosys.2015.07.006.

[78] L. Wang, L. Li, An effective differential evolution with level comparison for constrained engineering design, Struct. Multidiscip. Optim. 41 (2010) 947–963, http://dx.doi.org/10.1007/s00158-009-0454-5.

[79] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Struct. Multidiscip. Optim. 37 (2009) 395–413, http://dx.doi.org/10.1007/s00158-008-0238-3.

[80] A. Kaveh, T. Bakhshpoori, Water evaporation optimization: A novel physically inspired optimization algorithm, Comput. Struct. 167 (2016) 69–85, http://dx.doi.org/10.1016/j.compstruc.2016.01.008.

[81] M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search algorithm for solving optimization problems, Appl. Math. Comput. 188 (2007) 1567–1579, http://dx.doi.org/10.1016/j.amc.2006.11.033.

[82] H.S. Bernardino, H.J.C. Barbosa, A.C.C. Lemonge, A hybrid genetic algorithm for constrained optimization problems in mechanical engineering, in: 2007 IEEE Congr. Evol. Comput., 2007, pp. 646–653, http://dx.doi.org/10.1109/CEC.2007.4424532.

[83] C.A.C. Coello, R.L. Becerra, Efficient evolutionary optimization through the use of a cultural algorithm, Eng. Optim. 36 (2004) 219–236.

[84] H. Chen, M. Wang, X. Zhao, A multi-strategy enhanced sine cosine algorithm for global optimization and constrained practical engineering problems, Appl. Math. Comput. 369 (2020) 124872, http://dx.doi.org/10.1016/j.amc.2019.124872.

[85] Z. Xu, Z. Hu, A.A. Heidari, M. Wang, X. Zhao, H. Chen, X. Cai, Orthogonally-designed adapted grasshopper optimization: A comprehensive analysis, Expert Syst. Appl. 150 (2020) 113282, http://dx.doi.org/10.1016/j.eswa.2020.113282.

[86] E. Çelik, A powerful variant of symbiotic organisms search algorithm for global optimization, Eng. Appl. Artif. Intell. 87 (2020) 103294, http://dx.doi.org/10.1016/j.engappai.2019.103294.

[87] Z. Hu, X. Xu, Q. Su, H. Zhu, J. Guo, Grey prediction evolution algorithm for global optimization, Appl. Math. Model. 79 (2020) 145–160, http://dx.doi.org/10.1016/j.apm.2019.10.026.

[88] B. Kannan, S.N. Kramer, An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design, J. Mech. Des. 116 (1994) 405–411.

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

[90] E. Zahara, Y.-T. Kao, Hybrid nelder–mead simplex search and particle swarm optimization for constrained engineering design problems, Expert Syst. Appl. 36 (2009) 3880–3886, http://dx.doi.org/10.1016/j.eswa.2008.02.039.

[91] L. dos S. Coelho, Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems, Expert Syst. Appl. 37 (2010) 1676–1683, http://dx.doi.org/10.1016/j.eswa.2009.06.044.

[92] C.A.C. Coello, N.C. Cortés, Hybridizing a genetic algorithm with an artificial immune system for global optimization, Eng. Optim. 36 (2004) 607–634, http://dx.doi.org/10.1080/03052150410001704845.

[93] Q. Askari, I. Younas, M. Saeed, Political Optimizer: A novel socio-inspired meta-heuristic for global optimization, Knowl.-Based Syst. (2020) 105709, http://dx.doi.org/10.1016/j.knosys.2020.105709.

[94] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: A gravitational search algorithm, Spec. Sect. High Order Fuzzy Sets. 179 (2009) 2232–2248, http://dx.doi.org/10.1016/j.ins.2009.03.004.

[95] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, Comput. Methods Appl. Mech. Engrg. 194 (2005) 3902–3933, http://dx.doi.org/10.1016/j.cma.2004.09.007.

[96] S.H. Samareh Moosavi, V.K. Bardsiri, Poor and rich optimization algorithm: A new human-based and multi populations algorithm, Eng. Appl. Artif. Intell. 86 (2019) 165–181, http://dx.doi.org/10.1016/j.engappai.2019.08.025.

[97] L. Idoumghar, N. Chérin, P. Siarry, R. Roche, A. Miraoui, Hybrid ICA–PSO algorithm for continuous optimization, Appl. Math. Comput. 219 (2013) 11149–11170, http://dx.doi.org/10.1016/j.amc.2013.05.027.

[98] E. Mezura-Montes, C.A.C. Coello, Useful infeasible solutions in engineering optimization with evolutionary algorithms, in: A. Gelbukh, Á. de Albornoz, H. Terashima-Marín (Eds.), Adv. Artif. Intell., MICAI 2005, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 652–662.
[99] E. Mezura-Montes, J. Velazquez-Reyes, C.A. Coello Coello, Modified differential evolution for constrained optimization, in: 2006 IEEE Int. Conf. Evol. Comput, 2006, pp. 25–32, http://dx.doi.org/10.1109/CEC.2006.1688286.
[100] H. Liu, Z. Cai, Y. Wang, Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization, Appl. Soft Comput. 10 (2010) 629–640, http://dx.doi.org/10.1016/j.asoc.2009.08.031.
[101] M. Zhang, W. Luo, X. Wang, Differential evolution with dynamic stochastic selection for constrained optimization, Nat. Inspired Probl.-Solv. 178 (2008) 3043–3074, http://dx.doi.org/10.1016/j.ins.2008.02.014.
[102] M. Isiet, M. Gadala, Sensitivity analysis of control parameters in particle swarm optimization, J. Comput. Sci. 41 (2020) 101086, http://dx.doi.org/10.1016/j.jocs.2020.101086.
[103] A. Baykasoğlu, F.B. Ozsoydan, Adaptive firefly algorithm with chaos for mechanical design optimization problems, Appl. Soft Comput. 36 (2015) 152–164, http://dx.doi.org/10.1016/j.asoc.2015.06.056.
[104] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Eng. Comput. 29 (2013) 17–35, http://dx.doi.org/10.1007/s00366-011-0241-y.
[105] B. Akay, D. Karaboga, Artificial bee colony algorithm for large-scale problems and engineering design optimization, J. Intell. Manuf. 23 (2012) 1001–1014, http://dx.doi.org/10.1007/s10845-010-0393-4.
[106] A. Baykasoglu, Design optimization with chaos embedded great deluge algorithm, Appl. Soft Comput. 12 (2012) 1055–1067, http://dx.doi.org/10.1016/j.asoc.2011.11.018.
[107] W. Zhao, Z. Zhang, L. Wang, Manta ray foraging optimization: An effective bio-inspired optimizer for engineering applications, Eng. Appl. Artif. Intell. 87 (2020) 103300, http://dx.doi.org/10.1016/j.engappai.2019.103300.
[108] V.K. Kamboj, A. Nandi, A. Bhadoria, S. Sehgal, An intensify harris hawks optimizer for numerical and engineering optimization problems, Appl. Soft Comput. 89 (2020) 106018, http://dx.doi.org/10.1016/j.asoc.2019.106018.
[109] L. Gu, R. Yang, C.-H. Tho, M. Makowski, O. Faruque, Y. Li, Optimization and robustness for crashworthiness of side impact, Int. J. Veh. Des. 26 (2001) 348–360.
[110] A.H. Gandomi, X.-S. Yang, A.H. Alavi, Mixed variable structural optimization using firefly algorithm, Comput. Struct. 89 (2011) 2325–2336, http://dx.doi.org/10.1016/j.compstruc.2011.08.002.
[111] J. Huang, L. Gao, X. Li, An effective teaching-learning-based cuckoo search algorithm for parameter optimization problems in structure designing and machining processes, Appl. Soft Comput. 36 (2015) 349–356, http://dx.doi.org/10.1016/j.asoc.2015.07.031.
[112] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems, Comput.-Aided Des. 43 (2011) 303–315, http://dx.doi.org/10.1016/j.cad.2010.12.015.
[113] W. Gong, Z. Cai, D. Liang, Engineering optimization by means of an improved constrained differential evolution, Comput. Methods Appl. Mech. Engrg. 268 (2014) 884–904, http://dx.doi.org/10.1016/j.cma.2013.10.019.
[114] A.W. Mohamed, A novel differential evolution algorithm for solving constrained engineering optimization problems, J. Intell. Manuf. 29 (2018) 659–692, http://dx.doi.org/10.1007/s10845-017-1294-6.
[115] W. Zhao, L. Wang, Z. Zhang, Supply-demand-based optimization: A novel economics-inspired algorithm for global optimization, IEEE Access 7 (2019) 73182–73206, http://dx.doi.org/10.1109/ACCESS.2019.2918753.
[116] G. Dhiman, V. Kumar, Emperor penguin optimizer: A bio-inspired algorithm for engineering problems, Knowl.-Based Syst. 159 (2018) 20–50, http://dx.doi.org/10.1016/j.knosys.2018.06.001.
[117] A. Sadollah, A. Bahreininejad, H. Eskandar, M. Hamdi, Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems, Appl. Soft Comput. 13 (2013) 2592–2612, http://dx.doi.org/10.1016/j.asoc.2012.11.026.