


A novel chaotic Jaya algorithm for unconstrained numerical optimization

Anouar Farah · Akram Belazi 

Received: 20 June 2017 / Accepted: 4 April 2018 / Published online: 20 April 2018
© Springer Science+Business Media B.V., part of Springer Nature 2018

Abstract Jaya algorithm is one of the recent algorithms developed to solve optimization problems. The basic concept of this algorithm consists in moving the obtained solution, for a given problem, toward the best solution and avoiding the worst one. However, it severely suffers from premature convergence problem and therefore can be easily trapped in local optimums. This study aimed to alleviate these drawbacks and improve the performance of the original Jaya algorithm. Here, three new mutation strategies were implemented in the original Jaya to improve both its global and local search abilities. Chaotic maps were proved to be able to boost the search capabilities of meta-heuristic algorithms. Therefore, after demonstrating its chaotic behavior through the sensitivity to initial conditions, topological transitivity and the density of periodic points, we proposed a new 2D cross chaotic map. The chaotic sequences provided by the proposed chaotic map were embedded into the original Jaya algorithm to generate the initial population and control the search equations. It is worth mentioning that the modifications incorporated in the original algorithm did not affect its two essential characteristics, i.e.,

simplicity and nonrequirement of additional control parameters. As case studies, sixteen benchmark functions were used to evaluate the performance of the proposed chaotic Jaya algorithm (C-Jaya) regarding solution accuracy and convergence speed. Comparisons with some other meta-heuristic algorithms for low-, middle- and high-dimensional benchmark functions show that the proposed C-Jaya algorithm enhances the performance of original Jaya significantly. Moreover, it offers the fastest global convergence, the highest solution quality and it is the most robust on almost all the test functions among all the algorithms. Nonparametric statistical procedures, i.e., Friedman test, Friedman aligned ranks test and Quade test, conducted to analyze the obtained results, show the superiority of the proposed algorithm.

Keywords Chaos theory · 2D cross chaotic map · Optimization · Jaya algorithm · C-Jaya algorithm

1 Introduction

1.1 Research background

Several problems in various engineering domains can be formulated as optimization problems. Thus, the achievement of optimal solutions requires better optimization algorithms. Traditional optimization algorithms like dynamic programming, linear programming, steepest descent usually fail to reach optimal

A. Farah
CEM Laboratory, National Engineering School of Sfax,
Sfax, Tunisia
e-mail: farah.anouar@gmail.com

A. Belazi (✉)
RISC Laboratory, National Engineering School of Tunis,
University of Tunis El Manar, 1002 Tunis, Tunisia
e-mail: akram.belazi@enit.utm.tn

solutions for large-scale problems particularly with nondifferentiable, epistasis, i.e., correlated parameter, nonconvex and nonlinear objective function. Moreover, previous techniques fail to handle multimodal optimization problems. To overcome the aforementioned problems, heuristic algorithms have emerged as a robust method for finding optimum solutions for a given problem. Several classification criteria have been considered in the literature, such as deterministic, iterative-based, stochastic, population-based. The search process of population-based algorithms is initiated with the generation of random candidate solutions. The initial population is enhanced over time until a termination condition is satisfied. This kind of algorithms can be classified into two main groups: swarm intelligence-based algorithms and evolutionary algorithms (EA). Recently, some inventive techniques have been involved in improving the efficiency of the heuristic methods and the obtained algorithms are called meta-heuristic. The main advantages of these algorithms compared to conventional techniques are simplicity, better local optimum avoidance, gradient-free mechanism and flexibility. Researchers have made enormous efforts in this field, and nature-inspired meta-heuristic optimization algorithms have proved their efficiency in several optimization problems and thereby are extensively used.

1.2 Related works

Over the last three decades, well-known nature-inspired optimization algorithms have been developed: genetic algorithm (GA) [23] based on simulating living beings evolution and the survival of the fittest stated in Darwinian theory; particle swarm optimization (PSO) [32] based on the social behavior of fish schooling or bird flocking; artificial immune algorithm (AIA) [17] which is inspired by the behavior of the human being's immune system; ant colony optimization (ACO) [14] which imitates the ant colonies foraging behavior; biogeography-based optimization (BBO) [54] which simulates the island species migration behavior; shuffled frog Leaping (SFL) [15] which imitates the collaborative behavior of the frogs; artificial bee colony (ABC) [31] which is inspired by the honey bee foraging behavior; gravitational search algorithm (GSA) [51] which works on Newton gravity law; Grenade Explosion Method (GEM) [2] which is inspired by

the explosion of a grenade; and teaching-learning algorithm (TLA) [48,49] which imitates the teaching and learning processes.

All the aforementioned algorithms presented some limitations in their evolution process. The main weakness consists in the fact that the effectiveness of these algorithms is profoundly affected by the fixed control parameters [49]. In other words, the excellent selection of the parameters is crucial for the evolution process toward the optimum solution. For example, genetic algorithm provides near-optimal solutions owing to the difficulty to adjust the adequate controlling settings, such as mutation rate and crossover rate. The same applies to PSO, which uses cognitive and social parameters and inertia weight [28]. Similar to these two algorithms, ABC needs some control parameters, for instance, number of bees, limit. Similarly, BBO requires the probability of the habitat modification, mutation probability, habitat elitism parameter and population size. The design of an optimization algorithm that does not require control parameters has challenged researchers. This property was taken into account in this research.

Recently, there has been a surge of interest in the use of hybridization which is the combination of optimization algorithms to improve their quality [6–8,21,64]. The performance of the obtained algorithm is almost always superior to the original. Chaos is among the newest techniques applied in various engineering fields. Chaos theory concerns the study of chaotic dynamical systems which can be defined as nonlinear dynamical systems characterized by a high sensitivity to their initial conditions [38,42]. In other words, the outcome of the system is substantially affected by small changes in the initial conditions. Moreover, it can be recognized as pseudo-random behavior produced by a deterministic nonlinear system. Due to dynamic characteristics and ergodicity, chaos search emerges as a powerful technique for hybridization. The incorporation of chaos in a meta-heuristic algorithm can be divided into three classes; in the first class, the chaotic sequences generated by chaotic maps are used to substitute random numbers. In the second, local search approaches are implemented via chaotic map function, whereas the control parameters of algorithms are generated chaotically in the third one [30].

Being fascinated by the potential of chaos, many researchers have demonstrated that the introduction of chaos in optimization algorithms contributes to

improving their original version, such as chaotic differential evolution [27,43], chaotic genetic algorithms [37,61], chaotic simulated annealing [24,39], chaotic firefly algorithm [19,22], chaotic-bat swarm optimization [18] and chaotic biogeography-based optimization [52,53,59]. In 2010, Alatas embedded seven chaotic maps into harmony search (HS) algorithms [4]. He proved that chaos could improve their solution quality. In another work, Alatas demonstrated the benefit of introducing chaos in the ABC algorithm [3]. A chaos-improved version of the imperialist competitive algorithm (ICA) was proposed in 2012 by Talatahari et al. [57]. Mirjalili and Gandomi [40] pinpointed the problem of trapping in local optimum in gravitational search algorithm (GSA) and proposed a chaos-enhanced version of GSA. Also, Gao et al. [20] introduced chaos in GSA in two different manners. The first approach can be summarized in substituting random sequences by chaotic sequences generated by the chaotic map, while the second one uses chaos like a local search method. In [29], Jordehi developed three different versions of chaotic-bat swarm optimization (BSO) algorithm using eleven chaotic maps, and the best one was retained as the proposed BSO algorithm. Huang et al. [25] proposed a new chaotic cuckoo search (CCS) optimization algorithm by embedding chaotic approach into cuckoo search (CS) algorithm. In [16], Farah et al. proposed a new chaotic teaching-learning algorithm and applied it to a real-life problem. All the above studies prove the successful applicability of chaos approach in meta-heuristic algorithms.

1.3 Our contribution

The objective of the present work was to establish an efficient chaotic-based C-Jaya algorithm that alleviates premature convergence problem and outperforms the performance of its original version as well as those of the existing well-established meta-heuristic algorithms. Three new search equations were implemented for modifying the search process of the original Jaya algorithm which increases its search capabilities, improves the global convergence abilities and prevents the search process from sticking on a local optimum. Furthermore, a new 2D cross chaotic map was proposed and its characteristics were proved by Devaney definition. This chaotic map was firstly used to generate the initial population and secondly integrated into

the search equations. By doing so, the chaotic search would directly enhance the current solutions obtained by Jaya algorithm, lead to a satisfactory convergence speed and possibly allow a higher probability to escape from local solution.

1.4 Structure of the paper

The remainder of this paper is organized as follows. The proposed 2D cross chaotic map and its impact on Jaya algorithm are given in Sect. 2. Section 3 presents a brief description of the Jaya algorithm. Section 4 discusses the integration of the proposed chaotic map into the original Jaya algorithm to develop the proposed C-Jaya algorithm. Section 5 is devoted to testing the proposed approach through benchmark problems by comparing the results achieved by the proposed algorithm with those obtained via well-known meta-heuristic algorithms. Finally, a summary of our paper based on the comparison analysis is presented in Sect. 6.

2 The proposed 2D cross chaotic map and its impact on Jaya algorithm

2.1 The new 2D cross chaotic map

The proposed 2D cross chaotic map is defined as follows:

$$\begin{cases} x_{i+1} = \cos(k \arccos(y_i)) \\ y_{i+1} = 16x_i^5 - 20x_i^3 + 5x_i \end{cases} \quad (1)$$

where $x_i, y_i \in [-1, 1]$, $k > 1$ and (x_0, y_0) are the initial conditions. However, Chebyshev map $x_{i+1} = \cos(k \arccos(y_i))$ has been proven chaotic in [33]. The chaos proof of $G(x) = 16x^5 - 20x^3 + 5x$ based on Devaney definition [12] is given in ‘‘Appendix A.’’

2.2 The impact of the 2D cross map on Jaya algorithm

Before explaining the role of the new 2D cross map in the improvement of Jaya algorithm, two concepts must be defined: exploitation and exploration. The former consists in focusing more on a thoroughly narrow—but promising—area of the search space to improve the initial solution. This process allows to refine the solution

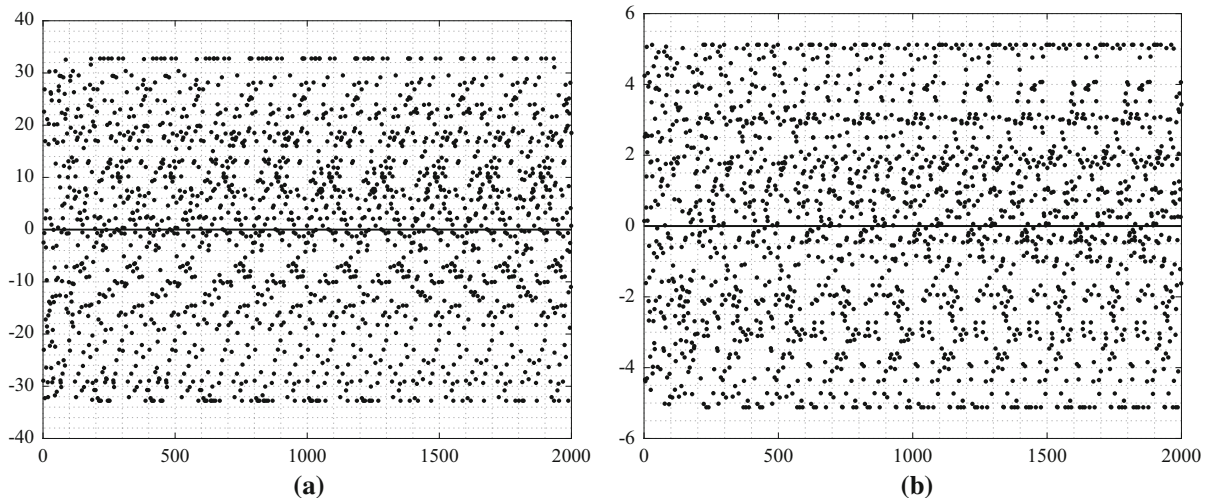


Fig. 1 The evaluated solutions by 2000 FEAs of Jaya algorithm when solving f_6 and f_7 problems **a** f_6 Ackley, **b** f_7 Rastrigin

and avoid big jumps in the search space. Therefore, exploitation is linked to local search as it improves the present solution by searching in its vicinity with a tiny perturbation. The latter deals with covering the whole search space to find other good solutions that can be enhanced. Indeed, it is linked with global search, i.e., search in diverse regions of the whole space, to gain new promising solutions and avoid being trapped in local optimums. A good optimization algorithm should strike a balance between exploitation and exploration [9, 13, 56, 58]. In fact, escaping local optimums requires a high level of diversification at the beginning of the algorithm and a lower one at its end. At the same time, the improvement of the current solution(s) requires a high level of intensification (refining) at the end of the algorithm and a lower one in its beginning. In short, the algorithm should favor global search in the beginning and local search at the end.

Recently, chaos has been extensively studied in the optimization field due to its dynamic properties, which support the algorithm to overcome local optimums [62, 63]. These properties include sensitivity to initial conditions, quasi-stochastic property and ergodicity [34]. They produce a high level of diversification in the algorithm that helps to explore the search space properly and then escape from any potential local optimum.

Moreover, the main idea of Jaya defined by Eq. (2) is as follows: A promising solution derived from a specific problem should approach the best solution and escape the worst one concurrently. This fact intensifies the local search and accelerates the convergence

rate of the algorithm. So, Jaya favors exploitation over exploration, which affects the population diversity and the capability of the algorithm to avoid local optimums. The evaluated solutions of 2000 function evaluations (FEs), of f_6 and f_7 10-dimensional functions (see Table 1), using the standard Jaya algorithm, in the scenario of minimization problems are shown in Fig. 1. It is clear that the algorithm was unable to escape local minima and then fails to converge to the best solution. The former effect transposes the main idea of Jaya (Eq. (2)), which favors exploitation rather than exploration. Moreover, the convergence visualizations of f_6 and f_7 10-dimensional functions are illustrated in Fig. 2 for the first five generations. One can note the weakness of Jaya algorithm to reach the best solution (here, the global minima, i.e., the (0, 0) pair.) Therefore, a balance between exploitation and exploration should be established. In this context, an improvement of Jaya algorithm based on the new 2D cross chaotic map was proposed. In fact, to maintain in equilibrium the exploration and exploitation capabilities of the search process, three mutation equations (Eqs. (3)–(5)) were used randomly for each solution in the improved algorithm. The first mutation (Eq. (3)) serves to enhance the population diversity as well as the global search capability. The individuals were guided by the best solution with the hope of finding other promising areas in the search space. Therefore, the algorithm becomes suitable for problems characterized by many local optimums. The second mutation (Eq. (4)) allows further increase in population diversity and an improvement in the current

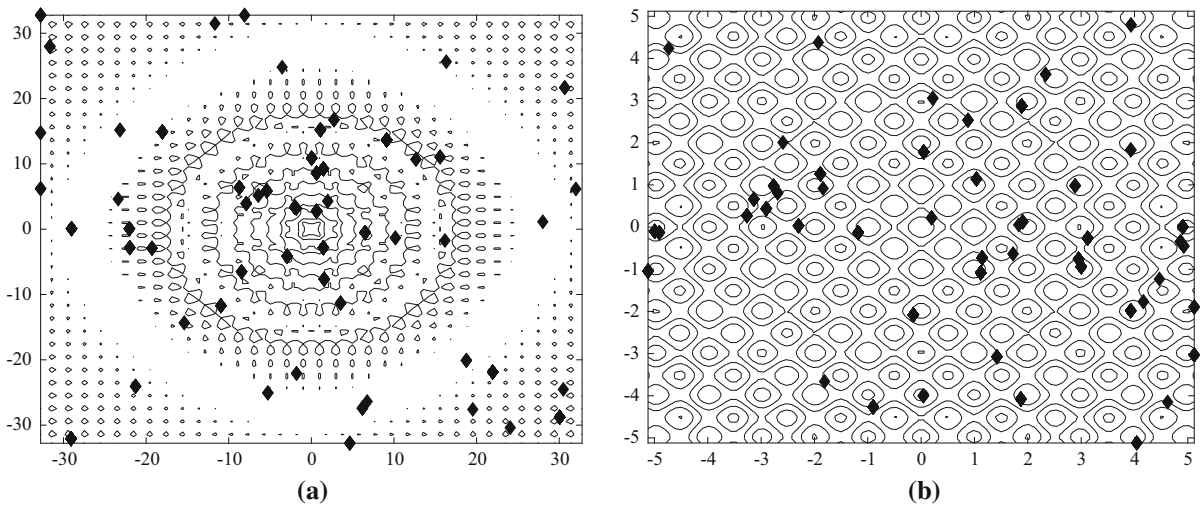


Fig. 2 Convergence visualization of solutions (for the first five generations) when solving f_6 and f_7 problems using Jaya algorithm **a** f_6 Ackley, **b** f_7 Rastrigin

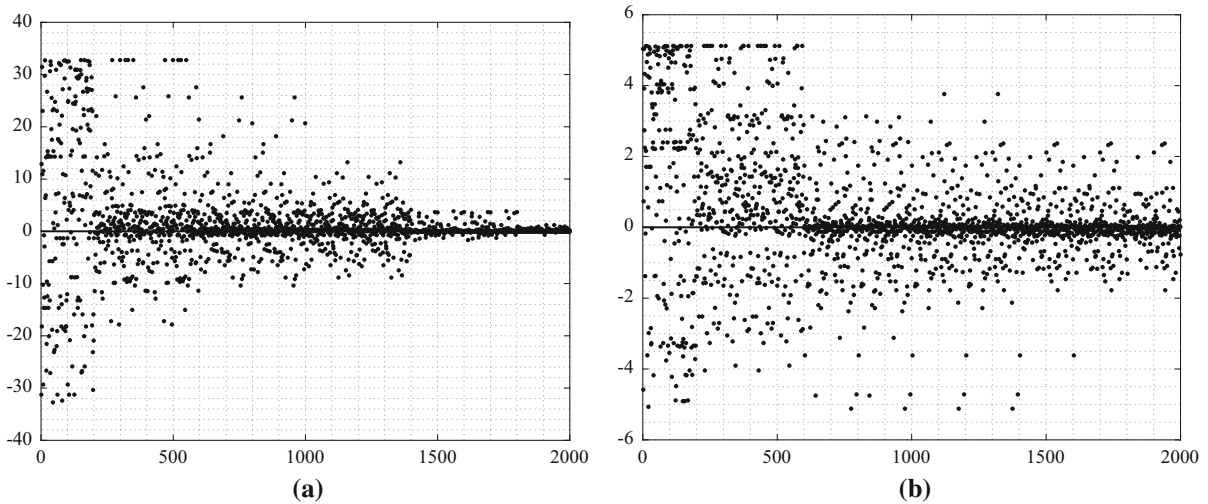


Fig. 3 The evaluated solutions by 2000 FE of C-Jaya algorithm when solving f_6 and f_7 problems **a** f_6 Ackley, **b** f_7 Rastrigin

solution by avoiding the worst one. The third mutation (Eq. (5)) favors local search around the best solution that implies a fast convergence speed. Additionally, each mutation was supported by the use of other solutions to improve the population diversity and then the exploration ability of the algorithm. The chaotic values derived from the 2D cross map have a dual task. First, a balance between the local and global search was achieved by using them to substitute the random mutation strategy. Second, the chaotic mutation further improves the capability to avoid being trapped in local optimums. The evaluated solutions of 2000 FE, of

f_6 and f_7 10-dimensional functions, using the chaotic Jaya (C-Jaya), in minimizing problems are shown in Fig. 3. It is observed that C-Jaya can escape local minima and achieve a satisfactory success level in leading to the best solution. In addition, the convergence visualizations of f_6 and f_7 10-dimensional functions are illustrated in Fig. 4 for the first five generations. It is clear that C-Jaya reaches the best solution. Thus, it presents an appropriate balance between local and global search. All the above results ensure that the 2D cross map plays a crucial role in enhancing the standard Jaya algorithm.

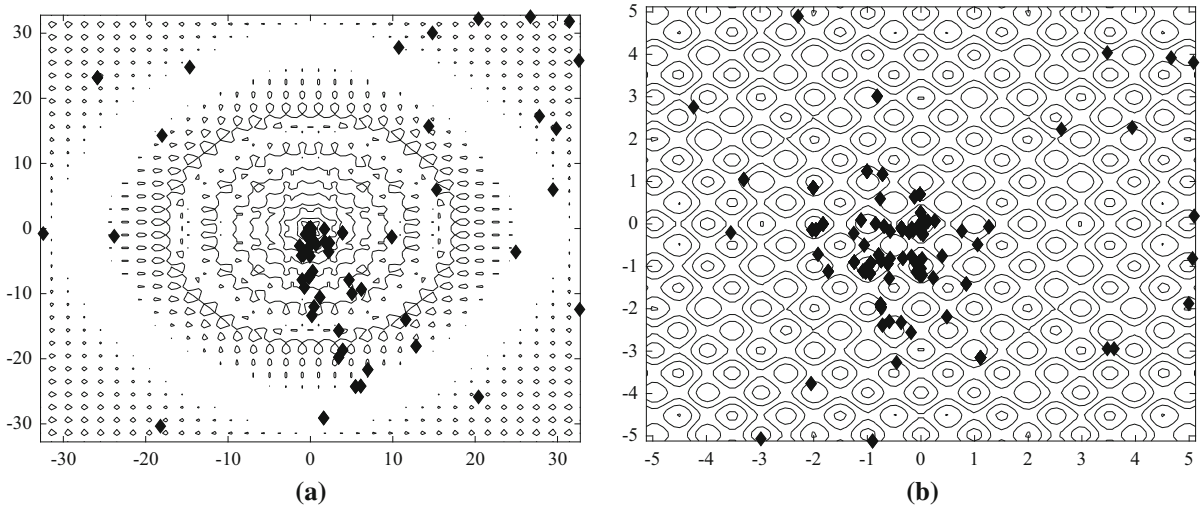


Fig. 4 Convergence visualization of solutions (for the first five generations) when solving f_6 and f_7 problems using C-Jaya algorithm **a** f_6 Ackley, **b** f_7 Rastrigin

3 Traditional Jaya algorithm

The Jaya algorithm was first introduced by Rao [44]. It was designed for handling both unconstrained and constrained optimization problems. The basic idea behind this algorithm is that the obtained solution for the optimization problem should avoid the worst solution and go toward the best one [46, 50]. Compared to other algorithms, Jaya algorithm does not require any additional control parameters except the common control parameters, i.e., the number of population and the number of function evaluations [1]. The details of Jaya algorithm performance are reported in [47].

Let $f(X)$ be the function to be optimized (i.e., objective function). The dimension of decision vector is m , and the population size is k . Let $f(X_{best})$ be the best value of the objective function produced by the best candidate. In the same manner, the worst candidate can be defined as the decision vector which produces the worst objective value (i.e., $f(X_{worst})$). The solution is updated according to the difference between the best candidate and the existing solution as well as the worst candidate and the existing solution as follows [45, 55, 60]:

$$X_{new,i,j} = X_{i,j} + r_{1i,j} (X_{best_j} - |X_{i,j}|) - r_{2i,j} (X_{worst_j} - |X_{i,j}|) \quad (2)$$

where X_{best_j} and X_{worst_j} are the values of the j th element of, respectively, the best and the worst can-

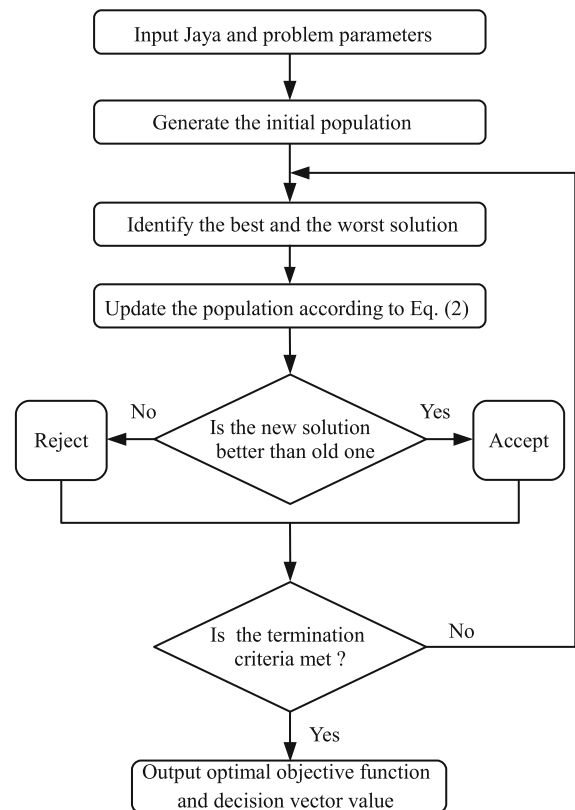
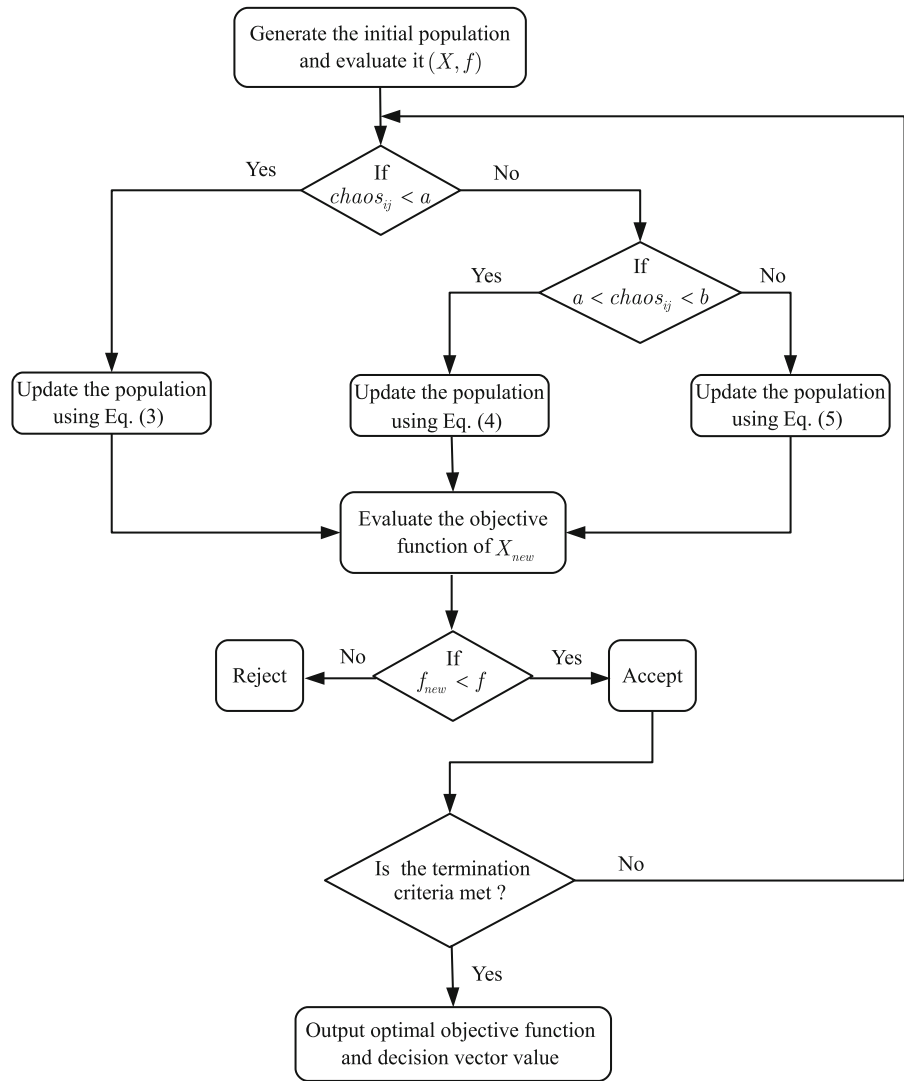


Fig. 5 Flowchart of the original Jaya algorithm

didate, $X_{new,i,j}$ is the update value of $X_{i,j}$, $r_{1i,j}$ and $r_{2i,j}$ are two different random numbers in the range

Fig. 6 Flowchart of the proposed chaotic Jaya algorithm



[0, 1]. The term $r_{1i,j} (X_{bestj} - |X_{i,j}|)$ reveals the tendency of the solution to move toward the best solution, whereas the term $-r_{2i,j} (X_{worstj} - |X_{i,j}|)$ reveals the tendency of the solution to escape from the worst solution. The acceptance criterion of the solutions is the improvement of the objective function. All the accepted function values at the end of the iteration are transferred to the next iteration. The name of the Jaya algorithm (i.e., victory) comes from the fact that this algorithm can achieve victories by attaining the best solution. The flowchart of the traditional Jaya algorithm is given in Fig. 5.

4 Chaotic Jaya algorithm (C-Jaya)

The population of the original Jaya algorithm suffers from the lack of diversity and premature convergence which may occur when the objective function converges to a local optimum. Therefore, to surmount the drawbacks of the original Jaya algorithm, the diversity of the population must be increased.

In addition, an efficient optimization algorithm needs a balance between exploitation and exploration [5,26,35,41]. The former refers to the ability of a population to converge as fast as possible to optimal solutions, whereas the latter can be defined as the ability of the algorithm to explore different regions in a search

Table 1 Details of test functions used

Name	Formula	Range	Acceptance
Sphere	$f_1(x) = \sum_{i=1}^D x_i^2$	$[-100, 100]$	1E-2
Quadric	$f_2(x) = \sum_{i=1}^D (\sum_{j=1}^i x_j)^2$	$[-100, 100]$	1E-5
Sum Squares	$f_3(x) = \sum_{i=1}^D (ix_i)^2$	$[-100, 100]$	1E-5
Zakharov	$f_4(x) = \sum_{i=1}^D x_i^2 + (\sum_{i=1}^D 0.5x_i)^2 + (\sum_{i=1}^n 0.5x_i)^4$	$[-10, 10]$	1E-5
Rosenbrock	$f_5(x) = \sum_{i=1}^{D-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$	$[-2.048, 2.048]$	50
Ackley	$f_6(x) = -20 \exp(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}) - \exp(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)) + 20 + e$	$[-32.768, 32.768]$	1E-5
Rastrigin	$f_7(x) = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10)$	$[-5.12, 5.12]$	1E-5
Weierstrass	$f_8(x) = \sum_{i=1}^D (\sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k (x_i + 0.5))]) - D \sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k \times 0.5)] a = 0.5b = 3k_{max} = 20$	$[-0.5, 0.5]$	1E-5
Griewank	$f_9(x) = \sum_{i=1}^D \frac{x_i^2}{4000} - \prod_{i=1}^D \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]$	1E-5
Rotated Sum Square	$f_{10}(x) = \sum_{i=1}^D iy_i^2 y = M \times x$	$[-100, 100]$	1E-5
Rotated Zakharov	$f_{11}(x) = \sum_{i=1}^D x_i^2 + (\sum_{i=1}^D 0.5iy_i)^2 + (\sum_{i=1}^n 0.5iy_i)^4$ $y = M \times x$	$[-10, 10]$	1E-5
Rotated Rosenbrock	$f_{12}(x) = \sum_{i=1}^{D-1} [100(y_i^2 - y_{i+1})^2 + (y_i - 1)^2] y = M \times x$	$[-2.048, 2.048]$	50
Rotated Ackley	$f_{13}(x) = -20 \exp(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D y_i^2}) - \exp(\frac{1}{D} \sum_{i=1}^D \cos(2\pi y_i)) + 20 + e$ $y = M \times x$	$[-32.768, 32.768]$	1E-5
Rotated Rastrigin	$f_{14}(x) = \sum_{i=1}^D (y_i^2 - 10 \cos(2\pi y_i) + 10) y = M \times x$	$[-5.12, 5.12]$	50
Rotated Weierstrass	$f_{15}(x) = \sum_{i=1}^D (\sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k (y_i + 0.5))]) - D \sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k \times 0.5)] a = 0.5b = 3k_{max} = 20 y = M \times x$	$[-0.5, 0.5]$	1E-5
Rotated Griewank	$f_{16}(x) = \sum_{i=1}^D \frac{y_i^2}{4000} - \prod_{i=1}^D \cos(\frac{y_i}{\sqrt{i}}) + 1 y = M \times x$	$[-600, 600]$	1E-5

space. Excessive exploitation will lead to local search only, whereas excessive exploration will result in a random search. To overcome the search, balance and convergence problems, we proposed three search equations for the original Jaya algorithm. The trial vectors X_{new} were generated using a random solution X_{rand} , and the best and worst solutions (X_{best} and X_{worst}) are, respectively, given as follows:

$$X_{new_{i,j}} = chaos_{i,j} X_{rand_{i,j}} + chaos_{i,j} (X_{i,j} - chaos_{i,j} X_{rand_{i,j}}) + chaos_{i,j} (X_{best_j} - chaos_{i,j} X_{rand_{i,j}}) \tag{3}$$

$$X_{new_{i,j}} = chaos_{i,j} X_{rand_{i,j}} + chaos_{i,j} (X_{i,j} - chaos_{i,j} X_{rand_{i,j}}) + chaos_{i,j} (X_{worst_j} - chaos_{i,j} X_{rand_{i,j}}) \tag{4}$$

$$X_{new_{i,j}} = chaos_{i,j} X_{best_j} + chaos_{i,j} (X_{rand_{i,j}} - S_F X_{best_j}) \tag{5}$$

where $chaos_{i,j}$ is the absolute value of a chaotic variable generated by the 2D cross chaotic map. The scaling factor (SF) can take two values (1 or 2) chaotically. When SF is equal to 1, Eq. (5) allows a search around the best solution, which favors local search ability and then reduces implementation time. However, a premature convergence is usually encountered when multimodal optimization problems are considered. Therefore, even if the local search is applied, global optimum cannot be reached, whereas, when SF equals 2, an important perturbation to current solutions is introduced. Consequently, premature convergence to the local optimum is avoided and the searching behavior

Table 2 Statistical results of 30 runs on 10-dimensional functions obtained by PSO, DE, HS, ABC, TLBO, Jaya and C-Jaya algorithms

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁							
Mean	3.34E−26	8.05E−42	3.39E−01	1.64E−16	8.52E−112	9.65E−27	0.00E+00
SD	1.01E−25	2.34E−41	1.90E−01	6.95E−17	2.22E−111	1.19E−26	0.00E+00
SEM	1.85E−26	4.27E−42	3.46E−02	1.27E−17	4.05E−112	2.17E−27	0.00E+00
<i>f</i> ₂							
Mean	2.99E−07	2.16E−01	1.79E+02	1.46E+02	1.29E−47	1.73E−02	0.00E+00
SD	5.02E−07	3.54E−01	1.42E+02	8.51E+01	5.90E−47	2.51E−02	0.00E+00
SEM	9.17E−08	6.47E−02	2.59E+01	1.55E+01	1.08E−47	4.57E−03	0.00E+00
<i>f</i> ₃							
Mean	7.32E−26	1.70E−41	1.91E+00	1.98E−16	1.87E−111	2.78E−26	0.00E+00
SD	2.34E−25	4.09E−41	1.68E+00	1.01E−16	5.20E−111	2.67E−26	0.00E+00
SEM	4.26E−26	7.46E−42	3.07E−01	1.85E−17	9.50E−112	4.87E−27	0.00E+00
<i>f</i> ₄							
Mean	5.50E−20	1.21E−09	1.54E+00	5.07E−01	3.98E−89	7.21E−13	0.00E+00
SD	1.82E−19	6.59E−09	1.62E+00	1.09E+00	1.61E−88	1.41E−12	0.00E+00
SEM	3.33E−20	1.20E−09	2.96E−01	1.99E−01	2.95E−89	2.57E−13	0.00E+00
<i>f</i> ₅							
Mean	4.31E+00	4.89E+00	4.78E+00	1.99E+00	4.64E+00	1.72E−01	8.81E+00
SD	1.10E+00	1.41E+00	2.93E+00	1.80E+00	6.59E−01	2.75E−01	2.59E−01
SEM	2.01E−01	2.58E−01	5.34E−01	3.29E−01	1.20E−01	5.02E−02	4.73E−02
<i>f</i> ₆							
Mean	8.17E−14	4.44E−15	4.46E−01	6.74E−13	4.44E−15	6.22E−14	8.88E−16
SD	1.71E−13	0.00E+00	2.24E−01	7.57E−13	0.00E+00	6.25E−14	0.00E+00
SEM	3.13E−14	0.00E+00	4.09E−02	1.38E−13	0.00E+00	1.14E−14	0.00E+00
<i>f</i> ₇							
Mean	1.12E+01	9.95E−02	1.55E−01	3.32E−02	3.94E+00	3.14E+01	0.00E+00
SD	6.46E+00	3.04E−01	1.08E−01	1.82E−01	2.25E+00	9.31E+00	0.00E+00
SEM	1.18E+00	5.54E−02	1.96E−02	3.32E−02	4.10E−01	1.70E+00	0.00E+00
<i>f</i> ₈							
Mean	5.00E−02	0.00E+00	4.86E−01	1.54E−15	0.00E+00	5.00E−02	0.00E+00
SD	2.74E−01	0.00E+00	1.38E−01	6.51E−15	0.00E+00	2.74E−01	0.00E+00
SEM	5.00E−02	0.00E+00	2.52E−02	1.19E−15	0.00E+00	5.00E−02	0.00E+00
<i>f</i> ₉							
Mean	8.35E−02	1.55E−03	3.99E−01	1.33E−02	7.53E−03	4.56E−01	0.00E+00
SD	3.46E−02	3.00E−03	1.50E−01	1.21E−02	1.42E−02	1.25E−01	0.00E+00
SEM	6.32E−03	5.47E−04	2.73E−02	2.21E−03	2.59E−03	2.28E−02	0.00E+00
<i>f</i> ₁₀							
Mean	2.50E−09	4.17E−13	4.59E+01	2.73E−02	6.68E−85	4.01E−18	0.00E+00
SD	1.37E−08	1.81E−12	4.83E+01	3.98E−02	3.39E−84	1.26E−17	0.00E+00
SEM	2.50E−09	3.31E−13	8.82E+00	7.27E−03	6.20E−85	2.29E−18	0.00E+00

Table 2 continued

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁₁							
Mean	5.20E−05	1.89E+00	2.18E+00	3.48E+02	2.91E−37	2.65E+02	0.00E+00
SD	2.63E−04	2.98E+00	2.91E+00	1.16E+02	8.16E−37	9.60E+01	0.00E+00
SEM	4.81E−05	5.45E−01	5.32E−01	2.12E+01	1.49E−37	1.75E+01	0.00E+00
<i>f</i> ₁₂							
Mean	8.89E+00	8.89E+00	9.26E+00	8.89E+00	8.89E+00	8.88E+00	8.96E+00
SD	6.08E−02	1.70E−02	4.36E−01	1.53E−02	1.13E−02	1.45E−02	1.68E−02
SEM	1.11E−02	3.10E−03	7.96E−02	2.79E−03	2.06E−03	2.65E−03	3.07E−03
<i>f</i> ₁₃							
Mean	9.69E−14	4.44E−15	1.84E+00	1.05E−08	4.44E−15	7.19E−14	8.88E−16
SD	1.96E−13	0.00E+00	7.66E−01	3.40E−08	0.00E+00	5.90E−14	0.00E+00
SEM	3.58E−14	0.00E+00	1.40E−01	6.20E−09	0.00E+00	1.08E−14	0.00E+00
<i>f</i> ₁₄							
Mean	3.17E+01	3.38E+01	2.98E+01	4.05E+01	2.44E+01	4.05E+01	0.00E+00
SD	7.73E+00	4.91E+00	4.05E+00	5.97E+00	8.25E+00	4.96E+00	0.00E+00
SEM	1.41E+00	8.97E−01	7.39E−01	1.09E+00	1.51E+00	9.06E−01	0.00E+00
<i>f</i> ₁₅							
Mean	8.74E−01	2.45E+00	1.70E+00	1.23E+00	0.00E+00	1.08E−04	0.00E+00
SD	2.58E+00	2.33E+00	7.71E−01	1.10E+00	0.00E+00	1.05E−04	0.00E+00
SEM	4.70E−01	4.26E−01	1.41E−01	2.01E−01	0.00E+00	1.91E−05	0.00E+00
<i>f</i> ₁₆							
Mean	5.94E−01	4.41E−01	6.71E−01	9.50E−02	1.64E−03	2.22E−02	0.00E+00
SD	1.56E−01	1.64E−01	8.14E−02	1.07E−01	6.16E−03	7.54E−02	0.00E+00
SEM	2.84E−02	3.00E−02	1.49E−02	1.95E−02	1.12E−03	1.38E−02	0.00E+00

is improved. The scaling factor is defined as follows: $S_F = \text{round}[1 + \text{chaos}_i]$.

In traditional Jaya algorithm, the candidate solutions are updated by involving the worst and the best solution simultaneously. This procedure can improve the convergence rate and the exploitation capability of the optimization algorithm. However, the prompt convergence rate may degrade the exploration ability and the population diversity of the algorithm. To maintain the balance between the exploitation and the exploration abilities, a new search strategy based on three mutually exclusive equations is introduced. In fact, the equations are selected randomly according to chaotic values $\text{chaos}_{i,j}$ and random numbers a and b at each iteration. These random numbers are generated by a Gaussian distribution. It can be seen that the introduced random numbers allow the choice between the three search equations which modulate the degree of avoiding the worst solution and approaching the best one. As a result, this

modulation enables the increase of convergence speed and the improvement of the current solution. Besides, a chaotic sequence is introduced to boost the quality of the best solution which attracts the candidate solutions to its region.

The first mutation (Eq. (3)) allowed an improvement of solutions for large-scale problems and increased the global search capabilities and the population diversity. The best solution was used as an attractor to guide the individuals to the most promising areas in a feasible search space. Unfortunately, in problems characterized by enormous local optimum, premature convergence may be encountered. The second mutation (Eq. (4)) further increased the population diversity and improved the global solution by avoiding the worst one. The third mutation (Eq. (5)) has a powerful local search around the best solution and provides a fast convergence speed. It is worth mentioning that the mutation strategy was selected randomly.

Table 3 Mean number of function evaluations, time consumed and success rate by comparative algorithms for 10-dimensional functions over 30 independent runs

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁							
MeanFEs	2690	2702	6160	2817	1117.33	4046.67	201.33
Time (s)	9.82E-02	1.19E-01	3.73E-01	3.19E-02	2.88E-02	4.75E-02	4.57E-03
SR (%)	100	100	3.33	100	100	100	100
<i>f</i> ₂							
MeanFEs	15160.67	NaN	NaN	NaN	3458.67	NaN	308.67
Time (s)	5.29E-01	NaN	NaN	NaN	7.99E-02	NaN	7.11E-03
SR (%)	100	0	0	0	100	0	100
<i>f</i> ₃							
MeanFEs	5274.67	4357.33	NaN	7488	1746.67	6467.33	286
Time (s)	1.84E-01	1.79E-01	NaN	7.50E-02	4.22E-02	6.62E-02	6.79E-03
SR (%)	100	100	0	100	100	100	100
<i>f</i> ₄							
MeanFEs	5826	13275.33	19640	NaN	2382.67	12422.67	294
Time (s)	2.05E-01	5.49E-01	1.26E+00	NaN	5.78E-02	1.31E-01	7.29E-03
SR (%)	100	100	3.33	0	100	100	100
<i>f</i> ₅							
MeanFEs	112	891.33	1480.67	925	246.67	1042.67	75.33
Time (s)	3.88E-03	3.46E-02	9.91E-02	1.03E-02	5.84E-03	1.04E-02	1.69E-03
SR (%)	100	100	100	100	100	100	100
<i>f</i> ₆							
MeanFEs	7801.33	5952.67	NaN	10393	2357.33	9194	396
Time (s)	2.88E-01	2.49E-01	NaN	1.32E-01	5.07E-02	9.76E-02	9.65E-03
SR (%)	100	100	0	100	100	100	100
<i>f</i> ₇							
MeanFEs	NaN	7201.48	18900	10128.57	13646.67	NaN	266.67
Time (s)	NaN	3.01E-01	1.22E+00	1.17E-01	2.97E-01	NaN	6.45E-03
SR (%)	0	90	3.33	93.33	10	0	100
<i>f</i> ₈							
MeanFEs	11323.45	7376	NaN	12622	3808	15211.03	457.33
Time (s)	2.63E+00	1.66E+00	NaN	9.63E+00	7.64E-01	5.49E+00	1.67E-01
SR (%)	96.67	100	0	100	100	96.67	100
<i>f</i> ₉							
MeanFEs	NaN	10836.52	NaN	12126.67	9308	NaN	324
Time (s)	NaN	4.70E-01	NaN	1.68E-01	2.30E-01	NaN	7.91E-03
SR (%)	0	76.67	0	30	66.67	0	100
<i>f</i> ₁₀							
MeanFEs	8124	10860	NaN	NaN	2212	8597.33	300.67
Time (s)	8.40E-01	1.18E+00	NaN	NaN	1.98E-01	1.53E-01	9.66E-03
SR (%)	100	100	0	0	100	100	100

Table 3 continued

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁₁							
MeanFEs	13022.22	NaN	NaN	NaN	5828	NaN	286
Time (s)	1.36E+00	NaN	NaN	NaN	4.84E−01	NaN	8.73E−03
SR (%)	90	0	0	0	100	0	100
<i>f</i> ₁₂							
MeanFEs	139.33	1145.33	502.67	1812	276	926.67	73.33
Time (s)	1.42E−02	1.21E−01	6.45E−02	6.65E−02	2.29E−02	1.66E−02	2.25E−03
SR (%)	100	100	100	100	100	100	100
<i>f</i> ₁₃							
MeanFEs	8283.33	7260.67	NaN	15322	2428	9210	350
Time (s)	8.41E−01	8.22E−01	NaN	5.87E−01	2.08E−01	1.63E−01	1.08E−02
SR (%)	100	100	0	100	100	100	100
<i>f</i> ₁₄							
MeanFEs	811.72	3438.67	1531.33	8231.03	878.67	5532.41	99.33
Time (s)	8.09E−02	3.98E−01	1.98E−01	3.18E−01	7.41E−02	9.95E−02	3.09E−03
SR (%)	96.67	100	100	96.67	100	96.67	100
<i>f</i> ₁₅							
MeanFEs	18900	NaN	NaN	NaN	5008	NaN	504
Time (s)	5.53E+00	NaN	NaN	NaN	1.40E+00	NaN	1.92E−01
SR (%)	3.33	0	0	0	100	0	100
<i>f</i> ₁₆							
MeanFEs	9060	NaN	NaN	NaN	5531.30	14805.45	276
Time (s)	9.16E−01	NaN	NaN	NaN	4.73E−01	2.75E−01	8.60E−03
SR (%)	3.33	0	0	0	76.67	73.33	100

4.1 Steps of the proposed C-Jaya

Based on the aforementioned formulation, the steps involved in the proposed C-Jaya algorithm can be encapsulated as follows:

- **Step 1** Initialization.
 - **Step 1.1** Initialize the value of chaos using the proposed 2D cross chaotic map ($x_0 = 0.2, y_0 = 0.3$).
 - **Step 1.2** Set FEsMAX and SN the maximum number of function evaluations and the population size, respectively.
- **Step 2** Generate chaotic sequences.
- **Step 3** Generate SN solutions chaotically, with length D (dimension of problems), to form an initial population as follows:

$$X_{i,j} = XL_i + chaos_{i,j}(XU_i - XL_i)$$

where $1 \leq i \leq SN$ and $1 \leq j \leq D$. XL_i and XU_i are the lowest and highest bounds of decision vector, respectively.

- **Step 4** Evaluate the objective function of the initial population and set FEs= SN, where FEs are the number of function evaluations.
- **Step 5** Update population: Let a and b be two random integers with $a < b$, and let $chaos_{i,j}$ be a chaotic value related to solution in the i th row and j th column. Herein, $1 \leq i \leq SN$ and $1 \leq j \leq D$.
 - **Step 5.1** Identify the worst and the best solutions in the population.
If $chaos_{i,j} < a$
 - **Step 5.2** Calculate the new solution $X_{new_{i,j}}$ using Eq. (3).
If $a < chaos_{i,j} < b$
 - **Step 5.3** Generate the new solution $X_{new_{i,j}}$ using Eq. (4).
If $chaos_{i,j} > b$

Table 4 Statistical results of 30 runs on 30-dimensional functions obtained by PSO, DE, HS, ABC, TLBO, Jaya and C-Jaya algorithms

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
f_1							
Mean	5.04E-13	2.89E-28	7.35E+00	1.57E-13	2.48E-179	2.36E-10	0.00E+00
SD	1.28E-12	1.74E-28	2.61E+00	1.87E-13	0.00E+00	1.25E-10	0.00E+00
SEM	2.34E-13	3.18E-29	4.77E-01	3.41E-14	0.00E+00	2.28E-11	0.00E+00
f_2							
Mean	3.43E+01	1.71E+04	2.98E+03	9.92E+03	2.73E-38	2.74E+04	0.00E+00
SD	1.75E+01	3.03E+03	8.11E+02	2.17E+03	6.58E-38	5.84E+03	0.00E+00
SEM	3.20E+00	5.53E+02	1.48E+02	3.96E+02	1.20E-38	1.07E+03	0.00E+00
f_3							
Mean	6.43E-12	3.04E-27	1.01E+02	5.48E-11	9.19E-178	3.32E-09	0.00E+00
SD	1.60E-11	1.34E-27	3.70E+01	7.99E-11	0.00E+00	2.12E-09	0.00E+00
SEM	2.92E-12	2.44E-28	6.75E+00	1.46E-11	0.00E+00	3.87E-10	0.00E+00
f_4							
Mean	3.81E-04	1.95E+02	1.39E+02	6.79E+03	9.00E-96	1.87E+03	0.00E+00
SD	5.70E-04	6.89E+01	4.31E+01	1.20E+03	3.59E-95	3.55E+02	0.00E+00
SEM	1.04E-04	1.26E+01	7.86E+00	2.19E+02	6.55E-96	6.48E+01	0.00E+00
f_5							
Mean	2.54E+01	2.47E+01	5.46E+01	1.50E+01	2.25E+01	1.72E+01	2.88E+01
SD	1.57E+00	6.05E-01	2.71E+01	5.44E+00	5.73E-01	1.12E+01	2.78E-01
SEM	2.86E-01	1.10E-01	4.94E+00	9.93E-01	1.05E-01	2.05E+00	5.07E-02
f_6							
Mean	1.46E-07	1.59E-14	1.33E+00	5.63E-08	6.10E-15	8.80E-06	8.88E-16
SD	2.63E-07	1.53E-15	2.81E-01	2.58E-08	1.80E-15	5.86E-06	0.00E+00
SEM	4.80E-08	2.79E-16	5.13E-02	4.72E-09	3.29E-16	1.07E-06	0.00E+00
f_7							
Mean	4.15E+01	3.62E+01	2.48E+00	5.92E-03	1.22E+01	2.04E+02	0.00E+00
SD	1.48E+01	4.16E+00	7.80E-01	3.07E-02	6.97E+00	2.29E+01	0.00E+00
SEM	2.70E+00	7.59E-01	1.42E-01	5.61E-03	1.27E+00	4.19E+00	0.00E+00
f_8							
Mean	4.45E-01	0.00E+00	2.47E+00	7.45E-06	0.00E+00	6.43E-01	0.00E+00
SD	8.03E-01	0.00E+00	5.10E-01	4.62E-06	0.00E+00	9.27E-01	0.00E+00
SEM	1.47E-01	0.00E+00	9.31E-02	8.44E-07	0.00E+00	1.69E-01	0.00E+00
f_9							
Mean	1.02E-02	0.00E+00	1.07E+00	2.92E-03	8.25E-10	8.19E-02	0.00E+00
SD	9.77E-03	0.00E+00	1.99E-02	6.45E-03	4.52E-09	1.24E-01	0.00E+00
SEM	1.78E-03	0.00E+00	3.62E-03	1.18E-03	8.25E-10	2.27E-02	0.00E+00
f_{10}							
Mean	1.05E+01	9.01E-03	8.09E+02	4.77E+04	2.98E-153	3.40E-02	0.00E+00
SD	2.66E+01	1.22E-02	3.88E+02	2.23E+04	6.12E-153	2.91E-02	0.00E+00
SEM	4.86E+00	2.23E-03	7.08E+01	4.07E+03	1.12E-153	5.31E-03	0.00E+00

Table 4 continued

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁₁							
Mean	2.21E+02	5.15E+03	4.73E+02	8.25E+03	3.89E−27	9.43E+03	0.00E+00
SD	4.14E+02	7.05E+02	1.39E+02	8.99E+02	1.23E−26	1.12E+03	0.00E+00
SEM	7.57E+01	1.29E+02	2.53E+01	1.64E+02	2.25E−27	2.05E+02	0.00E+00
<i>f</i> ₁₂							
Mean	2.91E+01	2.89E+01	3.02E+01	1.44E+02	2.89E+01	2.90E+01	2.90E+01
SD	1.36E−01	2.88E−02	4.52E−01	8.71E+01	1.58E−02	3.42E−02	1.60E−02
SEM	2.48E−02	5.25E−03	8.26E−02	1.59E+01	2.89E−03	6.25E−03	2.91E−03
<i>f</i> ₁₃							
Mean	5.23E−06	2.45E−13	2.95E+00	1.85E+00	5.39E−15	1.51E−05	8.88E−16
SD	8.89E−06	2.18E−13	3.48E−01	6.22E−01	1.60E−15	6.05E−06	0.00E+00
SEM	1.62E−06	3.97E−14	6.35E−02	1.14E−01	2.92E−16	1.11E−06	0.00E+00
<i>f</i> ₁₄							
Mean	1.69E+02	2.10E+02	1.87E+02	2.69E+02	1.27E+02	2.60E+02	0.00E+00
SD	1.19E+01	1.07E+01	1.07E+01	1.51E+01	5.07E+01	1.49E+01	0.00E+00
SEM	2.18E+00	1.95E+00	1.95E+00	2.75E+00	9.26E+00	2.72E+00	0.00E+00
<i>f</i> ₁₅							
Mean	2.97E+00	3.40E+01	7.02E+00	3.73E+01	0.00E+00	3.62E+01	0.00E+00
SD	3.49E+00	2.34E+00	8.93E−01	1.21E+00	0.00E+00	1.26E+00	0.00E+00
SEM	6.36E−01	4.28E−01	1.63E−01	2.21E−01	0.00E+00	2.30E−01	0.00E+00
<i>f</i> ₁₆							
Mean	4.56E−01	1.18E−03	1.06E+00	1.89E−01	0.00E+00	3.76E−01	0.00E+00
SD	3.63E−01	5.32E−04	2.99E−02	8.87E−02	0.00E+00	7.05E−02	0.00E+00
SEM	6.63E−02	9.72E−05	5.45E−03	1.62E−02	0.00E+00	1.29E−02	0.00E+00

- **Step 5.4** Generate the new solution $Xnew_{i,j}$ using Eq. (5).
- **Step 6** Evaluate the new fitness for the new decision vector $Xnew_i$ and set $FES = FES + 1$.
- **Step 7** Accept $Xnew_i$ if the objective function value is improved.
- **Step 8** If $FES < FESMAX$ go to Step 5, else stop and output the best solution so far.

The flowchart of the proposed C-Jaya is given in Fig. 6.

5 Experiments and comparisons

In this section, the accuracy and the performance of the proposed algorithm were investigated. In fact, it was applied in the optimization of 16 benchmark functions. The obtained results were compared with those of PSO, DE, HS, ABC, TLBO and Jaya algorithms.

In what follows, we will present the characteristics of the benchmark functions and the set of the obtained results.

5.1 Test functions

The effectiveness of C-Jaya for numerical optimization problems was proven by minimizing 16 benchmark functions. Besides, a comparison with traditional Jaya algorithm and other algorithms was performed. These functions belong to three different groups, namely: unimodal (f_1 – f_5), multimodal (f_6 – f_9) and rotated versions of f_3 – f_9 (f_{10} – f_{16}). Table 1 illustrates the specifications of these functions. The global optimum of all benchmark functions is equal to zero. The set of test functions were evaluated in 10, 30 and 100 dimensions.

Table 5 Mean number of function evaluations, time consumed and success rate by comparative algorithms for 30-dimensional functions over 30 independent runs

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁							
MeanFEs	24334.67	16872	NaN	15968	3048	37658.67	492
Time (s)	9.24E-01	1.13E+00	NaN	1.87E-01	8.46E-02	1.22E+00	2.85E-02
SR (%)	100	100	0	100	100	100	100
<i>f</i> ₂							
MeanFEs	NaN	NaN	NaN	NaN	17914.67	NaN	838.67
Time (s)	NaN	NaN	NaN	NaN	4.61E-01	NaN	5.47E+00
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₃							
MeanFEs	44868	27126.67	NaN	45266	4834.67	60441.33	701.33
Time (s)	1.70E+00	1.41E+00	NaN	5.23E-01	1.20E-01	1.84E+00	4.59E-02
SR (%)	100	100	0	100	100	100	100
<i>f</i> ₄							
MeanFEs	NaN	NaN	NaN	NaN	14474.67	NaN	838.67
Time (s)	NaN	NaN	NaN	NaN	3.74E-01	NaN	5.52E-02
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₅							
MeanFEs	1482.67	12194.67	34197.33	16624	1008	17252.41	206.67
Time (s)	5.57E-02	6.08E-01	4.30E+00	2.08E-01	2.40E-02	5.42E-01	1.19E-02
SR (%)	100	100	50	100	100	96.67	100
<i>f</i> ₆							
MeanFEs	59320	33666.67	NaN	62276	6048	77652.17	906.67
Time (s)	2.37E+00	1.80E+00	NaN	1.02E+00	1.37E-01	2.34E+00	5.27E-02
SR (%)	100	100	0	100	100	76.67	100
<i>f</i> ₇							
MeanFEs	NaN	NaN	NaN	63954.28	31826.67	NaN	681.33
Time (s)	NaN	NaN	NaN	1.05E+00	7.30E-01	NaN	3.98E-02
SR (%)	0	0	0	93.33	10	0	100
<i>f</i> ₈							
MeanFEs	73896	42934.67	NaN	77457.14	9394.67	NaN	1156
Time (s)	4.03E+01	2.92E+01	NaN	3.13E+02	1.40E+01	NaN	3.89E+00
SR (%)	50	100	0	70	100	0	100
<i>f</i> ₉							
MeanFEs	41214.55	26852	NaN	50440	5309.33	57488.89	682.67
Time (s)	1.57e+00	1.50E+00	NaN	1.55E+00	1.20E-01	1.91e+00	4.52E-02
SR (%)	36.67	100	0	80	100	30	100
<i>f</i> ₁₀							
MeanFEs	NaN	NaN	NaN	NaN	5592	NaN	744
Time (s)	NaN	NaN	NaN	NaN	2.20E+01	NaN	6.31E-02
SR (%)	0	0	0	0	100	0	100

Table 5 continued

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁₁							
MeanFEs	NaN	NaN	NaN	NaN	38528	NaN	813.33
Time (s)	NaN	NaN	NaN	NaN	5.86E+01	NaN	6.87E−02
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₁₂							
MeanFEs	1901.33	16626.67	11780	65880	1120	19910.67	214.67
Time (s)	6.68E−01	6.75E+00	6.29E+00	8.28E+00	4.22E−01	9.08E−01	1.63E−02
SR (%)	100	100	100	20	100	100	100
<i>f</i> ₁₃							
MeanFEs	70024	42104	NaN	NaN	6034.67	79340	900
Time (s)	2.43E+01	1.95E+01	NaN	NaN	2.02E+00	3.80E+00	6.83E−02
SR (%)	83.33	100	0	0	100	6.67	100
<i>f</i> ₁₄							
MeanFEs	NaN	NaN	NaN	NaN	61280	NaN	292
Time (s)	NaN	NaN	NaN	NaN	2.02E+01	NaN	2.21E−02
SR (%)	0	0	0	0	13.33	0	100
<i>f</i> ₁₅							
MeanFEs	NaN	NaN	NaN	NaN	10698.67	NaN	1238.67
Time (s)	NaN	NaN	NaN	NaN	1.01E+01	NaN	1.93E+00
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₁₆							
MeanFEs	NaN	NaN	NaN	NaN	5936	NaN	720
Time (s)	NaN	NaN	NaN	NaN	1.89E+00	NaN	6.52E−02
SR (%)	0	0	0	0	100	0	100

5.2 Parameter settings

All experiments were carried out using an i7-1.80 GHz CPU, 8-GB RAM and Windows XP with MATLAB 2017a. With the aim of mitigating statistical errors, all experiments were repeated 30 times. For all algorithms, the population size was set to 20 in 10-dimensional problems and 40 for 30-dimensional and 100-dimensional problems. The number of function evaluations (FEs) is considered as a stopping criterion for population-based algorithms. In our experiments, for the 10-dimensional problems, the maximum number of function evaluations was set as 20,000, whereas FEs were set to 80,000 for 30-dimensional and 100-dimensional functions.

PS In Tables 3, 5 and 7 “NaN” denotes that the algorithm cannot produce an acceptable solution during

30 independent runs, whereas boldface in Tables 2, 3, 4, 5, 6 and 7 indicates the best solutions.

5.3 Results for 10-dimensional problems

5.3.1 Comparison of solution accuracy

The performances of the PSO, DE, HS, ABC and C-Jaya algorithms on a large set of benchmark functions for 10D in 30 independent runs are given in Table 2.

The results for the unimodal functions f_1 – f_4 show that C-Jaya outperforms all the comparative algorithms regarding the best mean values, the standard deviations and the standard errors of means and succeeds to reach the global optimum. It is worth mentioning that the unimodal benchmark functions f_1 – f_4 present one global optimum without any local optimum; thus, they are

Table 6 Statistical results of 30 runs on 100-dimensional functions obtained by PSO, DE, HS, ABC, TLBO, Jaya and C-Jaya algorithms

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
f_1							
Mean	8.98E+00	1.43E-03	1.31E+04	2.99E-03	1.14E-162	2.38E+01	0.00E+00
SD	3.62E+00	2.53E-04	1.11E+03	2.76E-03	2.22E-162	8.75E+00	0.00E+00
SEM	6.61E-01	4.62E-05	2.03E+02	5.04E-04	4.06E-163	1.60E+00	0.00E+00
f_2							
Mean	2.17E+04	3.55E+05	3.15E+05	1.68E+05	1.85E-17	4.00E+05	0.00E+00
SD	5.94E+03	2.42E+04	3.40E+04	1.59E+04	3.71E-17	4.36E+04	0.00E+00
SEM	1.08E+03	4.41E+03	6.21E+03	2.90E+03	6.77E-18	7.96E+03	0.00E+00
f_3							
Mean	3.68E+02	5.52E-02	5.22E+05	1.14E-01	5.69E-161	1.41E+03	0.00E+00
SD	1.63E+02	1.33E-02	5.20E+04	8.46E-02	8.77E-161	6.96E+02	0.00E+00
SEM	2.98E+01	2.43E-03	9.50E+03	1.54E-02	1.60E-161	1.27E+02	0.00E+00
f_4							
Mean	2.47E+03	1.20E+05	5.94E+04	1.38E+05	1.28E-25	1.32E+05	0.00E+00
SD	4.22E+02	6.64E+03	5.85E+03	7.83E+03	5.17E-25	1.63E+04	0.00E+00
SEM	7.70E+01	1.21E+03	1.07E+03	1.43E+03	9.43E-26	2.97E+03	0.00E+00
f_5							
Mean	9.78E+01	9.64E+01	1.34E+03	2.48E+02	9.50E+01	1.72E+02	9.88E+01
SD	1.05E+00	5.08E-01	9.55E+01	4.67E+01	1.06E+00	4.65E+01	2.10E-01
SEM	1.92E-01	9.27E-02	1.74E+01	8.52E+00	1.93E-01	8.49E+00	3.84E-02
f_6							
Mean	1.85E+00	5.56E-03	1.22E+01	2.01E+00	7.99E-15	5.38E+00	8.88E-16
SD	3.06E-01	6.54E-04	3.40E-01	2.40E-01	0.00E+00	1.01E+00	0.00E+00
SEM	5.60E-02	1.19E-04	6.20E-02	4.38E-02	0.00E+00	1.85E-01	0.00E+00
f_7							
Mean	1.47E+02	6.06E+02	1.99E+02	5.15E+01	3.17E+00	6.94E+02	0.00E+00
SD	3.08E+01	2.03E+01	1.71E+01	8.72E+00	1.22E+01	1.50E+02	0.00E+00
SEM	5.62E+00	3.71E+00	3.11E+00	1.59E+00	2.24E+00	2.74E+01	0.00E+00
f_8							
Mean	8.64E+00	3.18E-01	3.68E+01	2.28E+00	0.00E+00	3.51E+01	0.00E+00
SD	3.14E+00	2.76E-02	2.17E+00	6.57E-01	0.00E+00	4.83E+00	0.00E+00
SEM	5.73E-01	5.03E-03	3.95E-01	1.20E-01	0.00E+00	8.82E-01	0.00E+00
f_9							
Mean	1.10E+00	9.46E-04	1.23E+02	6.62E-02	0.00E+00	1.26E+00	0.00E+00
SD	3.98E-02	3.47E-04	1.12E+01	5.51E-02	0.00E+00	1.31E-01	0.00E+00
SEM	7.26E-03	6.34E-05	2.05E+00	1.01E-02	0.00E+00	2.40E-02	0.00E+00
f_{10}							
Mean	5.74E+04	8.77E+05	7.81E+05	9.28E+06	7.21E-151	7.01E+04	0.00E+00
SD	1.77E+04	3.09E+05	7.14E+04	6.40E+05	1.12E-150	3.25E+04	0.00E+00
SEM	3.22E+03	5.63E+04	1.30E+04	1.17E+05	2.05E-151	5.94E+03	0.00E+00

Table 6 continued

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁₁							
Mean	2.60E+04	1.32E+05	8.00E+04	1.32E+05	7.81E+02	1.63E+05	0.00E+00
SD	9.39E+03	8.24E+03	6.10E+03	7.51E+03	4.67E+02	1.42E+04	0.00E+00
SEM	1.71E+03	1.50E+03	1.11E+03	1.37E+03	8.53E+01	2.60E+03	0.00E+00
<i>f</i> ₁₂							
Mean	1.15E+02	3.34E+02	8.46E+02	2.47E+04	9.88E+01	1.40E+02	9.89E+01
SD	3.60E+00	1.07E+02	9.12E+01	2.71E+03	2.04E-02	4.24E+01	2.16E-02
SEM	6.58E-01	1.95E+01	1.67E+01	4.96E+02	3.73E-03	7.75E+00	3.94E-03
<i>f</i> ₁₃							
Mean	3.91E+00	2.12E+00	1.24E+01	2.06E+01	7.76E-15	3.06E+00	8.88E-16
SD	2.33E-01	1.89E-01	4.46E-01	1.62E-01	9.01E-16	3.23E-01	0.00E+00
SEM	4.26E-02	3.44E-02	8.14E-02	2.96E-02	1.65E-16	5.89E-02	0.00E+00
<i>f</i> ₁₄							
Mean	8.17E+02	1.14E+03	9.64E+02	1.44E+03	2.95E+00	1.07E+03	0.00E+00
SD	3.02E+01	3.15E+01	2.67E+01	3.97E+01	6.13E+00	2.83E+01	0.00E+00
SEM	5.51E+00	5.74E+00	4.87E+00	7.25E+00	1.12E+00	5.17E+00	0.00E+00
<i>f</i> ₁₅							
Mean	4.94E+01	1.51E+02	8.81E+01	1.53E+02	0.00E+00	1.47E+02	0.00E+00
SD	1.06E+01	1.96E+00	2.92E+00	2.00E+00	0.00E+00	2.57E+00	0.00E+00
SEM	1.94E+00	3.58E-01	5.33E-01	3.65E-01	0.00E+00	4.70E-01	0.00E+00
<i>f</i> ₁₆							
Mean	1.09E+00	8.66E-01	1.22E+02	8.22E-01	0.00E+00	1.24E+00	0.00E+00
SD	3.41E-02	3.39E-02	1.06E+01	1.45E-01	0.00E+00	1.21E-01	0.00E+00
SEM	6.23E-03	6.18E-03	1.94E+00	2.65E-02	0.00E+00	2.21E-02	0.00E+00

obviously adequate for investigating the exploitation of algorithms. Therefore, these results prove that our proposed approach enhances, remarkably, the “exploitation” in Jaya algorithm.

For f_5 , all algorithms failed to attain the global optimum, and the Jaya algorithm produced the best result.

For f_6 , the proposed algorithm obtained the best mean, StdDev and SEM than other ones. However, PSO, DE, ABC and TLBO reached competitive results. Only C-Jaya, DE and ABC algorithms attained good results for f_7 , and in particular, C-Jaya attained the global optimum value.

DE, TLBO and C-Jaya produced better results than other algorithms for f_8 . Each of these algorithms has reached the global optimum value. Thus, DE, ABC and C-Jaya outperform the other algorithms in solving f_8 problem. For f_9 , only C-Jaya reached the global optimum value and the rest of the algorithms performed similarly.

The multimodal benchmark functions (f_6 to f_9), unlike the unimodal functions, have many local optimums. As a result, they are suitable for testing the exploration of a given algorithm. Meanwhile, with the increase of the local minima number of multimodal functions, the problem becomes very difficult to solve. According to the obtained results, C-Jaya significantly provides a good solution accuracy and a higher probability to escape from local optimums as well.

In rotated problems f_{10} , f_{11} , f_{14} and f_{16} only the proposed algorithm attained the global optimum value. Furthermore, TLBO algorithm produced the most competitive results compared to other ones. The Jaya algorithm has the best mean for f_{12} , whereas C-Jaya is the best one for solving f_{13} problem. However, all algorithms, except TLBO and C-Jaya, failed to reach the minimum value of f_{15} . We can conclude that the C-Jaya achieves good results regarding solutions quality for low-dimensional problems.

Table 7 Mean number of function evaluations, time consumed and success rate by comparative algorithms for 100-dimensional functions over 30 independent runs

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁							
MeanFEs	NaN	71777.33	NaN	63898	3597.33	NaN	592
Time (s)	NaN	3.62E+00	NaN	9.05E−01	1.05E−01	NaN	1.12E−01
SR (%)	0	100	0	100	100	0	100
<i>f</i> ₂							
MeanFEs	NaN	NaN	NaN	NaN	35658.67	NaN	1052
Time (s)	NaN	NaN	NaN	NaN	2.04E+00	NaN	2.98E−01
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₃							
MeanFEs	NaN	NaN	NaN	NaN	5816	NaN	884
Time (s)	NaN	NaN	NaN	NaN	1.85E−01	NaN	1.89E−01
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₄							
MeanFEs	NaN	NaN	NaN	NaN	50250.67	NaN	1412
Time (s)	NaN	NaN	NaN	NaN	1.36E+00	NaN	3.05E−01
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₅							
MeanFEs	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Time (s)	NaN	NaN	NaN	NaN	NaN	NaN	NaN
SR (%)	0	0	0	0	0	0	0
<i>f</i> ₆							
MeanFEs	NaN	NaN	NaN	NaN	6592	NaN	981.33
Time (s)	NaN	NaN	NaN	NaN	1.78E−01	NaN	2.12E−01
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₇							
MeanFEs	NaN	NaN	NaN	NaN	7794.28	NaN	774.67
Time (s)	NaN	NaN	NaN	NaN	2.06E−01	NaN	1.53E−01
SR (%)	0	0	0	0	93.33	0	100
<i>f</i> ₈							
MeanFEs	NaN	NaN	NaN	NaN	10133.33	NaN	1342.67
Time (s)	NaN	NaN	NaN	NaN	1.89E+01	NaN	5.34E+00
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₉							
MeanFEs	NaN	NaN	NaN	NaN	4992	NaN	784
Time (s)	NaN	NaN	NaN	NaN	1.45E−01	NaN	1.55E−01
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₁₀							
MeanFEs	NaN	NaN	NaN	NaN	6186.67	NaN	908
Time (s)	NaN	NaN	NaN	NaN	2.29E+01	NaN	3.26E−01
SR (%)	0	0	0	0	100	0	100

Table 7 continued

Function	PSO	DE	HS	ABC	TLBO	Jaya	C-Jaya
<i>f</i> ₁₁							
MeanFEs	NaN	NaN	NaN	NaN	NaN	NaN	1106.67
Time (s)	NaN	NaN	NaN	NaN	NaN	NaN	4.70E−01
SR (%)	0	0	0	0	0	0	100
<i>f</i> ₁₂							
MeanFEs	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Time (s)	NaN	NaN	NaN	NaN	NaN	NaN	NaN
SR (%)	0	0	0	0	0	0	0
<i>f</i> ₁₃							
MeanFEs	NaN	NaN	NaN	NaN	6653.33	NaN	978.67
Time (s)	NaN	NaN	NaN	NaN	2.44E+01	NaN	4.66E−01
SR(%)	0	0	0	0	100	0	100
<i>f</i> ₁₄							
MeanFEs	NaN	NaN	NaN	NaN	39501.33	NaN	337.33
Time(s)	NaN	NaN	NaN	NaN	1.45E+02	NaN	1.57E−01
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₁₅							
MeanFEs	NaN	NaN	NaN	NaN	10613.33	NaN	1418.67
Time (s)	NaN	NaN	NaN	NaN	6.00E+01	NaN	6.08E+00
SR (%)	0	0	0	0	100	0	100
<i>f</i> ₁₆							
MeanFEs	NaN	NaN	NaN	NaN	6568	NaN	777.33
Time (s)	NaN	NaN	NaN	NaN	2.57E+01	NaN	3.65E−01
SR (%)	0	0	0	0	100	0	100

5.3.2 Comparison of convergence

The speed of attaining global optimum is an important criterion in judging the performance of a given optimization algorithm. The effectiveness of the proposed algorithm was demonstrated via the computational effort expressed by the mean number of function evaluations (FEs), the CPU time and the success rate (SR) provided in Table 3. The proposed algorithm presents the highest success rate and the fewest number of FEs required to attain the global optimum. To further evaluate the exploitation property of C-Jaya algorithm compared to other ones, the convergence graphs of all used functions are illustrated in Figs. 7, 8, 9 and 10. It is worth mentioning that we shifted the values of the objective function by a gap of 10^{-3} , due to the existence of some zero values. Thus, we can plot them in semi-log scale. These figures show that the C-Jaya algorithm presents not only a good overall performance regard-

ing the quality of the obtained optimal solution but also faster convergence rates in most test problems. It is also clear that the C-Jaya algorithm outperforms other algorithms concerning the ultimate result similarly. As far as the convergence characteristic is concerned, the proposed algorithm converges very fast in most of the test functions compared to the other algorithms. Overall, the performance of C-Jaya is significantly superior to the other comparative algorithms, except for Rosenbrock function in which the ABC algorithm is the most effective.

To further validate the convergence performance of the proposed algorithm, a comparison of the computational time spent by all used algorithms to reach the first acceptable solution was made (Table 3). These CPU times allow a fair comparison of speed convergence among all algorithms. It is clear that the C-Jaya requires less CPU time to reach the first acceptable solution than other algorithms for all test functions.

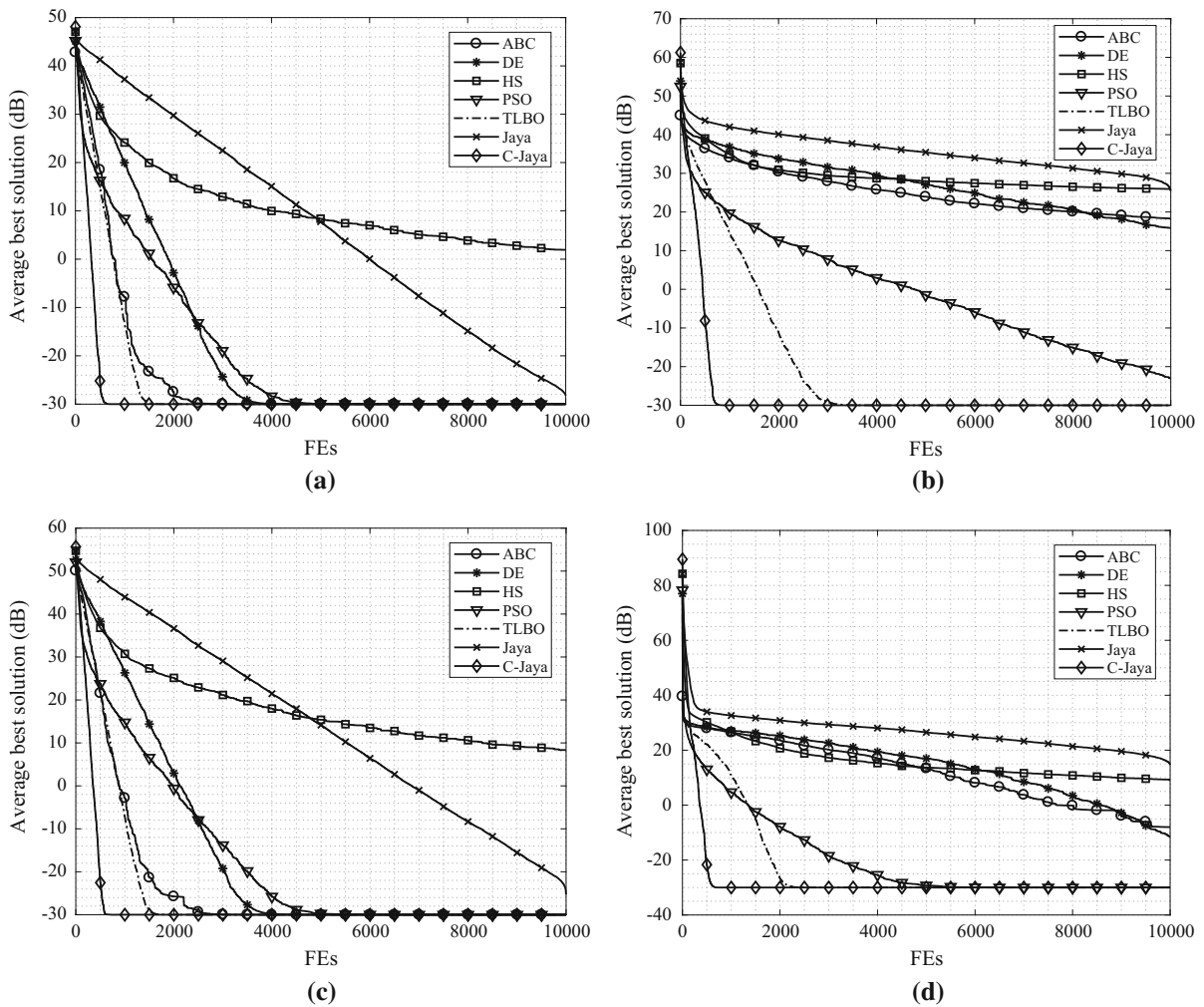


Fig. 7 Convergence performance of different algorithms on the $f_1 - f_4$ 10-dimensional functions **a** f_1 sphere, **b** f_2 quadric, **c** f_3 sum square, **d** f_4 Zakharov

Despite the fact that the proposed algorithm uses additional sequences compared with the original algorithm, C-Jaya ranks first among all the algorithms. In fact, it requires a minimum number of FEs to attain an acceptable solution.

In conclusion, C-Jaya is a powerful algorithm for solving complex optimization problems.

5.3.3 Statistical tests

We applied the Friedman test which is a well-known procedure for testing the differences between more than two algorithms [10, 11, 36]. To achieve this goal, we also used its two advanced versions which are the Fried-

man aligned ranks test and the Quade test. Figures 11 and 12 illustrate the ranking of algorithms used in the experiments based on the standard errors of means. As can be seen from these figures, C-Jaya obtained the best ranking, followed by TLBO algorithm. We can conclude that the proposed algorithm evidently outperforms the comparative algorithms.

5.4 Results for 30-dimensional problems

5.4.1 Comparison of solution accuracy

Table 4 indicates that C-Jaya reaches the global optimum for the unimodal functions $f_1 - f_4$ and the TLBO

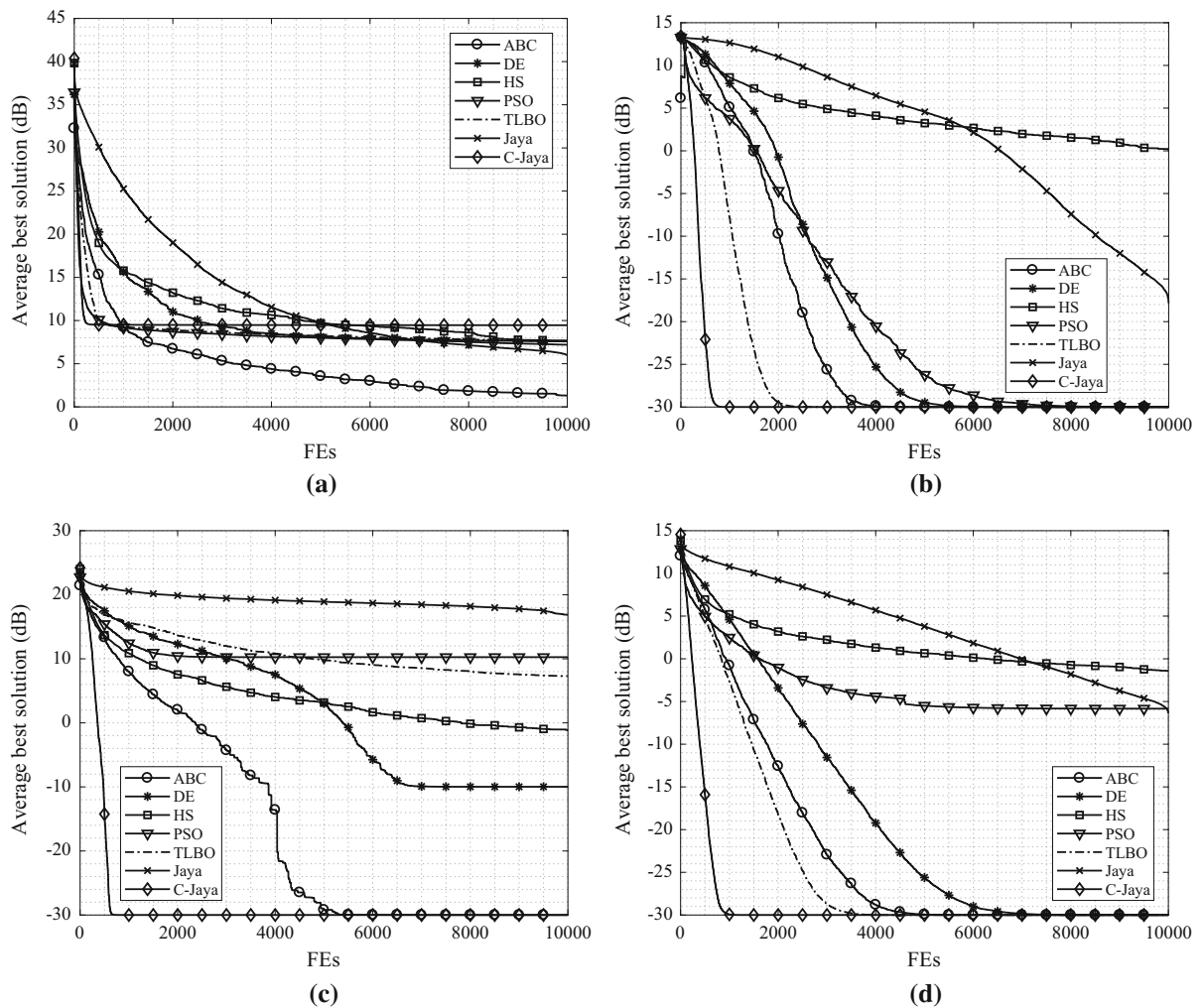


Fig. 8 Convergence performance of different algorithms on the $f_5 - f_8$ 10-dimensional functions **a** f_5 Rosenbrock, **b** f_6 Ackley, **c** f_7 Rastrigin, **d** f_8 Weierstrass

algorithm gives the best results among the other comparative algorithms. All the other algorithms, except HS algorithm, obtained acceptable solutions for the test function f_5 . Nevertheless, all algorithms failed to reach the global optimum for the same function. For the test function f_6 , DE, TLBO and C-Jaya obtained better results than other algorithms. The proposed algorithm gives the same output for the test functions $f_7 - f_{11}$ and $f_{14} - f_{16}$. The obtained solutions for the previous test problems are equal to the theoretical optimum. For the test function f_{12} all algorithms produced similar results, except ABC which does not reach an acceptable solution. Moreover, C-Jaya and TLBO again produce the best solution for f_{13} test function.

It should be noted that the modifications incorporated in the original Jaya algorithm improve its performance regarding solutions accuracy substantially.

5.4.2 Comparison of convergence

Table 5 shows the mean number of FEs, the CPU time and the success rate obtained in 30 independent runs. The success rates were calculated during each run by counting the number of fitness values less than the acceptable level (shown in Table 1), after the completion of 80,000 FEs.

As can be seen from this table, C-Jaya produces the highest success rate compared to other algorithms for

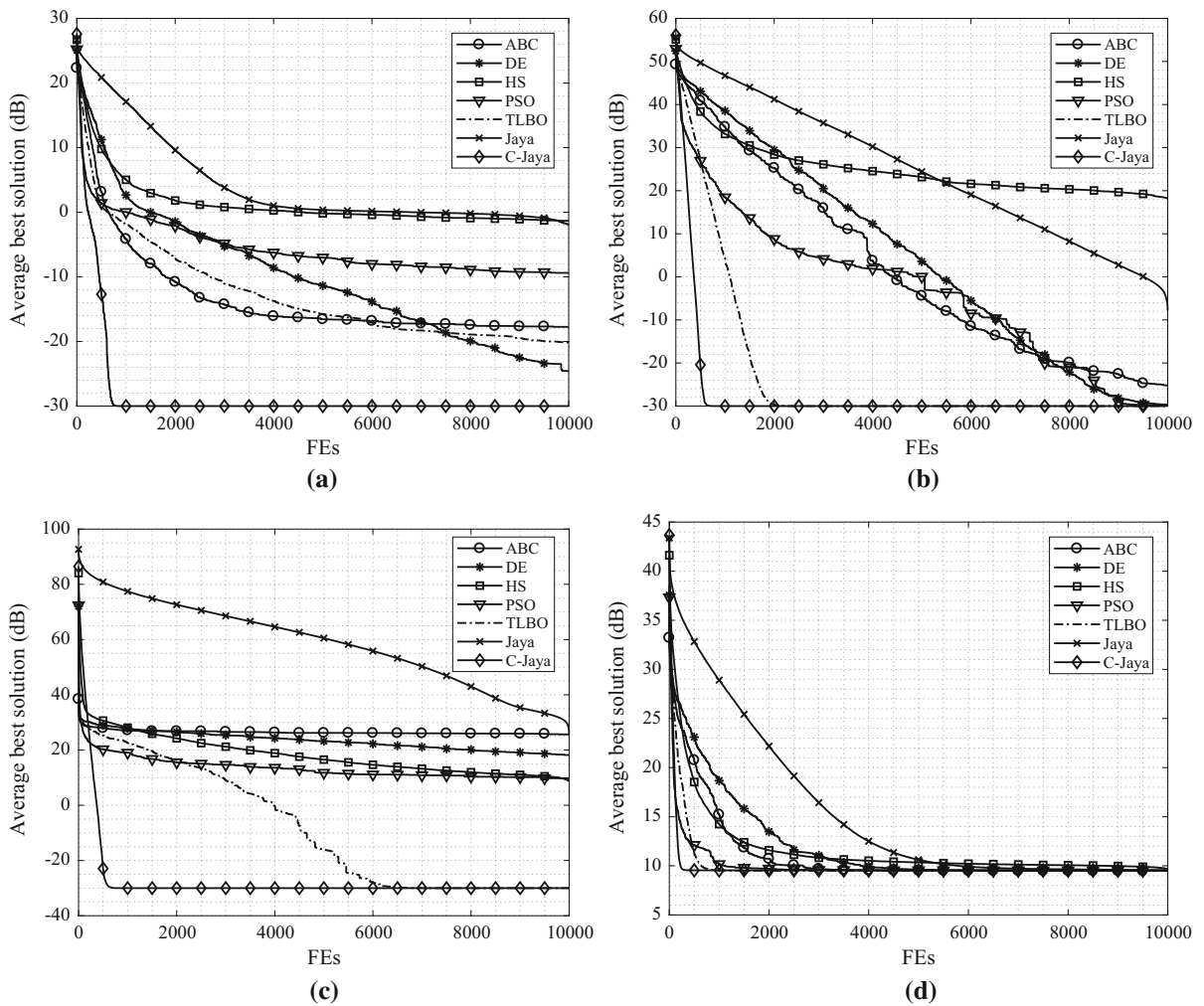


Fig. 9 Convergence performance of different algorithms on the $f_9 - f_{12}$ 10-dimensional functions **a** f_9 Griewank, **b** f_{10} rotated sum square, **c** f_{11} rotated Zakharov, **d** f_{12} rotated Rosenbrock

all test functions. Also, the smallest number of FEs required by C-Jaya to reach an acceptable solution, demonstrates its high speed of convergence in all test problems.

To further assess the convergence performance of the proposed C-Jaya, the CPU required to attain the first acceptable solution was calculated (Table 5). The obtained results for C-Jaya algorithm were compared with other algorithms. We can conclude that C-Jaya requires less CPU time for all test functions except for f_2 function which is ranked second best. In a first analysis, C-Jaya required the generation of chaotic sequences and may need more CPU time than the original one. However, the additional CPU time introduced by the

added operations in C-Jaya was compensated by the improvement of diversity characteristic.

5.4.3 Statistical tests

To prove the efficiency of the proposed algorithm in comparison with the other ones, we performed Friedman, Friedman aligned and Quade tests. Figures 13 and 14 show that C-Jaya is the best performing algorithm, for all tests, among all other algorithms. Thus, it is ranked first among all comparative algorithms. Therefore, C-Jaya can be considered as a very efficient algorithm for overcoming complex optimization problems.

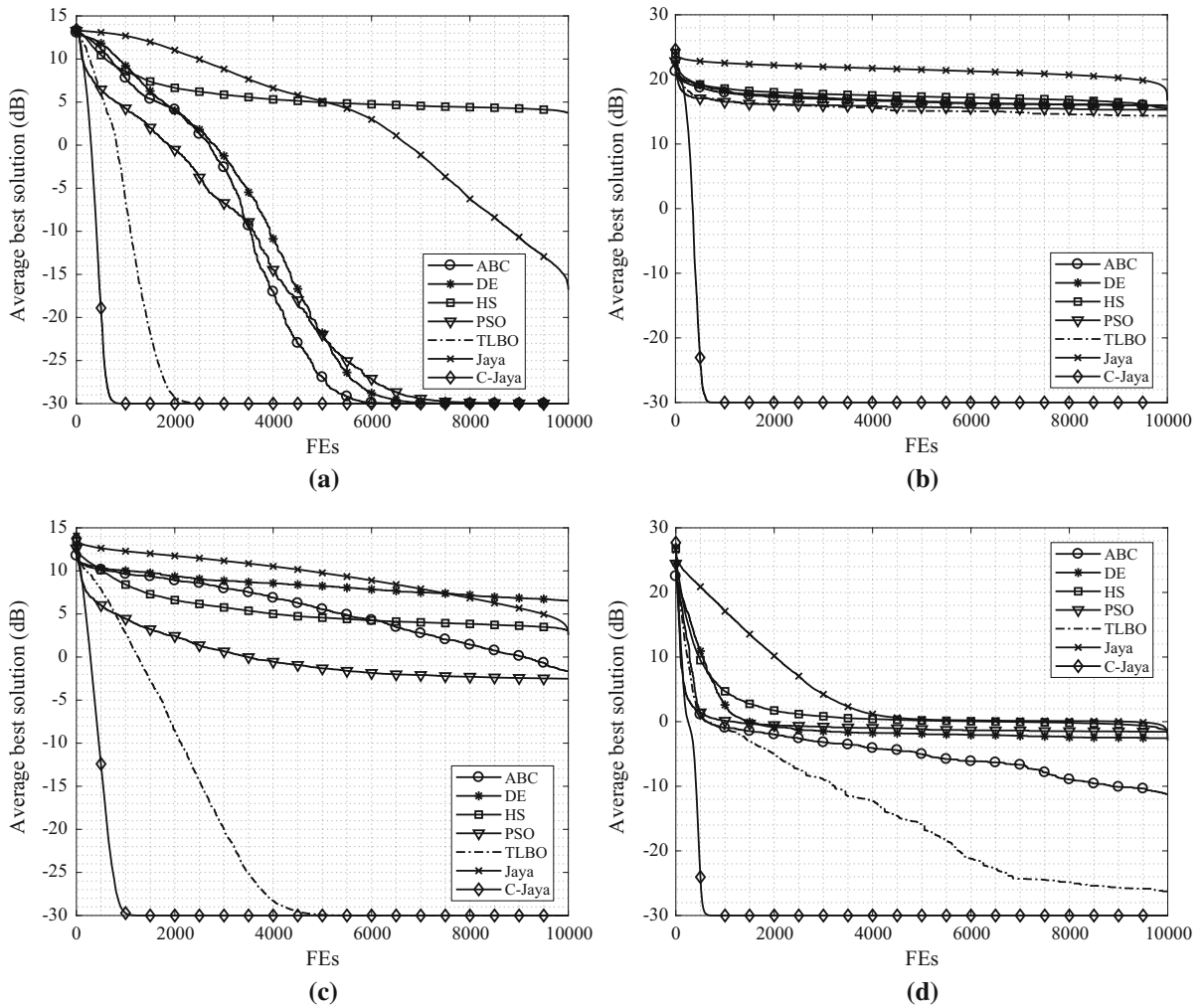


Fig. 10 Convergence performance of different on the $f_9 - f_{16}$ 10-dimensional functions **a** f_{13} rotated Ackley, **b** f_{14} rotated Rastrigin, **c** f_{15} rotated Weierstrass, **d** f_{16} rotated Griewank

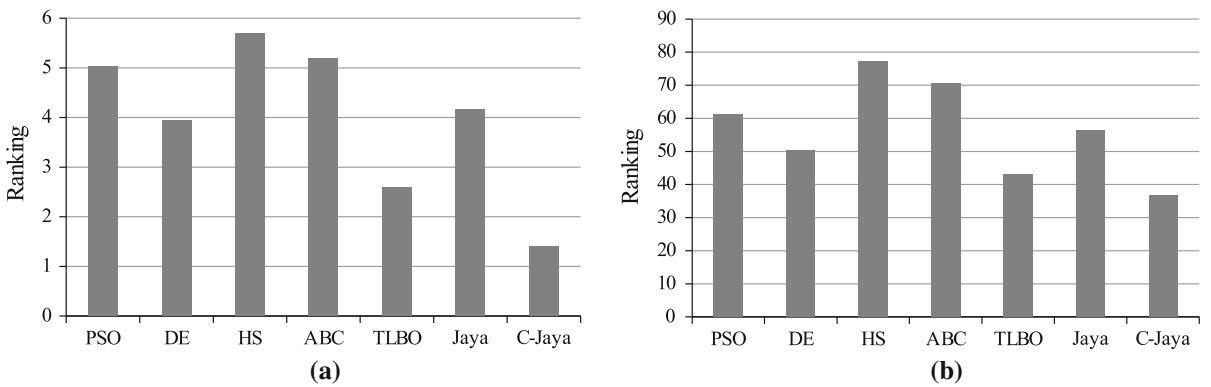


Fig. 11 Average ranking of comparative algorithms by Friedman test (a) and Friedman aligned test (b) for 10-dimensional problems

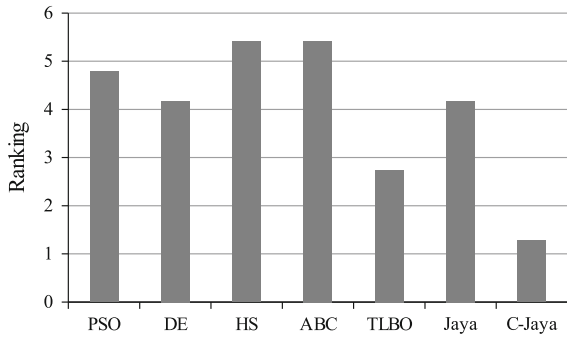


Fig. 12 Average ranking of comparative algorithms by Quade test for 10-dimensional problems

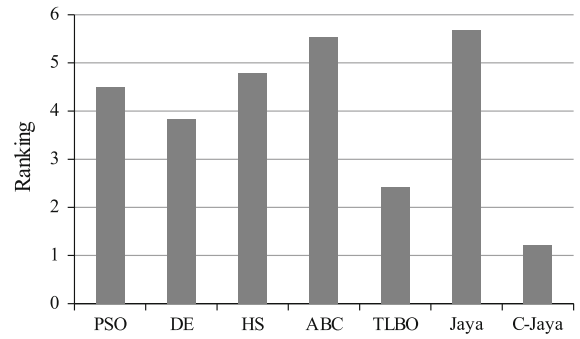


Fig. 14 Average ranking of comparative algorithms by Quade test for 30-dimensional problems

5.5 Results for 100-dimensional problems

5.5.1 Comparison of solution accuracy

The performances of the proposed C-Jaya, in high-dimensional problems, were evaluated by considering the aforementioned benchmark functions in 100 dimensions. The results presented in Table 6 give the means of the best solutions, the standard deviations and the standard errors of means obtained from 30 independent runs.

It can be seen that C-Jaya outperforms the comparative algorithms. We can again conclude that the TLBO produces the most competitive results among all other algorithms. It is worth noting that all comparative algorithms failed to attain the global optimum, except the TLBO which succeeded to reach the theoretical optimum in only four test functions. Thus, the C-Jaya is a very efficient algorithm for solving optimization problems.

5.5.2 Comparison of convergence

In this section, the computational effort and the success rate produced by all algorithms, in solving the considered problems, were studied. Indeed, experiments were conducted to obtain the FEs required by each algorithm to reach the optimal solution. The algorithms are terminated, if the global optimum is attained or if the FEs is completed. Table 7 shows results of all algorithms regarding the mean number of FEs, the CPU time and the success rate obtained in 30 independent runs.

In all test functions, the C-Jaya algorithm required the fewest number of FEs and as a consequence the least computational effort. Equally, the proposed algorithm reached 100% success rate for fourteen test functions, whereas the best among the other algorithms, i.e., TLBO algorithm, obtained the same result for twelve benchmark functions. Moreover, C-Jaya requires less CPU time for all test functions except for f_1 , f_3 , f_6 and f_9 functions which is ranked second best. It is worth

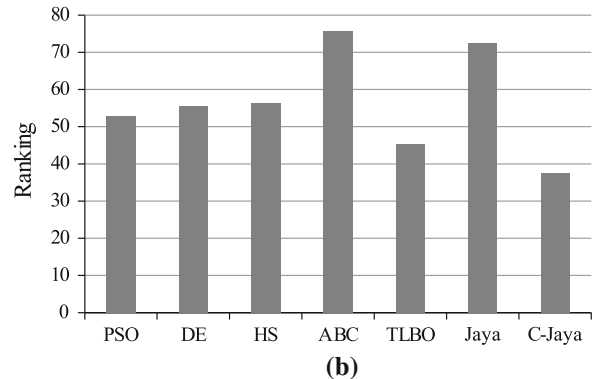
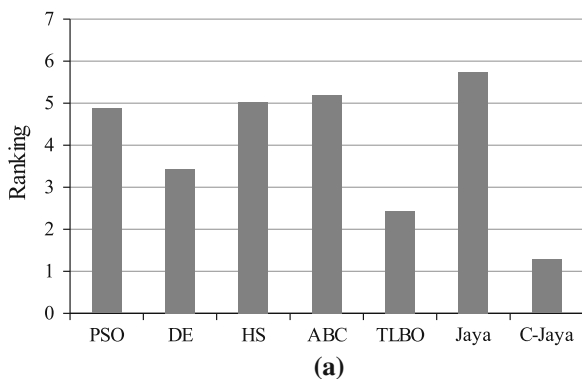


Fig. 13 Average ranking of comparative algorithms by Friedman test (a) and Friedman aligned test (b) for 30-dimensional problems

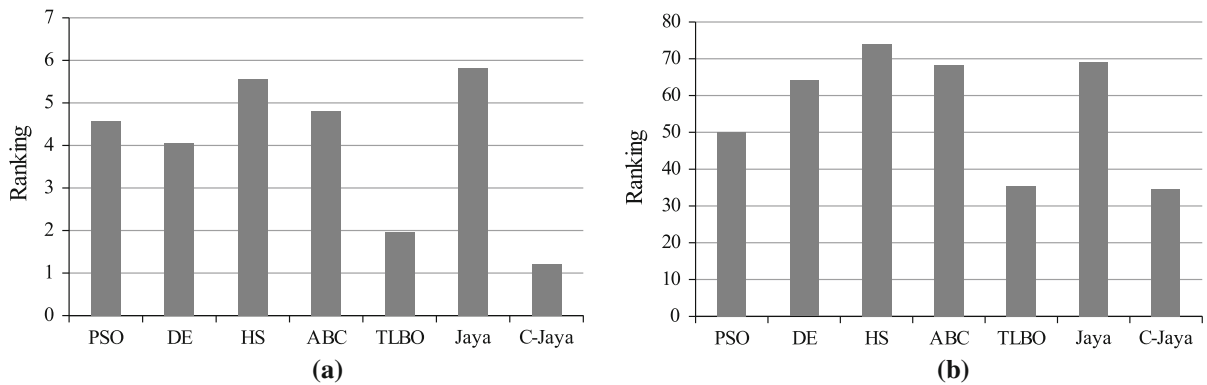


Fig. 15 Average ranking of comparative algorithms by Friedman test (a) and Friedman aligned test (b) for 100-dimensional problems

noting here that the original Jaya algorithm failed in all test problems, which reflects its weakness in global search. Thanks to the modifications introduced in Jaya, the obtained algorithm presents high efficiency and robustness.

To get a deep insight into the convergence performance of the C-Jaya algorithm and to make a fair comparison for high-dimensional problems, CPU times required by the studied algorithms were computed to achieve the first acceptable solution (Table 7). Regarding CPU time, C-Jaya algorithm is the most effective at finding the first acceptable solution for 14 from 16 benchmark functions. Moreover, TLBO algorithm is the second most effective, in attaining the first acceptable solution for 13 from 16 benchmark functions. Besides, the proposed algorithm is the best performing regarding CPU time in almost all test functions. Furthermore, it is observed from results that PSO, HS and Jaya algorithms fail to attain an acceptable solution for all test functions. Moreover, DE and ABC algorithms reach an acceptable solution for only one benchmark function.

5.5.3 Statistical analysis

To statistically quantify the performance of the algorithms used in the experiments, Friedman, Friedman aligned and Quade tests were performed based on standard errors of means. Figures 15 and 16 present the ranking of considered algorithms according to these tests. It is obvious from these histograms that C-Jaya is ranked first among all comparative algorithms. These results confirm the ability of the proposed algorithm for solving optimization problems.

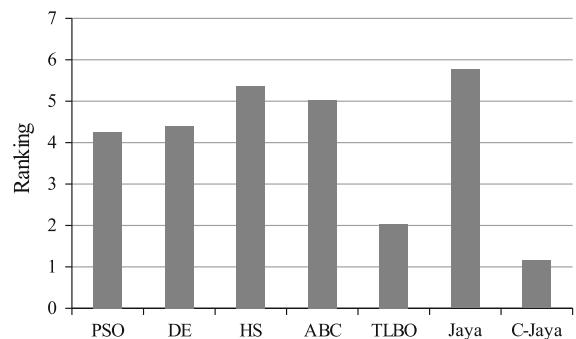


Fig. 16 Average ranking of comparative algorithms by Quade test for 100-dimensional problems

We can, therefore, conclude that the proposed C-Jaya has better-searching ability, robustness and efficiency for handling engineering design problems and that its performance surpasses all comparative algorithms.

6 Conclusion

In this work, we proposed a new chaotic Jaya algorithm that mitigates premature convergence problem of the original Jaya algorithm. Three new search equations were implemented to enhance the search abilities of the Jaya algorithm. Moreover, after investigating its chaotic behavior, the proposed 2D cross chaotic map was embedded in the original Jaya algorithm with the aim of improving both its exploration and exploitation abilities. It is worth mentioning that the modifications incorporated in Jaya algorithm maintained its simplicity and control parameter-free property. Experi-

tal results based on unimodal and multimodal benchmark functions prove that the proposed chaotic Jaya algorithm (C-Jaya) outperforms the original one. Also, comparing the results with other well-known optimization algorithms, i.e., PSO, DE, HS, ABC and TLBO, proves the predominance of the proposed C-Jaya concerning solution accuracy and speed convergence as well as nonparametric statistical tests.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix A: The proof of chaos for $G(x)$

Definition 1 (*Discrete chaos of Devaney*) Consider a discrete dynamical system in the following form:

$$y_{i+1} = g(y_i), \quad g : J \rightarrow J, \quad y_0 \in J \tag{6}$$

$g(x)$ is chaotic if the following conditions are satisfied [12].

(1) Sensitive to initial conditions

$$\exists \Omega > 0, \quad \forall y_0 \in J, \quad \omega > 0, \quad \exists n \in \mathbb{N}, \quad z_0 \in J \\ |y_0 - z_0| < \omega \Rightarrow |g^n(y_0) - g^n(z_0)| > \Omega. \tag{7}$$

(2) Topological transitivity

$$\forall J_1, J_2 \in J, \quad \exists y_0 \in J_1, \quad n \in \mathbb{N}, \quad g^n(y_0) \in J_2. \tag{8}$$

(3) Density of periodic points in J Let $K = \{k \in J | \exists n \in \mathbb{N} : g^n(k) = k\}$ be the set of periodic points of g . Therefore, K is dense in $J : \overline{K} = J$.

Definition 2 Let $f : I \rightarrow I$ and $g : J \rightarrow J$ be maps. f and g are topologically conjugated if there exists a homeomorphism $h : I \rightarrow J$ that makes $h \circ f = g \circ h$.

Theorem Let $f : I \rightarrow I$ and $g : J \rightarrow J$ be conjugate via h . If f is chaotic on I , then g is chaotic on J .

Proof (1) (Sensitive to initial conditions) Let f have sensitivity constant α . Let $I = [\omega_0, \omega_1]$. Assume $\alpha < \omega_1 - \omega_0$. Consider the function $|h(x + \alpha) - h(x)|$ where $x \in [\omega_0, \omega_1 - \alpha]$. This function has minimum value λ as it is continuous and positive. So, h maps intervals of length α to intervals of

length at least λ . We assume that g has sensitivity constant λ . Let $x_0 \in J$ and B be an open interval about x_0 . Therefore, $h^{-1}(B)$ is an open interval about $h^{-1}(x_0)$. By sensitive to initial conditions, there is $y_0 \in h^{-1}(B)$ and $m > 0$ which satisfy $|f^m(h^{-1}(x_0)) - f^m(y_0)| > \alpha$. Then,

$$\left| h \left(f^m \left(h^{-1}(x_0) \right) \right) - h \left(f^m(y_0) \right) \right| \\ = \left| g^m(x_0) - g^m(h(y_0)) \right| > \lambda.$$

(2) (Topological transitivity) Let A and B be open subintervals of J . So, $h^{-1}(A)$ and $h^{-1}(B)$ are open subintervals of I (h is a continuous function). As f is topologically transitive, there is $x_0 \in h^{-1}(A)$ that fulfills $f^n(x_0) \in h^{-1}(B)$ for some n . Therefore, $h(x_0) \in A$ and $g^n(h(x_0)) = h(f^n(x_0)) \in B$.

(3) (Density of periodic points) Let A be an open subinterval of J and consider $h^{-1}(A) \subset I$. Since periodic points of f are dense in I , there is a periodic point $x \in h^{-1}(A)$ of period m . We have $g \circ h = h \circ f$, so $g^m(h(x)) = h(f^m(x)) = h(x)$. Therefore, $h(x)$ is a periodic point of period m in A and periodic points are dense in J .

According to the definition of chaos Devaney [12], g is chaotic on J . □

It is known that $\phi(\theta) = 5\theta$ is chaotic [12] under the unit circle mapping $S^1 \rightarrow S^1$, so ϕ is sensitive to initial value, topologically transitive and dense in S^1 .

Considering $h(\theta) = \cos \theta$, we have $G \circ h = 16 \cos^5 \theta - 20 \cos^3 \theta + 5 \cos \theta = \cos(5\theta) = h \circ \phi$. So G is conjugated to ϕ in $y \in J = [-1, 1]$. Thus, as ϕ is chaotic on S^1 , then G is chaotic on $J = [-1, 1]$.

A.1 The randomness proof of $G(x)$

Because function $G(x) = 16x^5 - 20x^3 + 5x$ is a Chebyshev polynomial of degree 5 ($T_5(x) = \cos(5\theta)$, $x = \cos \theta$), so its invariant density is

$$\rho_G(x) = \frac{1}{\pi \sqrt{1-x^2}}, \quad -1 < x < 1 \tag{9}$$

A.1.1 The auto-correlation proof

It is known that

$$\begin{aligned} \bar{x} &= \int_{-1}^1 x\rho_G(x)dx \\ &\stackrel{x=\sin u}{=} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \sin u \frac{1}{\sqrt{1-\sin^2 u}} \cos u du = 0. \end{aligned}$$

Considering $x = \cos u$ and τ is the iterative times of $G(x)$. We have $G^\tau(x) = \cos(5^\tau u)$. When $\tau \neq 0$,

$$\begin{aligned} C_G(\tau) &= \int_{-1}^1 x\rho_G(x)G^\tau(x)dx - \bar{x}^2 \\ &= \int_{-1}^1 x \frac{1}{\pi\sqrt{1-x^2}} G^\tau(x)dx - \bar{x}^2 = 0. \end{aligned}$$

When $\tau = 0$,

$$\begin{aligned} C_G(0) &= \int_{-1}^1 x\rho_G(x)G^0(x)dx - \bar{x}^2 \\ &= \int_{-1}^1 x^2 \frac{1}{\pi\sqrt{1-x^2}} dx - \bar{x}^2 \\ &\stackrel{x=\cos u}{=} \int_{\pi}^0 \cos^2 u \frac{1}{\pi\sqrt{1-\cos^2 u}} \\ &\quad \times (-\sin u) du \\ &= \frac{1}{\pi} \int_0^\pi \frac{1+\cos(2u)}{2} du = 0.5. \end{aligned}$$

Therefore, auto-correlation function of $G(x)$ is:

$$C_G(\tau) = \begin{cases} 0.5 & \text{if } \tau = 0 \\ 0 & \text{if } \tau \neq 0 \end{cases} \tag{10}$$

The auto-correlation function of $G(x)$ shows that when the iterative times $\tau \neq 0$, the time sequences generated by $G(x)$ are independent.

A.1.2 The cross-correlation proof

Considering $x = \cos u$ and τ is the iterative times of $G(x)$. We have $G^\tau(x) = \cos(5^\tau u)$.

$$\begin{aligned} rr_G(\tau) &= \int_{-1}^1 \int_{-1}^1 x_0 G^\tau(x_1) \rho_{x_0}(x_0) \rho_{x_1}(x_1) \\ &\quad dx_0 dx_1 - \bar{x}^2 \\ &\stackrel{x_0=\cos u_0}{\stackrel{x_1=\cos u_1}{=}} \int_{\pi}^0 \int_{\pi}^0 \cos u_0 \cos(5^\tau u_1) \\ &\quad \times \frac{1}{\pi\sqrt{1-\cos^2 u_0}} \frac{1}{\pi\sqrt{1-\cos^2 u_0}} \end{aligned}$$

$$\begin{aligned} &\quad \times (-\sin u_0)(-\sin u_1) du_0 du_1 \\ &= \frac{1}{\pi^2} \int_0^\pi \int_0^\pi \cos u_0 \cos(5^\tau u_1) du_0 du_1 \\ &= \frac{1}{\pi^2} \int_0^\pi \cos u_0 du_0 \int_0^\pi \cos(5^\tau u_1) du_1 \\ &= \frac{1}{\pi^2} [\sin u_0]_0^\pi \left[\frac{\sin(5^\tau u_1)}{5^\tau} \right]_0^\pi = 0 \end{aligned}$$

Therefore, the cross-correlation function is given by

$$rr_G(\tau) = 0 \tag{11}$$

The result given in Eq. (11) shows that time sequences produced by $G(x)$ with different initial values have no relation to each other at any time. According to the above characteristics, the average value of $G(x)$ and the cross-correlation are 0, so the probability statistical characteristic is the same as the white noise and thus $G(x)$ function can be used as an ideal chaotic sequence generator.

A.2 The proof of the equal probability of the pseudo-random sequence

When $G(x)$ is iterated, a chaotic sequence is produced as follows: $g_0, g_1, \dots, g_p = G(x_{p-1})$ where p is an integer. The chaotic range $V = [-1, 1]$ is divided into M subdomains $v_i, i = 0, 1, 2, \dots, M - 1$. With $v_i = (t_i, t_{i+1})$ for $i = 0, 1, \dots, M - 1$. Here, t_i is defined by Eq. (12)

$$t_i = -\cos\left(\frac{i}{M}\pi\right), \quad i = 0, 1, 2, \dots, M - 1 \tag{12}$$

The initial conditions (x_0, y_0) of the cross chaotic map are used to generate a value of chaotic sequence $\{g_p\}_{p=1}^\infty$.

Definition 3 Mapping $S : V \rightarrow 0, 1, \dots, M - 1, x_p \rightarrow i = s_k, x_p \in v_i, p = 0, 1, \dots$, where s_p is described as Eq. (13), so the N phase pseudo-random sequence $\{s_p\}_{p=0}^\infty$ distributes in the N subdomains proportionally.

$$s_p = \begin{cases} \lfloor (1 - \arccos(g_p)/\pi) \times M \rfloor, & \text{if } g_p \in [-1, 1], \\ M - 1, & \text{if } g_p = 1 \end{cases} \tag{13}$$

Proof According to Eqs. (9) and (13), the probability of element i appearing in $\{s_p\}_{p=0}^{\infty}$ is:

$$Prob(i) = \int_{t_i}^{t_{i+1}} \rho(x) dx = \frac{1}{M}$$

□

Obviously $\{s_p\}_{p=0}^{\infty}$ obeys uniform distribution.

References

- Abhishek, K., Kumar, V.R., Datta, S., Mahapatra, S.S.: Application of JAYA algorithm for the optimization of machining performance characteristics during the turning of CFRP (epoxy) composites: comparison with TLBO, GA, and ICA. *Eng. Comput.* **33**, 1–19 (2016)
- Ahrari, A., Atai, A.: Grenade explosion method—a novel tool for optimization of multimodal functions. *Appl. Soft Comput.* **10**(4), 1132–1140 (2010)
- Alatas, B.: Chaotic bee colony algorithms for global numerical optimization. *Expert Syst. Appl.* **37**(8), 5682–5687 (2010)
- Alatas, B.: Chaotic harmony search algorithms. *Appl. Math. Comput.* **216**(9), 2687–2699 (2010)
- Alba, E., Dorronsoro, B.: The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Trans. Evol. Comput.* **9**(2), 126–142 (2005)
- Ali, M.Z., Awad, N.H., Suganthan, P.N., Duwairi, R.M., Reynolds, R.G.: A novel hybrid Cultural Algorithms framework with trajectory-based search for global numerical optimization. *Inf. Sci.* **334**, 219–249 (2016)
- Awad, N.H., Ali, M.Z., Suganthan, P.N., Reynolds, R.G.: CADE: a hybridization of cultural algorithm and differential evolution for numerical optimization. *Inf. Sci.* **378**, 215–241 (2017)
- Bai, Y.-Y., Xiao, S., Liu, C., Wang, B.-Z.: A hybrid IWO/PSO algorithm for pattern synthesis of conformal phased arrays. *IEEE Trans. Antennas Propag.* **61**(4), 2328–2332 (2013)
- Chen, J., Xin, B., Peng, Z., Dou, L., Zhang, J.: Optimal contraction theorem for exploration-exploitation tradeoff in search and optimization. *IEEE Trans. Syst. Man Cybern. Part A Syst. Hum.* **39**(3), 680–691 (2009)
- Derrac, J., García, S., Molina, D., Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **1**(1), 3–18 (2011)
- Derrac, J., García, S., Hui, S., Suganthan, P.N., Herrera, F.: Analyzing convergence performance of evolutionary algorithms: a statistical approach. *Inf. Sci.* **289**, 41–58 (2014)
- Devaney, R.: *An Introduction To Chaotic Dynamical Systems*. Westview Press, Boulder (2008)
- Dong, H., Song, B., Wang, P., Huang, S.: A kind of balance between exploitation and exploration on kriging for global optimization of expensive functions. *J. Mech. Sci. Technol.* **29**(5), 2121–2133 (2015)
- Dorigo, M., Di Caro, G.: Ant colony optimization: a new meta-heuristic. In: *Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99, vol. 2*, pp. 1470–1477. IEEE (1999)
- Eusuff, M., Lansey, K., Pasha, F.: Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. *Eng. Optim.* **38**(2), 129–154 (2006)
- Farah, A., Guesmi, T., Abdallah, H.H., Ouali, A.: A novel chaotic teaching-learning-based optimization algorithm for multi-machine power system stabilizers design problem. *Int. J. Electr. Power Energy Syst.* **77**, 197–209 (2016)
- Farmer, J.D., Packard, N.H., Perelson, A.S.: The immune system, adaptation, and machine learning. *Phys. D* **22**(1–3), 187–204 (1986)
- Gandomi, A.H., Yang, X.-S.: Chaotic bat algorithm. *J. Comput. Sci.* **5**(2), 224–232 (2014)
- Gandomi, A.H., Yang, X.-S., Talatahari, S., Alavi, A.H.: Firefly algorithm with chaos. *Commun. Nonlinear Sci. Numer. Simul.* **18**(1), 89–98 (2013)
- Gao, S., Vairappan, C., Wang, Y., Cao, Q., Tang, Z.: Gravitational search algorithm combined with chaos for unconstrained numerical optimization. *Appl. Math. Comput.* **231**, 48–62 (2014)
- Ghasemi, M., Ghavidel, S., Rahmani, S., Roosta, A., Falah, H.: A novel hybrid algorithm of imperialist competitive algorithm and teaching learning algorithm for optimal power flow problem with non-smooth cost functions. *Eng. Appl. Artif. Intell.* **29**, 54–69 (2014)
- Gokhale, S.S., Kale, V.S.: An application of a tent map initiated chaotic firefly algorithm for optimal overcurrent relay coordination. *Int. J. Electr. Power Energy Syst.* **78**, 336–342 (2016)
- Holland, J.H.: *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge (1992)
- Hong, W.-C.: Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm. *Neurocomputing* **74**(12), 2096–2107 (2011)
- Huang, L., Ding, S., Shouhao, Y., Wang, J., Ke, L.: Chaos-enhanced Cuckoo search optimization algorithms for global optimization. *Appl. Math. Model.* **40**(5), 3860–3875 (2016)
- Ji, X., Ye, H., Zhou, J., Yin, Y., Shen, X.: An improved teaching-learning-based optimization algorithm and its application to a combinatorial optimization problem in foundry industry. *Appl. Soft Comput.* **57**, 504–516 (2017)
- Jia, D., Zheng, G., Khan, M.K.: An effective memetic differential evolution algorithm based on chaotic local search. *Inf. Sci.* **181**(15), 3175–3187 (2011)
- Joorabian, M., Afzalan, E.: Optimal power flow under both normal and contingent operation conditions using the hybrid fuzzy particle swarm optimisation and Nelder-Mead algorithm (HFPSO-NM). *Appl. Soft Comput.* **14**, 623–633 (2014)
- Jordehi, A.R.: Chaotic bat swarm optimisation (CBSO). *Appl. Soft Comput.* **26**, 523–530 (2015)
- Jordehi, A.R.: A chaotic-based big bang-big crunch algorithm for solving global optimisation problems. *Neural Comput. Appl.* **25**(6), 1329–1335 (2014)

31. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.* **8**(1), 687–697 (2008)
32. Kennedy, J.: *Particle Swarm Optimization*, pp. 760–766. Springer, Boston (2010)
33. Kocarev, L., Tasev, Z.: Public-key encryption based on Chebyshev maps. In: *Proceedings of the 2003 International Symposium on Circuits and Systems*, 2003. ISCAS'03, vol. 3, pp. III–28–III–31. IEEE (2003)
34. Li, B., Wei-Sun, J.: Optimizing complex functions by chaos search. *Cybern. Syst.* **29**(4), 409–419 (1998)
35. Lin, L., Gen, M.: Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation. *Soft. Comput.* **13**(2), 157–168 (2009)
36. Liu, Z., Blasch, E., John, V.: Statistical comparison of image fusion algorithms: recommendations. *Inf. Fusion* **36**, 251–260 (2017)
37. Ma, Z.S.: Chaotic populations in genetic algorithms. *Appl. Soft Comput.* **12**(8), 2409–2424 (2012)
38. Majumdar, M., Mitra, T., Nishimura, K.: *Optimization and Chaos*, vol. 11. Springer, New York (2000)
39. Mingjun, J., Huanwen, T.: Application of chaos in simulated annealing. *Chaos Solitons Fractal* **21**(4), 933–941 (2004)
40. Mirjalili, S., Gandomi, A.H.: Chaotic gravitational constants for the gravitational search algorithm. *Appl. Soft Comput.* **53**, 407–419 (2017)
41. Mirjalili, S., Lewis, A.: The Whale optimization algorithm. *Adv. Eng. Softw.* **95**, 51–67 (2016)
42. Ott, E.: *Chaos in Dynamical Systems*. Cambridge University Press, Cambridge (2002)
43. Peng, C., Sun, H., Guo, J., Liu, G.: Dynamic economic dispatch for wind-thermal power system using a novel bi-population chaotic differential evolution algorithm. *Int. J. Electr. Power Energy Syst.* **42**(1), 119–126 (2012)
44. Rao, R.: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int. J. Ind. Eng. Comput.* **7**(1), 19–34 (2016)
45. Rao, R.V., More, K.C.: Design optimization and analysis of selected thermal devices using self-adaptive Jaya algorithm. *Energy Convers. Manage.* **140**, 24–35 (2017)
46. Rao, R.V., Saroj, A.: Constrained economic optimization of shell-and-tube heat exchangers using elitist-Jaya algorithm. *Energy* **128**, 785–800 (2017)
47. Rao, R.V., Waghmare, G.G.: A new optimization algorithm for solving complex constrained design optimization problems. *Eng. Optim.* **49**(1), 60–83 (2017)
48. Rao, V.R., Savsani, V.J., Vakharia, D.P.: Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. *Comput. Aided Des.* **43**(3), 303–315 (2011)
49. Rao, R.V., Savsani, V.J., Vakharia, D.P.: Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems. *Inf. Sci.* **183**(1), 1–15 (2012)
50. Rao, R.V., More, K.C., Taler, J., Ocloñ, P.: Dimensional optimization of a micro-channel heat sink using Jaya algorithm. *Appl. Therm. Eng.* **103**, 572–582 (2016)
51. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S.: GSA: a gravitational search algorithm. *Inf. Sci.* **179**(13), 2232–2248 (2009)
52. Saremi, S., Mirjalili, S.: Integrating chaos to biogeography-based optimization algorithm. *Int. J. Comput. Commun. Eng.* **2**(6), 655 (2013)
53. Saremi, S., Mirjalili, S., Lewis, A.: Biogeography-based optimisation with chaos. *Neural Comput. Appl.* **25**(5), 1077–1097 (2014)
54. Simon, D.: Biogeography-based optimization. *IEEE Trans. Evol. Comput.* **12**(6), 702–713 (2008)
55. Singh, P.S., Prakash, T., Singh, V.P., Babu, M.G.: Analytic hierarchy process based automatic generation control of multi-area interconnected power system using Jaya algorithm. *Eng. Appl. Artif. Intell.* **60**, 35–44 (2017)
56. Sun, L., Hong, L.J., Hu, Z.: Balancing exploitation and exploration in discrete optimization via simulation through a gaussian process-based search. *Oper. Res.* **62**(6), 1416–1438 (2014)
57. Talatahari, S., Farahmand Azar, B., Sheikholeslami, R., Gandomi, A.H., Gandomi, A.H.: Imperialist competitive algorithm combined with chaos for global optimization. *Commun. Nonlinear Sci. Numer. Simul.* **17**(3), 1312–1319 (2012)
58. Tan, K.C., Chiam, S.C., Mamun, A.A., Goh, C.K.: Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization. *Eur. J. Oper. Res.* **197**(2), 701–713 (2009)
59. Wang, X., Duan, H.: A hybrid biogeography-based optimization algorithm for job shop scheduling problem. *Comput. Ind. Eng.* **73**, 96–114 (2014)
60. Wang, S., Rao, R.V., Chen, P., Zhang, Y., Liu, A., Wei, L.: Abnormal breast detection in mammogram images by feed-forward neural network trained by jaya algorithm. *Fundam. Inf.* **151**(1–4), 191–211 (2017)
61. Yan, X.F., Chen, D.Z., Hu, S.X.: Chaos-genetic algorithms for optimizing the operating conditions based on RBF-PLS model. *Comput. Chem. Eng.* **27**(10), 1393–1404 (2003)
62. Yang, D., Liu, Z., Zhou, J.: Chaos optimization algorithms based on chaotic maps with different probability distribution and search speed for global optimization. *Commun. Nonlinear Sci. Numer. Simul.* **19**(4), 1229–1246 (2014)
63. Yuan, X., Zhao, J., Yang, Y., Wang, Y.: Hybrid parallel chaos optimization algorithm with harmony search algorithm. *Appl. Soft Comput.* **17**, 12–22 (2014)
64. Zhou, N., Zhang, A., Zheng, F., Gong, L.: Novel image compression-encryption hybrid algorithm based on key-controlled measurement matrix in compressive sensing. *Opt. Laser Technol.* **62**, 152–160 (2014)