Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems

R. Venkata Rao*

Department of Mechanical Engineering, S.V. National Institute of Technology, Ichchanath, Surat, Gujarat – 395 007, India

C H R O N I C L E

Article history:
Received July 21 2015
Received in Revised Format
August 16 2015
Accepted August17 2015
Available online
August18 2015

Keywords:
Jaya algorithm
Optimization
CEC 2006
Constrained benchmark problems
Unconstrained benchmark problems
Statistical tests

A B S T R A C T

A simple yet powerful optimization algorithm is proposed in this paper for solving the constrained and unconstrained optimization problems. This algorithm is based on the concept that the solution obtained for a given problem should move towards the best solution and should avoid the worst solution. This algorithm requires only the common control parameters and does not require any algorithm-specific control parameters. The performance of the proposed algorithm is investigated by implementing it on 24 constrained benchmark functions having different characteristics given in Congress on Evolutionary Computation (CEC 2006) and the performance is compared with that of other well-known optimization algorithms. The results have proved the better effectiveness of the proposed algorithm. Furthermore, the statistical analysis of the experimental work has been carried out by conducting the Friedman's rank test and Holm-Sidak test. The proposed algorithm is found to secure first rank for the 'best' and 'mean' solutions in the Friedman's rank test for all the 24 constrained benchmark problems. In addition to solving the constrained benchmark problems, the algorithm is also investigated on 30 unconstrained benchmark problems taken from the literature and the performance of the algorithm is found better.

1. Introduction

The population based heuristic algorithms have two important groups: evolutionary algorithms (EA) and swarm intelligence (SI) based algorithms.Some of the recognized evolutionary algorithms are: Genetic Algorithm (GA), Evolution Strategy (ES), Evolution Programming (EP), Differential Evolution (DE), Bacteria Foraging Optimization (BFO), Artificial Immune Algorithm (AIA), etc. Some of the well-known swarm intelligence based algorithms are:Particle Swarm Optimization (PSO), Shuffled Frog Leaping (SFL), Ant Colony Optimization (ACO), Artificial Bee Colony (ABC), Fire Fly (FF) algorithm, etc. Besides the evolutionary and swarm intelligence based algorithms, there are some other algorithms which work on the principles of different natural phenomena. Some of them are: Harmony Search (HS)

* Corresponding author. 91-261-2201661, Fax: 91-261-2201571
E-mail:ravipudirao@gmail.com(R. Venkata Rao)

doi: 10.5267/j.ijiec.2015.8.004

algorithm, Gravitational Search Algorithm (GSA), Biogeography-Based Optimization (BBO), Grenade Explosion Method (GEM), etc. (Rao and Patel, 2012, 2013).

All the evolutionary and swarm intelligence based algorithms are probabilistic algorithms and require common controlling parameters like population size, number of generations, elite size, etc. Besides the common control parameters, different algorithms require their own algorithm-specific control parameters. For example, GA uses mutation probability, crossover probability, selection operator; PSO uses inertia weight, social and cognitive parameters; ABC uses number of onlooker bees, employed bees, scout bees and limit; HS algorithm uses harmony memory consideration rate, pitch adjusting rate, and the number of improvisations. Similarly, the other algorithms such as ES, EP, DE, BFO, AIA, SFL, ACO, etc. need the tuning of respective algorithm-specific parameters. The proper tuning of the algorithm-specific parameters is a very crucial factor which affects the performance of the above mentioned algorithms. The improper tuning of algorithm-specific parameters either increases the computational effort or yields the local optimal solution. Considering this fact, Rao et al. (2011) introduced the teaching-learning-based optimization (TLBO) algorithm which does not require any algorithm-specific parameters. The TLBO algorithm requires only common controlling parameters like population size and number of generations for its working. The TLBO algorithm has gained wide acceptance among the optimization researchers (Rao, 2015).

Keeping in view of the success of the TLBO algorithm, another algorithm-specific parameter-less algorithm is proposed in this paper. However, unlike two phases (i.e. teacher phase and the learner phase) of the TLBO algorithm, the proposed algorithm has only one phase and it is comparatively simpler to apply. The working of the proposed algorithm is much different from that of the TLBO algorithm. The next section describes the proposed algorithm.

2. Proposed algorithm

Let $f(x)$ is the objective function to be minimized (or maximized). At any iteration i, assume that there are 'm' number of design variables (i.e. j=1,2,…,m), 'n' number of candidate solutions (i.e. population size, k=1,2,…,n). Let the best candidate *best* obtains the best value of $f(x)$ (i.e. $f(x)_{best}$) in the entire candidate solutions and the worst candidate *worst* obtains the worst value of $f(x)$ (i.e. $f(x)_{worst}$) in the entire candidate solutions. If $X_{j,k,i}$is the value of the j^{th} variable for the k^{th} candidate during the i^{th} iteration, then this value is modified as per the following Eq. (1).

$$X'_{j,k,i}= X_{j,k,i}+ r_{1,j,i} (X_{j,best,i}- \left|X_{j,k,i}\right|) - r_{2,j,i} (X_{j,worst,i}- \left|X_{j,k,i}\right|) \quad , \tag{1}$$

where, $X_{j,best,i}$is the value of the variable j for the *best* candidate and $X_{j,worst,i}$is the value of the variable j for the *worst* candidate. $X'_{j,k,i}$is the updated value of $X_{j,k,i}$ and $r_{1,j,i}$ and $r_{2,j,i}$ are the two random numbers for the j^{th} variable during the i^{th} iteration in the range [0, 1]. The term "$r_{1,j,i}$ (($X_{j,best,i}$- $\left|X_{j,k,i}\right|$)" indicates the tendency of the solution to move closer to the best solution and the term "-$r_{2,j,i}$ ($X_{j,worst,i}$- $\left|X_{j,k,i}\right|$)" indicates the tendency of the solution to avoid the worst solution. $X'_{j,k,i}$is accepted if it gives better function value. All the accepted function values at the end of iteration are maintained and these values become the input to the next iteration.

Fig.1 shows the flowchart of the proposed algorithm. The algorithm always tries to get closer to success (i.e. reaching the best solution) and tries to avoid failure (i.e. moving away from the worst solution). The algorithm strives to become victorious by reaching the best solution and hence it is named as **Jaya** (a Sanskrit word meaning **victory**). The proposed method is illustrated by means of an unconstrained benchmark function known as Sphere function in the next section.

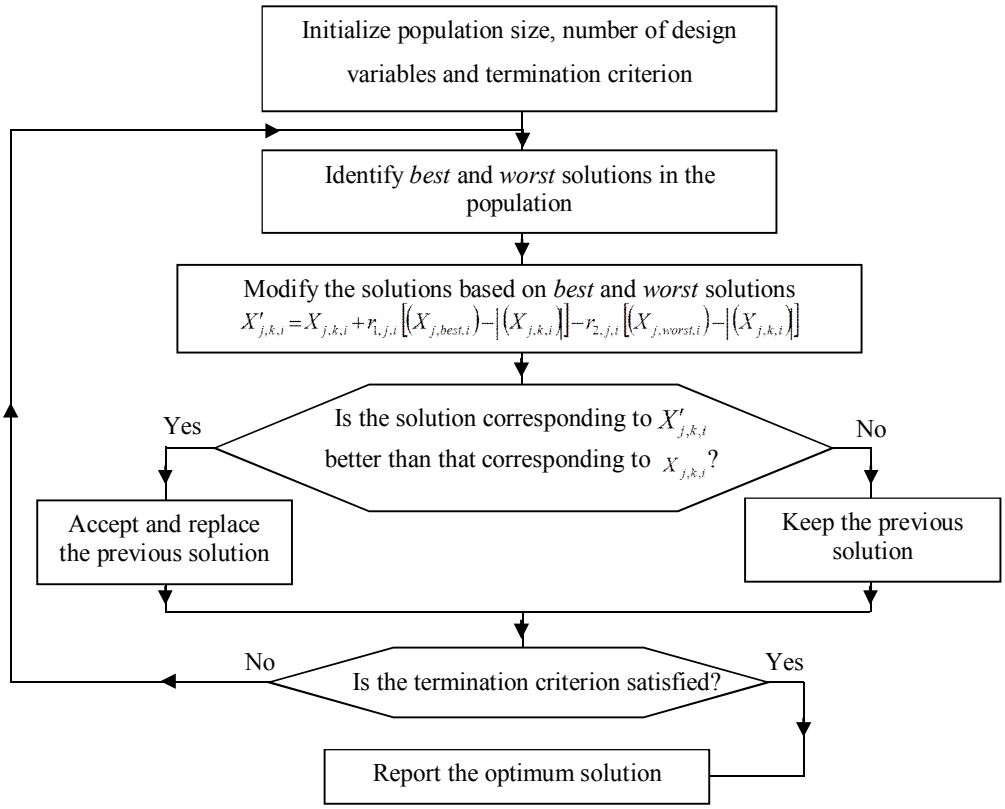

Fig. 1. Flowchart of the Jaya algorithm

2.1 Demonstration of the working of Jaya algorithm

To demonstrate the working of Jaya algorithm, an unconstrained benchmark function of Sphere is considered. The objective function is to find out the values of x_i that minimize the value of the Sphere function.

$$\min f(x_i)=\sum_{i=1}^{n} x_i^2$$
$$\text{subject to} \quad (2)$$
$$-100 \leq x_i \leq 100$$

The known solution to this benchmark function is 0 for all x_i values of 0. Now to demonstrate the Jaya algorithm, let us assume a population size of 5 (i.e. candidate solutions), two design variables x_1 and x_2 and two iterations as the termination criterion. The initial population is randomly generated within the ranges of the variables and the corresponding values of the objective function are shown in Table 1. As it is a minimization function, the lowest value of *f(x)* is considered as the best solution and the highest value of *f(x)* is considered as the worst solution.

Table 1
Initial population

Candidate	x_1	x_2	*f*(*x*)	*Status*
1	-5	18	349	
2	14	63	4165	
3	70	-6	4936	*worst*
4	-8	7	113	*best*
5	-12	-18	468	

From Table 1 it can be seen that the best solution is corresponding the 4th candidate and the worst solution is corresponding to the 3rd candidate. Now assuming random numbers $r_1 = 0.58$ and $r_2 = 0.81$ for x_1 and $r_1 = 0.92$ and $r_2 = 0.49$ for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(1) and

are placed in Table 2. For example, for the 1st candidate, the new values of x_1 and x_2 during the first iteration are calculated as shown below.

$X'_{1,1,1} = X_{1,1,1} + r_{1,1,1}(X_{1,4,1} - |X_{1,1,1}|) - r_{2,1,1}(X_{1,3,1} - |X_{1,1,1}|)$

$= -5 + 0.58(-8 - |-5|) - 0.81(70 - |-5|) = -65.19$

$X'_{2,1,1} = X_{2,1,1} + r_{1,2,1}(X_{2,4,1} - |X_{2,1,1}|) - r_{2,2,1}(X_{2,3,1} - |X_{2,1,1}|)$

$= 18 + 0.92(7 - |18|) - 0.49(-6 - |18|) = 19.64$

Similarly, the new values of x_1 and x_2 for the other candidates are calculated. Table 2 shows the new values of x_1 and x_2 and the corresponding values of the objective function.

Table 2
New values of the variables and the objective function during first iteration

Candidate	x_1	x_2	$f(x)$
1	-65.19	19.64	4635.466
2	-44.12	45.29	3997.76
3	24.76	0.8	613.697
4	-67.5	13.37	4735
5	-70.58	-16.36	5249.186

Now, the values of *f(x)* of Tables 1 and 2 are compared and the best values of *f(x)* are considered and placed in Table 3. This completes the first iteration of the Jaya algorithm.

Table 3
Updated values of the variables and the objective function based on fitness comparison at the end of first iteration

Candidate	x_1	x_2	$f(x)$	*Status*
1	-5	18	349	
2	-44.12	45.29	3997.76	*worst*
3	24.76	0.8	613.697	
4	-8	7	113	*best*
5	-12	-18	468	

From Table 3 it can be seen that the best solution is corresponding the 4th candidate and the worst solution is corresponding to the 2nd candidate. Now, during the second iteration, assuming random numbers $r_1 = 0.27$ and $r_2 = 0.23$ for x_1 and $r_1 = 0.38$ and $r_2 = 0.51$ for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(1). Table 4 shows the new values of x_1 and x_2 and the corresponding values of the objective function during the second iteration.

Table 4
New values of the variables and the objective function during second iteration

Candidate	x_1	x_2	$f(x)$
1	2.7876	-0.0979	7.7803
2	-37.897	30.74	2381.13
3	31.757	-19.534	1390.08
4	-0.3324	-12.528	157.06
5	-4.4924	-36.098	1323.247

Now, the values of *f(x)* of Tables 3 and 4 are compared and the best values of *f(x)* are considered and placed in Table 5. This completes the second iteration of the Jaya algorithm.

Table 5
Updated values of the variables and the objective function based on fitness comparison at the end of second iteration

Candidate	x_1	x_2	*f(x)*	*Status*
1	2.7876	-0.0979	7.7803	*best*
2	-37.897	30.74	2381.13	*worst*
3	24.76	0.8	613.697	
4	-8	7	113	
5	-12	-18	468	

From Table 5 it can be seen that the best solution is corresponding the 1st candidate and the worst solution is corresponding to the 2nd candidate. It can also be observed that the value of the objective function is reduced from 113 to 7.7803 in just two iterations. If we increase the number of iterations then the known value of the objective function (i.e. 0) can be obtained within next few iterations. Also, it is to be noted that in the case of maximization function problems, the best value means the maximum value of the objective function and the calculations are to be proceeded accordingly. Thus, the proposed method can deal with both minimization and maximization problems.

The above demonstration is for an unconstrained optimization problem. However, the similar steps can be followed in the case of constrained optimization problem. The main difference is that a penalty function is used in the constrained optimization problem to take care of the violation of each constraint and the penalty is operated upon the objective function. The next section deals with the experimentation of the proposed algorithm on 24 constrained benchmark problems given in CEC 2006 (Liang et al., 2006; Karaboga&Basturk, 2007; Karaboga&Akay, 2011).

3. Experiments on constrained benchmark problems

The objectives and constraints of the considered 24 benchmark functions of CEC 2006 have different characteristics such as linear, nonlinear, quadratic, cubic and polynomial. The number of design variables and their ranges are different for each problem (Liang et al., 2006; Karaboga and Basturk, 2007; Karaboga and Akay, 2011).

To evaluate the performance of the proposed Jaya algorithm, the results obtained by using the Jaya algorithm are compared with the results obtained by the other optimization algorithms such as homomorphous mapping (HM), adaptive segregational constraint handling evolutionary algorithm (ASCHEA), simple multi-membered evolution strategy (SMES), genetic algorithm (GA), particle swarm optimization (PSO), differential evolution (DE), artificial bee colony (ABC), biogeography based optimization (BBO) available in the literature (Karaboga&Basturk, 2007; Karaboga&Akay, 2011). However, these algorithms were experimented only on 13 functions of CEC 2006. Patel and Savsani (2015) extended the application of PSO, BBO, DE and ABC to the remaining 11 functions also. In addition, they had applied TLBO and heat transfer search (HTS) algorithms for all the 24 constrained benchmark functions.

Now, the computational experiments are conducted to identify the performance of the proposed Jaya algorithm and the performance of the algorithm is compared with the above-mentioned algorithms. A common platform is provided by maintaining the identical function evolutions for different algorithms considered for the comparison. Thus, the consistency in the comparison is maintained while comparing the performance of Jaya algorithm with other optimization algorithms. However, in general, the algorithm which requires less number of function evaluations to get the same best solution can be considered as better as compared to the other algorithms. However, in this paper, to maintain the consistency in the comparison of competitive algorithms, a common experimental platform described by Patel and Savsani (2015) is provided by setting the maximum number of function evaluations as 240000 for each benchmark function and the static penalty method is used as a constraint handling technique. Just like other algorithms, the proposed Jaya algorithm is executed 100 times for each benchmark function and the mean results obtained are compared with the other algorithms for the same number of runs.

3.1. Results and discussion on constrained benchmark functions

Table 6 presents the comparative results of G01-G13 test functions obtained by HM, ASCHEA, SMES, GA, PSO, DE, ABC, BBO, TLBO, HTS and Jaya algorithms for 240000 function evaluations averaged over 100 runs.

Table 6

Comparative results of G01-G13 benchmark functions obtained by different algorithms

Function		HM[a]	ASCHEA[b]	SMES[c]	GA[c]	PSO[c]	DE[c]	ABC[c]	BBO[b]	HTS[b]	TLBO[b]	TLBO[b] (Corrected results)	Jaya
G01 (-15.00)	B	-14.7864	-15.000	-15	14.44	-15	-15	-15	-14.977	−15	-15	-15	-15
	W	---	---	---	---	-13	-11.828	-15	-14.5882	- 15	-6	-15	-15
	M	-14.7082	-14.84	**-15**	-14.236	-14.71	-14.555	**-15**	-14.7698	**-15**	-10.782	**-15**	**-15**
G02 (-0.803619)	B	-0.79953	-0.785	-0.803601	-0.796321	-0.669158	-0.472	-0.803598	-0.7821	- 0 .7517	-0.7835	-0.7899	-0.803605
	W	---	---	---	---	-0.299426	-0.472	-0.749797	-0.7389	-0.5482	-0.5518	-0.5985	-0.7542
	M	-0.79671	-0.59	-0.785238	-0.78858	-0.41996	-0.665	-0.792412	-0.7642	-0.6437	-0.6705	-0.6882	**-0.7968**
G03 (-1.0005)	B	0.9997	1	1	0.99	1	-0.99393	-1	-1.0005	-1.0005	-1.0005	-1.0005	-1.005
	W	---	---	---	---	-0.464	-1	-1	-0.0455	0	0	-1	-1
	M	0.9989	0.99989	1	0.976	0.764813	**-1**	**-1**	-0.3957	-0.9004	-0.8	**-1**	**-1**
G04 (-30665.539)	B	-30664.5	-30665.5	-	-30626.053	-30665.539	-30665.539	-30665.539	-30665.539	-30665.5387	-30665.5387	-30665.5387	-30665.5387
	W	---	---	---	---	-30665.539	-30665.539	-30665.539	-29942.3	-30665.5387	-30665.5387	-30665.5387	-30665.5387
	M	-30655.3	**-30665.5**	-	-30590.455	**-30665.539**	**-30665.539**	**-30665.539**	-30411.865	**-30665.5387**	**-30665.5387**	**-30665.5387**	**-30665.5387**
G05 (5126.486)	B	NF	5126.5	5126.599	NF	5126.484	5126.484	5126.484	5134.2749	5126.486	5126.486	5126.486	5126.486
	W	NF	---	---	NF	5249.825	5534.61	5438.387	7899.2756	5126.6831	5127.714	5127.4197	5126.635
	M	NF	5141.65	5147.492	NF	5135.973	5264.27	5185.714	6130.5289	5126.5152	5126.6184	5126.5146	**5126.5061**
G06 (-6961.814)	B	-6952.1	-6961.81	-6961.814	-6952.472	-6961.814	-6954.434	-6961.814	-6961.8139	-6961.814	-6961.814	-6961.814	-6961.814
	W	---	---	---	---	-6961.814	-6954.434	-6961.805	-5404.4941	-6961.814	-6961.814	-6961.814	-6961.814
	M	-6342.6	**-6961.81**	-6961.284	-6872.204	**-6961.814**	-6954.434	-6961.813	-6181.7461	**-6961.814**	**-6961.814**	**-6961.814**	**-6961.814**
G07 (24.3062)	B	24.62	24.3323	24.327	31.097	24.37	24.306	24.33	25.6645	24.3104	24.3101	24.3100	24.3062
	W	---	---	---	---	56.055	24.33	25.19	37.6912	25.0083	27.6106	26.7483	24.8932
	M	24.826	24.6636	24.475	34.98	32.407	24.31	24.473	29.829	24.4945	24.837	24.6482	**24.3092**
G08 (-0.095825)	B	0.095825	0.095825	0.095825	0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
	W	---	---	---	---	-0.095825	-0.095825	-0.095825	-0.095817	-0.095825	-0.095825	-0.095825	-0.095825
	M	0.0891568	0.095825	0.095825	0.095799	**-0.095825**	**-0.095825**	**-0.095825**	-0.95824	**-0.095825**	**-0.095825**	**-0.095825**	**-0.095825**
G09 (680.6301)	B	680.91	680.63	680.632	685.994	680.63	680.63	680.634	680.6301	680.6301	680.6301	680.6301	680.6301
	W	---	---	---	---	680.631	680.63	680.634	721.0795	680.644	680.6456	680.6334	680.6301
	M	681.16	680.641	680.643	692.064	**680.63**	**680.63**	680.634	692.7162	680.6329	680.6336	680.6313	**680.6301**
G10 (7049.28)	B	7147.9	7061.13	7051.903	9079.77	7049.481	7049.548	7053.904	7679.0681	7049.4836	7250.9704	7052.329	7049.312
	W	---	---	----	---	7894.812	9264.886	7604.132	9570.5714	7252.0546	7291.3779	7152.0813	7119.632
	M	8163.6	7497.434	7253.047	10003.225	7205.5	7147.334	7224.407	8764.9864	7119.7015	7257.0927	7114.4893	**7104.6201**
G11 (0.7499)	B	0.75	0.75	0.75	0.75	0.749	0.752	0.75	0.7499	0.7499	0.7499	0.7499	0.7499
	W	---	---	---	---	0.749	1	0.75	0.92895	0.7499	0.7499	0.7499	0.7499
	M	**0.75**	**0.75**	**0.75**	**0.75**	**0.749**	0.901	**0.75**	0.83057	**0.7499**	**0.7499**	**0.7499**	**0.7499**
G12 (-1)	B	---	---	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
	W	---	---	---	---	-0.994	-1	-1	-1	-1	-1	-1	-1
	M	---	---	**-1**	**-1**	-0.998875	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**	**-1**
G13 (-0.05394)	B	---	---	0.054986	0.134057	0.085655	0.385	0.76	0.62825	0.37319	0.44015	0.003631	0.003625
	W	---	---	---	---	1.793361	0.99	1	1.45492	0.79751	0.95605	0.003632	0.003631
	M	---	---	0.166385	---	0.569358	0.872	0.968	1.09289	0.66948	0.69055	0.003631	**0.003627**

Result in boldface indicates better performing algorithm. (-) indicates that the results are not available. NF: means that no feasible solutions were found. The bold values indicate best results.
a: used decoder-based penalty method for constraint handling; b: used static penalty method for constraint handling.; c: used Deb's method for constraint handling.

For the TLBO algorithm, Patel and Savsani(2015) used a population size of 50 for all the 24 benchmark functions and thus reported inferior values for the TLBO algorithm for many benchmark functions even though the function evaluations were kept as 240000 in each case. It may be noted that population size and the number of generations are common control parameters (i.e. not algorithm-specific parameters) and variation in the common control parameters may also affect the results. Patel and Savsani (2015) had not considered different combinations of population sizes and the number of generations in the case of TLBO algorithm while maintaining the number of function evaluations as 240000. Hence, the results of TLBO algorithm are corrected now and included in Table 6. The corrected results for TLBO algorithm presented in this paper correspond to the population sizes of 100, 70, 100, 40, 60, 100, 50, 30, 100, 70, 50, 60, 80, 80, 50, 100, 70, 100, 50, 100, 100, 50, 70 and 80 respectively for G01-G24 functions and the number of generations are decided accordingly keeping the number of function evaluations same as 240000 in each case. It can be seen that the corrected results of TLBO algorithm are better than those reported by Patel and Savsani (2015). Employing a population size of 50 for all the algorithms for solving all benchmark functions by Patel and Savsani (2015) might not have revealed the true potential of the algorithms. It is important to keep the same number of function evaluations for all the algorithms for a benchmark function (instead of same population size for all the algorithms) and this procedure can be applied to other benchmark functions also.

The results of the proposed Jaya algorithm for each benchmark function by employing appropriate population sizes while maintaining the function evaluations of 240000 are included in the last column of Table 6. The Table 6 shows the comparative results of G01-G13 benchmark functions obtained by different algorithms for 240000 function evaluations averaged over 100 runs. The 'best (B)', 'worst (W)' and 'mean (M)' values of the 13 constrained benchmark functions (i.e. G01 – G13) attempted by HM, ASCHEA, SMES, GA, PSO, DE, ABC, BBO, HTS, TLBO and Jaya algorithm are shown. The global optimum values expected are given within brackets under each function. It can be observed that the proposed Jaya algorithm has obtained global optimum values for all the benchmark functions except for G10. However, in this case also, the global optimum value obtained by Jaya algorithm is superior to the remaining algorithms. Furthermore, it can be observed that the 'mean (M)' values of all the 13 benchmark functions obtained by Jaya algorithm are better than all other algorithms.

Table 7 shows the comparative results of G14-G24 benchmark functions obtained by different algorithms for 240000 function evaluations averaged over 100 runs. Here also it can be observed that the proposed Jaya algorithm has obtained global optimum values for all the benchmark functions except for G14, G19, G20, GG21, G22 and G23. However, in these cases also, the global optimum value obtained by Jaya algorithm for each of these functions is superior to those given by the remaining algorithms. Furthermore, it can also be observed that the 'mean (M)' values of all the 11 benchmark functions obtained by Jaya algorithm are better than all other algorithms. It may be mentioned here that the HTS algorithm has employed elitism while all other algorithms have not employed the concept of elitism. In fact, a fair comparison means that only the non-elitist algorithms should be compared. However, as only the results of elitist HTS are available in Patel and Savsani (2015), the comparison is made with the same. Even then it can be observed that Jaya algorithm has performed better than the other algorithms including the elitist HTS algorithm in the case of G14-G24 benchmark functions. It may be mentioned here that the elitist HTS algorithm is not an algorithm-specific parameter-less algorithm (as it contains conduction, convection and radiation factors and the results vary with the variation in the values of these factors).

Table 8 shows the success rates of various algorithms for G01-G24 functions over 100 runs. The comparison is made between PSO, BBO, DE, ABC, HTS and TLBO algorithms as these algorithms are applied for all the 24 benchmark functions. The success rate obtained by all the algorithms is 0 in the case of 8 benchmark functions (i.e. G02, G10, G13, G14, G19, G20, G22 and G23). In the case of all remaining 16 benchmark functions, the success rate obtained by the proposed Jaya algorithm is either equal or superior to the other algorithms of PSO, BBO, DE, ABC, HTS and TLBO algorithms.

Table 7
Comparative results of G14-G24benchmark functions obtained by different algorithms for 240000 function evaluations averaged over 100 runs

Function		PSO	BBO	DE	ABC	HTS	TLBO	TLBO	Jaya
G14 (–47.764)	Best	-44.9343	54.6679	-45.7372	-44.6431	-47.7278	-46.5903	- 46.7339	-47.7322
	Worst	-37.5000	257.7061	-12.7618	-23.3210	-45.0648	-17.4780	-18.1581	-46.2908
	Mean	-40.8710	175.9832	-29.2187	-40.1071	-46.4076	-39.9725	-40.2854	**-46.6912**
	SD	2.29E+00	7.90E+01	1.36E+01	7.14E+00	8.53E-01	1.15E+01	6.38E-01	2.98E-01
G15 (961.715)	Best	961.7150	962.6640	961.7150	961.7568	961.7150	961.7150	961.7150	961.7150
	Worst	972.3170	1087.3557	962.1022	970.3170	962.0653	964.8922	961.7150	961.7150
	Mean	965.5154	1001.4367	961.7537	966.2868	961.7500	962.8641	**961.7150**	**961.7150**
	SD	3.72E+00	4.74E+01	1.22E-01	3.12E+00	1.11E-01	1.49E+00	1.03E-01	0.9E-01
G16 (–1.9052)	Best	-1.9052	-1.9052	-1.9052	-1.9052	-1.9052	-1.9052	-1.9052	-1.9052
	Worst	-1.9052	-1.1586	-1.9052	-1.9052	-1.9052	-1.9052	-1.9052	-1.9052
	Mean	**-1.9052**	-1.6121	**-1.9052**	**-1.9052**	**-1.9052**	**-1.9052**	**-1.9052**	**-1.9052**
	SD	2.3E-16	2.58E-01	2.34E-16	2.34E-16	2.34E-16	2.134E-16	2.08E-16	1.62E-16
G17 (8853.5396)	Best	8857.5140	9008.5594	8854.6501	8859.7130	8853.5396	8853.5396	8853.5396	8853.5396
	Worst	8965.4010	9916.7742	8996.3215	8997.1450	8932.0712	8919.6595	8919.6595	8902.203
	Mean	8899.4721	9384.2680	8932.0444	8941.9245	8877.9175	8876.5071	8876.5071	**8872.5402**
	SD	3.79E+01	3.06E+02	4.68E+01	4.26E+01	3.09E+01	3.02E+01	2.87E+01	1.87E+01
G18 (–0.86603)	Best	-0.86603	-0.65734	-0.86531	-0.86603	-0.86603	-0.86603	-0.86603	-0.86603
	Worst	-0.51085	-0.38872	-0.85510	-0.86521	-0.67468	-0.86294	-0.86304	-0.86601
	Mean	-0.82760	-0.56817	-0.86165	-0.86587	-0.77036	-0.86569	-0.86601	**-0.86602**
	SD	1.11E-01	8.55E-02	3.67E-03	3.37E-04	1.01E-01	9.67E-04	8.45E-05	3.56E-06
G19 (32.6555)	Best	33.5358	39.1471	32.6851	33.3325	32.7132	32.7916	32.7194	32.6803
	Worst	39.8443	71.3106	32.9078	38.5614	33.2140	36.1935	35.9893	32.8776
	Mean	36.6172	51.8769	32.7680	36.0078	32.7903	34.0792	33.9912	**32.7512**
	SD	2.04E+00	1.12E+01	6.28E-02	1.83E+00	1.53E-01	9.33E-01	7.72E-01	1.87E-01
G20 (0.204979)	Best	0.24743	1.26181	0.24743	0.24743	0.24743	0.24743	0.24385	0.24139
	Worst	1.8720	1.98625	0.28766	1.52017	0.27331	1.84773	1.54249	0.25614
	Mean	0.97234	1.43488	0.26165	0.80536	0.25519	1.22037	1.03492	**0.24385**
	SD	6.34E-01	2.20E-01	1.91E-02	5.93E-01	1.25E-02	5.89E-01	1.12E-02	0.98E-02
G21 (193.274)	Best	193.7311	198.8151	193.7346	193.7343	193.7264	193.7246	193.6539	193.5841
	Worst	409.1320	581.2178	418.4616	330.1638	320.2342	393.8295	368.3786	202.3262
	Mean	345.6595	367.2513	366.9193	275.5436	256.6091	264.6092	260.4783	**193.7219**
	SD	6.36E+01	1.34E+02	9.13E+01	6.05E+01	6.63E+01	9.23E+01	4.82E+01	1.63E+01
G22 (236.4309)	Best	1.68E+22	1.02E+15	1.25E+18	2.82E+08	2.16E+03	4.50E+17	3.39E+16	5.66E+02
	Worst	3.25E+23	6.70E+16	2.67E+19	1.25E+18	1.33E+07	4.06E+19	3.97E+18	3.21E+06
	Mean	1.63E+23	1.41E+16	1.78E+19	4.10E+17	1.36E+06	1.61E+19	2.93E+18	**2.19E+03**
	SD	9.17E+22	1.96E+16	1.17E+19	4.72E+17	4.20E+06	1.51E+19	3.91E+09	1.91E+05
G23 (–400.055)	Best	-105.9826	2.3163	-72.6420	-43.2541	-390.6472	-385.0043	-390.8342	-391.5192
	Worst	0	74.6089	0	0	0	0	0	-132.49
	Mean	-25.9179	22.1401	-7.2642	-4.3254	-131.2522	-83.7728	-142.5837	**-381.2312**
	SD	4.30E+01	2.51E+01	2.30E+01	1.37E+01	1.67E+02	1.59E+02	2.37E+01	1.29E+01
G24 (–5.5080)	Best	-5.5080	-5.5080	-5.5080	-5.5080	-5.5080	-5.5080	-5.5080	-5.5080
	Worst	-5.5080	-5.4857	-5.5080	-5.5080	-5.5080	-5.5080	-5.5080	-5.5080
	Mean	**-5.5080**	-5.4982	**-5.5080**	**-5.5080**	**-5.5080**	**-5.5080**	**-5.5080**	**-5.5080**
	SD	9.3E-16	6.75E-03	9.36E-16	9.36E-16	9.36E-16	9.36E-16	9.36E-16	8.15E-17

The bold values indicate best results.

Table 9 shows the 'Mean number of function evaluations' required to reach global optimum value by the competitive algorithms for G01-G24 benchmark functions over 100 independent runs (except G02, G10, G13, G14, G19, G20, G22 and G23 functions for which the data is not available in the literature). Here also it can be observed that the proposed Jaya algorithm has obtained better results (i.e. minimum number of mean function evaluations required to reach global optimum value) for all the benchmark functions except for G01 and G12 as compared to PSO, BBO, DE, ABC, HTS and TLBO algorithms. The values of standard deviation of function evaluations for the Jaya algorithm are also comparatively better in the case of many functions.

Table 8
Success rate of various algorithms for G01-G24 functions over 100 runs

Function	PSO	BBO	DE	ABC	HTS	TLBO	TLBO (Corrected results)	Jaya
G01	38	0	94	100	100	26	100	100
G02	0	0	0	0	0	0	0	0
G03	59	23	41	67	86	74	81	96
G04	100	16	100	100	100	100	100	100
G05	61	0	93	28	95	92	96	98
G06	100	21	100	100	100	100	100	100
G07	21	0	26	28	37	23	34	41
G08	100	94	100	100	100	100	100	100
G09	84	26	95	89	96	91	92	100
G10	0	0	0	0	0	0	0	0
G11	100	57	19	100	100	100	100	100
G12	100	100	100	100	100	100	100	100
G13	0	0	0	0	0	0	0	0
G14	0	0	0	0	0	0	0	0
G15	53	0	73	42	83	81	86	87
G16	100	18	100	100	100	100	100	100
G17	0	0	0	0	26	58	61	74
G18	56	0	61	73	47	64	67	84
G19	0	0	0	0	0	0	0	0
G20	0	0	0	0	0	0	0	0
G21	12	0	24	36	48	35	51	64
G22	0	0	0	0	0	0	0	0
G23	0	0	0	0	0	0	0	0
G24	100	27	100	100	100	100	100	100

Table 9
Mean number of function evaluations required to reach global optimum value by comparative algorithms for G01-G24 over 100 independent runs

Function		PSO	BBO	DE	ABC	HTS	TLBO	TLBO (Corrected results)	Jaya
G01	Mean_FE	33750	---	6988.89	13200	12570	9750	9678	8972
	Std_FE	3872.01	---	157.674	708.676	6312.43	4171.93	4067.43	4038.56
G02	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	---	---	---	---
G03	Mean_FE	84610	157950	136350	121950	67400	178083	66985	61747
	Std_FE	35670	5939.7	78988.3	64296.1	29776.2	43819.3	28564.5	27612.1
G04	Mean_FE	14432	189475	14090	29460	10135	5470	4958	4210
	Std_FE	309.23	35390.7	1499.22	2619.25	1413.63	804.225	787.388	697.248
G05	Mean_FE	57921	---	108572	197749	43356	46888	42583	41734
	Std_FE	14277.4	---	41757.1	20576.8	3416.3	19623.2	3356.6	2849.2
G06	Mean_FE	14923	140150	17540	69310	15395	11600	10953	9372
	Std_FE	1789.32	22273.9	1214.91	3753.65	2566.61	2056.43	2011.84	1992.32
G07	Mean_FE	97742	---	147650	114351	92916.7	147550	90382.57	87237.65
	Std_FE	2984.2	---	4737.62	11384.4	17237.3	5020.46	4839.58	4695.01
G08	Mean_FE	622	4290	725	670	635	680	604	567
	Std_FE	189.78	4418.32	259.54	249.666	171.675	181.353	167.327	161.864
G09	Mean_FE	34877	194700	57205	149642	23235	37690	21949	20754
	Std_FE	12280.1	29557.1	10779.1	73436.8	10806.2	26350.6	9943.6	9746.3
G10	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	---	---	---	---
G11	Mean_FE	23312	35490	205250	29140	53270	3000	2957	2864
	Std_FE	1231.41	30627.4	8273.15	12982.5	18215.2	1354.83	1294	1163
G12	Mean_FE	1204	1865	1150	1190	2190	2480	2167	2086
	Std_FE	341.3	2240.54	263.523	747.514	824.554	917.484	811.135	789.326
G13	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	---	---	---	---
G14	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	----	---	---	---	---	---
G15	Mean_FE	41972	---	36391.7	157800	36756.3	52287.5	32593.6	30388.2
	Std_FE	4073.9	---	5509.21	57558.5	28670.6	47937.1	26399.1	24053.8
G16	Mean_FE	7114	85200	12565	19670	13045	7840	7612	6990
	Std_FE	643.3	16122	1155.19	714.998	1358.6	2709.74	1328.5	1287.4
G17	Mean_FE	---	---	---	---	65600	126980	62943	60483
	Std_FE	---	---	---	---	65053.8	46591.8	44936.3	42521.8
G18	Mean_FE	23769	---	170140	114120	35360	19226	18395	17943
	Std_FE	1009.78	---	20227.7	58105.8	7731.14	5762.16	5693.91	5395.23
G19	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	---	---	---	---
G20	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	---	---	---	---
G21	Mean_FE	39937	---	89500	99150	28037.5	108533	27664.8	26739.4
	Std_FE	4302.2	---	14283.6	3647.94	7032.35	8677.17	7011.76	6894.21
G22	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	---	---	---	---
G23	Mean_FE	---	---	---	---	---	---	---	---
	Std_FE	---	---	---	---	----	---	---	---
G24	Mean_FE	2469	84625	4855	5400	3715	2710	2704	2689
	Std_FE	245.5	2015.25	429.761	618.241	575.929	864.677	543.589	512.943

--- indicate that algorithm is failed to obtain a global optimum value for that function
Mean_FE= Mean number of function evaluations.
Std_FE=Standard deviation of function evaluations.

4. Statistical tests

It is observed from the results presented in Tables 6-9 that the performance of the proposed Jaya algorithm is better than the other competitive algorithms. However, it is necessary to conduct the statistical tests like Friedman rank test (Joaquin et al., 2011) and Holm-Sidak test (Holm, 1979) to prove the significance of the proposed algorithm. Table 10 shows Friedman rank test for the 'Best' and 'Mean' solutions obtained for G01-G13 functions. Table 11 shows the Friedman rank test for the 'Best' and 'Mean' solutions obtained for G14-G24 functions. The G05, G12, G13 functions were omitted by the previous researchers as the results of these functions are not available for some of the competitive algorithms and G22 function was omitted as none of the competitive algorithms had produced feasible solution for this function. Hence the same approach is used in the present work and it can be easily observed from Tables 10 and 11 that the proposed Jaya algorithm has got the 1st rank in the case of "Best" and "Mean' solutions of all benchmark functions considered. The corrected TLBO algorithm has obtained the 2nd rank. Table 12 shows the results of Friedman rank test for the 'Success Rate' solutions obtained. The proposed Jaya algorithm has obtained the 1st rank again followed by TLBO.

Table 10
Friedman rank test for the 'Best' and 'Mean' solutions obtained for G01-G13 functions

Test for best solution				Test for mean solution			
Algorithms	Friedman value	Normalized value	Rank	Algorithms	Friedman value	Normalized value	Rank
HM	85	2.61	10	HM	78	2.94	9
ASCHEA	70	2.15	9	ASCHEA	71	2.67	8
SMES	58.5	1.8	6	SMES	65	2.45	7
GA	92	2.83	11	GA	88	3.32	10
PSO	57	1.75	5	PSO	59	2.22	6
DE	60	1.84	7	DE	55.5	2.09	5
ABC	53	1.63	4	ABC	42.5	1.60	3
BBO	62	1.90	8	BBO	88	3.32	10
HTS	45.5	1.4	3	HTS	46	1.73	4
TLBO (corrected)	44.5	1.36	2	TLBO (corrected)	40.5	1.52	2
Jaya	32.5	1	1	Jaya	26.5	1	1

Table 11
Friedman rank test for the 'Best' and 'Mean' solutions obtained for G14-G24 functions

Test for best solution				Test for mean solution			
Algorithms	Friedman value	Normalized value	Rank	Algorithms	Friedman value	Normalized value	Rank
PSO	42.5	2.02	4	PSO	42	2.70	4
BBO	64	3.04	6	BBO	70	4.51	7
DE	42.5	2.02	4	DE	43	2.77	5
ABC	49.5	2.35	5	ABC	46	2.96	6
HTS	31.5	1.5	3	HTS	32	2.06	3
TLBO (corrected)	29	1.38	2	TLBO (corrected)	31.5	2.03	2
Jaya	21	1	1	Jaya	15.5	1	1

Table 12
Friedman rank test for the 'Success Rate' of the solutions obtained

Algorithms	Friedman value	Normalized value	Rank
PSO	106	1.58	6
BBO	136.5	2.03	7
DE	100	1.49	5
ABC	95.5	1.42	4
HTS	83	1.23	3
TLBO (corrected)	80	1.19	2
Jaya	67	1	1

Table 13
Holm-Sidak test for the 'Best' and the 'Mean' solutions obtained for G01-G13 functions

Test for best solution		Test for mean solution	
Algorithm[a]	*p*-value	Algorithm[a]	*p*-value
1-5	0.02019	1-5	0.03481
1-2	0.36309	1-9	0.03692
1-3	0.37918	1-2	0.4158
1-9	0.41125	1-3	0.6321
1-7	0.97138	1-4	0.8273
1-4	0.98204	1-6	0.9108
1-6	0.98327	1-7	0.9522
1-8	0.98502	1-11	0.9613
1-11	0.98575	1-8	0.9782
1-10	0.98689	1-10	0.9881

a 1-Jaya, 2-HM, 3-ASCHEA, 4-SMES, 5-GA, 6-PSO, 7-DE, 8-ABC, 9-BBO, 10-TLBO, 11-HTS

Table 14
Holm-Sidak test for the 'Best' and the 'Mean' solutions obtained for G14-G24 functions

Test for best solution		Test for mean solution	
Algorithm[a]	*p*-value	Algorithm[a]	*p*-value
1-3	0.01319	1-3	0.00071
1-5	0.15423	1-5	0.30019
1-4	0.21998	1-4	0.43427
1-2	0.22105	1-2	0.44549
1-7	0.96502	1-7	0.82612
1-6	0.97806	1-6	0.86512

a 1-Jaya, 2-PSO, 3-BBO, 4-DE, 5-ABC, 6-TLBO, 7-HTS

Table 15
Unconstrained benchmark functions considered

No.	Function	Formulation	D	Search range	C
F 1	Sphere	$F_{\min}=\sum_{i=1}^{D}x_i^2$	30	[-100, 100]	US
F 2	SumSquares	$F_{\min}=\sum_{i=1}^{D}ix_i^2$	30	[-10, 10]	US
F 3	Beale	$F_{\min}=\sum_{i=1}^{D}\left(1.5-x_1+x_1x_2\right)^2+\left(2.25-x_1+x_1x_2^2\right)^2+\left(2.625-x_1+x_1x_2^3\right)^2$	5	[-4.5, 4.5]	UN
F 4	Easom	$F_{\min}=-\cos\left(x_1\right)\cos\left(x_2\right)\exp\left(-\left(x_1-\pi\right)^2-\left(x_2-\pi\right)^2\right)$	2	[-100, 100]	UN
F 5	Matyas	$F_{\min}=0.26\left(x_1^2+x_2^2\right)-0.48x_1x_2$	2	[-10, 10]	UN
F 6	Colville	$F_{\min}=100\left(x_1^2-x_2\right)^2+\left(x_1-1\right)^2+\left(x_3-1\right)^2+90\left(x_3^2-x_4\right)+10.1\left(\left(x_2-1\right)^2+\left(x_4-1\right)^2\right)-0.48x_1x_2+19.8\left(x_2-1\right)\left(x_4-1\right)$	4	[-10, 10]	UN
F 7	Trid 6	$F_{\min}=\sum_{i=1}^{D}\left(x_i-1\right)^2-\sum_{i=2}^{D}x_ix_{i-1}$	6	[-D^2, D^2]	UN
F 8	Trid 10	$F_{\min}=\sum_{i=1}^{D}\left(x_i-1\right)^2-\sum_{i=2}^{D}x_ix_{i-1}$	10	[-D^2, D^2]	UN
F 9	Zakharov	$F_{\min}=\sum_{i=1}^{D}x_i^2+\left(\sum_{i=1}^{D}0.5ix_i\right)^2+\left(\sum_{i=1}^{D}0.5ix_i\right)^4$	10	[-5, 10]	UN
F 10	Schwefel 1.2	$F_{\min}=\sum_{i=1}^{D}\left(\sum_{j=1}^{i}x_j^2\right)^2$	30	[-100, 100]	UN
F 11	Rosenbrock	$F_{\min}=\sum_{i=1}^{D}[100(x_i^2-x_{i+1})^2+(1-x_i)^2]$	30	[-30, 30]	UN
F 12	Dixon-Price	$F_{\min}=\left(x_1-1\right)^2+\sum_{i=2}^{D}i\left(2x_i^2-x_{i-1}\right)^2$	30	[-10, 10]	UN

D: Dimension, C: Characteristic, U: Unimodal, M: Multimodal, S: Separable, N: Non-separable

Table 15
Unconstrained benchmark functions considered (Continued)

No.	Function	Formulation	D	Search range	C
F 13	Foxholes	$F_{\min}=\left[\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}\left(x_i-a_{ij}\right)^6}\right]^{-1}$	2	[-65.536, 65.536	
F 14	Branin	$F_{\min}=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	[-5, 10] [0, 15]	MS
F 15	Bohachevsky 1	$F_{\min}=x_1^2+2x_2^2-0.3\cos\left(3\pi x_1\right)-0.4\cos\left(4\pi x_2\right)+0.7$	2	[-100, 100]	MS
F 16	Booth	$F_{\min}=\left(x_1-2x_2-7\right)^2+\left(2x_1+x_2-5\right)^2$	2	[-10, 10]	MS
F 17	Michalewicz 2	$F_{\min}=-\sum_{i=1}^{D}\sin x_1\left(\sin\left(ix_i^2/\pi\right)\right)^{20}$	2	[0, π]	MS
F 18	Michalewicz 5	$F_{\min}=-\sum_{i=1}^{D}\sin x_1\left(\sin\left(ix_i^2/\pi\right)\right)^{20}$	5	[0, π]	MS
F 19	Bohachevsky 2	$F_{\min}=x_1^2+2x_2^2-0.3\cos\left(3\pi x_1\right)\left(4\pi x_2\right)+0.3$	2	[-100, 100]	MN
F 20	Bohachevsky 3	$F_{\min}=x_1^2+2x_2^2-0.3\cos\left(3\pi x_1+4\pi x_2\right)+0.3$	2	[-100, 100]	MN
F 21	GoldStein-Price	$F_{\min}=\left[1+\left(x_1+x_2+1\right)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]\left[30+\left(2x_1-3x_2\right)^2\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$	2	[-2, 2]	MN
F 22	Perm	$F_{\min}=\sum_{k=1}^{D}\left[\sum_{i=1}^{D}\left(i^k+\beta\right)\left(\left(x_i/i\right)^k-1\right)\right]^2$	4	[-D, D]	MN
F 23	Hartman 3	$F_{\min}=-\sum_{i=1}^{4}c_i\exp\left[-\sum_{j=1}^{3}a_{ij}\left(x_j-p_{ij}\right)^2\right]$	3	[0, 1]	MN
F 24	Ackley	$F_{\min}=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right)+20+e$	30	[-32, 32]	MN
F 25	Penalized 2	$F_{\min}=0.1\left[\sin^2(\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2\left\{1+\sin^2(3\pi x_{i+1})\right\}+(x_D-1)^2\left(1+\sin^2(2\pi x_D)\right)\right]+\sum_{i=1}^{D}u(x_i,5,100,4),\quad u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m & x_i>a,\\ 0, & -a\le x_i\le a,\\ k(-x_i-a)^m, & x_i<-a\end{cases}$	30	[-50, 50]	MN
F 26	Langerman 2	$F_{\min}=-\sum_{i=1}^{D}c_i\left(\exp\left(-\frac{1}{\pi}\sum_{j=1}^{D}\left(x_j-a_{ij}\right)^2\right)\cos\left(\pi\sum_{j=1}^{D}\left(x_j-a_{ij}\right)^2\right)\right)$	2	[0, 10]	MN
F 27	Langerman 5	$F_{\min}=-\sum_{i=1}^{D}c_i\left(\exp\left(-\frac{1}{\pi}\sum_{j=1}^{D}\left(x_j-a_{ij}\right)^2\right)\cos\left(\pi\sum_{j=1}^{D}\left(x_j-a_{ij}\right)^2\right)\right)$	5	[0, 10]	MN
F 28	Langerman 10	$F_{\min}=-\sum_{i=1}^{D}c_i\left(\exp\left(-\frac{1}{\pi}\sum_{j=1}^{D}\left(x_j-a_{ij}\right)^2\right)\cos\left(\pi\sum_{j=1}^{D}\left(x_j-a_{ij}\right)^2\right)\right)$	10	[0, 10]	MN
F 29	FletcherPowell 5	$F_{\min}=\sum_{i=1}^{D}\left(A_i-B_i\right)^2$, $A_i=\sum_{j=1}^{D}\left(a_{ij}\sin\alpha_j+b_{ij}\cos\alpha_j\right),\quad B_i=\sum_{j=1}^{D}\left(a_{ij}\sin x_j+b_{ij}\cos x_j\right)$	5	[-π, π]	MN
F 30	FletcherPowell 1	$F_{\min}=\sum_{i=1}^{D}\left(A_i-B_i\right)^2$, $A_i=\sum_{j=1}^{D}\left(a_{ij}\sin\alpha_j+b_{ij}\cos\alpha_j\right),\quad B_i=\sum_{j=1}^{D}\left(a_{ij}\sin x_j+b_{ij}\cos x_j\right)$	10	[-π, π]	MN

D: Dimension, C: Characteristic, U: Unimodal, M: Multimodal, S: Separable, N: Non-separable

Friedman rank test is used to rank the algorithms based on the results obtained by the algorithms. However, this test does not specify any statistical difference in the results and hence Holm-Sidak test is used to determine the statistical difference between the algorithms. Table 13 shows the Holm-Sidak test for the 'Best' and the 'Mean' solutions obtained for G01-G13 functions and Table 14 shows the

corresponding values for G14-G24 functions. The pairwise *p*-vaues obtained from the Holm-Sidak test for all the algorithms show the statistical difference between the proposed Jaya algorithm and the other algorithms. The statistical difference between the proposed Jaya algorithm and the TLBO algorithm is smaller.

Table 16
Results obtained by the Jaya algorithm for 30 bench mark functions over 30 independent runs with 500000 function evaluations

No.	Function	Optimum	Best	Worst	Mean	SD
F 1	Sphere	0	0	0	0	0.00E+00
F 2	SumSquares	0	0	0	0	0.00E+00
F 3	Beale	0	0	0	0	0.00E+00
F 4	Easom	-1	-1	-1	-1	0.00E+00
F 5	Matyas	0	0	0	0	0.00E+00
F 6	Colville	0	0	0	0	0.00E+00
F 7	Trid 6	-50	-50	-50	-50	0.00E+00
F 8	Trid 10	-210	-210	-210	-210	0.00E+00
F 9	Zakharov	0	0	0	0	0.00E+00
F 10	Schwefel 1.2	0	0	0	0	0.00E+00
F 11	Rosenbrock	0	0	0	0	0.00E+00
F 12	Dixon-Price	0	0	0	0	0.00E+00
F 13	Foxholes	0.998	0.998004	0.998004	0.998004	0.00E+00
F 14	Branin	0.398	0.397887	0.397887	0.397887	0.00E+00
F 15	Bohachevsky 1	0	0	0	0	0.00E+00
F 16	Booth	0	0	0	0	0.00E+00
F 17	Michalewicz 2	-1.8013	-1.801303	-1.801303	-1.801303	0.00E+00
F 18	Michalewicz 5	-4.6877	-4.687658	-4.645895	-4.680138	1.58E-02
F 19	Bohachevsky 2	0	0	0	0	0.00E+00
F 20	Bohachevsky 3	0	0	0	0	0.00E+00
F 21	GoldStein-Price	3	3	3	3	0.00E+00
F 22	Perm	0	0	0	0	0.00E+00
F 23	Hartman 3	-3.86	-3.862780	-3.862780	-3.862780	0.00E+00
F 24	Ackley	0	0	0	0	0.00E+00
F 25	Penalized 2	0	0	0	0	0.00E+00
F 26	Langerman 2	-1.08	-1.080938	-1.080938	-1.080938	0.00E+00
F 27	Langerman 5	-1.5	-1.5	-0.482871	-1.240500	0.312691
F 28	Langerman 10	NA	-1.5	-0.278661	-0.620503	0.319588
F 29	FletcherPowell 5	0	0	0.002794	0.000159	5.2E-04
F 30	FletcherPowell 10	0	0	5.33E-03	5.43E-04	9.87E-04

SD = Standard deviation

5. Experiments on unconstrained benchmark problems

The performance of the proposed Jaya algorithm is tested further on 30 unconstrained benchmark functions well documented in the optimization literature. These unconstrained functions have different characteristics like unimodality/multimodality, separability/non-separability, regularity/non-regularity, etc. The number of design variables and their ranges are different for each problem. Table 15 shows the 30 unconstrained benchmark functions.To evaluate the performance of the proposed Jaya algorithm, the results obtained by using the Jaya algorithm are compared with the results obtained by the other optimization algorithms such as GA, PSO, DE, ABC and TLBO. A common platform is provided by maintaining the identical function evaluations for different algorithms considered for the comparison. Thus, the consistency in the comparison is maintained while comparing the performance of Jaya algorithm with other optimization algorithms. However, in general, the algorithm which requires less number of function evaluations to get the same best solution can be considered as better as compared to the other algorithms. However, in this paper, to maintain the consistency in the comparison of competitive algorithms, a common experimental platform is provided by setting the maximum number of function evaluations as 500000 for each benchmark function. Just like other algorithms, the proposed Jaya algorithm is executed 30 times for each benchmark function by choosing suitable population sizes and the mean results obtained are compared with the other algorithms for the same number of runs.

Table 17
Comparative results of Jaya algorithm with other algorithms over 30 independent runs

		GA	PSO	DE	ABC	TLBO	Jaya			GA	PSO	DE	ABC	TLBO	Jaya
F1	M	1.11E+03	0	0	0	0	0	F13	M	0.998004	0.9980039	0.9980039	0.9980039	0.9980039	0.998004
	SD	74.214474	0	0	0	0	0.00E+00		SD	0	0	0	0	0	0.00E+00
F2	M	1.48E+02	0	0	0	0	0	F14	M	0.397887	0.3978874	0.3978874	0.3978874	0.3978874	0.397887
	SD	12.409289	0	0	0	0	0.00E+00		SD	0	0	0	0	0	0.00E+00
F3	M	0	0	0	0	0	0	F15	M	0	0	0	0	0	0
	SD	0	0	0	0	0	0.00E+00		SD	0	0	0	0	0	0.00E+00
F4	M	-1	-1	-1	-1	-1	-1	F16	M	0	0	0	0	0	0
	SD	0	0	0	0	0	0.00E+00		SD	0	0	0	0	0	0.00E+00
F5	M	0	0	0	0	0	0	F17	M	-1.8013	-1.5728692	-1.801303	-1.8013034	-1.801303	-1.801303
	SD	0	0	0	0	0	0.00E+00		SD	0.00E+00	0.11986	0	0	0	0.00E+00
F6	M	0.014938	0	0.0409122	0.0929674	0	0	F18	M	-4.64483	-2.4908728	-4.683482	-4.6876582	-4.6726578	-4.680138
	SD	0.007364	0	0.081979	0.066277	0	0.00E+00		SD	0.09785	0.256952	0.012529	0.00E+00	4.74E-02	1.58E-02
F7	M	-49.9999	-50	-50	-50	-50	-50	F19	M	0.06829	0.00	0.00	0.00	0.00	0
	SD	2.25E–5	0	0	0	0	0.00E+00		SD	0.078216	0.00	0.00	0.00	0.00	0.00E+00
F8	M	0.193417	0	0	0	0	-210	F20	M	0.00	0.00	0.00	0.00	0.00	0
	SD	0.035313	0	0	0	0	0.00E+00		SD	0.00	0.00	0.00	0.00	0.00	0.00E+00
F9	M	0.013355	0	0	0.0002476	0	0	F21	M	5.870093	3	3	3	3	3
	SD	0.004532	0	0	0.000183	0	0.00E+00		SD	1.071727	0	0	0	0	0.00E+00
F10	M	7.40E+03	0	0	0	0	0	F22	M	0.302671	0.0360516	0.0240069	0.0411052	0.0006766	0
	SD	1.14E+03	0	0	0	0	0.00E+00		SD	0.193254	0.048927	0.046032	0.023056	0.0007452	0.00E+00
F11	M	1.96E+05	15.088617	18.203938	0.0887707	1.62E-05	0	F23	M	-3.86278	-3.6333523	-3.862782	-3.8627821	-3.862782	-3.862780
	SD	3.85E+04	24.170196	5.036187	0.07739	3.64E-05	0.00E+00		SD	0.00E+00	0.116937	0	0	0	0.00E+00
F12	M	1.22E+03	0.6666667	0.6666667	0	0.6666667	0	F24	M	14.67178	0.1646224	0	0	0	0
	SD	2.66E+02	E–8	E–9	0	0	0.00E+00		SD	0.178141	0.493867	0	0	0	0.00E+00
F25	M	125.0613	0.0076754	0.0021975	0	2.34E-08	0	F28	M	-0.63644	-0.0025656	-1.0528	-0.4460925	-0.64906	-0.620503
	SD	12.0012	0.016288	0.004395	0	0	0.00E+00		SD	0.374682	0.003523	0.302257	0.133958	0.1728623	0.319588
F26	M	-1.08094	-0.679268	-1.080938	-1.0809384	-1.080938	-1.080938	F29	M	0.004303	1457.8834	5.988783	0.1735495	2.2038134	0.000159
	SD	0	0.274621	0	0	0	0.00E+00		SD	0.009469	1269.3624	7.334731	0.068175	4.3863209	5.2E-04
F27	M	0.287548	0.213626	0	0.000208	1.55E-05	-1.240500	F30	M	29.57348	1364.4556	781.55028	8.2334401	35.971004	5.43E-04
	SD	0.052499	0.039003	0	3.80E–05	2.83E-06	0.312691		SD	16.02108	1325.3797	1048.8135	8.092742	71.284369	9.87E-04

5.1. Results and discussion on unconstrained benchmark functions

The results of Jaya algorithm corresponding to each benchmark function are presented in Table 16 in the form of best solution, worst solution, mean solution and standard deviation obtained in 30 independent runs on each benchmark function. The performance of Jaya algorithm is compared with the other well-known optimization algorithms such as GA, PSO, DE, ABC and TLBO and the results are given in Table 17. The results of GA, PSO, DE and ABC are taken from Karaboga and Akay (2009) and the results of TLBO are taken from Rao and Patel (2013) where the authors had experimented benchmark functions each with 500000 function evaluations with best setting of algorithm-specific parameters (except for TLBO for which, like Jaya algorithm, there are no algorithm-specific parameters). It can be observed from Tables 16 and 17 that the proposed Jaya algorithm has obtained better results in terms of "best', "mean' and 'worst' values of each objective function and 'standard deviation'. Furthermore, it can be observed that the performance of the Jaya algorithm is found either equal or better than all other algorithms in all the 30 unconstrained benchmark functions.

6. Conclusions

All the evolutionary and swarm intelligence based algorithms require proper tuning of algorithm-specific parameters in addition to tuning of common controlling parameters. A change in the tuning of the algorithm-specific parameters influences the effectiveness of the algorithm. The recently proposed TLBO algorithm does not require any algorithm-specific parameters and it only requires the tuning of the common controlling parameters of the algorithm for its working. Keeping in view of the success of the TLBO algorithm in the field of optimization, another algorithm-specific parameter-less algorithm is proposed in this paper and is named as 'Jaya algorithm'. However, unlike two phases (i.e. teacher phase and the learner phase) of the TLBO algorithm, the proposed algorithm has only one phase and it is comparatively simpler to apply.

The proposed algorithm is implemented on 24 well defined constrained optimization problems having different characteristics given in CEC 2006 competition. The results obtained by the proposed Jaya algorithm are compared with the results of well-known optimization algorithms such as HM, ASCHEA, SMES, GA, PSO, DE, ABC, BBO, HTS and TLBO algorithms for the considered constrained benchmark problems. Results have shown the satisfactory performance of Jaya algorithm for the constrained optimization problems. The statistical tests have also supported the performance supremacy of the proposed method.

The proposed algorithm is also implemented on 30 well defined unconstrained optimization problems documented in the optimization literature. These unconstrained optimization problems have different characteristics and the results obtained by the proposed Jaya algorithm are compared with the results of well-known optimization algorithms such as GA, PSO, DE, ABC and TLBO algorithms. Results have shown the satisfactory performance of Jaya algorithm for the considered unconstrained optimization problems. It may be concluded that the proposed Jaya algorithm can be used for solving the constrained as well as unconstrained optimization algorithms.

It is emphasized here that the proposed Jaya algorithm is not claimed as the 'best' algorithm among all the optimization algorithms available in the literature. In fact, there may not be any such 'best' algorithm existing for all types and varieties of problems! However, the Jaya algorithm is a newly proposed algorithm and is believed to have strong potential to solve the constrained and unconstrained optimization problems. If the algorithm is found having certain limitations then the efforts of the researchers should be to find out the ways to overcome the limitations and to further strengthen the algorithm. The efforts should not be in the form of destructive criticism. What can be said with more confidence at present about the Jaya algorithm is that it is simple to apply, it has no algorithm-specific parameters and it provides the optimum results in comparatively less number of function evaluations. Researchers are

encouraged to make improvements to the Jaya algorithm so that the algorithm can become much more powerful with much improved performance. It is hoped that the researchers belonging to different disciplines of engineering and sciences (physical, life and social) will find the Jaya algorithm as a powerful tool to optimize the systems and processes. Readers interested to get the MATLAB code of the Jaya algorithm may contact the author. Readers interested to get the MATLAB code of the Jaya algorithm may refer to https://sites.google.com/site/jaya-algorithm/

Acknowledgement

The author gratefully acknowledges the support of his present Ph.D. students (particularly, Mr. Dhiraj P. Rai for helping him in executing the code and Mr. G. G. Waghmare for providing the corrected TLBO results).

References

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandanavian Journal of Statistics*, 6(2), 65-70.

Joaquin, D., Salvador, G., Daniel, M., & Francisco, H. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1), 3–18.

Karaboga, D., &Basturk, B. (2007). Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems, LNAI, SpringerVerlag Berlin, 4529, 789-798.

Karaboga, D., &Akay, B. (2009). A comparative study of Artificial Bee Colony algorithm. *Applied Mathematics and Computation,* 214(1) 108-132.

Karaboga, D., &Akay, B. (2011). A modified Artificial Bee Colony (ABC) algorithm for constrained optimization problems. *Applied Soft Computing*, 11, 3021-3031.

Liang, J.J., Runarsson, T.P., Mezura-Montes, E., Clerc, M., Suganthan, P.N., Coello, C.A.C., & Deb, K. (2006). Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session On Constrained Real-Parameter Optimization, *Technical Report, Nanyang Technological University, Singapore, http://www.ntu.edu.sg/home/EPNSugan.*

Patel, V.K., &Savsani, V.J. (2015). Heat transfer search (HTS): a novel optimization algorithm. *Information Sciences*, 324, 217-246.

Rao, R.V. (2015). Teaching Learning Based Optimization And Its Engineering Applications. Springer Verlag, London.

Rao, R.V., & Patel, V. (2012). An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems. *International Journal of Industrial Engineering Computations*, 3(4), 535-560.

Rao, R.V., & Patel, V. (2013). Comparative performance of an elitist teaching-learning-based optimization algorithm for solving unconstrained optimization problems. *International Journal of Industrial Engineering Computations,* 4(1), 29-50.

Rao, R.V., Savsani, V.J. &Vakharia, D.P. (2011). Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design,* 43(3), 303-315.