Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Rao algorithms: Three metaphor-less simple algorithms for solving optimization problems

Ravipudi Venkata Rao[a*]

[a]*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 June 1 2019
Received in Revised Format
June 4 2019
Accepted June 9 2019
Available online
July 7 2019

Keywords:
Metaphor-less algorithms
Optimization
Benchmark functions

A B S T R A C T

Three simple metaphor-less optimization algorithms are developed in this paper for solving the unconstrained and constrained optimization problems. These algorithms are based on the best and worst solutions obtained during the optimization process and the random interactions between the candidate solutions. These algorithms require only the common control parameters like population size and number of iterations and do not require any algorithm-specific control parameters. The performance of the proposed algorithms is investigated by implementing these on 23 benchmark functions comprising 7 unimodal, 6 multimodal and 10 fixed-dimension multimodal functions. Additional computational experiments are conducted on 25 unconstrained and 2 constrained optimization problems. The proposed simple algorithms have shown good performance and are quite competitive. The research community may take advantage of these algorithms by adapting the same for solving different unconstrained and constrained optimization problems.

1. Introduction

In recent years the field of population based meta-heuristic algorithms is flooded with a number of 'new' algorithms based on *metaphor* of some natural phenomena or behavior of animals, fishes, insects, societies, cultures, planets, musical instruments, etc. Many new optimization algorithms are coming up every month and the authors claim that the proposed algorithms are 'better' than the other algorithms. Some of these newly proposed algorithms are dying naturally as there are no takers and some have received success to some extent. However, this type of research may be considered as a threat and may not contribute to advance the field of optimization (Sorensen, 2015). *It would be better if the researchers focus on developing simple optimization techniques that can provide effective solutions to the complex problems instead of looking for developing metaphor based algorithms*. Keeping this point in view, three simple metaphor-less and algorithm-specific parameter-less optimization algorithms are developed in this paper. The next section describes the proposed algorithms.

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

2020 Growing Science Ltd.
doi: 10.5267/j.ijiec.2019.6.002

2. Proposed algorithms

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, '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 equations.

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

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

$$X'_{j,k,i} = X_{j,k,i} + r_{1,j,i}\,(X_{j,best,i} - |\,X_{j,worst,i}\,|) + r_{2,j,i}\,(\,|\,X_{j,k,i} \text{ or } X_{j,l,i}\,| - (X_{j,l,i} \text{ or } X_{j,k,i})), \tag{3}$$

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 during the i^{th} iteration. $X'_{j,k,i}$ 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].

In Eqs.(2) and (3), the term $X_{j,k,i}$ or $X_{j,l,i}$ indicates that the candidate solution k is compared with any randomly picked candidate solution l and the information is exchanged based on their fitness values. If the fitness value of kth solution is better than the fitness value of lth solution then the term "$X_{j,k,i}$ or $X_{j,l,i}$" becomes $X_{j,k,i}$. On the other hand, if the fitness value of lth solution is better than the fitness value of kth solution then the term "$X_{j,k,i}$ or $X_{j,l,i}$" becomes $X_{j,l,i}$. Similarly, if the fitness value of kth solution is better than the fitness value of lth solution then the term "$X_{j,l,i}$ or $X_{j,k,i}$" becomes $X_{j,l,i}$. If the fitness value of lth solution is better than the fitness value of kth solution then the term "$X_{j,l,i}$ or $X_{j,k,i}$" becomes $X_{j,k,i}$.

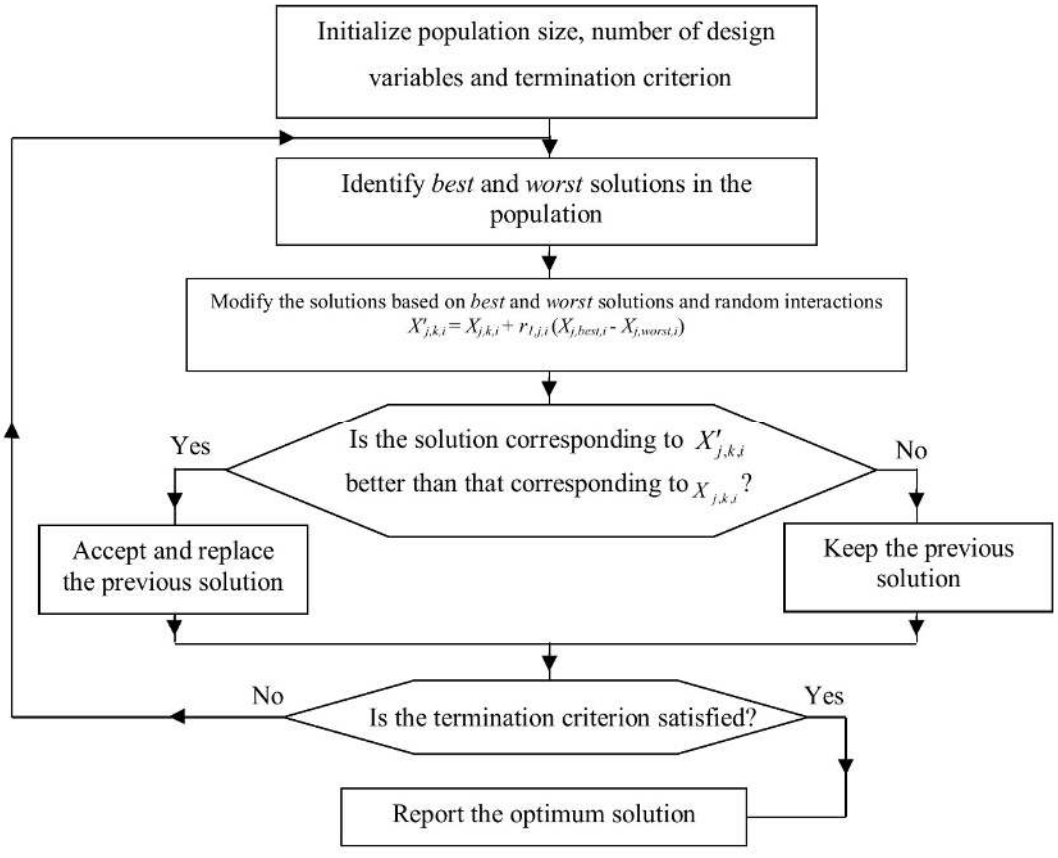

Fig. 1. Flowchart of Rao-1 algorithm

These three algorithms are based on the *best* and *worst* solutions in the population and the random interactions between the candidate solutions. Just like TLBO algorithm (Rao, 2015) and Jaya algorithm (Rao, 2016; Rao, 2019), these algorithms do not require any algorithm-specific parameters and thus the designer's burden to tune the algorithm-specific parameters to get the best results is eliminated. These algorithms are named as Rao-1, Rao-2 and Rao-3 respectively. Fig. 1 shows the flowchart of Rao-1 algorithm. The flowchart will be same for Rao-2 and Rao-3 algorithms except that the Eq. (1) shown in the flowchart will be replaced by Eq. (2) and Eq. (3) respectively. The proposed algorithms are illustrated by means of an unconstrained benchmark function known as Sphere function.

2.1 Demonstration of the working of proposed Rao-1 algorithm

To demonstrate the working of proposed algorithms, 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.

Benchmark function: Sphere

$$\min f(x_i) = \sum_{i=1}^{n} x_i^2$$

Range of variables: $-100 \leq x_i \leq 100$ (4)

The known solution to this benchmark function is 0 for all x_i values of 0. Now to demonstrate the proposed algorithms, 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	33	1285	*Worst*
3	30	-6	936	
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 2nd candidate. Using the initial solutions of Table 1 and assuming random number $r_1 = 0.10$ for x_1 and $r_2 = 0.50$ for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(1) and 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.

$X'_{1,1,1} = X_{1,1,1} + r_{1,1,1}(X_{1,4,1} - X_{1,2,1}) = -5 + 0.10\ (-8-14) = -7.2,$
$X'_{2,1,1} = X_{2,1,1} + r_{2,1,1}(X_{2,4,1} - X_{2,2,1}) = 18 + 0.50\ (7-33) = 5.$

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 (Rao-1)

Candidate	x_1	x_2	*f*(*x*)
1	-7.2	5	76.84
2	11.8	20	539.24
3	27.8	-19	1133.84
4	-10.2	-6	140.04
5	-14.2	-31	1162.64

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

Table 3
Updated values of the variables and the objective function based on fitness comparison at the end of first iteration (Rao-1)

Candidate	x_1	x_2	*f*(*x*)	*Status*
1	-7.2	5	76.84	*best*
2	11.8	20	539.24	
3	30	-6	936	*worst*
4	-8	7	113	
5	-12	-18	468	

From Table 3 it can be seen that the best solution is corresponding the 1st candidate and the worst solution is corresponding to the 3rd candidate. In the first iteration, the value of the objective function is improved from 113 to 76.84 and the worst value of the objective function is improved from 1285 to 936. Now, assuming random number $r_1 = 0.80$ for x_1 and $r_2 = 0.1$ 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 4. Table 4 shows the corresponding values of the objective function also.

Table 4
New values of the variables and the objective function during second iteration (Rao-1)

Candidate	x_1	x_2	*f*(*x*)
1	-36.96	6.1	1403.2516
2	-17.96	21.1	767.7716
3	0.24	-4.9	24.0676
4	-37.76	8.1	1491.4276
5	-41.76	-16.9	2029.5076

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 Rao-1 algorithm.

Table 5
Updated values of the variables and the objective function based on fitness comparison at the end of second iteration (Rao-1)

Candidate	x_1	x_2	*f*(*x*)	*Status*
1	-7.2	5	76.84	
2	11.8	20	539.24	*worst*
3	0.24	-4.9	24.0676	*best*
4	-8	7	113	
5	-12	-18	468	

It can be observed that at the end of second iteration, the value of the objective function is improved from 113 to 24.0676 and the worst value of the objective function is improved from 1285 to 539.24. 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. This 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 for violation of each constraint and the penalty value is operated upon the objective function.

2.2 Demonstration of the working of proposed Rao-2 algorithm

Using the initial solutions of Table 1, and assuming random numbers $r_1 = 0.10$ and $r_2 = 0.50$ for x_1 and $r_1 = 0.60$ and $r_2 = 0.20$ for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(2) and

placed in Table 6. For example, for the 1st candidate, the new values of x_1 and x_2 during the first iteration are calculated as shown below. Here the 1st candidate has interacted with the 2nd candidate. The fitness value of the 1st candidate is better than the fitness value of the 2nd candidate and hence the information exchange is from 1st candidate to 2nd candidate.

$$X'_{1,1,1} = X_{1,1,1} + r_{1,1,1}(X_{1,4,1} - X_{1,2,1}) + r_{2,1,1}(|X_{1,1,1}| - |X_{1,2,1}|)$$
$$= -5 + 0.10\,(-8-14) + 0.50\,(5-14) = -11.7$$
$$X'_{2,1,1} = X_{2,1,1} + r_{1,2,1}(X_{2,4,1} - X_{2,2,1}) + r_{2,2,1}(|X_{2,1,1}| - |X_{2,2,1}|)$$
$$= 18 + 0.60\,(7-33) + 0.20\,(18-33) = -0.6$$

Similarly, the new values of x_1 and x_2 for the other candidates are calculated. Here the random interactions are taken as 2 vs. 5, 3 vs. 1, 4 vs. 2 and 5 vs. 4. Table 6 shows the new values of x_1 and x_2 and the corresponding values of the objective function.

Table 6
New values of the variables and the objective function during first iteration (Rao-2)

Candidate	x_1	x_2	$f(x)$
1	-11.7	-0.6	137.25
2	10.8	14.4	324
3	15.3	-19.2	602.73
4	-13.2	-13.8	364.68
5	-16.2	-35.8	1544.08

Now, the values of *f(x)* of Table 1 and Table 6 are compared and the best values of *f(x)* are considered and placed in Table 7. This completes the first iteration of the Rao-2 algorithm.

Table 7
Updated values of the variables and the objective function based on fitness comparison at the end of first iteration (Rao-2)

Candidate	x_1	x_2	$f(x)$	*Status*
1	-11.7	-0.6	137.25	
2	10.8	14.4	324	
3	15.3	-19.2	602.73	*worst*
4	-8	7	113	*best*
5	-12	-18	468	

From Table 7 it can be seen that the best solution is corresponding the 4th candidate and the worst solution is corresponding to the 3rd candidate. Now, during the second iteration, assuming random numbers r_1 = 0.01 and r_2 = 0.10 for x_1 and r_1 = 0.10 and r_2 = 0.50 for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(2). Here the random interactions are taken as 1 vs. 4, 2 vs. 3, 3 vs. 5, 4 vs. 2 and 5 vs. 1. Table 8 shows the new values of x_1 and x_2 and the corresponding values of the objective function during the second iteration.

Table 8
New values of the variables and the objective function during second iteration (Rao-2)

Candidate	x_1	x_2	$f(x)$
1	-12.303	5.22	178.612
2	10.117	14.62	316.098
3	14.737	-17.18	512.331
4	-8.513	5.92	107.517
5	-12.263	-24.08	730.227

Now, the values of *f(x)* of Tables 7 and 8 are compared and the best values of *f(x)* are considered and placed in Table 9. This completes the second iteration of the Rao-2 algorithm.

Table 9
Updated values of the variables and the objective function based on fitness comparison at the end of second iteration (Rao-2)

Candidate	x_1	x_2	$f(x)$	*Status*
1	-11.7	-0.6	137.25	
2	10.117	14.62	316.098	
3	14.737	-17.18	512.331	*worst*
4	-8.513	5.92	107.517	*best*
5	-12	-18	468	

From Table 9 it can be seen that the best solution is corresponding the 2nd candidate and the worst solution is corresponding to the 5nd candidate. It can be observed that the value of the objective function is improved from 113 to 107.517 in two iterations. Similarly, the worst value of the objective function is improved from 1285 to 512.331 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, just like Rao-1, the proposed Rao-2 can deal with both unconstrained and constrained minimization as well as maximization problems.

2.3 Demonstration of the working of proposed Rao-3 algorithm

Now assuming random numbers $r_1 = 0.10$ and $r_2 = 0.50$ for x_1 and $r_1 = 0.60$ and $r_2 = 0.20$ for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(3) and placed in Table 10. For example, for the 1st candidate, the new values of x_1 and x_2 during the first iteration are calculated as shown below. Here the 1st candidate has interacted with the 2nd candidate. The fitness value of the 1st candidate is better than the fitness value of the 2nd candidate and hence the information exchange is from 1st candidate to 2nd candidate.

$$X'_{1,1,1} = X_{1,1,1} + r_{1,1,1}(X_{1,4,1} - |X_{1,2,1}|) + r_{2,1,1}(|X_{1,1,1}| - X_{1,2,1})$$
$$= -5 + 0.10\,(-8-14) + 0.50\,(5-14) = -11.7$$
$$X'_{2,1,1} = X_{2,1,1} + r_{1,2,1}(X_{2,4,1} - |X_{2,2,1}|) + r_{2,2,1}(|X_{2,1,1}| - X_{2,2,1})$$
$$= 18 + 0.60\,(7-33) + 0.20\,(18-33) = -0.6$$

Similarly, the new values of x_1 and x_2 for the other candidates are calculated. Here the random interactions are taken as 2 vs. 5, 3 vs. 1, 4 vs. 2 and 5 vs. 4. Table 10 shows the new values of x_1 and x_2 and the corresponding values of the objective function.

Table 10
New values of the variables and the objective function during first iteration (Rao-3)

Candidate	x_1	x_2	$f(x)$
1	-11.7	-0.6	137.25
2	10.8	14.4	324
3	15.3	-16.8	516.33
4	-13.2	-13.8	364.68
5	-4.2	-28.6	835.6

Now, the values of *f(x)* of Tables 1 and 10 are compared and the best values of *f(x)* are considered and placed in Table 11. This completes the first iteration of the Rao-3 algorithm. From Table 11 it can be seen that the best solution is corresponding the 4th candidate and the worst solution is corresponding to the 3rd candidate. Now, during the second iteration, assuming random numbers $r_1 = 0.01$ and $r_2 = 0.10$ for x_1 and $r_1 = 0.10$ and $r_2 = 0.50$ for x_2, the new values of the variables for x_1 and x_2 are calculated using Eq.(3). Here the random interactions are taken as 1 vs. 4, 2 vs. 3, 3 vs. 5, 4 vs. 2 and 5 vs. 1. Table 12

shows the new values of x_1 and x_2 and the corresponding values of the objective function during the second iteration.

Table 11
Updated values of the variables and the objective function based on fitness comparison at the end of first iteration (Rao-3)

Candidate	x_1	x_2	$f(x)$	*Status*
1	-11.7	-0.6	137.25	
2	10.8	14.4	324	
3	15.3	-16.8	516.33	*worst*
4	-8	7	113	*best*
5	-12	-18	468	

Table 12
New values of the variables and the objective function during second iteration (Rao-3)

Candidate	x_1	x_2	$f(x)$
1	-9.963	2.22	104.189
2	10.117	29.02	944.514
3	14.737	-0.38	217.323
4	-8.513	2.32	77.853
5	-9.863	-9.68	190.981

Now, the values of *f(x)* of Tables 11 and 12 are compared and the best values of *f(x)* are considered and placed in Table 13. This completes the second iteration of the Rao-3 algorithm.

Table 13
Updated values of the variables and the objective function based on fitness comparison at the end of second iteration (Rao-3)

Candidate	x_1	x_2	$f(x)$	*Status*
1	-9.963	2.22	104.189	
2	10.8	14.4	324	*worst*
3	-14.737	-0.38	217.323	
4	-8.513	2.32	77.853	*best*
5	-9.863	-9.68	190.981	

From Table 13 it can be seen that the best solution is corresponding the 2nd candidate and the worst solution is corresponding to the 5nd candidate. It can be observed that the value of the objective function is improved from 113 to 77.853 in just two iterations. Similarly, the worst value of the objective function is improved from 1285 to 324 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, just like Rao-1 and Rao-2, the proposed Rao-3 can also deal with both unconstrained and constrained minimization as well as maximization problems. It may be noted that the above three demonstrations with random numbers are just to make the readers familiar with the working of the proposed algorithms. While executing the algorithms different random numbers will be generated during different iterations and the computations will be done accordingly. The next section deals with the experimentation of the proposed algorithms on the benchmark optimization problems.

3. Computational experiments on unimodal, multi-modal and fixed-dimension multimodal optimization problems

The computational experiments are first conducted on 23 benchmark functions including 7 unimodal, 6 multimodal and 10 fixed-dimension multimodal functions. Table 14 shows these benchmark functions.

Table 14
Unimodal, multimodal and fixed-dimension multimodal functions (Mirjalili, 2014)

Sr. No	*Function*	*D*	*Range*	f_{min}
1	$f_1(x)=\sum_{i=1}^{n} x_i^2$	30	[-100,100]	0
2	$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[-10,10]	0
3	$f_3(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	30	[-100,100]	0
4	$f_4(x)=\max_i\left\{\lvert x_i\rvert,\ 1\le i\le n\right\}$	30	[-100,100]	0
5	$f_5(x)=\sum_{i=1}^{n-1}\left[100\left(x_{i+1}-x_i^2\right)^2+\left(x_i-1\right)^2\right]$	30	[-30,30]	0
6	$f_6(x)=\sum_{i=1}^{n}\left(\lvert x_i+0.5\rvert\right)^2$	30	[-100,100]	0
7	$f_7(x)=\sum_{i=1}^{n} i x_i^4+random[0,1)$	30	[-1.28,1.28]	0
8	$f_8(x)=\sum_{i=1}^{n} -x_i\sin\left(\sqrt{\lvert x_i\rvert}\right)$	30	[-500,500]	-418.9829 ×30
9	$f_9(x)=\sum_{i=1}^{n}\left[x_i^2-10\cos(2\pi x_i)+10\right]$	30	[-5.12,5.12]	0
10	$f_{10}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	30	[-32,32]	0
11	$f_{11}(x)=\frac{1}{4000}\sum_{i=1}^{n} x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	30	[-600,600]	0
12	$f_{12}(x)=\frac{\pi}{n}\left\{10\sin^2(\pi.y)+\sum_{i=1}^{n-1}(y_i-1)^2\left[1+10\sin^2(\pi.y_{i+1})\right]+(y_n-1)^2\right\}+\sum_{i=1}^{n} u(x_i,10,100,4)$ $y_i=1+\left(\frac{x_i+1}{4}\right)$ $u(x_i,a.k,m)=\begin{cases}k(x_i-a)^m & x_i>a\\ 0 & -a<x_i<a\\ k(-x_i-a)^m & x_i<-a\end{cases}$	30	[-50,50]	0
13	$f_{13}(x)=0.1\left\{\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right]+(x_n-1)^2\left[1+\sin^2(2\pi x_n)\right]\right\}+\sum_{i=1}^{n} u(x_i,5,100,4)$	30	[-50,50]	0
14	$f_{14}(x)=\left(\frac{1}{500}+\sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}\left(x_i-a_{ij}\right)^6}\right)^{-1}$	2	[-65,65]	0.998
15	$f_{15}(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1\left(b_i^2+b_i x_2\right)}{b_i^2+b_i x_3+x_4}\right]^2$	4	[-5,5]	0.0003

16	$f_{16}(x)=4x_1^2+2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	2	[-5,5]	-1.0316
17	$f_{17}(x)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10$	2	[-5,5]	0.398
18	$f_{18}(x)=\left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]\times\left[30+(2x_1-3x_2)^2\times\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2\right)\right]$	2	[-2,2]	3
19	Hartman 3 $f_{19}(x)=-\sum_{i=1}^{4}C_i\exp\left(-\sum_{j=1}^{3}a_{ij}\left(x_j-P_{ij}\right)^2\right)$	3	[0,1]	-3.86
20	Hartman 6 $f_{20}(x)=-\sum_{i=1}^{4}C_i\exp\left(-\sum_{j=1}^{6}a_{ij}\left(x_j-P_{ij}\right)^2\right)$	6	[0,1]	-3.32
21	Shekel 5 $f_{21}(x)=-\sum_{i=1}^{5}\left[\sum_{j=1}^{4}\left(x_j-a_{ij}\right)^2+c_i\right]^{-1}$	4	[0,10]	-10.1532
22	Shekel 7 $f_{22}(x)=-\sum_{i=1}^{7}\left[\sum_{j=1}^{4}\left(x_j-a_{ij}\right)^2+c_i\right]^{-1}$	4	[0,10]	-10.4029
23	Shekel 10 $f_{23}(x)=-\sum_{i=1}^{10}\left[\sum_{j=1}^{4}\left(x_j-a_{ij}\right)^2+c_i\right]^{-1}$	4	[0,10]	-10.5364

D: Dimensions (i.e., no. of design variables); f_{min}: Global optimum value

The benchmark functions 1-7 are the unimodal functions (for checking the exploitation capability of the algorithms), 8-13 are the multimodal functions that have many local optima which increase with the increase in the number of dimensions (for checking the exploration capability of the algorithms) and 14-23 are the fixed-dimension multimodal benchmark functions (for checking the exploration capability of the algorithms in the case of fixed dimension optimization problems). The global optimum values of the benchmark functions are also given in Table 15 to give an idea to the readers about the performances of the proposed algorithms.

The performance of the proposed algorithms is tested on the 23 benchmark functions listed in Table 14. To evaluate the performance of the proposed algorithms, a common experimental platform is provided by setting the maximum number of function evaluations as 30000 for each benchmark function with 30 runs for each benchmark function. The results of each benchmark function are presented in Table 15 in the form of best solution, worst solution, mean solution, standard deviation obtained in 30 independent runs, mean function evaluations, and the population size used for each benchmark function. The results of the proposed algorithms are compared with the already established Grey Wolf Optimization (GWO) algorithm (Mirjalili, 2014) and Ant Lion Optimization (ALO) algorithm (Mirjalili, 2015).

It may be mentioned here that the GWO algorithm was already shown competitive to the other advanced optimization algorithms like particle swarm optimization (PSO), gravitational search algorithm (GSA), differential evolution (DE) and fast evolutionary programming (FEP) (Mirjalili, 2014). The ALO algorithm was also shown competitive to PSO, states of matter search (SMS), bat algorithm (BA), flower pollination algorithm (FPA), cuckoo search (CS) and firefly algorithm (FA) (Mirjalili, 2015). Hence in this paper the results of other advanced optimization algorithms are not shown. The GWO algorithm was used for solving 23 benchmark functions (Mirjalili, 2014) and ALO was used for solving 13 benchmark functions (Mirjalili, 2014). The results of application of the proposed algorithms are shown in Table 15. Mirjalili (2014, 2015) had shown the results of only mean solutions and standard deviations. However, the results of the proposed algorithms are presented in Table 15 in terms of the best (B), worst (W), mean (M), standard deviation (SD), mean function evaluations (MFE) and the population size (P) used for obtaining the results within the maximum function evaluations of 30000. The values shown in bold in Table 15 indicate the comparatively better mean results of the respective algorithms.

Table 15
Results of the proposed algorithms for 23 benchmark functions considered (30000 function evaluations)

Func.	f_{min}		*GWO (Mirjalili, 2014)*	*ALO (Mirjalili, 2015)*	*Rao-1*	*Rao-2*	*Rao-3*
1	0	B			4.84E-25	1.40E-15	1.58E-50
		W			3.28E-21	3.47E-11	6.29E-41
		M	6.59E-28	2.59E-10	3.59E-22	3.57E-12	**6.71E-42**
		SD	6.34E-05	1.65E-10	7.33E-22	7.95E-12	1.56E-41
		MFE			29998	29953	29991
		P			10	10	10
2	0	B			2.04E-15	0.000121792	6.32E-24
		W			7.60E-11	10.00121716	2.10E-19
		M	7.18E-17	1.84E-06	4.07E-12	0.678178098	**9.33E-21**
		SD	0.029014	6.58E-07	1.40E-11	2.534459078	3.84E-20
		MFE			29994	29882	29983
		P			10	20	20
3	0	B			5.31E-45	7.92E-29	4.93E-64
		W			1.35E-38	3.79E-15	5.00E-52
		M	3.29E-06	6.07E-10	8.34E-40	1.27E-16	**1.68E-53**
		SD	79.14958	6.34E-10	2.90E-39	6.93E-16	9.12E-53
		MFE			29993	29975	29959
		P			10	10	20
4	0	B			0.494772	5.742890	0.001209
		W			5.572192	29.514839	0.285619
		M	5.61E-07	**1.36E-08**	2.119522	16.563950	0.081469
		SD	1.315088	1.81E-09	1.150517	5.632224	0.078402
		MFE			29882	28845	29899
		P			30	20	20
5	0	B			0.403869	0.002873	0.006485
		W			108.778761	85.487340	88.373496
		M	26.81258	**0.346772**	31.604357	11.474080	29.206289
		SD	69.90499	0.109584	28.406665	16.683870	29.093295
		MFE			29609	28925	28922
		P			20	10	20
6	0	B			4.70E-25	3.27E-12	2.196020
		W			4.22E-20	1.41E-06	3.680173
		M	0.816579	2.56E-10	**2.63E-21**	1.09E-07	2.919904
		SD	0.000126	1.09E-10	7.87E-21	3.09E-07	0.399770
		MFE			29993	29945	20023
		P			10	10	30
7	0	B			0.029805	0.018737	0.004610
		W			0.132753	0.234932	0.038987
		M	**0.002213**	0.004292	0.058328	0.087804	0.015770
		SD	0.100286	0.005089	0.027453	0.044495	0.008669
		MFE			26785	25354.66667	24044
		P			20	20	30

Table 15
Results of the proposed algorithms for 23 benchmark functions considered (30000 function evaluations)

Func.	f_{min}		*GWO (Mirjalili, 2014)*	*ALO (Mirjalili, 2015)*	*Rao-1*	*Rao-2*	*Rao-3*
8	-12569	B			-10250.82586	-12352.34695	-12135.20714
		W			-3879.49856	-5960.01496	-5751.10732
		M	-6123.1	-1606.276	-8685.17016	-8757.58136	**-9664.70182**
		SD	-4087.44*	314.4302	1690.54881	1896.34347	1544.65568
		MFE			21166	22377	28385
		P			10	10	20
9	0	B			25.868920	68.121702	29.889988
		W			183.605714	232.791997	197.125802
		M	0.310521	**7.71E-06**	87.013555	148.949496	84.122877
		SD	47.35612	8.45E-06	32.317490	41.526656	38.179200
		MFE			26015	24754	27934
		P			10	10	10
10	0	B			4.41E-07	1.43E-02	7.57E-10
		W			2.131898	1.350810	3.24E-07
		M	1.06E-13	**3.73E-15**	0.619739	0.170688	7.97E-08
		SD	0.077835	1.50E-15	0.695792	0.318320	8.69E-08
		MFE			29929	29881	29919
		P			40	20	50
11	0	B			3.90E-13	4.44E-15	0
		W			0.063900	0.243692	0.162637
		M	**0.004485**	0.018604	0.011455	0.044885	0.028906
		SD	0.006659	0.009545	0.014397	0.066572	0.042806
		MFE			29971	29406	21654
		P			20	10	20
12	0	B			1.48E-14	0.000165	0.314068
		W			6.639524	27.399757	1.820371
		M	0.053438	**9.75E-12*** **	1.549523	6.222186	0.791997
		SD	0.020734	9.33E-12	1.497920	7.075035	0.372832
		MFE			29957	28537	26432
		P			20	20	50
13	0	B			1.48E-06	3.12E-10	6.31E-13
		W			0.408911	2.301389	0.108359
		M	0.654464	**2.00E-11*** **	0.024281	0.458132	0.009724
		SD	0.004474	1.13E-11	0.078964	0.638728	0.026098
		MFE			29927	29996.33333	29947
		P			30	10	50
14	0.998	B			0.998004	0.998004	0.998004
		W			0.998004	0.998004	0.999089
		M	4.042493		**0.998004**	**0.998004**	0.998116
		SD	4.252799		8.25E-17	2.43E-08	2.51E-04
		MFE			12013	24069	14583
		P			20	20	50

Table 15
Results of the proposed algorithms for 23 benchmark functions considered (30000 function evaluations)

15	0.0003	B		0.00037651	0.000307486	0.000307489
		W		0.02036792	0.001667376	0.001656898
		M	**0.000337**	0.001429471	0.000665627	0.000485752
		SD	0.000625	0.003589047	0.000514761	0.000326366
		MFE		21826.66667	23386	21737
		P		100	20	30
16	-1.0316	B		-1.031628	-1.031628	-1.031628
		W		-1.031605	-1.031594	-1.031628
		M	**-1.03163**	-1.031627	-1.031626	-1.031628
		SD	-1.03163*	4.36E-06	7.39E-06	8.39E-08
		MFE		2577	4612	20283
		P		10	5	5
17	0.397887	B		0.397887	0.397887	0.397887
		W		0.397887	0.397887	0.397887
		M	0.397889	**0.397887**	**0.397887**	**0.397887**
		SD	0.397887	0	0	0
		MFE		995	695	692
		P		10	10	10
18	3	B		3	3	3
		W		3	3	3.000160
		M	3.000028	**3**	**3**	3.000021
		SD	3	9.00E-16	6.06E-16	3.30E-05
		MFE		10031	18098	22145.6
		P		10	20	10
19	-3.86	B		-3.86278	-3.86278	-3.86278
		W		-3.86278	-3.86278	-3.86278
		M	-3.86263	**-3.86278**	**-3.86278**	**-3.86278**
		SD	-3.86278*	1.56E-15	3.11E-15	3.06E-15
		MFE		575	4093	6680
		P		5	20	30
20	-3.32	B		-3.322368	-3.322368	-3.322368
		W		-3.140792	-3.132710	-3.203162
		M	-3.28654	-3.286657	**-3.297920**	-3.278659
		SD	-3.25056*	0.056640	0.057190	0.058427
		MFE		8003	2799	6916
		P		20	10	30
21	-10.1532	B		-10.153200	-10.153200	-10.153200
		W		-2.626968	-5.055198	-2.630472
		M	**-10.1514****	-7.566177	**-8.405803**	-8.168698
		SD	-9.14015*	2.413688	2.391694	2.693478
		MFE		11371	11016	13321
		P		20	20	30

Table 15
Results of the proposed algorithms for 23 benchmark functions considered (30000 function evaluations)

22	-10.4029	B		-10.402941	-10.402941	-10.402941
		W		-2.765897	-5.128823	-7.863835
		M	**-10.4015****	-8.760775	**-10.108301**	-9.976039
		SD	-8.58441*	2.146664	1.004131	0.626313
		MFE		13592	17633	22713
		P		20	50	100
23	-10.5364	B		-10.536410	-10.536410	-10.536410
		W		-5.175647	-9.647597	-9.025835
		M	**-10.5343****	-9.570118	-10.470286	**-10.486057**
		SD	-8.55899*	1.598056	0.212811	0.275792
		MFE		16652	26983	18602
		P		20	100	50

*Func.: Function; f_{min}: Global optimum value; *: This may be the W value of GWO (as the standard deviation can not be negative);; **:This may be the B value of GWO; ***:This may be the B value of ALO; The results of ALO are available only for 1-13 benchmark functions.*

It may be observed from Table 15 that the proposed algorithms are *not origin-biased* as it can be seen that these algorithms have obtained the global optimum solutions in the case of benchmark functions 8 and 14-23 whose optima are not at origin. The performance of the proposed algorithms is appreciable on the benchmark functions considered. It may also be observed that the standard deviation results of GWO for objective functions 8,16,19-23 (Mirjalili, 2014) are incorrect as the standard deviation value can not be negative. Furthermore, it seems that the values given by GWO as mean solutions for benchmark functions 21-23 may not be corresponding to the mean solutions and these may be corresponding to the best solutions of GWO. That is why, even though the "mean solutions" of GWO are shown in bold for the functions 21-23, the mean solutions of functions 21 and 22 given by Rao-2 algorithm, and the mean solution of function 23 by given by Rao-3 algorithm are also shown in bold.

In terms of the mean solutions, GWO algorithm has performed better (compared to ALO, Rao-1, Rao-2 and Rao-3 algorithms) on functions 7,11,15, 16 (and 21-23?). The results corresponding to functions 21-23 may be corresponding to the "best (B)" solutions of GWO algorithm. The mean results of ALO algorithm are comparatively better for functions 4,5,9,10 (and 12 and 13?). The mean results of Rao-1 algorithm are better for functions 6,14,17,18 and 19. The mean results of Rao-2 algorithm are better for functions 14,17,18,19,20 (and 21 and 22?). The mean results of Rao-3 algorithm are better for functions 1-3, 8,17,19,(and 23?). Thus, the proposed three algorithms can be said competitive to the existing advanced optimization algorithms in terms of better results for solving the unimodal, multimodal and fixed-dimension multimodal optimization problems with better exploitation and exploration potential.

If an intra-comparison is made among the proposed three algorithms in terms of the "best (B)" solutions obtained, Rao-3 algorithm has obtained the best solutions in 17 functions; Rao-2 has obtained the best solutions in 9 functions and Rao-1 in 9 functions. In terms of the 'worst (W)" solutions obtained, Rao-3 performs better in 14 functions, Rao-2 in 8 functions and Rao-1 in 7 functions.

The MATLAB codes of Rao-1, Rao-2 and Rao-3 algorithms are given in Appendix-1, Appendix-2 and Appendix-3 respectively. The code is developed for the objection function "Sphere function". The user may copy and paste this code in a MATLAB file and run the program. *The user may replace the portion of the code corresponding to the Sphere function with the objective function of the optimization problem considered by him/her to get the results.*

4. Additional experiments on unconstrained optimization problems

The performance of the proposed three algorithms is tested further on 25 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 16 shows the details of 25 unconstrained benchmark functions.

Table 16
Unconstrained benchmark functions considered

No.	Function	Formulation	D	Search range	C
1	Sphere	$F_{\min}=\sum_{i=1}^{D}x_i^2$	30	[-100, 100]	US
2	SumSquares	$F_{\min}=\sum_{i=1}^{D}ix_i^2$	30	[-10, 10]	US
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
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
5	Matyas	$F_{\min}=0.26\left(x_1^2+x_2^2\right)-0.48x_1x_2$	2	[-10, 10]	UN
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
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
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
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
10	Schwefel 1.2	$F_{\min}=\sum_{i=1}^{D}\left(\sum_{j=1}^{i}x_j^2\right)^2$	30	[-100, 100]	UN
11	Rosenbrock	$F_{\min}=\sum_{i=1}^{D}[100(x_i^2-x_{i+1})^2+(1-x_i)^2]$	30	[-30, 30]	UN
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
13	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
14	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
15	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
16	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
17	Booth	$F_{\min}=\left(x_1-2x_2-7\right)^2+\left(2x_1+x_2-5\right)^2$	2	[-10, 10]	MS
18	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
19	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
20	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
21	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
22	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

Table 16
Unconstrained benchmark functions considered (Continued)

No.	Function	Formulation	D	Search range	C
23	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]	MS
24	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
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

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

To evaluate the performance of the proposed algorithms, a common experimental platform is provided by setting the maximum number of function evaluations as 500000 for each benchmark function with 30 runs for each benchmark function. The results of each benchmark function are presented in Table 17 in the form of best solution, worst solution, mean solution, standard deviation obtained in 30 independent runs and the mean function evaluations on each benchmark function. The global optimum values of the benchmark functions are also given in Table 17 to give an idea to the readers about the performances of the proposed algorithms.

Table 17
Results of the proposed algorithms for the unconstrained benchmark functions

S. No.	Function	Optimum		Rao-1	Rao-2	Rao-3
1	Sphere	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	499976	499791	277522
2	SumSquares	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	499975	499851	276556
3	Beale	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	9805	7612	7325
4	Easom	-1	B	-1	-1	-1
			W	0	-1	-1
			M	-0.5667	-1	-1
			SD	0.5040	0	0
			MFE	3010	11187	14025
5	Matyas	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	77023	110544	143088
6	Colville	0	B	0	0	0
			W	0	5.35E-23	1.32E-25
			M	0	1.80E-24	7.87E-27
			SD	0	9.76E-24	2.61E-26
			MFE	385066	477753	488127

Table 17
Results of the proposed algorithms for the unconstrained benchmark functions (Continued)

S. No.	Function	Optimum		Rao-1	Rao-2	Rao-3
7	Trid 6	-50	B	-50	-50	-50
			W	-50	-50	-50
			M	-50	-50	-50
			SD	0	0	0
			MFE	17485	37209	34796
8	Trid 10	-210	B	-210	-210	-210
			W	-210	1171	-210
			M	-210	-30.8587	-210
			SD	0	4.13E+02	0
			MFE	48231	144156	142253
9	Zakharov	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	345615	499767	258451
10	Schwefel 1.2	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	301513	499849	144367
11	Rosenbrock	0	B	8.95E-26	1.86E-16	1.40E-14
			W	3.9866	22.191719	22.191719
			M	0.6644	0.739724	0.739728
			SD	1.51E+00	4.05E+00	4.05E+00
			MFE	489811	478410	478420
12	Dixon-Price	0	B	0.666667	2.81E-30	0.666667
			W	0.666667	0.666667	0.667019
			M	0.666667	0.288889	0.666686
			SD	0	3.36E-01	7.39E-05
			MFE	75427	113638	159231
13	Branin	0.397887	B	0.397887	0.397887	0.397887
			W	0.397931	0.397933	0.397888
			M	0.397892	0.397891	0.397887
			SD	1.05E-05	1.03E-05	1.44E-07
			MFE	102785	41263	80683
14	Bohachevsky 1	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	3129	4751	3435
15	Bohachevsky 2	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	2963	4272	3191
16	Bohachevsky 3	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	4725	12337	6821
17	Booth	0	B	0	0	0
			W	0	0	0
			M	0	0	0
			SD	0	0	0
			MFE	5583	4485	4312

Table 17
Results of the proposed algorithms for the unconstrained benchmark functions (Continued)

S. No.	Function	Optimum		Rao-1	Rao-2	Rao-3
18	Michalewicz 2	-1.8013	B	-1.801303	-1.801303	-1.801303
			W	-1.801303	-1.801303	-1.801303
			M	-1.801303	-1.801303	-1.801303
			SD	0	0	0
			MFE	3863	2694	2751
19	Michalewicz 5	-4.6877	B	-4.687658	-4.687658	-4.687658
			W	-4.537656	-3.116841	-3.495893
			M	-4.674306	-4.429948	-4.492183
			SD	3.09E-02	3.60E-01	2.79E-01
			MFE	39710	67252	58401
20	GoldStein-Price	3	B	3	3	3
			W	3	84	3
			M	3	5.7	3
			SD	0	1.48E+01	0
			MFE	180121	176933	353893
21	Perm	0	B	0	0	0
			W	3.71E-09	0	0
			M	1.45E-10	0	0
			SD	6.78E-10	0	0
			MFE	82792	3139	4453
22	Ackley	0	B	1.51E-14	7.99E-15	4.44E-15
			W	2.220970	1.51E-14	1.51E-14
			M	0.566540	1.04E-14	6.69E-15
			SD	7.41E-01	3.14E-15	2.38E-15
			MFE	129392	417741	76352
23	Shekel's Foxholes	0.998004	B	0.998004	0.998004	0.998004
			W	0.998004	0.998004	0.998004
			M	0.998004	0.998004	0.998004
			SD	0	0	0
			MFE	18839	95983	243748
24	Hartmann 3	-3.86278	B	-3.86278	-3.86278	-3.86278
			W	-3.86278	-3.86278	-3.86278
			M	-3.86278	-3.86278	-3.86278
			SD	0	0	0
			MFE	4459	3022	3271
25	Penalized 2	0	B	1.35E-32	1.35E-32	1.35E-32
			W	0.010987	1.597462	0.141320
			M	0.001465	0.057915	0.016008
			SD	3.80E-03	2.91E-01	3.50E-02
			MFE	173661	115593	55637

B: Best Solution; W: Worst Solution; M: Mean Solution; SD: Standard Deviation; MFE: Mean Function Evaluations.

Table 18 shows the number of instances the results of each algorithm are either better or equal to the performance other algorithms in terms of best solution (B), worst solution (W), mean solution (M), standard deviation (SD) and mean function evaluations (MFE).

Table 18
Comparison of the results in terms of number of instances a particular algorithm is better than or equal in performance to other algorithms

	Rao-1	Rao-2	Rao-3
B	24	22	24
W	21	18	20
M	20	17	20
SD	21	16	20
MFE	13	4	10

It can be observed from Tables 17 and 18 that the algorithms are not origin-biased as it can be seen that these algorithms have obtained the global optimum solutions in the case of benchmark functions 4, 7, 8, 13, 18, 19, 20, 23 and 24 whose optima are not at origin. The performance of the proposed algorithms is appreciable on 25 unconstrained benchmark functions considered. Out of the 25 unconstrained benchmark functions, the proposed algorithms have obtained the same results in 14 functions (i.e., in terms of best solution, worst solution, mean solution, standard deviation and mean function evaluations). Even though Rao-2 has obtained the best solution in the case of function nos. 8 and 20 but the worst solutions obtained are not good and hence the mean solution values are increased. In the case of function no. 12, Rao-1 and Rao-3 have not obtained the best solution but the best solution obtained by Rao-2 is comparatively better.

5. Experiments on constrained optimization problems

The performance of the proposed three algorithms is tested further on 2 constrained benchmark functions as part of the investigations. The details of the functions are given below.

1. Himmelblau function: It is a continuous and non-convex multi-modal function.
   Min. $f(x,y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2$
   Subjected to the constraints of:
   $26 - (x-5)^2 - y^2 \geq 0$
   $20 - 4x - y \geq 0$
   $x \in [-5, 5]$; $y \in [-5, 5]$

2. Min. $f(x,y) = (x - 10)^3 + (y - 20)^3$
   Subjected to the constraints of:
   $100 - (x - 5)^2 - (y - 5)^2 \geq 0$
   $(x - 6)^2 + (y - 5)^2 - 82.81 \geq 0$
   $x \in [13, 100]$; $y \in [0, 100]$

The results of application of the proposed algorithms on the above two benchmark functions are given in Table 19. The number of runs is 30 and the maximum function evaluations are 500000.

Table 19
Results of constrained benchmark functions

Function	Optimum		Rao-1	Rao-2	Rao-3
1	0	B	0	0	0
		W	0.000012	0	0
		M	0.000002	0	0
		SD	0.000003	0	0
		MFE	74980	9881	118858
2	-6961.814	B	-6961.813876	-6961.81388917	-6961.81388947
		W	-6961.813876	-6961.81388914	-6961.81388914
		M	-6961.813876	-6961.81388915	-6961.81388916
		SD	1.734E-10	6.69E-09	5.75E-08
		MFE	217739	487953	484997

B: Best Solution; W: Worst Solution; M: Mean Solution; SD: Standard Deviation; MFE: Mean Function Evaluations.

In the case of constrained benchmark functions, it can be observed from Table 19 that Rao-2 and Rao-3 have obtained comparatively better results than Rao-1. It may be noted that Rao-1 algorithm, given by Eq. (1), is a very simple algorithm and is based only on the difference between the best and worst

solutions. Even then, it can be observed that its performance is appreciable in quite a good number of unconstrained and constrained functions.

6. Conclusions

It is proved in this paper that it is possible to develop potential optimization algorithms without the need of using metaphors related to the behavior of animals, birds, insects, societies, cultures, planets, musical instruments, chemical reactions, physical reactions, etc. The proposed three optimization algorithms are not based on any metaphor or algorithm-specific parameters. These require only the tuning of the common controlling parameters of the algorithm for working (e.g., population size and the number of iterations). The proposed algorithms are implemented first on 23 unconstrained optimization problems including 6 unimodal, 7 multimodal and 10 fixed-dimension multimodal problems. Additional computational experiments are carried out on 25 well defined unconstrained optimization problems having different characteristics and 2 standard constrained optimization problems. The proposed three simple algorithms have given satisfactory performance and are believed to have potential to solve the complex optimization problems as well.

The results of the proposed algorithms presented in this paper are based on the preliminary investigations. Detailed investigations are planned to be carried out in the coming days. These investigations will include testing the performance of the proposed algorithms on various complex and computationally expensive benchmark functions involving a large number of dimensions. The results of detailed experimentation will be compared with the results of other existing well established optimization algorithms and the statistical tests will also be conducted. The researchers working in the field of optimization are requested to make improvements to these three algorithms so that these algorithms will become much more powerful. If these algorithms are found having certain limitations then the researchers may suggest the ways to overcome the limitations, instead of making destructive criticism, to further strengthen the algorithms.

Acknowledgement

The author gratefully acknowledges the support of his students Mr. Rahul Pawar and Mr. Hameer Singh for helping him in executing the codes.

References

Mirjalili, S. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46-61.

Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software, 83*, 80-98.

Rao, R.V. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations, 7*(1), 19-34.

Rao, R.V. (2019). *Jaya: An Advanced Optimization Algorithm And Its Engineering Applications*. Springer International Publishing, Switzerland.

Rao, R.V. (2015). *Teaching Learning Based Optimization And Its Engineering Applications*. Springer International Publishing, Switzerland.

Sorensen, K. (2015). Metaheuristics – the metaphor exposed. *International Transactional in Operational Research, 22*, 3-18.

Appendix-1: MATLAB code for Rao-1 algorithm

```
%% MATLAB code of Rao-1 algorithm
%% Unconstrained optimization
%% Sphere function
function Rao-1 ()
clc
clear all
pop = 10;                          % Population size
var = 30;    % Number of design variables
maxFes = 30000;                % Maximum functions evaluation
maxGen = floor(maxFes/pop);     % Maximum number of iterations
mini = -100*ones(1,var);
maxi = 100*ones(1,var);
[row,var] = size(mini);
x = zeros(pop,var);
for i=1:var
   x(:,i) = mini(i)+(maxi(i)-mini(i))*rand(pop,1);
end
f = objective(x);
gen=1;
while(gen <= maxGen)
   xnew = updatepopulation(x,f);
   xnew = trimr(mini,maxi,xnew);
   fnew = objective(xnew);
   for i=1:pop
      if(fnew(i)<f(i))
         x(i,:) = xnew(i,:);
         f(i) = fnew(i);
      end
   end
   disp(['Iteration No. = ',num2str(gen)])
   disp('%%%%%%%% Final population %%%%%%%%%')
   disp([x,f])
   fnew = []; xnew = [];
   [fopt(gen),ind] = min(f);
   xopt(gen,:)= x(ind,:);
   gen = gen+1;
end
[val,ind] = min(fopt);
Fes = pop*ind;
disp(['Optimum value = ',num2str(val,10)])
end

%%The objective function is given below.
function [f] = objective(x)
[r,c]=size(x);
for i=1:r
   y=0;
   for j=1:c
      y=y+(x(i,j))^2;        % Sphere function
   end
   z(i)=y;
end
f=z';
end
```

```
function [xnew] = updatepopulation(x,f)
[row,col]=size(x);
[t,tindex]=min(f);
Best=x(tindex,:);
[w,windex]=max(f);
worst=x(windex,:);
xnew=zeros(row,col);
for i=1:row
   for j=1:col
      xnew(i,j)=(x(i,j))+rand*(Best(j)-worst(j));
   end
end
end

function [z] = trimr(mini,maxi,x)
[row,col]=size(x);
for i=1:col
   x(x(:,i)<mini(i),i)=mini(i);
   x(x(:,i)>maxi(i),i)=maxi(i);
end
z=x;
end
```

Appendix-2: MATLAB code for Rao-2 algorithm

```
%% MATLAB code of Rao-2 algorithm
%% Unconstrained optimization
%% Sphere function
function Rao-2 ()
clc
clear all
pop = 10;                    % Population size
var = 30;       % Number of design variables
maxFes = 30000;    % Maximum functions evaluation
maxGen = floor(maxFes/pop);      % Maximum number of iterations
mini = -100*ones(1,var);
maxi = 100*ones(1,var);
[row,var] = size(mini);
x = zeros(pop,var);
for i=1:var
   x(:,i) = mini(i)+(maxi(i)-mini(i))*rand(pop,1);
end
f = objective(x);
gen=1;
while(gen <= maxGen)
   xnew = updatepopulation(x,f);
   xnew = trimr(mini,maxi,xnew);
   fnew = objective(xnew);
   for i=1:pop
      if(fnew(i)<f(i))
         x(i,:) = xnew(i,:);
         f(i) = fnew(i);
      end
   end
   disp(['Iteration No. = ',num2str(gen)])
```

```
    disp('%%%%%%%% Final population %%%%%%%%%%')
    disp([x,f])
    fnew = []; xnew = [];
    [fopt(gen),ind] = min(f);
    xopt(gen,:)= x(ind,:);
    gen = gen+1;
end
[val,ind] = min(fopt);
Fes = pop*ind;
disp(['Optimum value = ',num2str(val,10)])
end


%%The objective function is given below.
function [f] = objective(x)
[r,c]=size(x);
for i=1:r
    y=0;
    for j=1:c
        y=y+(x(i,j))^2;          % Sphere function
    end
    z(i)=y;
end
f=z';
end

function [xnew]=updatepopulation(x,f)
[row,col]=size(x);
[t,tindex]=min(f);
Best=x(tindex,:);
[w,windex]=max(f);
worst=x(windex,:);
xnew=zeros(row,col);
for i=1:row
    k=randi(row);
    while (k==i)
        k=randi(row);
    end
    if (f(i)<f(k))
        for j=1:col
            r=rand(1,2);
            xnew(i,j)=x(i,j)+r(1)*(Best(j)-worst(j))+r(2)*(abs(x(i,j))-abs(x(k,j)));
        end
    else
        for j=1:col
            r=rand(1,2);
            xnew(i,j)=x(i,j)+r(1)*(Best(j)-worst(j))+r(2)*(abs(x(k,j))-abs(x(i,j)));
        end
    end
end
end

function [z] = trimr(mini,maxi,x)
[row,col]=size(x);
for i=1:col
    x(x(:,i)<mini(i),i)=mini(i);
```

```
    x(x(:,i)>maxi(i),i)=maxi(i);
end
z=x;
end
```

Appendix-3: MATLAB code for Rao-3 algorithm

```
%% MATLAB code of Rao-3 algorithm
%% Unconstrained optimization
%% Sphere function
function Rao-3 ()
clc
clear all
pop = 10;      % Population size
var = 30;                    % Number of design variables
maxFes = 30000;              % Maximum functions evaluation
maxGen = floor(maxFes/pop);       % Maximum number of iterations
mini = -100*ones(1,var);
maxi = 100*ones(1,var);
[row,var] = size(mini);
x = zeros(pop,var);
for i=1:var
    x(:,i) = mini(i)+(maxi(i)-mini(i))*rand(pop,1);
end
f = objective(x);
gen=1;
while(gen <= maxGen)
    xnew = updatepopulation(x,f);
    xnew = trimr(mini,maxi,xnew);
    fnew = objective(xnew);
    for i=1:pop
        if(fnew(i)<f(i))
            x(i,:) = xnew(i,:);
            f(i) = fnew(i);
        end
    end
    disp(['Iteration No. = ',num2str(gen)])
    disp('%%%%%%%% Final population %%%%%%%%%%')
    disp([x,f])
    fnew = []; xnew = [];
    [fopt(gen),ind] = min(f);
    xopt(gen,:)= x(ind,:);
    gen = gen+1;
end
[val,ind] = min(fopt);
Fes = pop*ind;
disp(['Optimum value = ',num2str(val,10)])
end

%%The objective function is given below.
function [f] = objective(x)
[r,c]=size(x);
for i=1:r
    y=0;
    for j=1:c
        y=y+(x(i,j))^2;          % Sphere function
```

```
    end
    z(i)=y;
end
f=z';
end

function [xnew]=updatepopulation(x,f)
[row,col]=size(x);
[t,tindex]=min(f);
Best=x(tindex,:);
[w,windex]=max(f);
worst=x(windex,:);
xnew=zeros(row,col);
for i=1:row
    k=randi(row);
    while (k==i)
        k=randi(row);
    end
    if (f(i)<f(k))
        for j=1:col
            r=rand(1,2);
            xnew(i,j)=x(i,j)+r(1)*(Best(j)-abs(worst(j)))+r(2)*(abs(x(i,j))-x(k,j));
        end
    else
        for j=1:col
            r=rand(1,2);
            xnew(i,j)=x(i,j)+r(1)*(Best(j)-abs(worst(j)))+r(2)*(abs(x(k,j))-x(i,j));
        end
    end
end
end

function [z] = trimr(mini,maxi,x)
[row,col]=size(x);
for i=1:col
    x(x(:,i)<mini(i),i)=mini(i);
    x(x(:,i)>maxi(i),i)=maxi(i);
end
z=x;
end
```