 https://doi.org/10.3311/PPci.20067

Periodica Polytechnica Civil Engineering, 66(3), pp. 694–709, 2022

Enhanced Rao Algorithms for Optimization of the Structures Considering the Deterministic and Probabilistic Constraints

Ali Kaveh[1*], Ataollah Zaerreza[1]

[1] School of Civil Engineering, Iran University of Science and Technology, Hengam St., Resalat Square, 16846-13114 Tehran, Iran
[*] Corresponding author, e-mail: alikaveh@iust.ac.ir

Received: 27 February 2022, Accepted: 28 March 2022, Published online: 30 March 2022

Abstract

Rao algorithms are metaheuristic algorithms that are based on population and do not have metaphors. These algorithms are extremely simple and do not require the use of any parameters that are dependent on the problem. Although these algorithms have some other benefits to, they are vulnerable of being trapped in local optima. The present work proposes Enhanced Rao algorithms denoted by ERao as a means of alleviating this drawback. In the ERao algorithms, the modified version of the statistically regenerated mechanism is added. Additionally, the mechanism that sticks the candidate solution to the border of the search space is modified. The efficiency of the ERao algorithms is tested on three structural design optimization problems with probabilistic and deterministic constraints. The optimization results are compared to those of the Rao algorithms and some other state-of-art optimization methods. The results show that the proposed optimization method can be an effective tool for solving structural design problems with probabilistic and deterministic constraints.

Keywords

metaheuristic algorithms, Rao algorithms, statistically regenerated mechanism, probabilistic and deterministic constraints

1 Introduction

Optimization methods have been very popular in the recent three decades. Especially engineers have employed optimization approaches to solve design challenges in order to obtain the optimal solution [1]. Thus, new optimization algorithms and their improvements are developed in order to achieve a better outcome for optimization problems. Gradient-based approaches and metaheuristic methods are two types of optimization algorithms [2]. Gradient-based methods can find the exact solution. However, the results are dependent on the starting point and can quickly get trapped in the local optimum [3]. Metaheuristics techniques are created to address these issues. Metaheuristic algorithms are not dependent on the start point and can quickly get out of the local optimum. Also, implementation of the metaheuristic algorithms is easier than gradient-based methods. Hence, metaheuristic algorithms are prevalent than gradient-based methods.

A metaheuristic optimization algorithm comprises of two conflicting abilities: exploration and exploitation. Exploration is the ability to explore entirely new regions of the solution space, whereas exploitation is the ability to search around solutions with higher fitness [4]. The agents of a metaheuristic may trap into a local optimum without exploratory behavior. On the other hand, the lack of exploitation behavior decreases the performance of the metaheuristic. In this case, the metaheuristic can never converge to the optimum solution. Thus, a good balance between these two behaviors results in an efficient metaheuristic. Nevertheless, some optimization problems require extreme exploration behavior, and others need extreme exploitation behavior to find the optimum solution [5].

According to the no-free lunch theory [6], the metaheuristic algorithms cannot perform well in all the optimization problems. To this goal, researchers try to improve the different metaheuristic algorithms to be capable in other areas or test the performance of the algorithms in the different fields. For instance, Kaveh et al. [7] improved the Ray optimization algorithm, and Kaveh and Zakian [8] enhanced the Bat algorithm. Yoo and Han [9] modified the ant colony optimization method to solve problems involving dynamic topology optimization. Kaveh and Vazirinia [10] upgraded the sine cosine algorithm for tower crane selection and layout problems. Kazemzadeh Azad et al. [11] improved a metaheuristic by using the

Cite this article as: Kaveh, A., Zaerreza, A. "Enhanced Rao Algorithms for Optimization of the Structures Considering the Deterministic and Probabilistic Constraints", Periodica Polytechnica Civil Engineering, 66(3), pp. 694–709, 2022. https://doi.org/10.3311/PPci.20067

upper bound technique. Kaveh, et al. [12] tested the performance of the four metaheuristic algorithms in the optimum design of the portal frames. Artificial bee colony has been enhanced by Latif and Saka [13] for the optimal design of tied-arch bridges. Abdollahzadeh et al. [14] enhanced the binary slime mold algorithm for solving the knapsack problem. Kaveh and Zaerreza [15] investigate the ability of SSOA in the layout optimization of trusses. Das and Dhang [16] improved the hybrid teaching-learning-based – particle swarm optimization technique to identify structural damage. Makiabadi and Maheri [17] developed the enhanced symbiotic organisms search for optimum design of the truss structures.

In this study, we have concentrated on the Rao algorithms. These algorithms are metaphor-less based on the best candidate solution, worst candidate solution, and interactions between the candidate solutions [18]. Rao algorithms have been applied in various fields; several of them are summarized here. Premkumar et al. [19] estimated photovoltaic cell parameters using Rao algorithms. Rao and Pawar [20] applied the Rao algorithms for the optimum design of mechanical system components. Kalemci and Ikizler [21] optimized the cantilever retaining wall using the Rao-3 algorithm. Manam et al. [22] employed the Rao algorithms for the economic dispatch problem. Rao and Pawar [23] created the self-adaptive multi-population Rao algorithms for benchmark problem optimization. Hassan et al. [24] made modifications to the second Rao algorithm for solving the optimal power flow problem.

Rao algorithms are highly capable in finding the optimum solution of different problems. However, they can be trapped in the local optimum due to not having a mechanism for escaping from local and sticking the solution to the search space boundaries when the solution violates the search space. Thus, a modified statistically regenerated mechanism is added to the basic Rao algorithms. Also, the mechanism that keeps the solution in search space is modified. These modifications have enhanced the performance of the basic Rao algorithms to get out of the local optimum; thus, the modified algorithms are named ERao algorithms. ERao algorithms performances are tasted on three structural design problems. One of these problems has probabilistic constraints, and two of them have deterministic constraints. The statistical results indicate that ERao algorithms are robust compared to the basic Rao algorithms.

The rest of the paper is organized as follows: in Section 2, Rao's algorithms are provided. In Section 3, ERao algorithms are presented. Three structure design examples containing one example with probabilistic constraints and two examples with deterministic constraints are given in Section 4. Finally, the conclusion is provided in Section 5.

2 Algorithms of Rao

Rao developed three metaphor-less optimization algorithms and called them Rao-1, Rao-2, and Rao-3 in 2020 [18]. These methods are easy to implement and need only basic control parameters. Algorithms of Rao consist of 5 steps, and only step 2 is different in the Rao algorithms.

Step 1: Initialization

Similar to other metaheuristic algorithms, the initial candidate solutions of Rao algorithms are generated randomly in the search space using the following equation.

$$X_{i,j}^{0} = X_{j}^{min} + r \times \left(X_{j}^{min} - X_{j}^{max} \right); \quad i = 1,2,\ldots,n; \;\; j = 1,2,\ldots,m, \tag{1}$$

where, $X_{i,j}^{0}$ is initial value of the *j*th variable of the *i*th candidate solution; X^{min} and X^{max} represent the search space's lower and upper bounds, respectively; *n* and *m* are the number of candidate solutions and design variables, respectively; *r* is the random number generator that produces numbers in the range [0,1].

Step 2: Generate the new candidate solution

The step size used to generate the new solution is different in the Rao algorithms; yet, they are very similar. To generate the new solution in all Rao algorithms, the worst and best solution of the entire solution should be determined. In the Rao-1, only the difference between the best and worst solution is utilized to generate the new solution, as given in Eq. (2).

$$X_{i,j}^{new} = X_{i,j} + r \times \left(X_{i,j}^{best} - X_{i,j}^{worst} \right), \tag{2}$$

where, $X_{i,j}^{new}$ indicates the *j*th variable of the *i*th solution which generated; $X_{i,j}^{best}$ and $X_{i,j}^{worst}$ are the best and worst solution of the entire solution.

Rao-2, like Rao-1, requires the difference between the best and worst solutions to generate a new solution. However, the interaction between the candidate solutions is added to the equation of Rao-2, as shown in Eq. (3).

$$X_{i,j}^{new} = X_{i,j} + r \times \left(X_{i,j}^{best} - X_{i,j}^{worst} \right) + \begin{cases} r \times \left(\left| X_{i,j} \right| - \left| X_{k,j} \right| \right) & \text{if } X_{i,j} \text{ better than } X_{k,j} \\ r \times \left(\left| X_{k,j} \right| - \left| X_{i,j} \right| \right) & \text{if } X_{k,j} \text{ better than } X_{i,j} \end{cases}, \tag{3}$$

where $X_{k,j}$ is the candidate solution randomly selected for $X_{i,j}$.

The equation used to generate the new candidate solution in the Rao-3 algorithm is similar to the equation used in the Rao-2 algorithm. However, the location where the absolute value operator is applied differs. The equation to generate the new candidate solution in Rao-3 is given in Eq. (4).

$$X_{i,j}^{new} = X_{i,j} + r \times \left(X_{i,j}^{best} - \left| X_{i,j}^{worst} \right| \right) + \begin{cases} r \times \left(\left| X_{i,j} \right| - X_{k,j} \right) & \text{if } X_{i,j} \text{ better than } X_{k,j} \\ r \times \left(\left| X_{k,j} \right| - X_{i,j} \right) & \text{if } X_{k,j} \text{ better than } X_{i,j} \end{cases}, \quad (4)$$

Step 3: Checking the search space boundaries which are not violated

After generating the new candidate solution, the new candidate solutions are checked to find out if the candidates are in the search space or not. If any variable of the new solutions is not in the search space, it will stick to the closest boundary of the search space.

Step 4: Replacement strategy

First, the new candidate solutions are evaluated. Then, their new objective values are compared to their old objective values. If the new objective value of a candidate solution is better than the old one, the old candidate solution is omitted. Otherwise, the new candidate solution is deleted.

Step 5: Checking termination condition

The termination condition of the algorithms of Rao is the maximum number of iterations (*MaxIt*), and the optimization process stops when the iteration is reached *MaxIt*. Otherwise, the algorithm goes to step 2 for the next optimization iteration.

As further clarity, the flowchart of the Rao-1 algorithm is presented in Fig. 1.

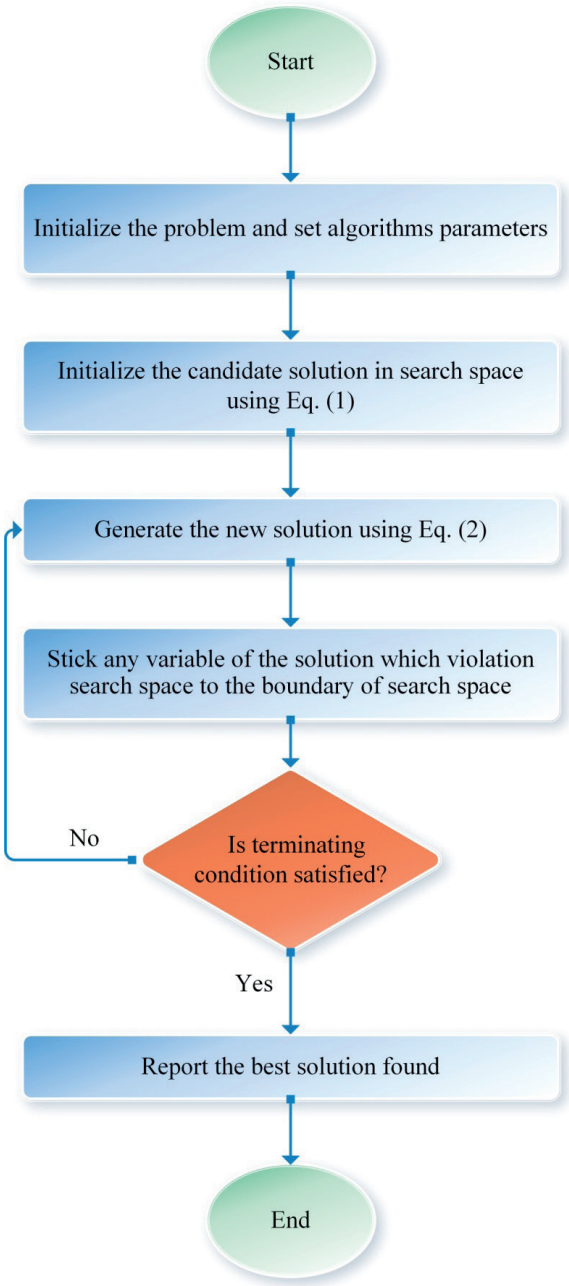

Fig. 1 Flowchart of the Rao-1 algorithm

3 Enhanced Rao algorithms

Although the algorithms of Rao do not require user-defined parameters and are easy to implement, however, they can be trapped in local optimums. This difficulty arises from the absence of a mechanism to escape from the local optimum and stick the solution to the nearest search space border if the search space boundary is violated. Hence, these two difficulties result in algorithms being trapped in local optimums. Here, an enhanced version of the Rao algorithms named Enhanced Rao algorithms (ERao) is introduced. A modified version of the statistically regenerated mechanism is used in the ERao as the method for escaping from the local optimum. Additionally, the technique that keeps the solutions in the search space to have acceptable results is improved to overcome the trapping in the local optimum of the algorithms of Rao.

3.1 Modified statistically regenerated mechanism

Statistically regenerated mechanism (SRM) is a powerful mechanism for escaping from local optimum introduced by Kaveh et al. [25]. In SRM, 80 percent of the candidate solutions are selected randomly, then 20 percent of their variables are regenerated using the statistic information of the solutions. The average and standard deviation of each variable of candidate solutions are used as the statistic information in this mechanism. The general formulation of the SRM is as follows:

$$x_{i,s}^{new} = UNFIRAND\begin{pmatrix} Mean_s - std_s - sigma_s, \\ Mean_s + std_s + sigma_s \end{pmatrix}, \quad (5)$$

where *UNFIRAND* is the operator that generates a random integer from a continuous uniform distribution whose lower and upper bounds are defined by $Mean_s - std_s - sigma_s$ and $Mean_s + std_s + sigma_s$, respectively; $Mean_s$ and std_s are the mean and standard deviation of the sth design variables of candidate solutions. $sigma_n$ is a parameter that aids the functioning of the SRM and is defined as follows:

$$sigma_s = \begin{cases} 0.01\times\left(X_s^{max} - X_s^{min}\right) & if\ std_s < \\ & 0.01\times\left(X_s^{max} - X_s^{min}\right) \\ 0 & otherwise \end{cases} \quad (6)$$

The SRM performs well in the escape from the local optimum, however, when the variables of candidate solutions are close to each other in the early iterations of algorithm, the standard deviation of the solution becomes a small value. Thus, the SRM performing the local search, and the algorithm can be trapped in the local optimum. To this end, to increase the performance of the SRM, the modified statistically regenerated mechanism (MSRM) is introduced in this study. In MSRM, only the value of the $sigma_s$ is modified to increase the performance of the SRM. Sigma is regarded five times larger in MSRM than in SRM and linearly reduces to zero, as demonstrated in Eq. (7).

$$sigma_s = \begin{cases} 0.05\times\left(X_s^{max} - X_s^{min}\right) & if\ std_s < 0.05\times\left(X_s^{max} - X_s^{min}\right) \\ \times\left(1 - \dfrac{it}{MaxIt}\right) & \times\left(1 - \dfrac{it}{Maxit}\right) \\ 0 & otherwise \end{cases} \quad (7)$$

where, *it* is the current iteration number; *MaxIt* is the maximum iteration number.

By this modification, at the start of the optimization process, solutions are regenerated at least in the ten percent of the search space. Thus, this modification helps the mechanism to perform better exploitation than the previous version in the early iterations. The linear decrease of the sigma causes the algorithm to perform the local search when the number of iterations gets close to the maximum number of iterations to find the better solution, so unlike the SRM, the local search never performs in the early iterations of the algorithm. The method MSRM is employed the same as SRM, so 80 percent of the candidate solutions in the ERao algorithms are picked randomly, and 20 percent of their variables are regenerated using the MSRM.

3.2 Improved mechanism for keeping solution in search space

In the Rao algorithms, if any variable of the solutions violates the search space, it is attached to the nearest border of the search space. This mechanism is the simplest method of ensuring for the solution to remain in the search space. However, if there is a local optimum in the border of the search space, the solution is trapped in this local optimum and cannot easily get out of it. On the other hand, it is possible that the optimal solution to be found on the border of the search space. In this case, this simple mechanism helps the optimization method to find the optimum solution faster. As a result, this method has both benefits and downsides, and in order to eliminate disadvantages, it must be modified.

Another mechanism is added to the primary mechanism in order to enhance it. The mechanism added is the regeneration mechanism that regenerates the solution randomly in search space. This regeneration technique enables the optimization algorithms to avoid being trapped at the boundary of search space. Thus, this mechanism omits the disadvantage of being trapped in the border of the search space. The search space of the optimization problem is unknown, and the best solution can be in the border of the search space, so there is no method to say which of these mechanisms is has more probability of happening. To this end, the probability of occurring of each of these mechanisms is considered as 50 percent. For further clarify the improved mechanism is defined in Eq. (8).

$$\begin{cases} Stick\ the\ variable\ of\ the\ solution\ to\ the\ closest\ boundary & if\ r < 0.5 \\ Regenerate\ the\ variable\ of\ the\ solution\ on\ the\ search\ space & if\ r \geq 0.5 \end{cases} \quad (8)$$

For further clarify, the flowchart of the ERao-1 algorithm is presented in Fig. 2.

4 Design examples

In this study, four structural design examples are considered to investigate the ability of the ERao algorithms. These examples include two examples with probabilistic constraint and two examples with deterministic constraint. Sequential optimization and reliability assessment-double metaheuristic (SORA-DM) framework of Kaveh and Zaerreza [26] is utilized to handle the probabilistic constraint for reliability-based design optimization (RBDO). In this framework, the reliability assessment and optimization process are performed using the metaheuristic algorithms. Thus,

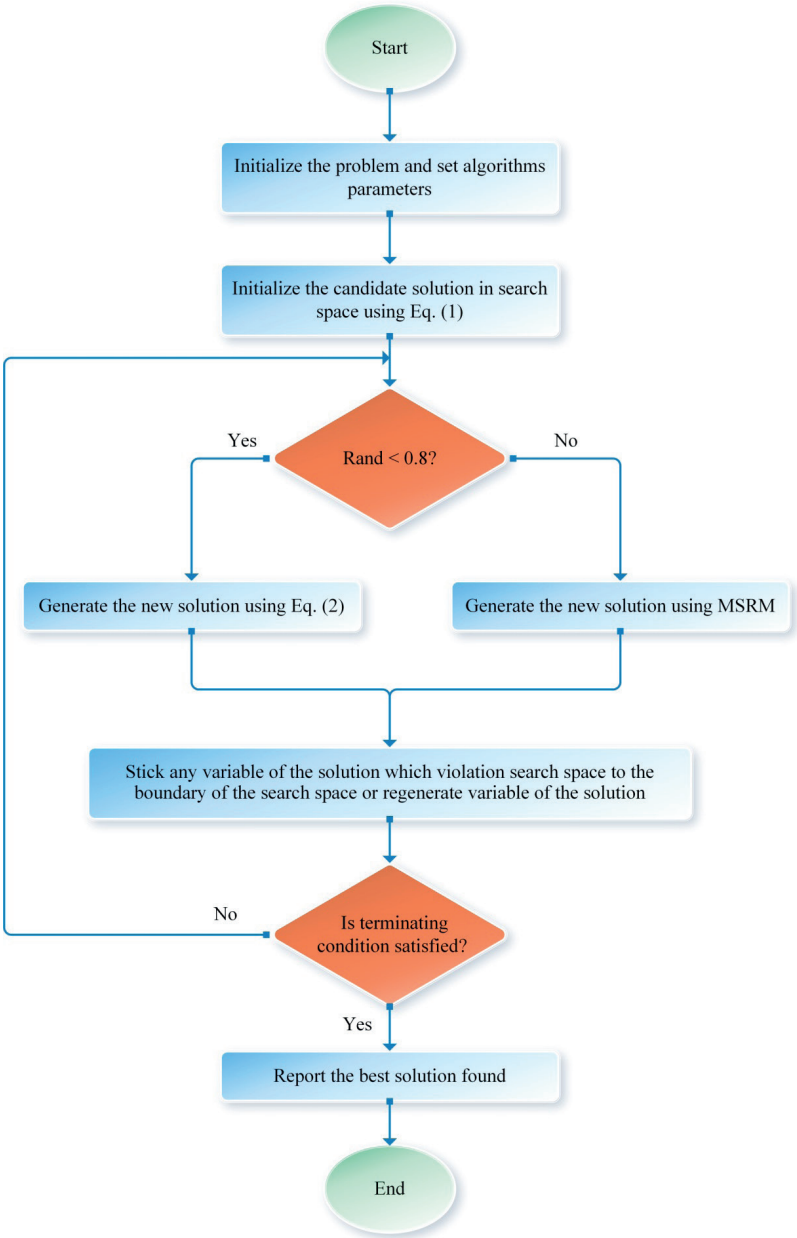

Fig. 2 Flowchart of the ERao-1 algorithm

Rao's algorithms in problem with probabilistic constraints are named SORA-DRao-1, SORA-DRao-2, and SORA-DRao-3, and ERao algorithms in these problems are named SORA-DERao-1, SORA-DERao-2, and SORA-DERao-3. The difference between Rao-2 and Rao-3 algorithms is only where the absolute value operator is applied. On the other hand, there is no negative value for design variables in structural optimization. Hence, both Rao-2 and Rao-3 have the same performance in these examples, and thus only ability of the Rao-2 algorithm is investigated.

The first example includes a 120-bar dome truss design problems with probabilistic constraints. The maximum number of iterations of the 120-bar dome truss design problem is set to 400, respectively, the same as the previous study [26]. In the 3-Bay 24-Story Frame and 1016-bar double-layer grid optimization problems, the maximum number of iterations is set to 1000. The number of candidate solutions is the same for all design examples and it is set to 20.

4.1 A 120-bar dome truss design problem

For the first time, Kaveh and Zaerreza [26] introduced probabilistic constraints based on displacement for the 120-bar dome truss design problem. These probabilistic constraints were imposed on the z-direction displacements of nodes 1, 2, 14, and 15. As illustrated in Fig. 3, the

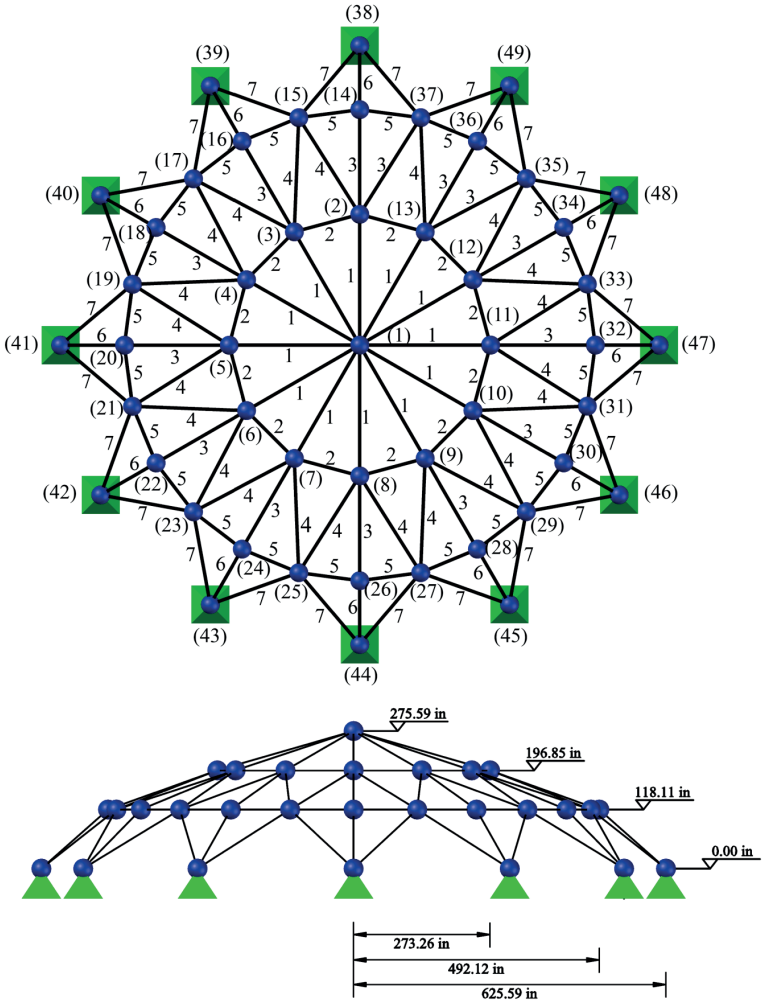

Fig. 3 Schematic of the 120-bar dome truss structure

truss elements are classified into seven classes. As with the previous example, the cross-sectional area of the elements (A) were considered as a random design variables with a normal distribution, and their mean value (μ_A) were considered as a design variable with a coefficient of variation of 0.05. Only the elasticity modulus (E) was considered as a random design parameter in this example. It has a normal distribution, an average value of 30450 ksi, and a coefficient of variation of 0.05. The applied load was considered as deterministic values, as given in Table 1. The material density was assumed to be the deterministic value of equal to 0.288 lb/in^3. The following formulas express the problem mathematically.

Table 1 Loading condition of the 120-bar dome truss

Nodes	F_x	F_y	F_z
1	0	0	-13.49 kip
2–13	0	0	-6.744 kip
14–37	0	0	-2.248 kip

$$
\begin{aligned}
&find: \mu_A = \{\mu_{A_1}, \mu_{A_2}, \ldots, \mu_{A_7}\} \\
&minimize: f(\mu_A) = \rho \sum_{i=1}^{7} A_i L_i \\
&Subjected\ to: PROB\left(g_j(x) \le 0\right) \le \Phi\left(\beta_j^t\right) \quad j = 1,2,3,4 \\
&Where: g_1(\mu_A, E) = 0.2 - \left|u_1^Z(\mu_A, E)\right| \\
&g_2(\mu_A, E) = 0.2 - \left|u_2^Z(\mu_A, E)\right| \\
&g_3(\mu_A, E) = 0.2 - \left|u_{14}^Z(\mu_A, E)\right| \\
&g_4(\mu_A, E) = 0.2 - \left|u_{15}^Z(\mu_A, E)\right| \\
&0.775\,in^2 \le \mu_A \le 20\,in^2 \\
&\beta_1^t = \beta_2^t = \beta_3^t = \beta_4^t = 3.0
\end{aligned}
\tag{9}
$$

The results of the SORA-DRao-1, SORA-DRao-2, SORA-DERao-1, and SORA-DERao-2 are provided in Table 2. SORA-DERao-1 outperforms other methods, and SORA-DERao-2 outperforms standard Rao algorithms, SORA-DESSOA, and SORA-DTLBO. The NFE required in the optimization parts for standard Rao and ERao

Table 2 Comparative results of the ERao and Rao algorithms with other methods in the 120-bar dome truss

Design variable	SORA-DTLBO [26]	SORA-DESSOA [26]	Present study			
			SORA-DRao-1	SORA-DRao-2	SORA-DERao-1	SORA-DERao-2
A_1	2.2692	2.2942	2.2786	2.2539	2.2802	2.2731
A_2	16.9854	17.1781	17.0643	16.8723	17.0620	17.0172
A_3	6.6318	6.4836	6.6082	6.5750	6.5797	6.5981
A_4	2.9090	2.9592	2.9042	2.9500	2.9178	2.9525
A_5	11.5758	11.4084	11.4040	11.5554	11.4758	11.4481
A_6	4.0656	4.0365	4.1812	4.0848	4.0694	4.1296
A_7	2.2492	2.3002	2.2892	2.2745	2.2827	2.2500
Best weight (lb)	37278.11	37276.82	37278.17	37265.72	37263.88	37265.11
NFE in the optimization part	49020	49020	26540	26540	26540	26540
NFE in reliability analysis part	114720	72780	84860	54220	105860	64340
Total NFE	163920	121800	111460	80820	132400	90880
β_1^{MCS}	3.01	3.01	3.0069	3.0084	3.0073	3.0107
β_2^{MCS}	3.01	3.01	3.0071	3.0012	3.0030	3.0035
β_3^{MCS}	Infinite	Infinite	Infinite	Infinite	Infinite	Infinite
β_4^{MCS}	Infinite	Infinite	Infinite	Infinite	Infinite	Infinite
Mean weight (lb)	37285.75	37282.28	37290.06	N/A	37277.59	37276.12
Standard deviation (lb)	9.3667	2.8442	8.8307	N/A	6.9959	11.8642

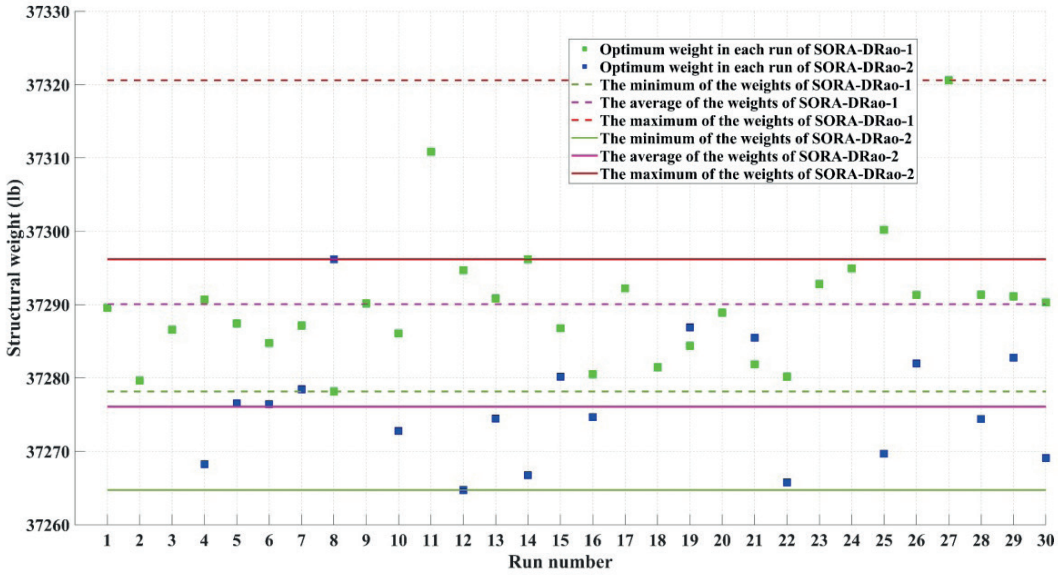

Fig. 4 The structural weight of each independent run of the Rao algorithms for the 120-bar dome truss design problem

algorithms is the same and less than SORA-DESSOA and SORA-DTLBO. ERao algorithms need more NFE than Rao algorithms in the reliability assessment part. The total function evaluation of the SORA-DRao-1, SORA-DRao-2, and SORA-DERao-2 are less than SORA-DESSOA and SORA-DTLBO. Although the SORA-DRao-2 required less NFE than other methods, SORA-DRao-2 fails to do reality assessment correctly in 37 percent of the 30 independent runs and is trapped in the local optimum, as shown in Fig. 4. However, SORA-DERao-2 do reliability assessment correct for all 30 independent runs, as shown in Fig. 5. This indicated that the modification improved Rao's algorithms perfectly, and ERao algorithms can easily escape from the local optimum. Like the previous example, the results are verified using the MCS. The reliability indexes found by MCS are more than 3 for all constraint functions. Rao and ERao algorithms need three optimization cycles to obtain the optimum result, as shown in Fig. 6.

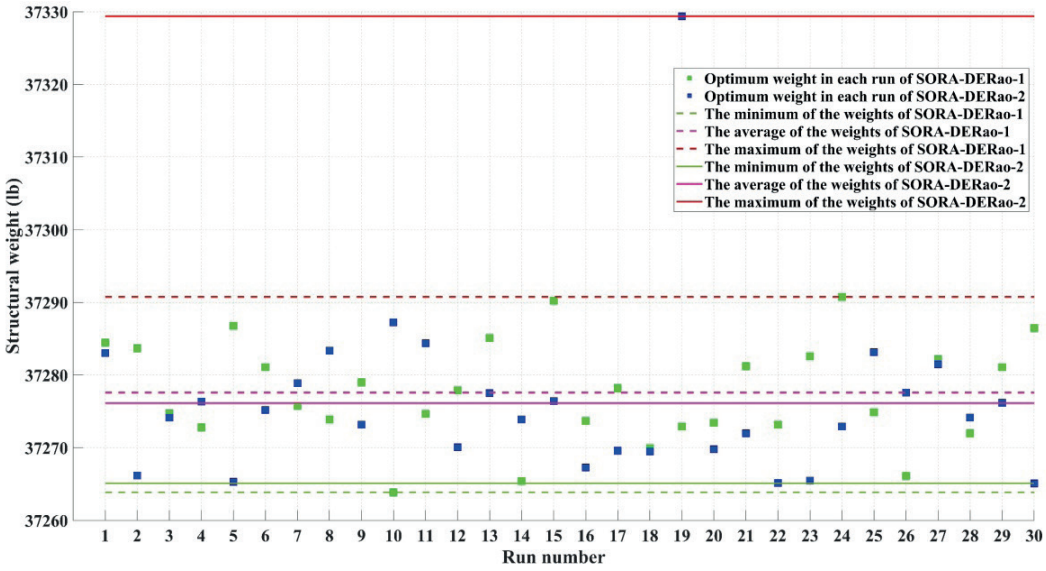

Fig. 5 The structural weight for each independent run of the ERao algorithms for the 120-bar dome truss design problem

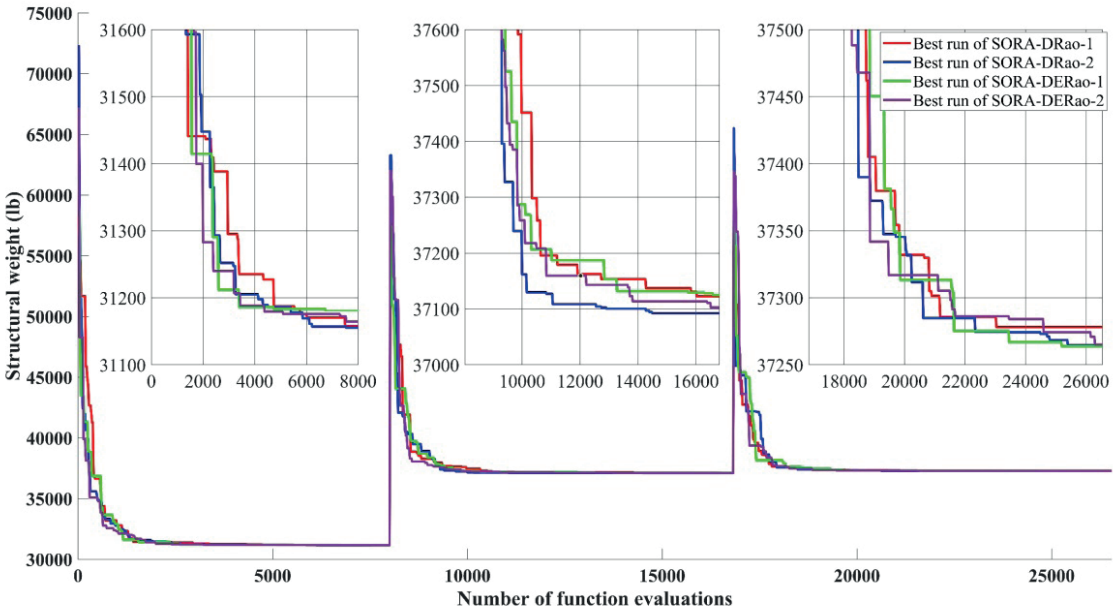

Fig. 6 Convergence histories of the best run of the Rao and ERao algorithms for the 120-bar dome truss design problem

4.2 A 3-bay 24-story frame design problem

The second example is the 3-bay 24-story frame design problem. This example is one of the well-known examples of structural optimization. This structure contains 96 columns and 72 beams, as depicted in Fig. 7. The beam elements are divided into 4 groups, and they are selected from W 14 sections. The columns are divided into 16 groups, and they are selected from the 267 W-shape section. The elasticity modulus (E) and yield stress (F_y) are equal to 29732 ksi and 33.4 ksi, respectively. The effective length factors of the member's sway-permitted frame (k_x) are calculated, and the out-of-plane effective length factor (k_y) is set to 1. All columns and beams are considered as non-braced along their lengths. According to the AISC-LRFD, this example contains three constraints: maximum lateral displacements, inter-story displacements, and strength constraints. The formulation of constraints is as follows.

(a) *The maximum lateral displacement*

$$\frac{\Delta_T}{H} - R \le 0\,, \tag{10}$$

where Δ_T is the maximum lateral displacement, H is the height of the frame structure, and R is the maximum drift index which is equal to 1/300.

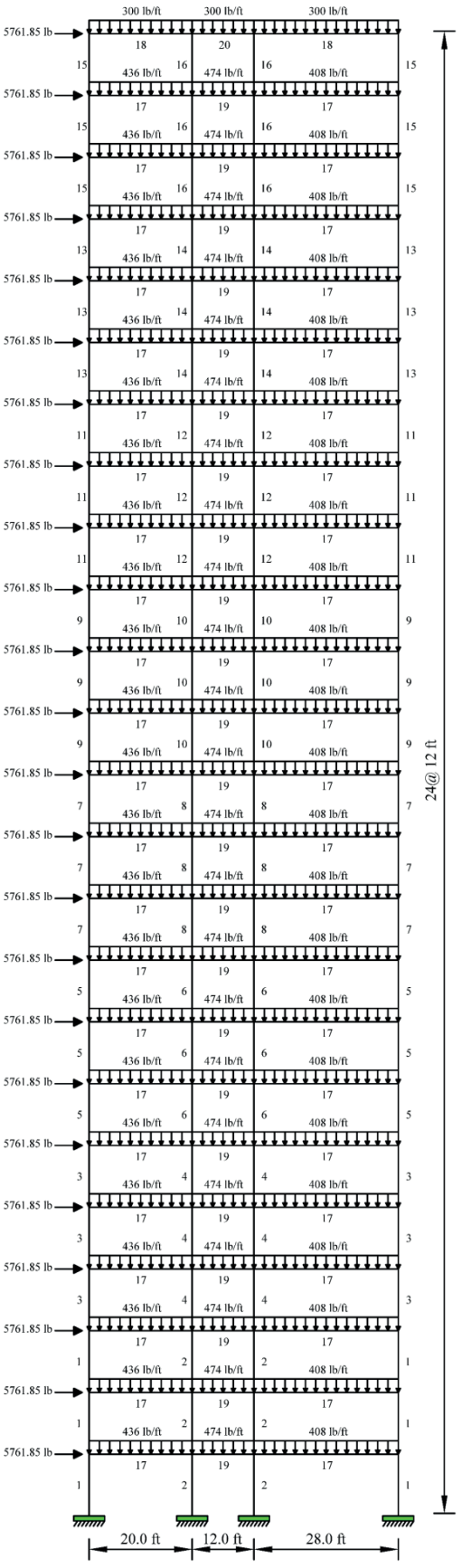

Fig. 7 The schematic of the 3-bay 24-story steel frame

(b) *The inter-story displacements*

$$\frac{d_i}{h_i} - R_i \le 0\,, \tag{11}$$

where d_i, h_i, and R_i are inter-story drift, the story height, and allowable inter-story drift index of the ith story.

(c) *The strength constraints*

$$\begin{aligned} &\frac{P_u}{2\phi_c P_n} + \left(\frac{M_{ux}}{\phi_b M_{nx}} + \frac{M_{uy}}{\phi_b M_{ny}}\right) - 1 \le 0; \ \ if \ \ \frac{P_u}{\phi_c P_n} < 0.2, \\ &\frac{P_u}{\phi_c P_n} + \frac{8}{9} \times \left(\frac{M_{ux}}{\phi_b M_{nx}} + \frac{M_{uy}}{\phi_b M_{ny}}\right) - 1 \le 0; \ \ if \ \ \frac{P_u}{\phi_c P_n} \ge 0.2, \end{aligned} \tag{12}$$

where P_u is the required tension or compression strength; ϕ_c is the resistance factor, and it is equal to 0.9 for tension and 0.85 for compression; M_{ux} and M_{uy} are respectively the required flexural strength in the x and y directions, while M_{nx} and M_{ny} are the nominal flexural strength in the x and y directions, respectively, and M_{ny} is equal to zero due to a two-dimensional structure. ϕ_b is the flexural resistance reduction factor, and its value is set to 0.9.

From Table 3, it can see that the ERao-2 found the smallest weight among the other methods [27–29]. The weights found by ERao-1 and ERao-2 are better than those found by Rao-1 and Rao-2, respectively. Although there is no significant difference between the required number of function evaluations of the Rao and ERao algorithms, the average and standard deviation of the results found by the ERao algorithms are significantly better than Rao algorithms. Also, the worst weight found by ERao algorithms is significantly smaller than Rao algorithms, as shown in Figs. 8 and 9. This difference in the statistic results indicate that the Rao algorithms are trapped in the local optimum. However, the ERao algorithms can easily escape from the local optimum and find better results. The convergence histories of the Rao and ERao algorithms for finding the best solution is given in Fig. 10.

4.3 A 1016-bar double-layer grid

The last example of this study investigates a 1016-bar double-layer grid with the configuration of the square on the diagonal grid. This structure has 320 nodes and 12 supports, as shown in Fig. 11. The connections of the members are assumed to be ball-jointed, thus the members only sustain axial forces. The top layer joints are subjected to the gravity forces of 30 kN. The material density, modulus of elasticity (E), and yield stress (F_y) are 7833.413 kg/m^3, 205 GPa, and 248.2 MPa, respectively. The cross-sectional

Table 3 Comparative results of the ERao and Rao algorithms with other methods in the 3-bay 24-story frame structure

Element group	Optimal cross-sectional areas (W shapes)						
	ES-DE [27]	Acceleratd WEO [28]	IBH [29]	Present study			
				Rao-1	Rao-2	ERao-1	ERao-2
1	W14×145	W14×159	W14 × 132	W14×159	W14×159	W14×159	W14×159
2	W14×99	W14×132	W14 × 99	W14×109	W14×132	W14×109	W14×132
3	W14×109	W14×99	W14 × 109	W14×99	W14×109	W14×99	W14×109
4	W14×132	W14×109	W14 × 109	W14×74	W14×90	W14×82	W14×74
5	W14×99	W14×68	W14 × 109	W14×68	W14×68	W14×68	W14×68
6	W14×109	W14×38	W14 × 99	W14×61	W14×48	W14×53	W14×38
7	W14×145	W14×30	W14 × 90	W14×34	W14×30	W14×38	W14×34
8	W14×68	W14×22	W14 × 90	W14×22	W14×22	W14×22	W14×22
9	W14×109	W14×90	W14 × 68	W14×90	W14×90	W14×90	W14×90
10	W14×68	W14×99	W14 × 74	W14×109	W14×99	W14×109	W14×99
11	W14×48	W14×99	W14 × 53	W14×99	W14×90	W14×99	W14×90
12	W14×68	W14×74	W14 × 53	W14×99	W14×82	W14×90	W14×90
13	W14×38	W14×68	W14 × 30	W14×74	W14×68	W14×74	W14×68
14	W14×61	W14×61	W14 × 38	W14×48	W14×53	W14×53	W14×61
15	W14×30	W14×34	W14 × 22	W14×34	W14×34	W14×30	W14×34
16	W14×22	W14×22	W14 × 22	W14×22	W14×22	W14×22	W14×22
17	W30×90	W30×90	W30 × 90	W30×90	W30×90	W30×90	W30×90
18	W21×55	W8×18	W6 × 16	W8×18	W8×18	W8×18	W8×18
19	W21×48	W24×55	W24 × 55	W24×55	W24×55	W24×55	W24×55
20	W10×45	W6×8.5	W6 × 9	W6×8.5	W6×8.5	W6×12	W6×8.5
Best weight (lb)	212479.17	202194.02	208719	201978.03	201618.03	201588.04	201186.03
NFE	12500	11300	10000	19480	14940	19740	13120
Mean weight (lb)		203412.88	211471	210272.24	206672.81	204450.89	203594.58
Standard deviation (lb)		N/A	2934	8395.97	6654.71	1669.76	1978.29

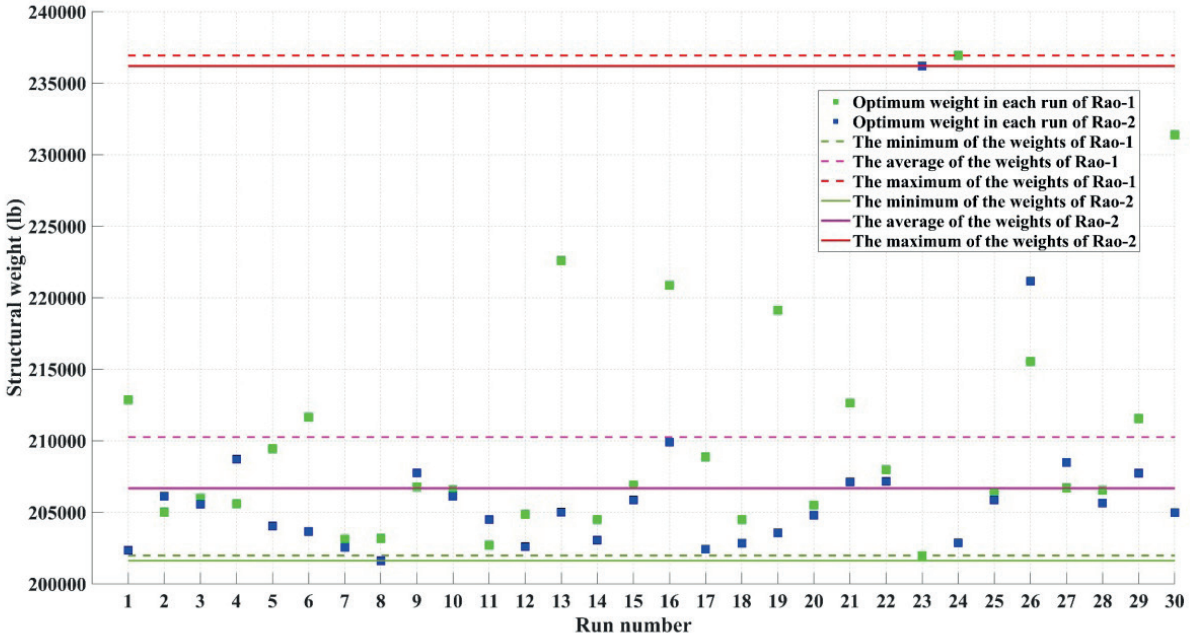

Fig. 8 The structural weight of each independent run of Rao algorithms for the 3-bay 24-story frame structure

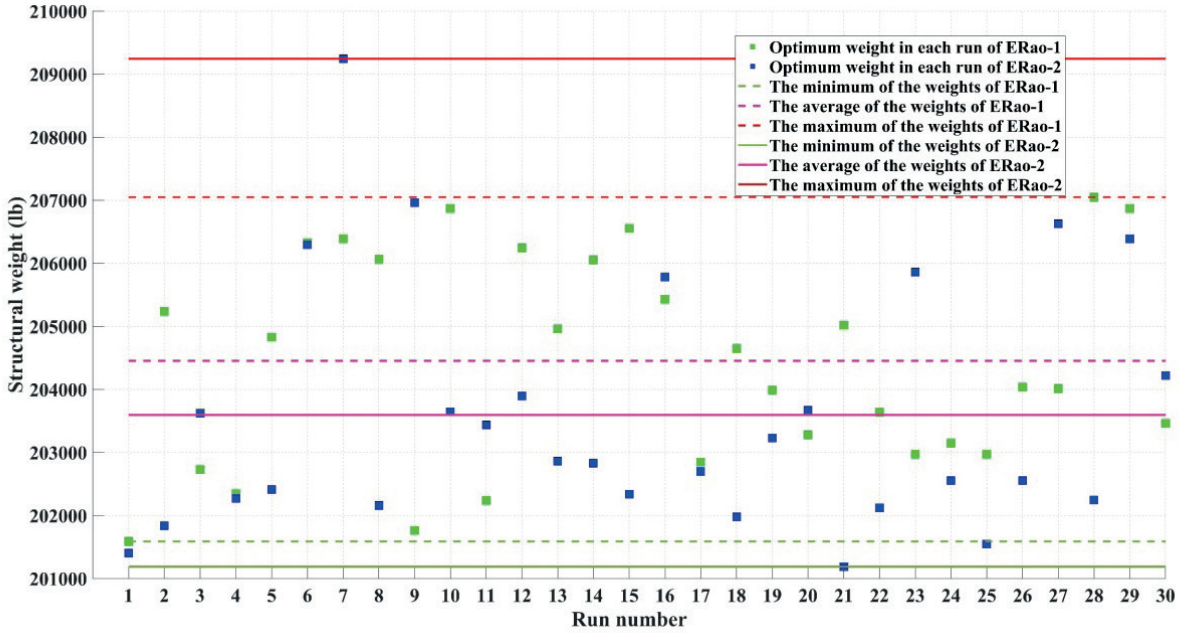

Fig. 9 The structural weight for each independent run of the ERao algorithms for the 3-bay 24-story frame structure

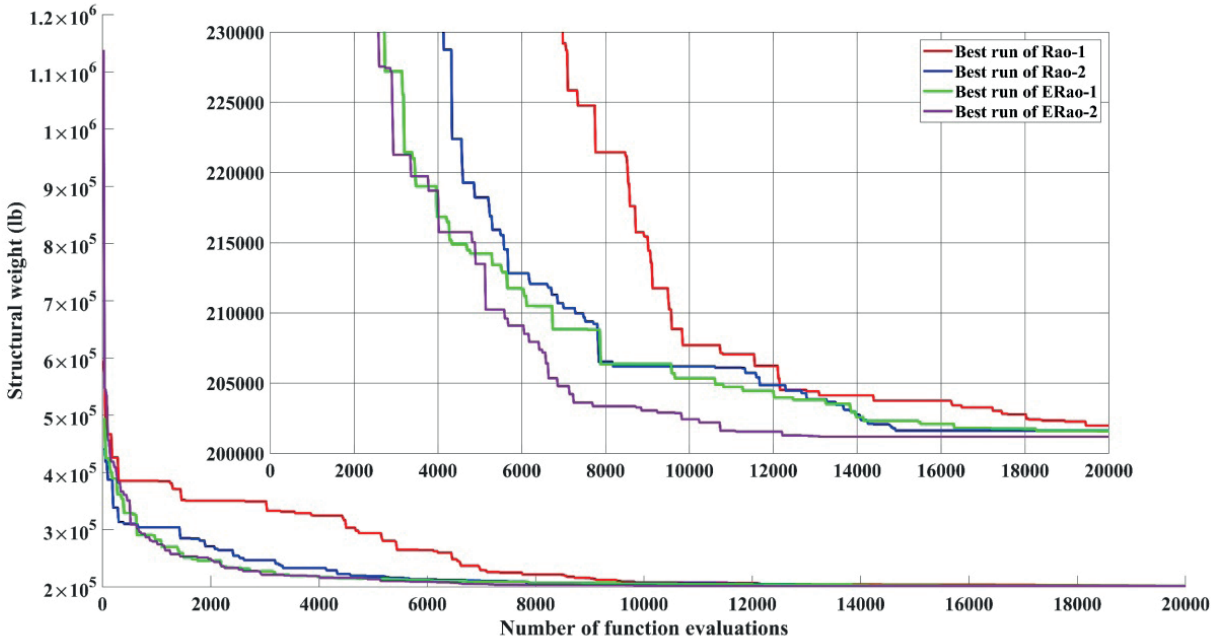

Fig. 10 Convergence histories of the best run of the Rao and ERao algorithms for the 3-bay 24-story frame structure

area of the members is selected from the list of steel pipe sections from AISC-LRFD, which is available in [30]. This structure has three constraints: the vertical displacement of all nodes, the slender ratio of the members, and the stresses for the members.

Displacement of all the nodes are limited to 20/6 cm. Slenderness of the tension members are limited to 300, and slenderness of the compression members are limited to 200. The tension and compression stresses of the members are limited according to AISC-LRFD. The tension constraints of the members are defined using Eq. (14).

$$p_u \le p_r;\ \ p_r = \min \begin{cases} \varnothing_t F_y A_g;\ \varnothing_t = 0.9 \\ \varnothing_t F_u A_e;\quad \varnothing_t = 0.75 \end{cases}, \tag{13}$$

where, p_u and p_r are the required strength and nominal axial strengths; A_g and A_e are the gross cross-sectional area and the effective net cross-sectional area of the members; F_y and F_u are the yield and ultimate tensile stresses.

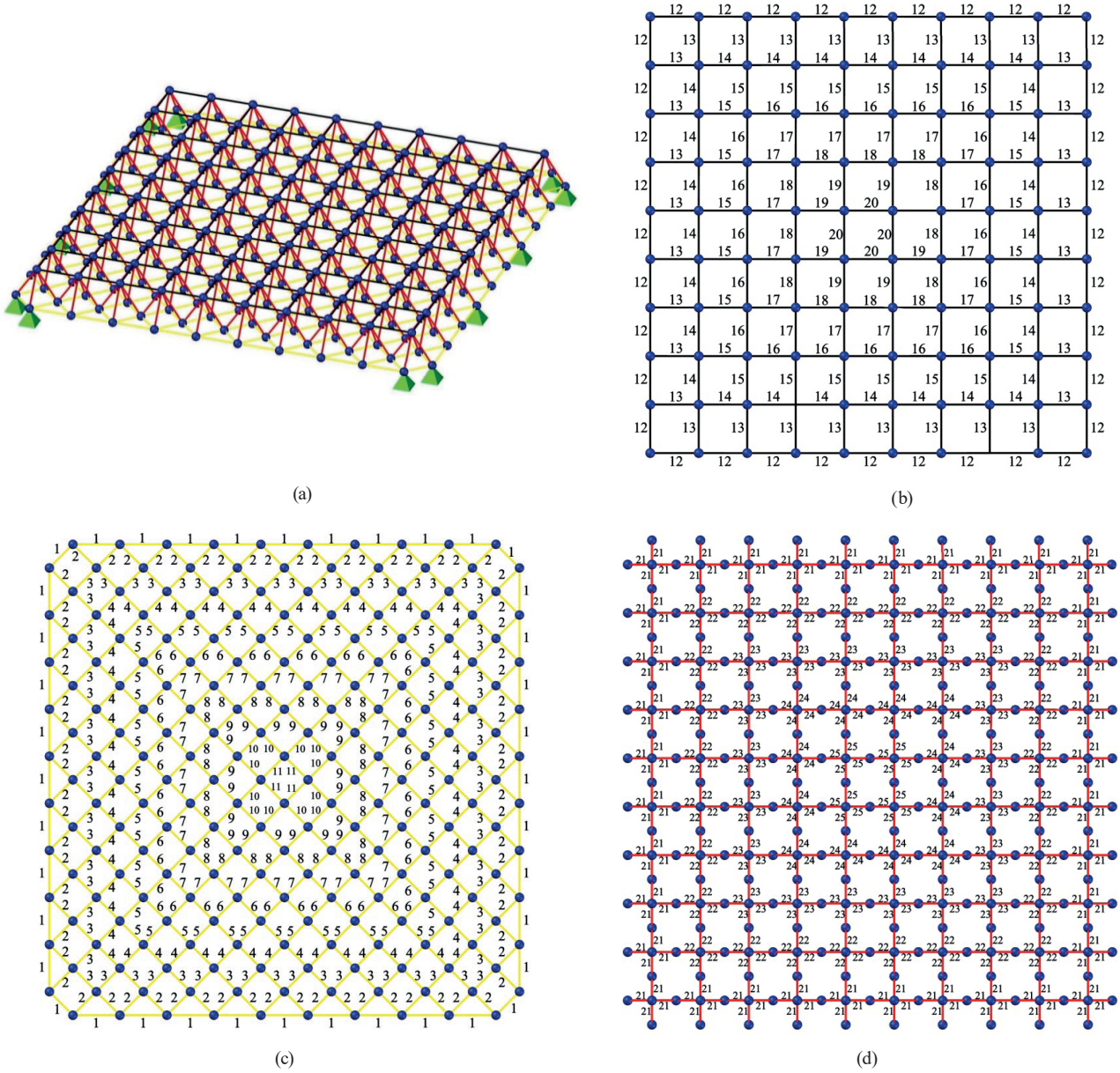

Fig. 11 Schematic of the 1016-bar double layer grid; (a) 3D view, (b) Top layer members, (c) Bottom layer members, and (d) Web members

The compression constraint of the member is defined using Eq. (15).

$$p_u \le p_r;\ \ p_r = \varnothing_c F_{cr} A_g;\ \ \varnothing_c = 0.9\,, \tag{14}$$

$$F_{cr} = \begin{cases} \left(0.658^{\frac{F_y}{F_e}}\right) F_y; & \dfrac{KL}{r} \le 4.71\sqrt{\dfrac{E}{F_y}} \\ 0.877^{*} F_e; & \dfrac{KL}{r} > 4.71\sqrt{\dfrac{E}{F_y}} \end{cases}, \tag{15}$$

$$F_e = \frac{\pi^2 E}{\left(\dfrac{KL}{r}\right)^2},$$

where, F_e and F_{cr} are the elastic bulking stress and critical stress of the members, respectively; L is the length of the member; r is the corresponding radius of gyration; K is the effective length factor, which is taken as 1.

Table 4 compares the results obtained by the Rao and ERao algorithms to some other optimization methods. The result obtained by ERAO-2 is better than those of the other optimization methods [30-31]. Although the ERao algorithms are required more NFE than other stat-art optimization methods, ERao algorithms have significantly better statistic results than other methods. Also, the averages of the 30 independent runs of the ERao algorithms are better than the best run of the other methods. On the other

Table 4 Comparative results of the ERao and Rao algorithms with other in the 1016-bar double layer grid structure

Element group	ECBO [30]	MDVC-UVPS [30]	SSOA [31]	ESSOA [25]	Present study			
					Rao-1	Rao-2	ERao-1	ERao-2
1	EST 5	DEST 4	EST 5	ST 6	EST 6	EST 5	ST 6	ST 6
2	EST 5	DEST 3	ST 5	ST 5	ST 5	DEST 3	EST 4	EST 4
3	ST 3	ST 3½	ST 4	EST 3	ST 3½	EST 3	ST 3½	ST 3½
4	ST 3 ½	ST 2½	EST 2 ½	EST 2 ½	EST 2½	ST 2½	ST 2½	ST 2½
5	ST 2 ½	ST3	ST 3 ½	ST 3	ST 3	ST 2½	EST 2½	ST 2½
6	ST 2	EST 1½	EST 1 ½	EST 1 ½	EST 1½	EST 2	EST 1½	EST 1
7	DEST 2	EST 1½	EST 1 ½	EST 1 ½	ST 2	EST 1¼	EST 2	EST 2
8	DEST 2	EST 2½	EST 1 ½	ST 2 ½	EST 1½	EST 2	DEST 2	ST 3
9	EST 2	ST 3½	ST 4	EST 3	EST 2½	ST 3	ST 3	DEST 2
10	ST 6	DEST 2	DEST 2 ½	EST 2 ½	DEST 2	EST 2½	EST 3	EST 2½
11	ST 2	DEST 2½	ST 2 ½	EST 4	EST 12	DEST 8	DEST 2	ST 2½
12	EST 8	EST 8	ST 10	ST 10	ST 10	ST 12	ST 12	ST 12
13	EST 3 ½	EST 4	EST 4	ST 4	ST 5	ST 4	ST 4	ST 4
14	ST 5	ST 4	ST 4	ST 5	ST 5	ST 5	ST 5	ST 5
15	ST 4	ST 5	EST 4	EST 4	ST 5	EST 4	ST 5	ST 5
16	EST 5	ST 4	ST 6	ST 6	ST 4	ST 4	ST 6	EST 5
17	ST 5	ST 6	ST 5	EST 4	EST 4	ST 6	ST 6	EST 4
18	EST 5	ST 6	EST 5	ST 5	EST 4	ST 8	EST 4	ST 6
19	EST 5	EST 6	DEST 4	EST 6	ST 6	ST 5	EST 5	EST 5
20	ST 8	EST 6	DEST 4	EST 6	EST 8	ST 12	EST 6	DEST 5
21	ST 5	ST 5	ST 6	ST 6	EST 5	ST 5	ST 5	ST 5
22	ST 3	ST 3½	ST 3 ½	ST 3 ½	ST 3½	ST 3	ST 3	ST 3½
23	EST 2 ½	EST 2½	ST 3 ½	ST 3 ½	ST 3½	ST 3½	ST 3½	ST 3½
24	ST 5	ST 2½	ST 2 ½	EST 2 ½	EST 2½	ST 2½	ST 3½	ST 2½
25	ST 4	ST 2½	ST 3 ½	EST 1 ½	EST 2	EST 1½	EST 2	EST 1½
Best Weight (kg)	67,839	65,826	68,398	67,079	71,018	66,089	64,971	64,597
NFE	15,760	3,142	12,020	11,680	20,000	20,000	20,000	20,000
Average weight (kg)	73,042	70,488	72,084	70,408	650,811	828,792	67,200	66,955
Standard deviation (kg)	9,158	5,018	1,802	2703	1,149,928	1,896,425	1,189	1,071

hand, Rao algorithms, same as the previous examples, are trapped in the local optimum. Also, Fig. 12 shows that the Rao algorithms in some of the runs cannot get out of the penalized part of the search space. However, the ERao algorithms do not get trapped in the penalized part of the search space in any of the 30 independent runs, as shown in Fig. 13. The convergence histories of the Rao and ERao algorithms for finding the best solution is given in Fig. 14.

5 Conclusions

This study introduced the enhanced versions of the Rao algorithms, called ERao algorithms. In the ERao algorithms, a modified statistically regenerated mechanism (MSRM) is incorporated. MSRM avoids algorithms from being trapped in the local optima. Also, the technique that keeps the solution in the search space is modified. This change prevents the stuck at the border of the search space. The performance of the ERao algorithms is tested in the three structural design problem. One of these examples includes a 120-bar dome truss design problem with probabilistic constraints. To handle the probabilistic constraint, sequential optimization and reliability assessment-double metaheuristic (SORA-DM) is utilized. The other examples include a 3-bay 24-story frame design problem and a 1016-bar double-layer grid with deterministic constraints.

In all design examples examined, ERao algorithms surpassed Rao algorithms in terms of best weight, average weight, worst weight, and standard deviation. These results

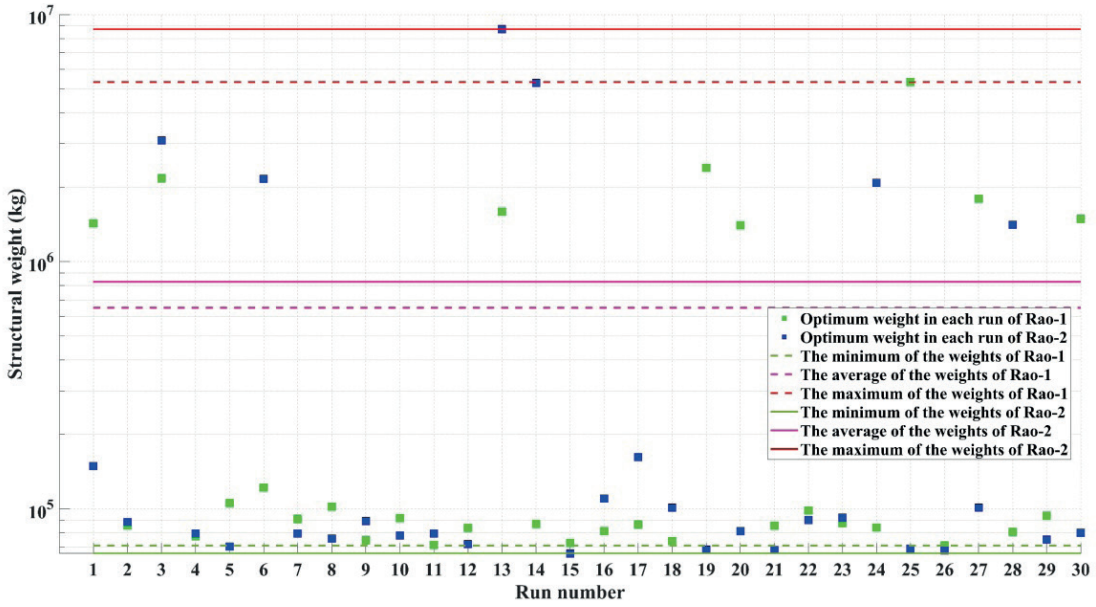

Fig. 12 The structural weight of each independent run of the Rao algorithms in the 1016-bar double layer grid structure

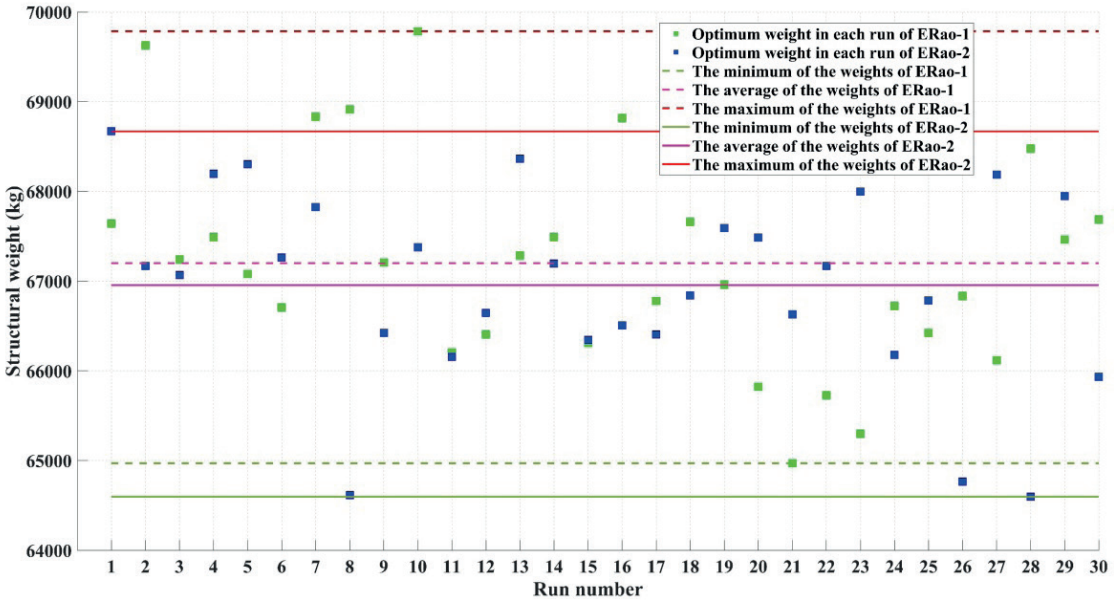

Fig. 13 The structural weight of each independent run of the ERao algorithms in the 1016-bar double layer grid structure

demonstrated that ERao algorithms were superior and more robust than Rao algorithms. Additionally, the second Rao algorithm did not perform reliability assessment in some of the runs and becomes trapped in the local optimum. However, the second ERao method performed reliability evaluation flawlessly in all the runs. Moreover, comparing the results obtained by ERao algorithms to those obtained by certain other state-of-the-art metaheuristics indicated the superiority of ERao algorithms' in optimizing problems with deterministic and probabilistic constraints.

Conflict of interest

No potential conflict of interest was reported by the authors.

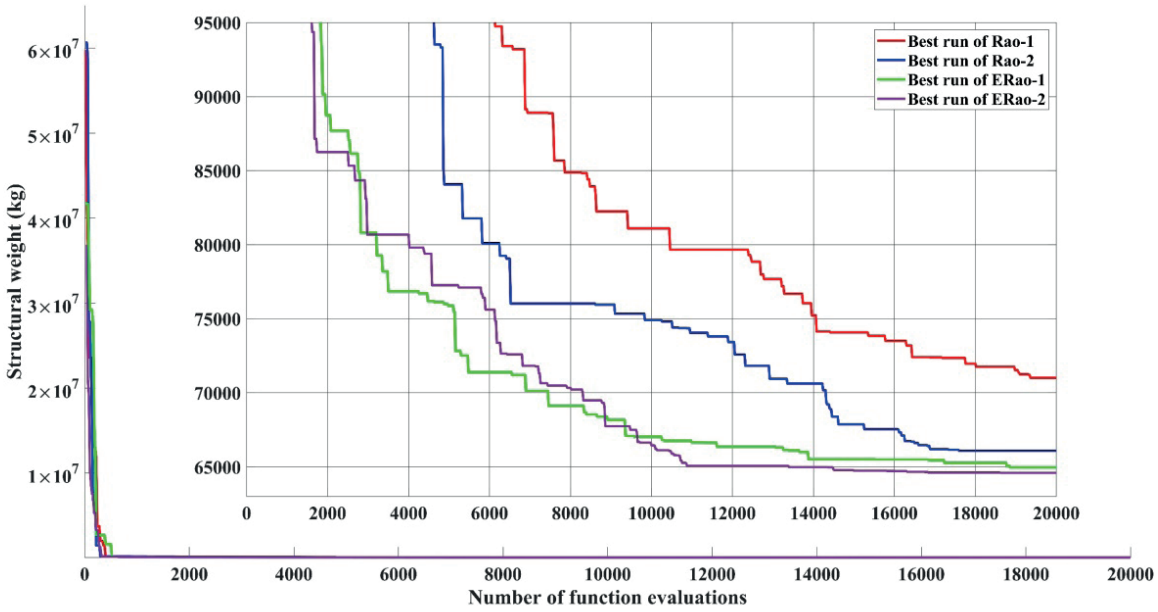

Fig. 14 Convergence histories of the best run of the Rao and ERao algorithms in the 1016-bar double layer grid structure

References

[1] Kaveh, A. "Advances in Metaheuristic Algorithms for Optimal Design of Structures", Springer, Cham, Switzerland, 2021. https://doi.org/10.1007/978-3-319-05549-7

[2] Abualigah, L., Yousri, D., Elaziz, M. A., Ewees, A. A., Al-qaness, M. A. A., Gandomi, A. H. "Aquila Optimizer: A novel meta-heuristic optimization algorithm", Computers & Industrial Engineering, 157, Article number: 107250, 2021. https://doi.org/10.1016/j.cie.2021.107250

[3] Kirsch, U. "Efficient reanalysis for topological optimization", Structural Optimization, 6(3), pp. 143–150, 1993. https://doi.org/10.1007/BF01743505

[4] Al-Betar, M. A., Alyasseri, Z. A. A., Awadallah, M. A., Doush, I. A. "Coronavirus herd immunity optimizer (CHIO)", Neural Computing and Applications, 33, pp. 5011–5042, 2021. https://doi.org/10.1007/s00521-020-05296-6

[5] Kaveh, A., Talatahari, S. "A novel heuristic optimization method: charged system search", Acta Mechanica, 213, pp. 267–289, 2010. https://doi.org/10.1007/s00707-009-0270-4

[6] Wolpert, D. H., Macready, W. G. "No Free Lunch Theorems for Search", Santa Fe Institute, Santa Fe, NM, USA, Working Paper 1995-02-010, 1995. [online] Available at: https://www.santafe.edu/research/results/working-papers/no-free-lunch-theorems-for-search

[7] Kaveh, A., Ilchi Ghazaan, M., Bakhshpoori, T. "An improved ray optimization algorithm for design of truss structures", Periodica Polytechnica Civil Engineering, 57(2), pp. 97–112, 2013. https://doi.org/10.3311/PPci.7166

[8] Kaveh, A., Zakian, P. "Enhanced bat algorithm for optimal design of skeletal structures", Asian Journal of Civil Engineering, 15(2), pp. 179–212, 2014. [online] Available at: https://www.sid.ir/en/Journal/ViewPaper.aspx?ID=353420

[9] Yoo, K.-S., Han, S.-Y. "A modified ant colony optimization algorithm for dynamic topology optimization", Computers & Structures, 123, pp. 68–78, 2013. https://doi.org/10.1016/j.compstruc.2013.04.012

[10] Kaveh, A., Vazirinia, Y. "An Upgraded Sine Cosine Algorithm for Tower Crane Selection and Layout Problem", Periodica Polytechnica Civil Engineering, 64(2), pp. 325–343, 2020. https://doi.org/10.3311/PPci.15363

[11] Kazemzadeh Azad, S., Hasançebi, O., Kazemzadeh Azad, S. "Upper bound strategy for metaheuristic based design optimization of steel frames", Advances in Engineering Software, 57, pp. 19–32, 2013. https://doi.org/10.1016/j.advengsoft.2012.11.016

[12] Kaveh, A., Karimi Dastjerdi, M. I., Zaerreza, A., Hosseini, M. "Discrete Optimum Design of Planar Steel Curved Roof and Pitched Roof Portal Frames Using Metaheuristic Algorithms", Periodica Polytechnica Civil Engineering, 65(4), pp. 1092–1113, 2021. https://doi.org/10.3311/PPci.18425

[13] Latif, M. A., Saka, M. P. "Optimum design of tied-arch bridges under code requirements using enhanced artificial bee colony algorithm", Advances in Engineering Software, 135, Article number: 102685, 2019. https://doi.org/10.1016/j.advengsoft.2019.102685

[14] Abdollahzadeh, B., Barshandeh, B., Javadi, H., Epicoco, N. "An enhanced binary slime mould algorithm for solving the 0–1 knapsack problem", Engineering with Computers, 2021. https://doi.org/10.1007/s00366-021-01470-z

[15] Kaveh, A., Zaerreza, A. "Size/Layout Optimization of Truss Structures Using Shuffled Shepherd Optimization Method", Periodica Polytechnica Civil Engineering, 64(2), pp. 408–421, 2020. https://doi.org/10.3311/PPci.15726

[16] Das, S., Dhang, N. "Damage identification of structures using incomplete mode shape and improved TLBO-PSO with self-controlled multi-stage strategy", Structures, 2021.
https://doi.org/10.1016/j.istruc.2021.07.089

[17] Makiabadi, M. H., Maheri, M. R. "An enhanced symbiotic organisms search algorithm for design optimization of trusses with frequency constraints", Advances in Structural Engineering, 24(14), pp. 3315–3337, 2021.
https://doi.org/10.1177/13694332211026219

[18] Rao, R. V. "Rao algorithms: Three metaphor-less simple algorithms for solving optimization problems", International Journal of Industrial Engineering Computations, 11, pp. 107–130, 2020.
https://doi.org/10.5267/j.ijiec.2019.6.002

[19] Premkumar, M., Babu, T. S., Umashankar, S.,Sowmya, R. "A new metaphor-less algorithms for the photovoltaic cell parameter estimation", Optik, 208, Article number: 164559, 2020.
https://doi.org/10.1016/j.ijleo.2020.164559

[20] Rao, R. V., Pawar, R. B. "Constrained design optimization of selected mechanical system components using Rao algorithms", Applied Soft Computing, 89, Article number: 106141, 2020.
https://doi.org/10.1016/j.asoc.2020.106141

[21] Kalemci, E. N., Ikizler, S. B. "Rao-3 algorithm for the weight optimization of reinforced concrete cantilever retaining wall", Geomechanics and Engineering, 20(6), pp. 527–536, 2020.
https://doi.org/10.12989/gae.2020.20.6.527

[22] Manam, R., Sangu, R., Pamidi, L., Reddy Karri, M. K. "RA 123 s: Three metaphor-less Algorithms for Economic Load Dispatch Solution", Journal of Electrical Engineering & Technology, 2021.
https://doi.org/10.1007/s42835-021-00922-2

[23] Rao, R. V., Pawar, R. B. "Self-adaptive Multi-population Rao Algorithms for Engineering Design Optimization", Applied Artificial Intelligence, 34(3), pp. 187–250, 2020.
https://doi.org/10.1080/08839514.2020.1712789

[24] Hassan, M. H., Kamel, S., Selim, A., Khurshaid, T., Domínguez-García, J. L. "A Modified Rao-2 Algorithm for Optimal Power Flow Incorporating Renewable Energy Sources", Mathematics, 9(13), Article number: 1532, 2021.
https://doi.org/10.3390/math9131532

[25] Kaveh, A., Zaerreza, A., Hosseini, S. M. "An enhanced shuffled Shepherd Optimization Algorithm for optimal design of large-scale space structures", Engineering with Computers, 2021.
https://doi.org/10.1007/s00366-021-01292-z

[26] Kaveh, A., Zaerreza, A. "A new framework for reliability-based design optimization using metaheuristic algorithms", Structures, 38, pp. 1210–1225, 2022.
https://doi.org/10.1016/j.istruc.2022.02.069

[27] Talatahari, S., Gandomi, A. H., Yang, X.-S., Deb, S. "Optimum design of frame structures using the Eagle Strategy with Differential Evolution", Engineering Structures, 91, pp. 16–25, 2015.
https://doi.org/10.1016/j.engstruct.2015.02.026

[28] Kaveh, A., Bakhshpoori, T. "An accelerated water evaporation optimization formulation for discrete optimization of skeletal structures", Computers & Structures, 177, pp. 218–228, 2016.
https://doi.org/10.1016/j.compstruc.2016.08.006

[29] Gholizadeh, S., Razavi, N., Shojaei, E. "Improved black hole and multiverse algorithms for discrete sizing optimization of planar structures", Engineering Optimization, 51(10), pp. 1645–1667, 2019.
https://doi.org/10.1080/0305215X.2018.1540697

[30] Kaveh, A., Ilchi Ghazaan, M. "Meta-heuristic Algorithms for Optimal Design of Real-Size Structures", Springer, Cham, Switzerland, 2018.
https://doi.org/10.1007/978-3-319-78780-0

[31] Kaveh, A., Zaerreza, A. "Shuffled shepherd optimization method: a new Meta-heuristic algorithm", Engineering Computations, 37(7), pp. 2357–2389, 2020.
https://doi.org/10.1108/EC-10-2019-0481