Structures 56 (2023) 104933

Contents lists available at ScienceDirect

Structures

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

Random update particle swarm optimizer (RUPSO): A novel robust optimization algorithm

H. Dadashi, M. Mohammadi *

International Institute of Earthquake Engineering and Seismology, (IIEES), Tehran, Iran

ARTICLE INFO

Keywords:
Structural optimization
Metaheuristic algorithm
Truss structures
Discrete and continuous variables
Random update

ABSTRACT

Achieving an optimal design, especially those with numerous variables, is a tedious process. It is regularly achieved by mathematical programming or Metaheuristic algorithm methods. A novel metaheuristic optimization algorithm is presented in this research. The proposed algorithm is based on the well-known Particle Swarm Optimization (PSO) algorithm, in which the updating process has been improved. The efficiency of the new method, named Random Update Particle Swarm Optimizer (RUPSO), has been verified for problems with continuous variables as well as problems with discrete variables. It is shown that RUPSO converges to better optimum solutions than PSO and other existing metaheuristic algorithms. It is for its capability of searching through the entire feasible search spaces and going over the local optima. Random updating process, which has been utilized in RUPSO, decreases the sensitivity of the algorithm and increases its stability in solving optimization problems. High stability of the method is also shown in the paper. The RUPSO algorithm finds the optimized answers with a very high convergence rate for problems with discrete or even continuous variables.

1. Introduction

In design procedures, there are a lot of variables affecting structural weight or performance. How to achieve the most optimized structure among all feasible designs is a challenging subject. Since the cost of materials is one of the major factors in the construction of a structure, the optimum structure is determined by minimizing the weight of the structural system while considering the restrictions and criteria specified by the design codes. In other words, the optimum structure is usually achieved by selecting the design variables in such a way that the objective function is minimized, while all other design constraints are satisfied [1]. Achieving an optimal design without using optimization algorithms, is an exhausting or even impossible process, especially in structures with numerous design variables. Many great developments in structural optimization took place in the early 1960 s. From then on, various general approaches were developed and adopted for structural optimization. Engineering optimization is carried out using gradient methods and metaheuristic algorithms. While gradient-based methods are more applicable for design variables in continuous space, metaheuristic algorithms could also be employed. For discrete design variables, metaheuristic algorithms are far more effective and practical. One of the advantages of metaheuristic algorithms, which makes them easy to apply, is that derivability of the objective functions are not required. The main idea behind designing a metaheuristic algorithm is to tackle complex optimization problems where other optimization methods fail to be effective. These methods are now recognized as one of the most practical approaches for solving many real-world problems. Metaheuristic methods are the most general form of stochastic optimization applied to engineering problems. In recent years, metaheuristics are the most wide-used means of achieving optimized solutions in complex optimization problems. There are three main categories in structural optimization: (a) sizing optimization (cross-sectional areas of the members considered as design variables [2–6]), (b) shape optimization (nodal coordinates considered as design variables [4,5]), and (c) topology optimization (the location of links - where nodes are connected - considered as design variables [5–7]). Numerous metaheuristic algorithms have been introduced throughout the years, among which are the genetic algorithm (GA) [8], ant colony optimization (ACO) [9], particle swarm optimization (PSO) [10], harmony search (HS) [11], big bang-big crunch (BB-BC) [12], grey wolf optimization [13], Colliding Bodies Optimization [14] and firefly algorithm (FA) [15]. Lamberti and Pappalettere [16] gave a comprehensive review of metaheuristics and their applications in the field of structural optimization. The use of metaheuristic algorithms for damage detection [17], evaluating the

* Corresponding author.
E-mail addresses: hashem.dadashi@stu.iiees.ac.ir (H. Dadashi), Mohammadi@iiees.ac.ir (M. Mohammadi).

https://doi.org/10.1016/j.istruc.2023.104933
Received 5 February 2023; Received in revised form 13 June 2023; Accepted 19 July 2023
Available online 11 August 2023

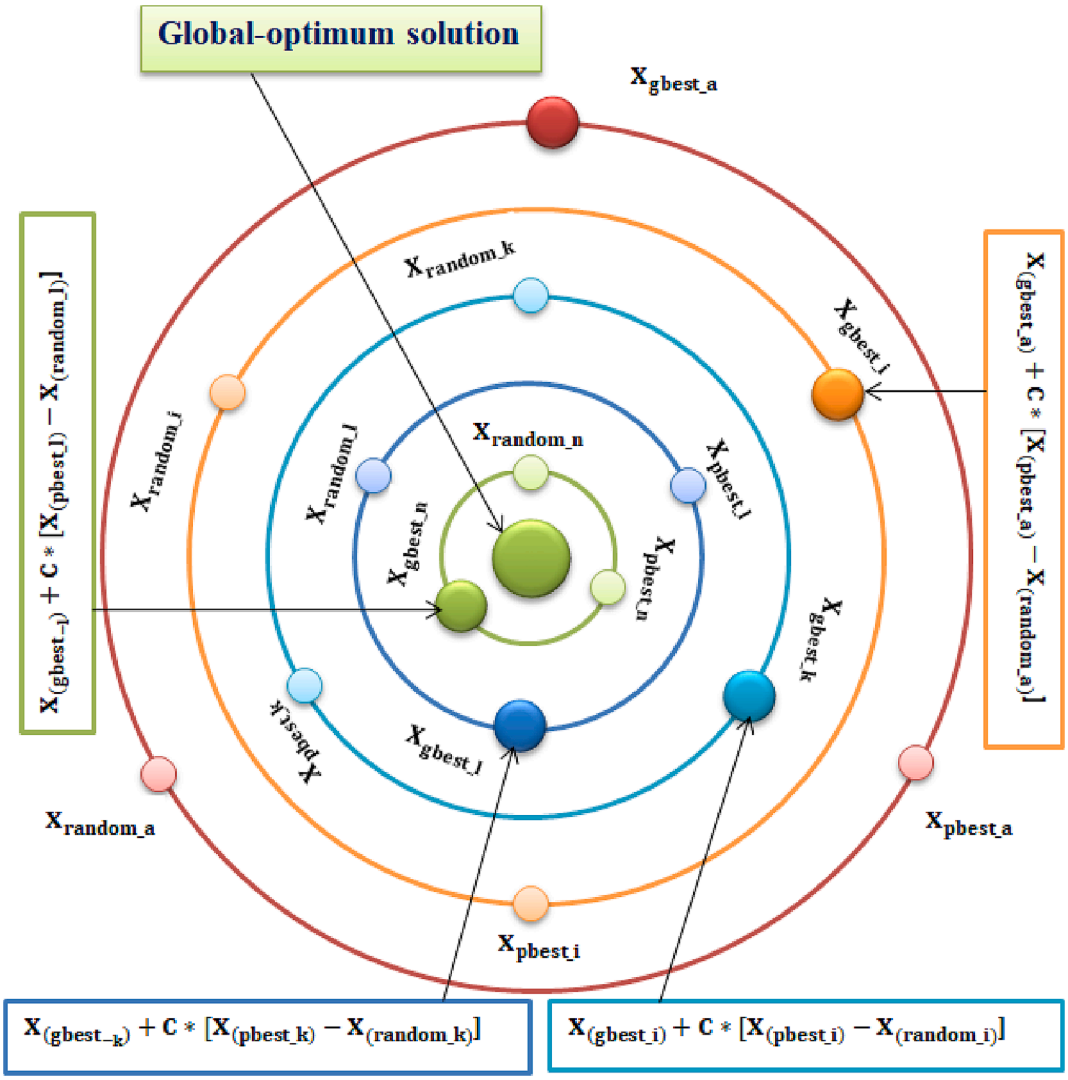

Fig. 1. Schematic view of random updating of RUPSO algorithm.

Table 1
Comparison RUPSO with existing algorithms for example No.1 with continuous variable.

Objective function value	Past Studies and various algorithms						Present study
	EA [37]	CDE [38]	FSA [39]	GA [40]	NM-PSO [41]	P SOLVER [42]	RUPSO
F(x) =	−6961.81388	−6961.81388	−6961.81388	−6961.81388	−6961.8240	−6961.8244	−6961.926207

Table 2
Optimal solutions for example No.2 with continuous variable.

Objective function value	Past Studies and various algorithms						Present study
	HM [42]	EP [43]	SR [44]	EECA [45]	HPSO [46]	P SPLVER [42]	RUPSO
F(x) =	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825041

performance of structural components [18], predicting the behavior of structural systems [19], and performance-based optimum design of structures [30] is very useful and has become more popular. Ma et al. [18] showed that the GWO-ELM algorithm had a better performance in evaluating the performance of composite beams than the GWO and ELM algorithms. Morasaei et al. [19] have simulated the behavior of steel and concrete composite roof system at high temperatures using meta-heuristic algorithms. They showed that the ELM-GWO technique had a better prediction of composite roof system response than the ELM-PSO technique. Combining different metaheuristic algorithms improves the optimization procedure, leading to a more efficient method for solving complex optimization problems, especially those involving structures with discrete variables [20–29]. Gholizadeh and Dadashi [30] have used metaheuristic algorithms for performance-based optimum design of steel moments frames (SMFs). They have shown that the PSO algorithm performed better than the genetic algorithm (GA), ant colony optimization (ACO), and harmony search (HS) algorithms. The PSO has been proposed to simulate the motion of bird swarms as a part of a socio-

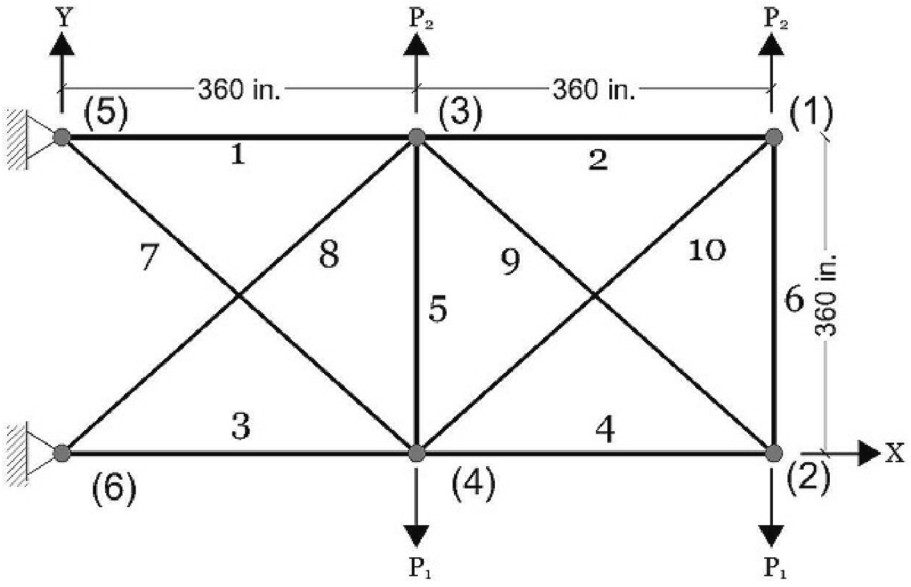

Fig. 2. 10-bar truss structure.

Table 3
Comparison of optimal designs for the 10-bar truss structure (Case 1).

Element Group	Past Studies and various algorithms						Present study
	PSOPC [47]	HS [48]	HPSO [49]	PSO [48]	IPSO [1]	FA [48]	RUPSO
A1	30.00	30.00	30.00	33.50	33.50	33.50	33.50
A2	1.80	1.8	1.62	1.62	1.62	1.62	1.62
A3	26.50	22.90	22.90	22.90	22.90	22.90	22.90
A4	15.50	13.50	13.50	14.20	14.20	14.20	14.20
A5	1.62	1.62	1.62	1.62	1.62	1.62	1.62
A6	1.62	1.62	1.62	1.62	1.62	1.62	1.62
A7	11.50	11.50	7.97	7.97	7.97	7.97	7.97
A8	18.80	22.00	26.50	22.90	22.90	22.90	22.90
A9	22.00	22.90	22.00	22.00	22.00	22.00	22.00
A10	3.09	1.92	1.80	1.62	1.62	1.62	1.62
Weight (lb)	5593.44	5544.6	5531.98	5490.74	5490.74	5490.74	5490.74

Table 4
Comparison of optimal designs for the 10-bar truss structure (Case 2).

Element Group	Past Studies and various algorithms						Present study
	PSOPC [47]	HS [48]	HPSO [49]	PSO [48]	IPSO [1]	FA [48]	RUPSO
A1	25.5	28.5	31.5	29.5	29.5	31.0	31.0
A2	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A3	23.5	21.5	24.5	24.0	24.0	23.0	23.0
A4	18.5	14.0	15.5	15.0	15.0	15.0	15.0
A5	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A6	0.5	0.5	0.5	0.5	0.5	0.5	0.5
A7	7.5	8.0	7.5	7.5	7.5	7.5	7.5
A8	21.5	23.0	20.5	21.5	21.5	21.0	21.0
A9	23.5	23.5	20.5	21.5	21.5	21.5	21.5
A10	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Weight (lb)	5133.16	5104.9	5073.51	5067.33	5067.33	5060.6	5060.6

cognitive study. The appropriate exploration of the PSO algorithm compared to other algorithms motivates researchers to apply this algorithm in solving complex optimization problems. However, PSO has a low convergence rate. To improve search efficiency, researchers have developed multiple PSO variants [31,32]. Jafari and Salajegheh et al. [32] have improved the search efficiency of the PSO algorithm by using a hybrid method in the optimization of the truss structures. They have shown that the use of the proposed method leads to a more optimal solution than the PSO algorithm. In this paper to improve the search efficiency, a new method called the Random Update Particle Swarm Optimization (RUPSO), is proposed. The method is based on the Particle Swarm Optimization (PSO) algorithm which itself is based on the social behavior of animals such as fish, insects, and bird flocking. Since the RUPSO algorithm searches for the optimized answer among the existing population, it is capable of going through the entire search space. RUPSO algorithm does not produce additional population in any step to bypass the issue of local optimum points, meaning that it can go over local optimal points, conducting a fast and efficient search with the lowest computational cost. RUPSO is a novel robust metaheuristic algorithm for the optimization of both discrete and continuous variables.

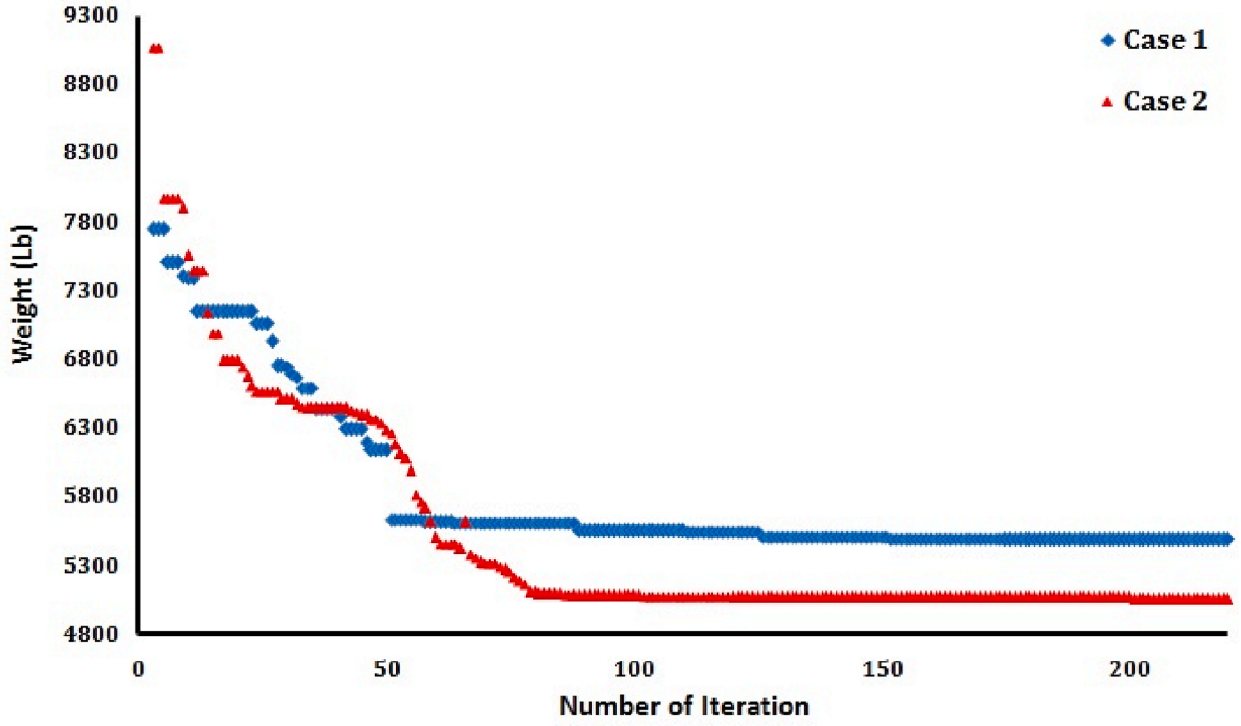

Fig. 3. Weight (lb) evolution history for the 10-bar truss using RUPSO: Case 1 and Case 2.

It is shown that the proposed method produces results that are more optimized than those of previous optimizers. The concept and formulation of the method are presented and how to select the required parameters is completely described. The proposed method is validated by finding optimal solutions for benchmark examples and it is shown that it converges to better answers than other algorithms.

2. Problem formulation

In structural optimization problems, the main purpose of optimization is to minimize an objective function, while observing some constraints/conditions. This problem can be expressed as follows:

$$\text{To minimize objective function } F(\{X\}) = \sum_{i}^{ne} \rho_i . A_i . L_i \tag{1}$$

$$\text{Subject To } g_i(X) \le 0, i = 1, ..., m \tag{2}$$

$$X_j^l \le X_j \le X_j^u, j = 1, ..., n \tag{3}$$

where X is the vector of design variables; F(X) is the objective function that has to be minimized;ρ_i, A_i and L_i are the material density, cross-sectional area, and length of the i^{th} structural element, respectively. $g_i(x)$ is the i^{th} behavioral constraints, and X_j^l and X_j^u are the lower and upper bounds on a typical design variable namely X_j. To minimize the objective function in relation.1, each of the parameters A and L or both of them can be considered as variables of the optimization problem. In this study, the nodal coordinates in the finite element model of structural examples are considered fixed and only the parameter A is considered as objective function variable. Therefor, to minimize the objective function, the size of the structural element is designed optimally, so that none of the displacement of the nodes and the stress of the elements exceed their limits. Parameter A can have different combinations of X vectors as design variables. For each of the design vectors, there is a possibility of violating the constraints of the optimization problem. Instead of applying constraints to different combinations of design vectors, depending on the amount of violation of the constraints of the problem, the objective function corresponding to that vector is penalized. The amount of the penalty should be enough large to improve the composition of the variables of the penalized vectors during the optimization process. The most famous method of imposing a penalty is the external penalty function, which is proportional to the square of the problem constraint violation. In this study, the exterior penalty function method (EPFM) [8] has been employed to transform the constrained structural optimization problem into an unconstrained one. For this purpose, the following formula is applied:

$$\Phi(X, r_p) = F(X) + r_p \sum_{i=1}^{m} [\max\{0, g_i(X)\}]^2 \tag{4}$$

where, Φ and r_p are the pseudo-objective function and the positive penalty parameter, respectively.

3. RUPSO algorithm

3.1. Basic concepts of PSO as the basis of RUPSO

PSO was originally proposed to simulate the motion of bird swarms as a part of a socio-cognitive study. The PSO makes use of several particles (collectively called a "swarm") that are randomly initialized in the search space. Each particle in the swarm represents a potential solution for the optimization problem. The particles fly through the search space, with their positions being constantly updated based on the best position of individual particles and that of the entire swarm in each iteration. The objective function is evaluated for each particle at each grid point, and the fitness values of the particles are obtained to determine the best position in the search space [33]. In iteration t, the swarm is updated using the following equations:

$$X_i^{t+1} = X_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (P_g^t - X_i^t) \tag{5}$$

$$V_i^{t+1} = \omega^t V_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (P_g^t - X_i^t \tag{6}$$

$$X_i^{t+1} = X_i^t + V_i^{t+1} \tag{7}$$

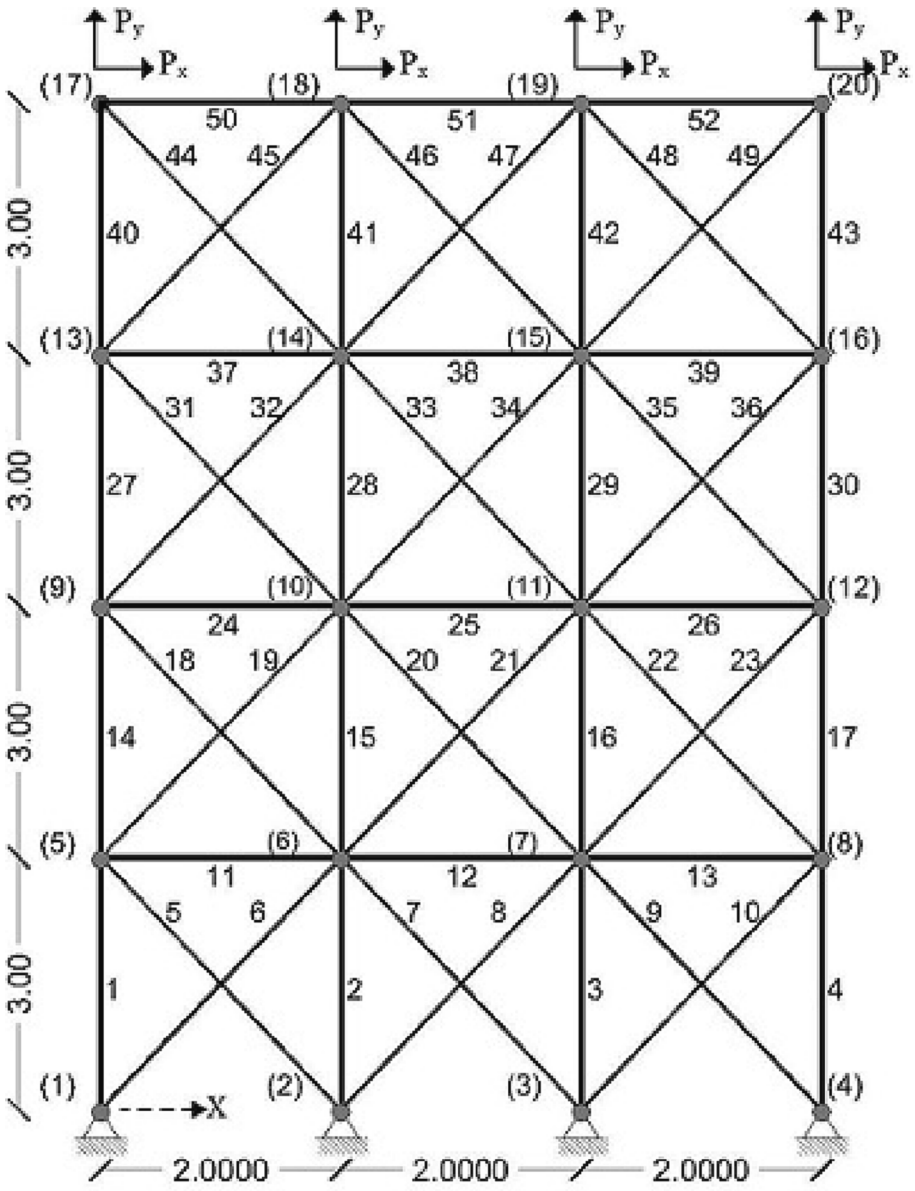

Fig. 4. 52-bar planar truss.

where X_i and V_i represent the current position and the velocity of the i^{th} particle, respectively. P_i is the best position of the i^{th} particle in the previous iteration (pbest) and P_g is the best global position of all the particles in the swarm (gbest, r_1 and r_2 are two uniform random sequences generated within the interval [0, 1]; c_1 and c_2 are the cognitive and social scaling parameters, respectively. The inertia weight used to discount the previous velocity of the particle preserved is represented by the parameter ω. Parameter ω controls the movement speed of the particles toward the optimal points and prevents large fluctuations of particles around the optimal points.

Due to the importance of ω in achieving an efficient search, the updating criterion can be expressed as follows [10]:

$$\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{t_{max}} \tag{8}$$

where ω_{max} and ω_{min} are the maximum and minimum values of ω, respectively. Also, t_{max} and t are, respectively, the numbers of maximum iterations and present iterations.Many successful applications of PSO have been reported in the literature on structural engineering, some of them can be found in [34–36,56].

3.2. Concepts of RUPSO

3.2.1. Introduction to RUPSO

To solve optimization problems with discrete design variables, different combinations of variables cause different minimum solutions for the objective function. Therefor, in finding optimal design of structures, there are several local optimal solutions. As mentioned, the Random Update Particle Swarm Optimization (RUPSO) is based on the PSO algorithm, in which random updating and mutation procedures are improved. Furthermore, for random updating of the design vectors, entire design space is searched which leads to achieving more optimal solution with higher convergence rate.

3.2.2. Random update procedure

Random updating procedure in RUPSO generates a new mutated position vector based on the following relationships:

$$X(i)_{(t+1)} = X(X_g)_{(t)} + C^*(X(X_p)_{(t)} - X_r(i)) \tag{9}$$

Table 5
The available cross-section areas of the AISC norm.

No $in^2 mm^2$	No. $in^2 mm^2$	No. $in^2 mm^2$	No. $in^2 mm^2$
1 0.111 71.613	17 1.563	33 3.840	49 11.500
2 0.141 90.968	1008.385	2477.414	7419.340
3 0.196	18 1.620	34 3.870	50 13.500
126.451	1045.159	2496.769	8709.660
4 0.250	19 1.800	35 3.880	51 13.900
161.290	1161.288	2503.221	8967.724
5 0.307	20 1.990	36 4.180	52 14.200
198.064	1283.868	2696.769	9161.272
6 0.391	21 2.130	37 4.220	53 15.500
252.258	1374.191	2722.575	9999.980
7 0.442	22 2.380	38 4.490	54 16.000
285.161	1535.481	2896.768	10322.560
8 0.563	23 2.620	39 4.590	55 16.900
363.225	1690.319	2961.284	10903.204
9 0.602	24 2.630	40 4.800	56 18.800
388.386	1696.771	3096.768	12129.008
10 0.766	25 2.880	41 4.970	57 19.900
494.193	1858.061	3206.445	12838.684
11 0.785	26 2.930	42 5.120	58 22.000
506.451	1890.319	3303.219	14193.520
12 0.994	27 3.09	43 5.740	59 22.900
641.289	1993.544	3703.218	14774.164
13 1.000	28 3.130	44 7.220	60 24.500
645.160	2019.351	4658.055	15806.420
14 1.228	29 3.380	45 7.970	61 26.500
792.256	2180.641	5141.925	17096.740
15 1.266	30 3.470	46 8.530	62 28.000
816.773	2283.705	5503.215	18064.480
16 1.457	31 3.550	47 9.300	63 30.000
939.998	2290.318	5999.988	19354.800
	32 3.630	48 10.850	64 33.500
	2341.931	6999.986	21612.860

If

$$F(X(i)_{(t+1)}) < F(X(i)_{(t)}) => (X(i)_{(new)}) = X(i)_{(t+1)} \quad (10)$$

Else

$$X(i)_{(new)}) = X(i)_{(t)} \quad (11)$$

where C is a number between 0.5 and 1 and depends on the domain of the design variables. If the design variables are close to each other, C takes on a value close to 0.5 and if the design variables are far from each other, it is closer to 1.

$X_r(i)$ is an arbitrary position vector among the existing vectors.

An unlimited number of acceptable answers could exist in the search space, with some being better than others. Each vector in the population matrix conserves its best location to reach a better answer. Until a specific iteration, the best location of each vector is shown by $X(x_p)$, and The best answer among vectors is shown by $X(x_g)$.

The Vector with the best location is used as the benchmark for the swarm to reach a better point. In the end, a better location is found for each vector. The purpose of this approach is to generate reliable solutions and prevent the algorithm from getting stuck on the local extreme. This way, the location of the best vector is used as the best point. The mutation equation is iterated over for each of the vectors, with the objective function being investigated at each iteration. When a point better than the previous iteration is found, it is considered as the main point and the mutation process is repeated. Using the information regarding the best locations of vectors inside the population matrix, all of the vectors of the matrix gradually approach the optimum due to the comparison and updating procedures in each iteration. Finally, the vectors of the matrix circle around a better answer, and with several iterations, they approach a much better state. The procedure by which different particles are updated and finally reach the optimum points - i.e., g_{best} and P_{best} - is displayed in Fig. 1. As it is obvious, every level can be a local optimum point and by applying the mutation equation and iterating of the optimization procedure, the g_{best} and P_{best} are updated and eventually reach their optimum values. These minimum/maximum values are better than the ones obtained in the previous levels.

3.2.3. Main steps in the RUPSO

The main steps of the RUPSO algorithm are as follows:

(a): Initialize a swarm of particles by a uniform random selection procedure of the particles from the design space.

(b): Analyzing the swarm to evaluate the fitness value of each particle.

In this step, the objective function corresponding to each of the population vectors is obtained. Also, it is checked whether each of the constraints of the problem has been violated for each of the vectors. In case of violation of any of the constraints, the objective function corresponding to that vector is penalized according to relation (4).

(c): Determine X_{pbest} and X_{gbest} based on the fitness values obtained.

After calculating the objective function in step (b) for each vector, the best position of each vector up to a certain iteration that minimized the objective function, is considered as X_{pbest}. The vector for which the objective function is minimized compared to the objective function corresponding to other vectors considered as X_{gbest}.

(d): Updating X_{pbest} and X_{gbest} based on the application of random update procedure according to formula (9–11).

(e): Repeating (b) to (d) until a termination condition is satisfied.

Based on the initial assumption (assumed swarm of particles in step (a), RUPSO leads to different optimum answers, which are very close to each other. This will be shown in structural examples as follows in this paper.

Table 6
Comparison of results for the 52-bar truss obtained using various algorithms.

Variables (mm^2)	Past studies and various algorithms							Present study
	PSO [50]	PSOPC [50]	SGA [50,51]	HS [50,52]	HPSO [50]	DHPSACO [50]	MBA [50]	RUPSO
A_1–A_4	4658.055	5999.988	4658.055	4658.055	4658.055	4658.055	4658.055	4658.055
A_5–A_{10}	1374.190	1008.380	1161.288	1161.288	1161.288	1161.288	1161.288	1161.288
A_{11}–A_{13}	1858.060	2696.770	645.160	506.451	363.225	494.193	494.193	494.193
A_{14}–A_{17}	3206.440	3206.440	3303.219	3303.219	3303.219	3303.219	3303.219	3303.219
A_{18}–A_{23}	1283.870	1161.290	1045.159	940.000	940.000	1008.385	940.000	940.000
A_{24}–A_{26}	252.260	729.030	494.193	494.193	494.193	285.161	494.193	494.193
A_{27}–A_{30}	3303.220	2238.710	2477.414	2290.318	2238.705	2290.318	2283.705	2283.705
A_{31}–A_{36}	1045.160	1008.380	1045.159	1008.385	1008.385	1008.385	1008.385	1008.385
A_{37}–A_{39}	126.450	494.190	285.161	2290.318	388.386	388.386	494.193	494.193
A_{40}–A_{43}	2341.93	1283.870	1696.771	1535.481	1283.868	1283.868	1283.868	1283.868
A_{44}–A_{49}	1008.38	1161.290	1045.159	1045.159	1161.288	1161.288	1161.288	1161.288
A_{50}–A_{52}	1045.16	494.190	641.289	506.451	729.256	506.451	494.193	494.193
Weight (kg)	2230.16	2146.63	1970.142	1906.76	1905.495	1904.83	1902.605	1902.605

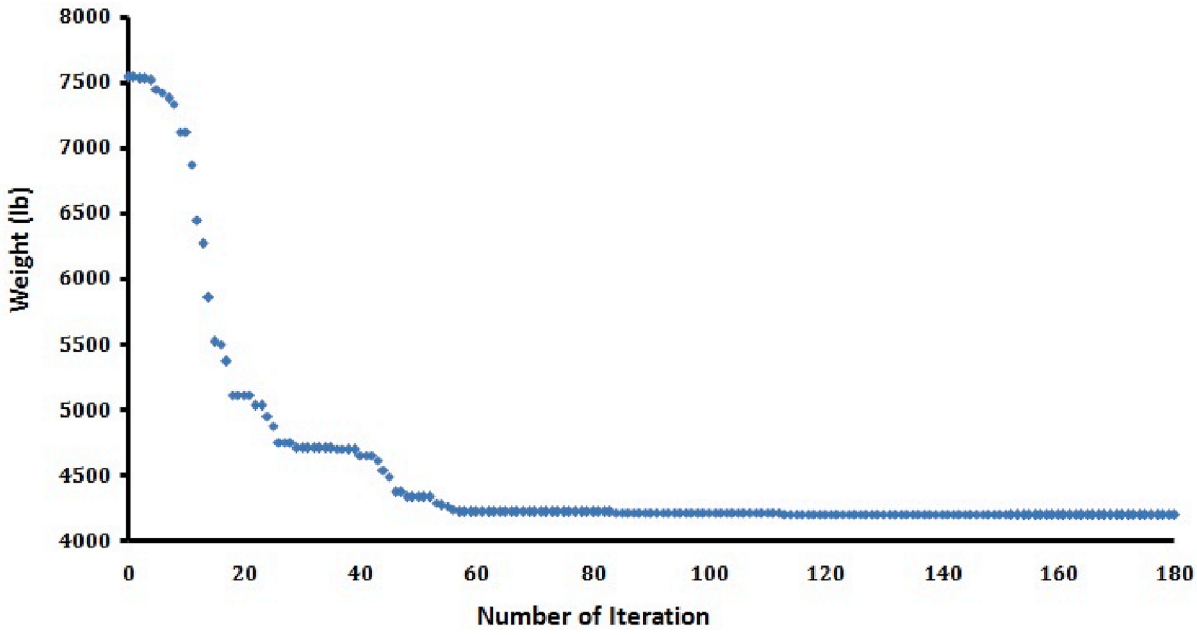

Fig. 5. Weight (lb) evolution history for the 52-bar truss using RUPSO.

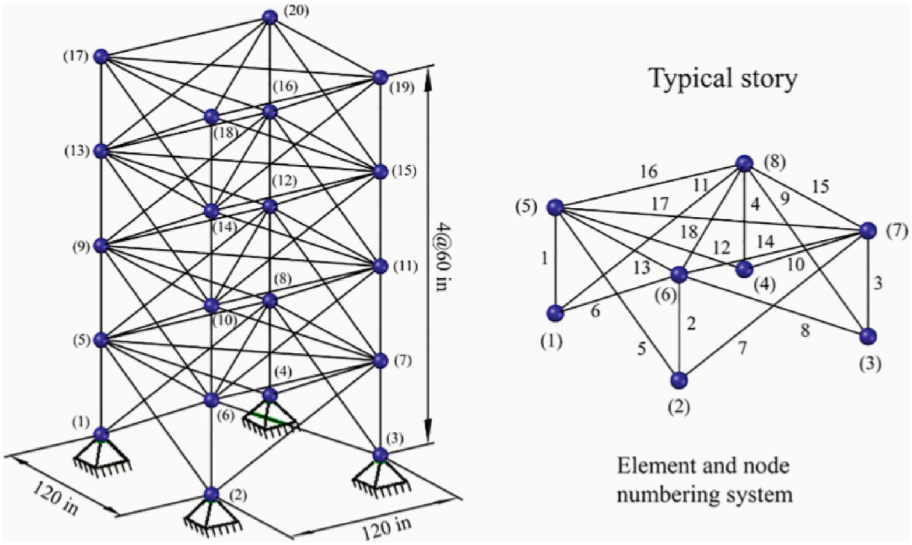

Fig. 6. 72-bar spatial truss.

Table 7
Load cases for the 72-bar spatial truss.

Nodes	Load Case 1			Load Case 2		
	P_x(kips)	P_y(kips)	P_z(kips)	P_x(kips)	P_y(kips)	P_z(kips)
17	5	5	−5	0	0	−5
18	0	0	0	0	0	−5
19	0	0	0	0	0	−5
20	0	0	0	0	0	−5

4. Numerical results

In the present paper, the efficiency and computational performance of the RUPSO algorithm have been evaluated through five optimization problems: Two mathematical equations and three size optimization examples – i.e., two planar truss structures with 10 and 52 bar elements, and a space truss consisting of 72 bar elements. Two mathematical equations have continuous design variables. In equation. (12), the objective was to minimize the function, as well as in the equation. (13), the objective was to maximize the function. Three truss size optimization examples have discrete design variables. In all three truss examples, the objective was to minimize the weight of the trusses. The selected test examples have been solved by other researchers, whose results are compared to the obtained answers. For all examples, the size of the population was considered to be 20 and the maximum number of iterations is 200. MATLAB has been applied for the calculations on a personal computer with a Pentium IV 3000 MHz processor.

4.1. Numerical examples

4.1.1. Optimization problem with continuous variables

Optimization problems with continuous variables, such as harmonic functions or mathematical equations with higher-order variables, have local optimum solutions. It is normally very difficult to achieve a more

Table 8
Comparison of results obtained using different algorithms for Case 1 of the 72-bar truss.

Element Group (elements)	Past studies and various algorithms							Present study
	PSOPC [50]	SGA [50,51]	HPSO [50]	HS [50,52]	PSO [48]	DHPSACO [50]	MBA [50]	RUPSO
A1 (1–4)	3.0	1.5	2.1	1.9	2.0	1.9	2.0	1.9
A2 (5–12)	1.4	0.7	0.6	0.5	0.5	0.5	0.6	0.5
A3 (13–16)	0.2	0.1	0.1	0.1	0.1	0.1	0.4	0.1
A4 (17–18)	0.1	0.1	0.1	0.1	0.1	0.1	0.6	0.1
A5 (19–22)	2.7	1.3	1.4	1.4	1.2	1.3	0.5	1.2
A6 (23–30)	1.9	0.5	0.5	0.6	0.5	0.5	0.5	0.5
A7 (31–34)	0.7	0.2	0.1	0.1	0.1	0.1	0.1	0.1
A8 (35–36)	0.8	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A9 (37–40)	1.4	0.5	0.5	0.6	0.6	0.6	1.4	0.4
A10 (41–48)	1.2	0.5	0.5	0.5	0.5	0.5	0.5	0.6
A11 (49–52)	0.8	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A12 (53–54)	0.1	0.2	0.1	0.1	0.1	0.1	0.1	0.1
A13 (55–58)	0.4	0.2	0.2	0.2	0.2	0.2	1.9	0.1
A14 (59–66)	1.9	0.5	0.5	0.5	0.6	0.6	0.5	0.5
A15 (67–70)	0.9	0.5	0.3	0.4	0.4	0.4	0.1	0.4
A16 (71–72)	1.3	0.7	0.7	0.6	0.6	0.6	0.1	0.5
Weight (lb)	1069.79	400.66	388.94	387.94	385.54	385.54	385.54	372.5486

Table 9
Comparison of results obtained using various algorithms for Case 2 of the 72-bar truss.

Element Group (elements)	Past studies and various algorithms						Present study
	SGA [50,51]	DHPSACO [50]	MBA [50]	PSO [48]	IAOA [21]	IS-Jaya [22]	RUPSO
A1 (1–4)	0.196	1.800	0.196	1.800	1.990	1.990	1.800
A2 (5–12)	0.602	0.442	0.563	0.563	0.563	0.563	0.563
A3 (13–16)	0.307	0.141	0.442	0.111	0.111	0.111	0.111
A4 (17–18)	0.766	0.111	0.602	0.111	0.111	0.111	0.111
A5 (19–22)	0.391	1.228	0.442	1.266	1.228	1.228	1.266
A6 (23–30)	0.391	0.563	0.442	0.442	0.563	0.563	0.442
A7 (31–34)	0.141	0.111	0.111	0.111	0.111	0.111	0.111
A8 (35–36)	0.111	0.111	0.111	0.111	0.111	0.111	0.111
A9 (37–40)	1.800	0.563	1.266	0.563	0.563	0.563	0.6020
A10 (41–48)	0.602	0.563	0.563	0.563	0.442	0.442	0.563
A11 (49–52)	0.141	0.111	0.111	0.111	0.111	0.111	0.111
A12 (53–54)	0.307	0.250	0.111	0.111	0.111	0.111	0.111
A13 (55–58)	1.563	0.196	1.800	0.196	0.196	0.196	0.1960
A14 (59–66)	0.766	0.563	0.602	0.563	0.563	0.563	0.563
A15 (67–70)	0.141	0.442	0.111	0.442	0.391	0.391	0.442
A16 (71–72)	0.111	0.563	0.111	0.602	0.563	0.563	0.563
Weight (lb)	427.203	393.380	390.73	389.45	389.3342	389.3342	389.07

optimum solution if the local optimum solutions are close to each other. In the process of optimization with metaheuristic algorithms, it is possible to achieve different local optimum solutions according to the formation of different variables in particle vectors. Random updates and the possibility of searching in the entire feasible search space is one of the advantages of the RUPSO algorithm to achieve more optimal solutions in solving complex optimization problems with continuous variables.

4.1.2. Example No.1

The first minimization problem, demonstrated by Eq. (12), has two design variables and two inequality constraints:

$$\mathrm{Min}F(x) = (x_1 - 10)^3 + (x_2 - 20)^3 \tag{12}$$

$$s.t.g_1(x) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0 \tag{12a}$$

$$g_2(x) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0 \tag{12b}$$

$$13 \le x_1 \le 100 \tag{12c}$$

$$0 \le x_2 \le 100 \tag{12d}$$

Table 1 compares the results of RUPSO with existing optimization algorithms for example No. 1. The example has an optimal solution at $x = (14.095, 0.84296)$, with a corresponding function value of $F(x) = -6961.81388$. The problem has been previously solved using EA [37], CDE [38], FSA [39], GA [40], NM–PSO [41], and PSOLVER [42] methods. Among those studies, the best solution was reported by Ali Haydar Kayhan et al. [42], arriving at an objective function value of $F(x) = -6961.8244$ using the PSOLVER algorithm. RUPSO algorithm achieved the optimized solution at $x = (14.09494, 0.842861)$, with the corresponding objectvie function value of $F(x) = -6961.926207$. This solution, which is more optimized than other abovementioned solvers, was achieved after 200 iterations.

Therefore, RUPSO algorithm provided a better solution than the other algorithms.

4.1.3. Example No.2

The second example has two design variables and two inequality constraints, as given in Eq. (13):

$$\mathrm{Max}\ F(x) = \frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)} \tag{13}$$

$$s.t.g_1(x) = x_1^2 - x_2 + 1 \le 0 \tag{13a}$$

$$g_2(x) = 1 - x_1 + (x_2 - 4)^2 \le 0 \tag{13b}$$

$$0 \le x_1 \le 10 \tag{13c}$$

$$0 \le x_2 \le 10 \tag{13d}$$

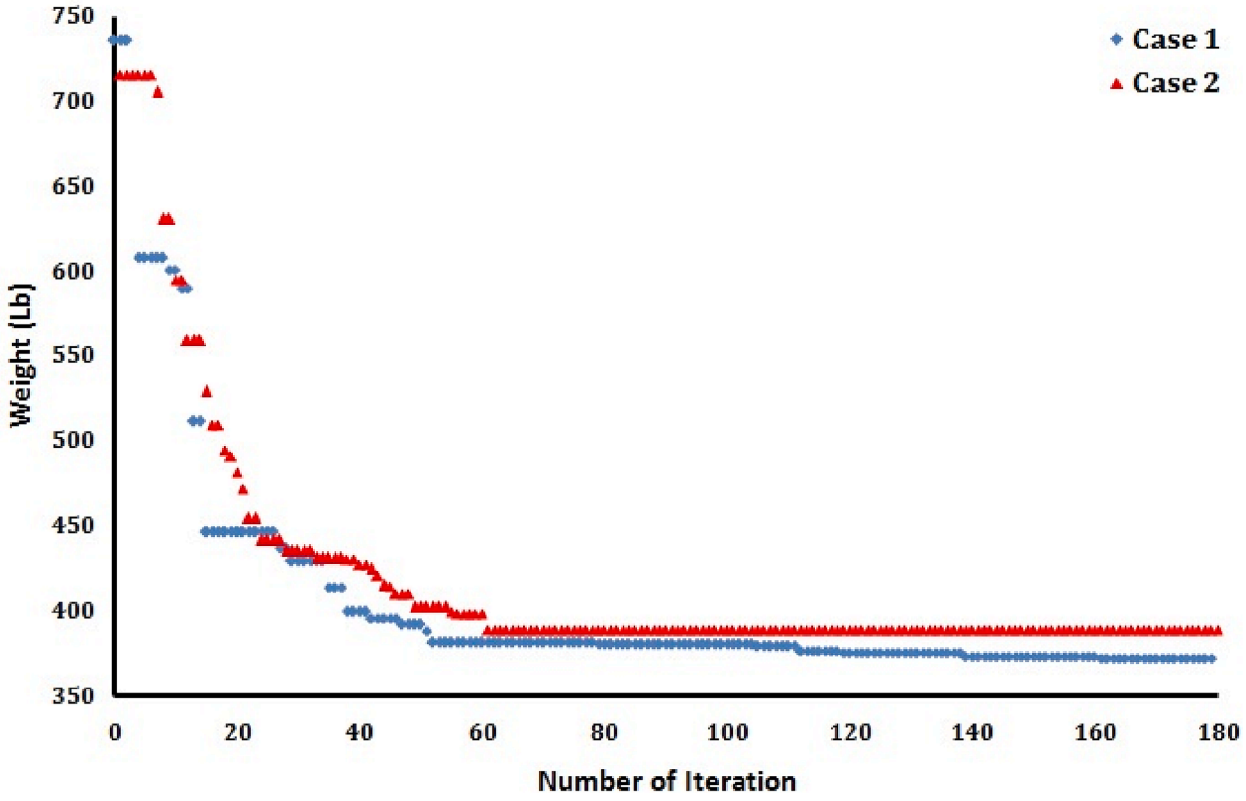

Fig. 7. Weight (lb) evolution history for the 72-bar truss using RUPSO: Case 1 and Case 2.

Table 2 provides the optimum solutions obtained with several algorithms. Example No. 2 has a global optimum at $x = (1.2279713, 4.2453733)$, with a corresponding.

function value of $F(x) = 0.095825$. The function was previously solved using the HM [42], EP [43], SR [44], EECA [45], HPSO [46] and PSOLVER [42] (after 308 function evaluations) algorithms. RUPSO algorithm solved the problem, obtaining an optimum solution a $x = (1.22797135, 4.24537336)$, with a corresponding objective function value of $F(x) = 0.095825041$. The solution was achieved after 250 function evaluations. Therefore, for example 2, RUPSO algorithm provided the same solution with fewer function evaluations.

4.2. Structural optimization with discrete variables

Since truss structures are widely used in structural engineering, optimizing their parameters is always important. Regarding the optimal design of truss structures, the objective is to find the best feasible structure with a minimum weight [20,54,55].

The constraints considered for the truss structures in this article are described as:

$$g_1(x): \frac{F_i}{A_i} = \sigma_i \le \sigma_{allowed} \tag{14}$$

$$g_2(x): D_l \le D_{allowed} \tag{15}$$

where F is the axial force, A is the cross-sectional area of the element, and i is the number of the structure's components. $\sigma_{allowed}$ is the allowable stress in the structure, D_l is the displacement of node l, and $D_{allowed}$ is the allowable displacement of each non-constrained node.

4.2.1. Optimization of a 10-bar planar truss

The 10-bar truss structure is shown in Fig. 2. The loads applied to the truss include $P1 = 10^5$ lbs and $P2 = 0$. The material density is 0.1 lb/in^3 and the modulus of elasticity is 10,000 ksi. The members are subjected to stress limitations of ± 25 ksi. All nodes in both directions are subjected to displacement limitations of ± 2.0. The maximum number of iterations is 200.

10 design variables, areas of the bar members, were considered in this example. They are selected from the set D1 = {1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.63, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.50, 13.50, 13.90, 14.20, 15.50, 16.00, 16.90, 18.80, 19.90, 22.00, 22.90, 26.50, 30.00, 33.50} (in^2). In addition to this case, called Case 1, another case (Case 2) is considered, in which the design variables are selected from the set D2 = {0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0, 10.5, 11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0, 14.5, 15.0, 15.5, 16.0, 16.5, 17.0, 17.5, 18.0, 18.5, 19.0, 19.5, 20.0, 20.5, 21.0, 21.5, 22.0, 22.5, 23.0, 23.5, 24.0, 24.5, 25.0, 25.5, 26.0, 26.5, 27.0, 27.5, 28.0, 28.5, 29.0, 29.5, 30.0, 30.5, 31.0, 31.5} (in^2).

The obtained results of different algorithms for this problem are shown in Tables 3 and 4, for Case 1 and Case 2, respectively. The results show that for both cases, the RUPSO algorithm is one of the best algorithms; the most optimized weight of the considered structure is 5490.74 lb for Case 1 and 5060.6 lb for Case 2. RUPSO is better than PSOPC [47], HS [48], and HPSO [49] for Case 1, and it finds the optimized answers of Case 2 better than PSOPC [47], HS [48], HPSO [49], PSO [48] and IPSO [1] algorithms. Based on the initial assumption (step a, specified in section 3.2.3), RUPSO leads to different optimum answers. Applying RUPSO 10 times leads to different answers between 5490.75(lb) to 5593.44(lb) for case1, and between 5060.6(lb) to 5084.10(lb) for case 2. The evolution history diagrams for the two load cases for the 10-bar truss for the RUPSO algorithm are displayed in Fig. 3.

4.2.2. Optimization of 52-bar planar truss

The 52-bar truss, shown in Fig. 4, was studied by multiple researchers, namely Wu and Chow [37], Lee et al. [40], Li et al. [35],

Kaveh and Talatahari [43], and Sadollah and Bahreininejad [42]. The vertical loads were equal to Px = 100 kN, Py = 200 kN. The material density and the modulus of elasticity are 7860 kg/m^3 and E = 2.07 × 10^5 MPa, respectively. The stress limitation for each member of this structure is equal to ± 180 MPa. This truss has 12 design variables, since its members are divided into 12 groups (members of each group have equal cross-sectional areas): (1) A_1–A_4, (2) A_5–A_{10}, (3) A11–A13, (4) A14–A17, (5) A18–A23, (6) A24–A26, (7) A27–A30, (8) A31–A36, (9) A37–A39, (10) A40–A43, (11) A44–A49, and (12) A50–A52.

The discrete variables could be selected from Table 5, which is based on the data provided by the American Institute of Steel Construction (AISC). In general, the problem has a variable dimensionality of 12, and a constraint dimensionality of 144 (including 52 tension constraints, 52 compression constraints, and 40 displacement constraints). The maximum number of iterations for the problem was 180.

Table 6 shows the results obtained by different algorithms for the optimal design of the truss; The weight of the best solution is 1902.605 (kg); RUPSO was better than PSO [50], PSOPC [50], SGA [51], HS [52], HPSO [50], and DHPSACO [50] algorithms. By Applying RUPSO 10 times, 1902.605(kg) and 1906.76(kg), were obtained as the best and worst solutions respectively. Fig. 5 illustrates the evolution history diagram of the RUPSO algorithm for the 52-bar truss.

4.2.3. Optimization of 72-bar spatial truss

The 72-bar spatial truss, shown in Fig. 6, was also studied by numerous researchers. These include Wu and Chow [51], Lee et al. [47], Kaveh and Talatahari [53], and Li et al. [52]. The material density is 0.1 lb/in.3 and the modulus of elasticity is 10,000 ksi. Stress limitations of ± 25 ksi were imposed on the members of the structure. Also, the uppermost nodes are subjected to displacement limits of ± 0.25 in both the × and y directions. Hence, the problem has a variable dimensionality of 16 and a constraint dimensionality of 198 (72 tension constraints, 72 compression constraints, and 54 displacement constraints). Furthermore, two load cases were considered, as described in Table 7. The 72 members were divided into 16 groups, as follows: (1) A1–A4, (2) A5–A12, (3) A13–A16, (4) A17–A18, (5) A19–A22, (6) A23–A30 (7) A31–A34, (8) A35–A36, (9) A37–A40, (10) A41–A48, (11) A49–A52, (12) A53–A54, (13) A55–A58, (14) A59–A66 (15) A67–A70, and (16) A71–A72. Two optimization cases were studied: Case 1: the discrete variables were selected from the set D = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0, 3.1,3.2] ($in.^2$), Case 2: the discrete variables were selected from Table 3. To compare the results to other algorithms, a maximum number of 180 iterations was considered. Tables 8 and 9 compare the results obtained by different algorithms with those generated by the proposed RUPSO algorithm respectively, for load case 1 and load case 2. By inspecting Table 8, it is clear that the proposed RUPSO algorithm method is superior to the other algorithms concerning its solution for load case 1. Table 9 contains the result of different optimizers for the optimal design of the 72-bar truss for load case 2. The obtained results show the RUPSO algorithm was more effective than the SGA [51], DHPSACO [50], MBA [50], PSO [48], IAOA [21], and IS-Jaya [22] algorithms. For the 72-bar truss, the best and worst solutions in the optimization process for case1 were 372.5486 (lb) and 385.54 (lb), respectively. Similarly, for load case 2, these values are, respectively, equal to 389.07 (lb) and 390.73 (lb). The evolution history diagrams for the two load cases for the RUPSO algorithm are displayed in Fig. 7.

5. Conclusions

This paper presents a new population-based optimization technique, the Random Update Particle Swarm Optimization (RUPSO). Concepts and ideas underlying the formulation of this method is similar to that of PSO, which itself simulates the motion of bird swarms as a part of a socio-cognitive study. RUPSO optimizer can be used for problems not only with continuous variables but also with discrete variables. To verify the efficiency of the proposed method, it is firstly used for optimization of some truss structures and mathematical equations, then the obtained answers are compared with the results of other optimization algorithms. It is shown that for all problems, RUPSO gives more optimized answers than all other methods; In problems with continues variables of relation (12), RUPSO is converged to −6961.926207 where the most optimized value of the other method is −6961.8244, for equation (13), all algorithms including RUPSO gives the same maximum value.

Regarding the problems with discrete variables, for the 10-bar truss structure with case 1 and 2, the optimal designs obtained by the RUPSO algorithm have respectively 0.745 and 0.132 percent less weight than the best values obtained by other algorithms. The most optimal answer of the other methods for the 52-bar truss structure is improved as 0.1116 percent. For the 72-bar spatial truss structure, applying RUPSO improves the answers as 3.37 percent and 0.0678 percent for case 1 and case 2, respectively.

High convergence rate and low computational cost of RUPSO increases its attraction for solving complex optimization problems. Another advantage of this method is its usage of random updating and capability to search through the entire feasible search spaces, which prevents it from getting fixed on local optima. In summary and regarding high advantages of RUPSO, it can be utilized in global optimization problems.

Declaration of Competing Interest

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

Acknowledgments

The authors would like to acknowledge International Institute of Earthquake Engineering and Seismology under Grant No. 7421 as well as Dr. B. Alinejad, associate professor of Maragheh University, for his invaluable consultation.

References

[1] Gholizadeh S. Optimum design of structures by an improved particle swarm algorithm. Asian J Civil Eng 2010;11:779–96.
[2] Kaveh A, Talatahari S. Size optimization of space trusses using Big Bang-Big Crunch algorithm. Comput Struct 2009;87(17-18):1129–40.
[3] Kaveh A, Kamalinejad M, Biabani Hamedani K, Arzani H. Quantum Teaching-Learning-Based Optimization algorithm for sizing optimization of skeletal structures with discrete variables. Structures 2021;32:1798–819.
[4] Kaveh A, Zolghadr A. Shape and size optimization of truss structures With frequency constraints using enhanced Charged system search algorithm. Asian J Civil Eng 2011;12:487–509.
[5] Degertekin S, Lamberti L, Ugur I. Discrete sizing/layout/topology optimization of truss structures with an advanced Jaya algorithm. Appl Soft Comput 2019;79:363–90.
[6] Rahami H, Kaveh A, Gholipour Y. Sizing, geometry and topology optimization of trusses via force method and genetic algorithm. Eng Struct 2008;30(9):2360–9.
[7] Rasmussen MH, Stolpe M. Global optimization of discrete truss topology design problems using a parallel cut-and-branch method. Comput Struct 2008;86(13-14):1527–38.
[8] Holland JH. Adaptation in Natural and Artificial Systems. Ann Arbor: University of Michigan Press; 1975.
[9] Dorigo M. Optimization, learning and natural algorithms. IT: Dipartimento di Elettronica, Politecnico di Milano; 1992. PhD Thesis.
[10] Eberhart RC, Kennedy J. A new optimizer using particle swarm theory, Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya: IEEE Press, 39–43, 1995.
[11] Geem ZW, Kim JH, Loganathan GV. A new heuristic optimization algorithm: harmony search. Simulations 2001;76:60–8.
[12] Erol OK, Eksin I. A new optimization method: big bang–big crunch. Adv Eng Softw 2006;37(2):106–11.
[13] Mirjalili S, Mirjalili SM, Lewis A. Grey Wolf Optimizer. Adv Eng Softw 2014;69:46–61.
[14] Kaveh A, Mahdavi VR. Colliding Bodies Optimization method for optimum design of truss structures with continuous variables. Adv Eng Softw 2014;70:1–12.
[15] Yang XS. Firefly algorithms for multimodal optimization, in: Stochastic Algorithms: Foundations and Applications (Eds O. Watanabe and T. Zeugmann), SAGA 2009,

Lecture Notes in Computer Science, 5792, Springer-Verlag, Berlin, 2009, pp. 169–78.
[16] Lamberti L, Pappalettere C. Metaheuristic design optimization of skeletal structures: a review. Comput Technol Rev 2011;4:1–32.
[17] Ding Z, Li J, Hao H. Structural damage identification using improved Jaya algorithm based on sparse regularization and Bayesian inference. Mech Syst Sig Process 2019;132:211–31.
[18] Ma R, Karimzadeh M, Ghabussi A, Zandi Y, Baharom Sh, Selmi A, Maureira-Carsalade N: Assessment of composite beam performance using GWO–ELM metaheuristic algorithm: Engineering with Computers 2021.
[19] Morasaei A, Ghabussi A, Aghlmand S, Yazdani M, Baharom Sh, Assilzadeh H. Simulation of steel–concrete composite floor system behavior at elevated temperatures via multi-hybrid metaheuristic framework: Engineering with Computers 2021.
[20] Kazemzadeh AS. Optimum design of structures using an improved firefly algorithm. Int J Optim Civil Eng 2011;2:327–40.
[21] Kaveh A, Biabani Hamedani K. Improved arithmetic optimization algorithm and its application to discrete structural optimization. Structures 2022;35:748–64.
[22] Kaveh A, Hosseini SM, Zaerreza A. Improved Shuffled Jaya algorithm for sizing optimization of skeletal structures with discrete variables. Structures 2021;29:107–28.
[23] Jalili S, Hosseinzadeh Y. Design optimization of truss structures with continuous and discrete variables by hybrid of biogeography-based optimization and differential evolution methods. Struct Des Tall Spec Build 2018;27(14):e1495.
[24] Thang LD, Dac-Khuong B, Duc NT, Quoc-Hung N, Nguyen- XH. A novel hybrid method combining electromagnetism-like mechanism and firefly algorithms for constrained design optimization of discrete truss structures. Comput Struct 2019;212:20–42.
[25] Chaudhary R, Banati H. Swarm bat algorithm with improved search (SBAIS). Soft Comput 2019;23(22):11461–91.
[26] Kaveh A, Talatahari S. A hybrid particle swarm and ant colony optimization for design of truss structures. Asian J Civil Eng :2008; 9: 329-348.
[27] Kaveh A, Talatahari S, Alami M.T. A new hybrid meta-heuristic for optimum design of frame structures. Asian J Civil Eng:2012; 13: 705-717.
[28] Kaveh A, Shahrouzi M. Dynamic selective pressure using hybrid evolutionary and ant system strategies for structural optimization. Int J Numer Meth Engng 2008;73(4):544–63.
[29] Farshchin M, Maniat M, Camp CV, Pezeshk S. School based optimization algorithm for design of steel frames. Eng Struct 2018;171:326–35.
[30] Gholizadeh S, Kamyab R, Dadashi H. Performance-based design optimization of Steel moment frames. Int J Optimiz Civil Eng 2013;3:327–43.
[31] Cao H, Qian X, Chen Z, Zhu H. Enhanced particle swarm optimization for size and shape optimization of truss structures. Eng Optim 2017;49(11):1939–56.
[32] Jafari M, Salajegheh E, Salajegheh J. Optimal design of truss structures using a hybrid method based on particle swarm optimizer and cultural algorithm. Structures 2021;32:391–405.
[33] Kennedy J, Eberhart RC. Particle swarm optimization. International Conference on Neural Networks, Perth, Australia. 1995.
[34] Gholizadeh S, Salajegheh E. Optimal seismic design of steel structures by an efficient soft computing based algorithm. J Constr Steel Res 2010;66(1):85–95.
[35] Gomes HM. Truss optimization with dynamic constraints using a particle swarm algorithm. Expert Syst Appl 2011;38(1):957–68.
[36] Doğan E, Saka MP. Optimum design of unbraced steel frames to LRFD–AISC using particle swarm optimization. Adv Eng Softw 2012;46(1):27–34.
[37] Runarsson TP, Yao X. Search biases in constrained evolutionary optimization. IEEE Trans Syst Man Cybern 2005;35(2):233–43.
[38] Becerra RL, Coello CAC. Cultured differential evolution for constrained optimization. Comput Methods Appl Mech Eng 2006;195(33–36):4303–22.
[39] Hedar AD, Fukushima M. Derivative-free filter simulated annealing method for constrained continuous global optimization. J Glob Optim 2006;35(4):521–49.
[40] Chootinan P, Chen A. Constraint handling in genetic algorithms using a gradient-based repair method. Comput Oper Res 2006;33(8):2263–81.
[41] Zahara E, Kao YT. Hybrid Nelder-Mead simplex search and particle swarm optimization for constrained engineering design problems. Expert Syst Appl 2009;36:3880–6.
[42] Kayhan AH, Huseyin Ceylan M, Ayvaz T, Gurarslan G. PSOLVER: A new hybrid particle swarm optimization algorithm for solving continuous optimization problems. Expert Syst Appl 2010;37(2010):6798–808.
[43] Koziel S, Michalewicz Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evol Comput 1999;7:19–44.
[44] Runarsson TP, Yao X. Stochastic ranking for constrained evolutionary optimization. IEEE Trans Evol Comput 2000;4(3):284–92.
[45] Coello CAC, Becerra RL. Efficient evolutionary optimization through the use of a cultural algorithm. Eng Optim 2004;36(2):219–36.
[46] He Q, Wang L. A hybrid particle swarm optimization with a feasibility based rule for constrained optimization. Appl Math Comput 2007;186:1407–22.
[47] Li LJ, Huang ZB, Liu F, Wu QH. A heuristic particle swarm optimizer for optimization of pin connected structures. Comput Struct 2007;85(7-8):340–9.
[48] Gholizadeh S, Barati H. A comparative study of three metaheuristics for optimum design of trusses. Int J Optim Civil Eng 2012;3:423–41.
[49] Zahara E, Hu CH. Solving constrained optimization problems with hybrid particle swarm optimization. Eng Optim 2008;40(11):1031–49.
[50] Sadollah A, Bahreininejad A, Eskandar H, Hamdi M. Mine blast algorithm for optimization of truss structures with discrete variables. Comput Struct 2012;102-103:49–63.
[51] Wu S-J, Chow P-T. Steady-state genetic algorithms for discrete optimization of trusses. Comput Struct 1995;56(6):979–91.
[52] Lee KS, Geem ZW, Lee SH, Bae KW. The harmony search heuristic algorithm for discrete structural optimization. Eng Optimiz 2005;37(7):663–84.
[53] Kaveh A, Talatahari S. A particle swarm ant colony optimization for truss structures with discrete variables. J Constr Steel Res 2009;65(8-9):1558–68.
[54] Rajeev S, Krishnamoorthy CS. Discrete optimization of structures using genetic algorithms. J Struct Eng 1992;118(5):1233–50.
[55] Ringertz UT. On methods for discrete structural constraints. Eng Optimiz 1988;13(1):47–64.
[56] Li LJ, Huang ZB, Liu F. A heuristic particle swarm optimization method for truss structures with discrete variables. Comput Struct 2009;87(7-8):435–43.