Journal of Soft Computing in Civil Engineering 5-2 (2021) 86-119

Contents lists available at SCCE

Journal of Soft Computing in Civil Engineering

Journal homepage: www.jsoftcivil.com

A Hybrid Generalized Reduced Gradient-Based Particle Swarm Optimizer for Constrained Engineering Optimization Problems

H. Varaee [1], N. Safaeian Hamzehkolaei [2*], M. Safari [3]

1. Assistant Professor, Department of Civil Engineering, Ale Taha Institute of Higher Education, Tehran, Iran
2. Assistant Professor, Department of Civil Engineering, Bozorgmehr University of Qaenat, Qaen, Iran
3. MSc, Department of Civil Engineering, Ale Taha Institute of Higher Education, Tehran, Iran

Corresponding author: *nsafaeian@buqaen.ac.ir*

https://doi.org/10.22115/SCCE.2021.282360.1304

ARTICLE INFO

Article history:
Received: 21 April 2021
Revised: 19 June 2021
Accepted: 12 July 2021

Keywords:
Hybrid global-local search engine;
Particle swarm optimization (PSO);
Generalized reduced gradient (GRG) algorithm;
k-nearest neighbors (k-NN) algorithm;
Purely uniform distributed swarm.

ABSTRACT

A hybrid algorithm is presented that combines strong points of Particle Swarm Optimization (PSO) and Generalized Reduced Gradient (GRG) algorithm to keep a good compromise between exploration and exploitation. The hybrid PSO-GRG quickly approximates the optimum solution using PSO as a global search engine in the first phase of the search process. The solution accuracy is then improved during the second phase of the search process using the GRG algorithm to probe locally for a proper solution(s) in the vicinity of the current best position obtained by PSO. The k-nearest neighbors (k-NN)-based Purely Uniform Distributed (PUD) initial swarm is also applied to increase the convergence speed and reduce the number of function evaluations (NFEs). Hybridization between both algorithms allows the proposed algorithm to accelerate throughout the early stages of optimization using the high exploration power of PSO whereas, promising solutions will possess a high probability to be exploited in the second phase of optimization using the high exploitation ability of GRG. This prevents PUD-based hybrid PSO-GRG from becoming trapped in local optima while maintaining a balance between exploration and exploitation. The competence of the algorithm is compared with other state-of-the-art algorithms on benchmark optimization problems having a wide range of dimensions and varied complexities. Appraising offered algorithm performance revealed great competitive results on the Multiple Comparison Test (MCT) and Analysis of Variance (ANOVA) test. Results demonstrate the superiority of hybrid PSO-GRG compared to standard PSO in terms of fewer NFEs, fast convergence speed, and high escaping ability from local optima.

How to cite this article: Varaee H, Safaeian Hamzehkolaei N, Safari M. A Hybrid Generalized Reduced Gradient-Based Particle Swarm Optimizer for Constrained Engineering Optimization Problems. J Soft Comput Civ Eng 2021;5(2):86-119. https://doi.org/10.22115/scce.2021.282360.1304.

1. Introduction

Complex engineering optimization problems usually include a large number of non-convex, non-linear and non-differentiable constraints and objective functions. Exact optimization algorithms do not efficiently solve Constrained Optimization Problems (COPs) that have nonlinear and non-differentiable search space, objective and constraint functions [1]. Recently, many metaheuristic algorithms derived from nature have been developed and employed to cope with COPs [2–4]. Among them, the Swarm Intelligence Optimization (SIO) algorithms, inspired by natural phenomena and biological behaviors, are considered as a kind of bionic random method, which can deal with certain high-dimensional intricate and variable optimization problems due to its better computing performance and simple model [5]. Swarm intelligence systems typically comprise simple agents, that follow extremely simple rules and interact with each other and their surroundings. Although each agent alone can be considered unintelligent, interactions between multiple agents lead to the emergence of intelligent collective behavior [6]. Particle Swarm Optimization (PSO) [7], Ant Colony Optimization (ACO) [8], Artificial Bee Colony (ABC) algorithms [9], Firefly Algorithm (FA) [10], Glow Worm Optimization (GWO) algorithm [11], Bat Algorithm (BA) [12], Lion Optimization Algorithm (LOA) [13], Grey Wolf Optimization (GWO) algorithm [14], Monarch Butterfly Optimization (MBO) [15], Krill Herd Optimization (KHO) algorithm [16], Elephant Herding Optimization (EHO) [17], Cuckoo Search (CS) [18] are in the class of SIO algorithms. These algorithms have been analyzed over time by researchers in various areas [19–28]. The SIOs have exhibited good performance in different engineering fields including feature selection [29], structural weight minimization [30–34], shape and topology optimization [35–37], damage detection [38–40], and performance-based design optimization [41].

Among all global search algorithms, PSO has been applied and proven useful on a wide range of engineering COPs such as the optimal design of truss structures [42–45], structural damage detection [46], topology optimization [47–49], and reliability-based design optimization [50–52]. PSO algorithm can efficiently handle non-linear, non-convex, and non-differentiable design spaces since it does not require prior knowledge about the search space, internal variable transformations, or other manipulations to handle constraints [53].

Despite the efficiency of the SIO algorithms, none of these algorithms is capable of offering adequately superior performance to solve all optimization problems [2,23]. There are also some disadvantages to SIO utilization. First, effective parameters tuning of SIO algorithms is a challenging task for various swarm-based algorithms. Premature convergence and/or trapping in local optima is also another problem encountered when using SIO algorithms. For instance, although BA is potent in local search, occasionally it may get trapped in some local optimum, thus it is not capable of carrying out global search well [54,55], PSO can sometimes find local optima or exhibit slow convergence speed.

In recent years, many algorithms with different strategies have been proposed to cope with the above-mentioned issues [56–59]. Parameter tuning, hybridization, and better initializing are the most common methods that have been used in literature. Gupta and Deep have proposed a hybrid

algorithm based on the combination of the ABC with the sine cosine algorithm aiming to improve both the local and global search capabilities of the standard ABC algorithm [60]. In 2020, Yildizdan and Baykan have suggested a new hybrid BA-ABC algorithm to improve the diversity and global search capability of the BA using the ABC algorithm, while the inertia weight was also added to the velocity formula to enhance the exploration ability of BA [61]. Yue and Zhang [62] were proposed a hybrid Grasshopper Optimization Algorithm (GOA) with BA for global optimization. In this study, the local search operation of the BA and the Levy flight with variable coefficient together with the random search strategy was employed to balance the exploration and exploitation capability of the proposed hybrid algorithm. Yue et al. have introduced a novel hybrid algorithm named FWGWO, which accordingly, the exploration capability of the fireworks algorithm with the exploitation ability of the GWO has been combined through the setting a balance coefficient [63]. Authors in [64–66] employed several PSO-based hybridized algorithms such as the PSO-SA, PSO-GA, and PSO-GSA, for different optimization problems. Fuzzy logic, Chaos strategy, Elitism approach, Quantum strategy, and opposite-based learning are some other methods that have been utilized by researchers to ameliorate the performance of the standard PSO [67–69].

In this study, a new hybrid optimization algorithm is proposed based on the PSO and Generalized Reduced Gradient (GRG) algorithm to improve the local search ability of the standard PSO. The Purely Uniform Distributed (PUD) initial swarm is also implemented as an efficient strategy to enhance the convergence speed of the optimization procedure. The proposed hybrid PSO-GRG algorithm with PUD operator is called the PGP method and introduced in detail and implemented successfully for some mathematical and engineering COPs having various dimensions and varied complexities. The efficiency and accuracy of the proposed algorithm are also compared with other state-of-the-art algorithms by performing the Multiple Comparison Test (MCT) and Analysis of Variance (ANOVA) test.

The remainder of the paper is organized as follows. Details of the original PSO and GRG algorithms are presented in Sect. 2 and 3, respectively. In Sect. 4, the proposed hybrid PSO-GRG with the PUD operator is presented. In Sect. 5, the experimental results are provided for the nonlinear benchmark functions. Finally, the summary and concluding remarks are discussed in Sect. 6.

2. Particle swarm optimization (PSO)

The PSO algorithm was proposed by Kennedy and Eberhart [7] based on the flocking behavior and social cooperation of birds. In this algorithm, the position of each particle is considered as a potential solution to the optimization problem. In PSO, the position of each particle in a swarm approaches the optimum solution using its velocity vector (V), personal experience $(Pbest)$, and the best experience of the swarm $(Gbest)$.

In the first iteration of the PSO, the initial swarm with P particles is generated by distributing a uniform random population in the search space. During the optimization process, the position vector $X_i = [x_{i1}, \dots, x_{iD}]$ and velocity vector $V_i = [v_{i1}, \dots, v_{iD}]$ of each particle is updated concerning

the personal best position $Pbest_i = \left[x_{i1}^{Pbest}, \dots, x_{iD}^{Pbest}\right]$, and best position of the swarm $Gbest_i = \left[x_{i1}^{Gbest}, \dots, x_{iD}^{Gbest}\right]$ as follows:

$$V_i^{k+1}(j) = \lambda\left[\omega^k V_i^k(j) + r_1 c_1\left(Pbest_i^k(j) - X_i^k(j)\right) + r_2 c_2\left(Gbest_i^k(j) - X_i^k(j)\right)\right] \quad (1)$$

$$X_i^{k+1}(j) = X_i^k(j) + V_i^{k+1}(j) \quad (2)$$

where P is the swam size; D is the dimension of the search space, k is the current iteration number; $X_i^k(j)$ and $V_i^k(j)$ are the position and velocity of the i-th particle $(i = 1, 2, \dots, P)$ in the j-th dimension, respectively $(j = 1, 2, \dots, D)$; λ is the constriction factor; c_1 and c_2, respectively, are the personal (cognitive) and social learning constant; r_1 and r_2 are random numbers between [0-1], and ω is the inertia weight factor used to keep a balance between the exploration and exploitation power of the algorithm. The linearly decreasing inertia weight factor is applied in this study [51,52].

$$\omega(k) = \omega_{max} - \frac{(\omega_{max} - \omega_{min}) \times k}{K} \quad (3)$$

where K is the maximum iteration number; and ω_{max} and ω_{min} denoted the maximum and minimum values for inertia weight factor, used in the first and last iterations, respectively.

In this algorithm, the velocity vector is limited to $10 - 20\%$ of each dimension size to control the particle's step size of each particle. After updating the velocity and position, the existent particles within the search space are evaluated. If the objective function related to the current position is better than the individual best position, the $Pbest$ of each particle will be replaced by the current position. Moreover, if a particle position is better than the current best solution obtained by the entire swarm, the $Gbest$ will also be updated. The search process will be continued until the stop conditions are met.

3. Generalized reduced gradient (GRG) algorithm

The GRG algorithm is robust local search algorithms, which is based on the linearizing of the non-linear objective function and constraints at a local solution by applying the Taylor expansion equation and the linear optimization methods [70]. Given that inequality constraints can always be converted to equalities through the addition of slack variables (s), an equality-constrained NLP model can be formed as follows:

$$\begin{aligned} &Min: f(w) \\ &Subject\ to: h_i(w) = 0 \quad i = 1,2, \dots, neq \\ &l \le w \le u \end{aligned} \quad (4)$$

where $w = [w_1, w_2, \dots, w_n]^T$ contains the original design variables x and the slack variables s, and the vectors l and u denote the lower and upper bounds for s, respectively. The gradient of the f can be defined as:

$$\nabla^T f(z) = \left[\frac{\partial f}{\partial z_1}, \frac{\partial f}{\partial z_2}, \dots, \frac{\partial f}{\partial z_{NI}}\right] \quad (5)$$

$$\nabla^T f(y) = \left[\frac{\partial f}{\partial y_1}, \frac{\partial f}{\partial y_2}, \dots, \frac{\partial f}{\partial y_{ND}}\right] \quad (6)$$

where two vectors z with NI elements, and y with ND elements are partitioned from the vector w. The Jacobian matrix (J) of the constraints is also partitioned in the same manner. The differential of the constraints and objective function can then be written as follows:

$$df = \nabla^T f(z)dz + \nabla^T f(y)dy \quad (7)$$

$$dh = J_z dz + J_y dy = 0 \quad (8)$$

where dz and dy are vectors of differential displacements in z and y, respectively. Solving for dy in terms of dz gives:

$$dy = -J_y^{-1} J_z dz \quad (9)$$

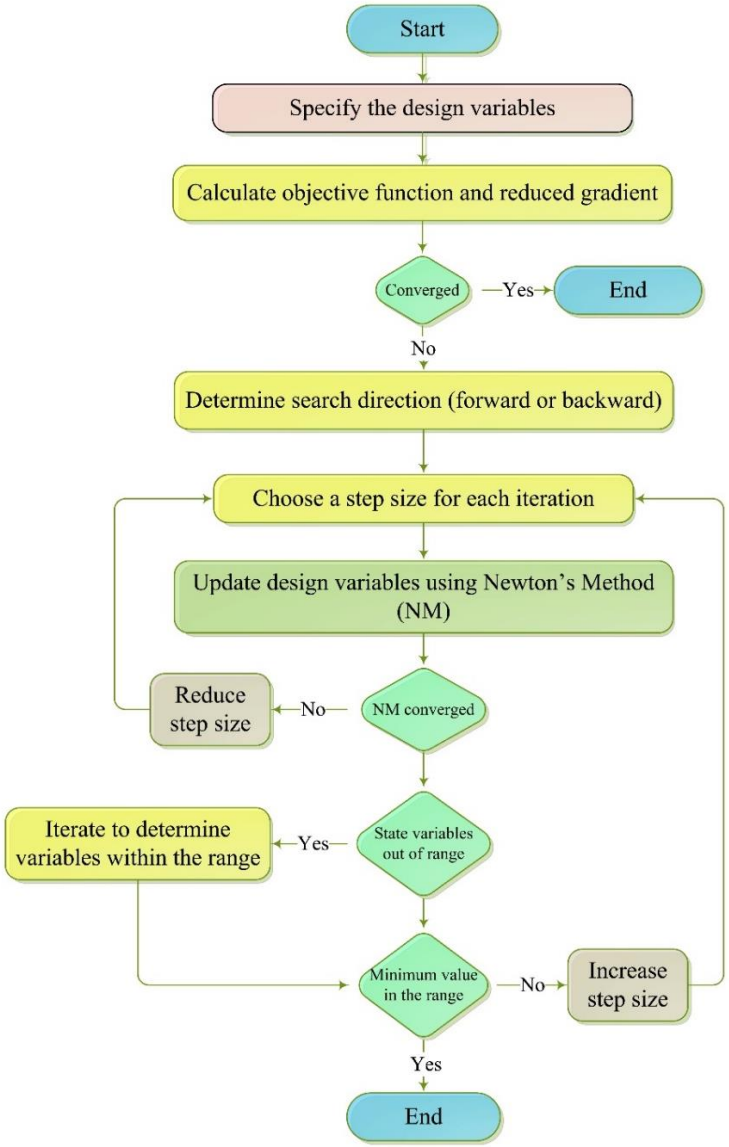


Fig. 1. Flowchart of the GRG algorithm.

Substituted dy from Eq. (9) into Eqs. (7) and (8) and rearranging the results, the reduced gradient $\nabla_r^T f$ can be defined as:

$$\nabla_r^T f(z) = \nabla^T f(z) - \nabla^T f(y) J_y^{-1} J_z \tag{10}$$

The potential constraint strategy can also be employed to treat all constraints in the sub-problem as equality constraints [71]. Accordingly, a search direction is found so that for any small movement, the present active constraints stay in an exactly active manner. The Newton–Raphson algorithm is employed to draw back into the constraint boundary if some active constraints are not precisely satisfied due to the nonlinearity of the constraint functions. Thusly, the GRG method works nearly like the gradient projection algorithm [72].

Fig. 1, shows the general flowchart of the GRG algorithm. More details about GRG have been presented in [73,74].

4. Proposed hybrid PSO-GRG algorithm

In this study, the GRG algorithm is used to enhance the local search and exploitation ability of the PSO in some iterations of the search process. Furthermore, the PUD operator is also applied to increase the convergence rate and reduce the total Number of Function Evaluations (NFEs). Given that COPs are mostly complex and time-consuming, reducing the NFEs is an imperative issue. Details of the proposed PUD-based hybrid algorithm are presented in the next subsections.

4.1. Purely uniform distributed swarm

Although, the standard PSO starts with a group of randomly generated particles, however, the search space may not effectively be covered by a uniformly distributed swarm. As a result, as shown in Fig. 2, random generation of the swarm may lead to creating the particle density in some subspaces of the search spaces, so that some adjacent particles may practically exist at a very close distance, which will achieve relatively similar fitness during the search process. Furthermore, as shown in Fig. 2, some subspaces may not even cover by random generation of the initial swarm. Therefore, the size of the population should be greatly increased to efficiently cover the overall search space for high dimension and/or complex problems. The existence of such conditions leads to an increase in the total NFEs and, as a result, increases the computation time of the optimization process.

In this study, the purely uniform distributed random particles are generated to probe the more efficiently the search domain. The promise of employing the PUD operator for generating the initial swarm is that the performance of the algorithm could be enhanced by avoiding checking particles with the same fitness so that no two particles are evaluated at a distance less than a certain radius from each other. For this purpose, the distance between the two adjacent particles is calculated using a k-nearest neighbors (k-NN) method that is a non-parametric algorithm for both classification and regression [75]. Also, the devoid subspaces of the search space should be covered as much as possible by particles from the initial swarm. Accordingly, after producing an initial swarm, all particles of the swarm are evaluated at a specified neighborhood during the first

stage, and then the dense particles in the neighborhood are exited from the swarm. Ultimately, these modifications result in the generation of an initial swarm so that the particles with the same fitness functions have been removed. Fig. 3(a), illustrates the PUD-based generated initial swarm for a two-dimensional search space. To efficiently explore the problem space as completely as possible in the second stage, a certain number of particles are randomly generated through the uniform distribution and added to the current swarm. The vicinity evaluation process is performed again using k-NN for all particles of the swarm. As shown in Fig. 3(b), this process considerably increases the probability of covering the entire search space. Figs. 4, shows the flowchart of the proposed k-NN-based PUD operator.

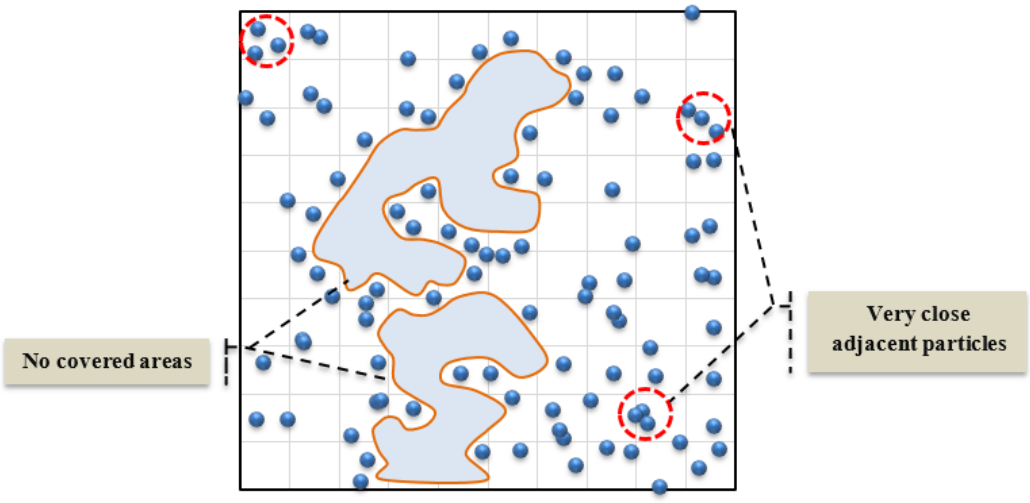
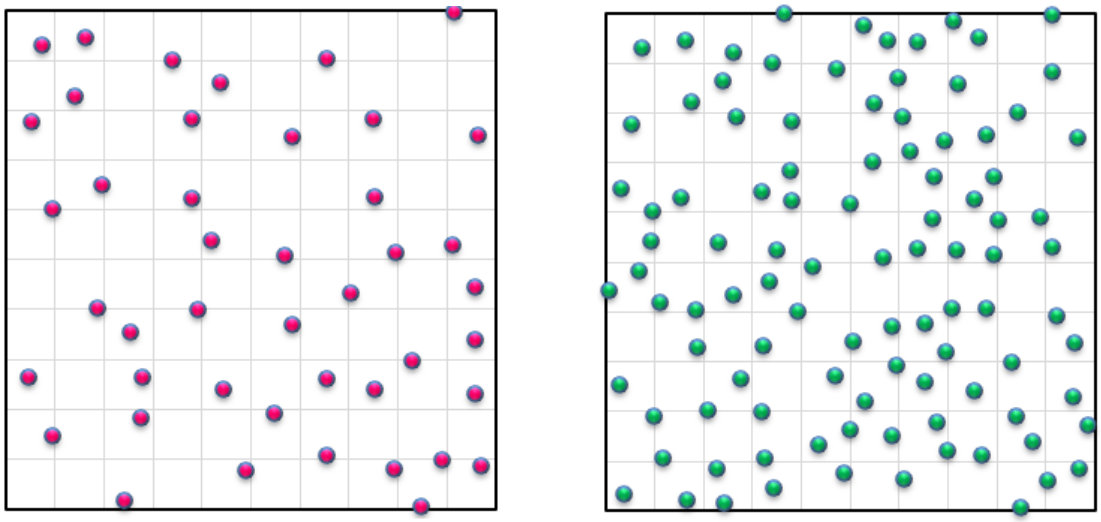


Fig. 2. Initial swarm generated by randomly uniform distribution (n=100).



a) b)

Fig. 3. Purely uniform distributed initial random swarm; a) n=44, b) n=100.

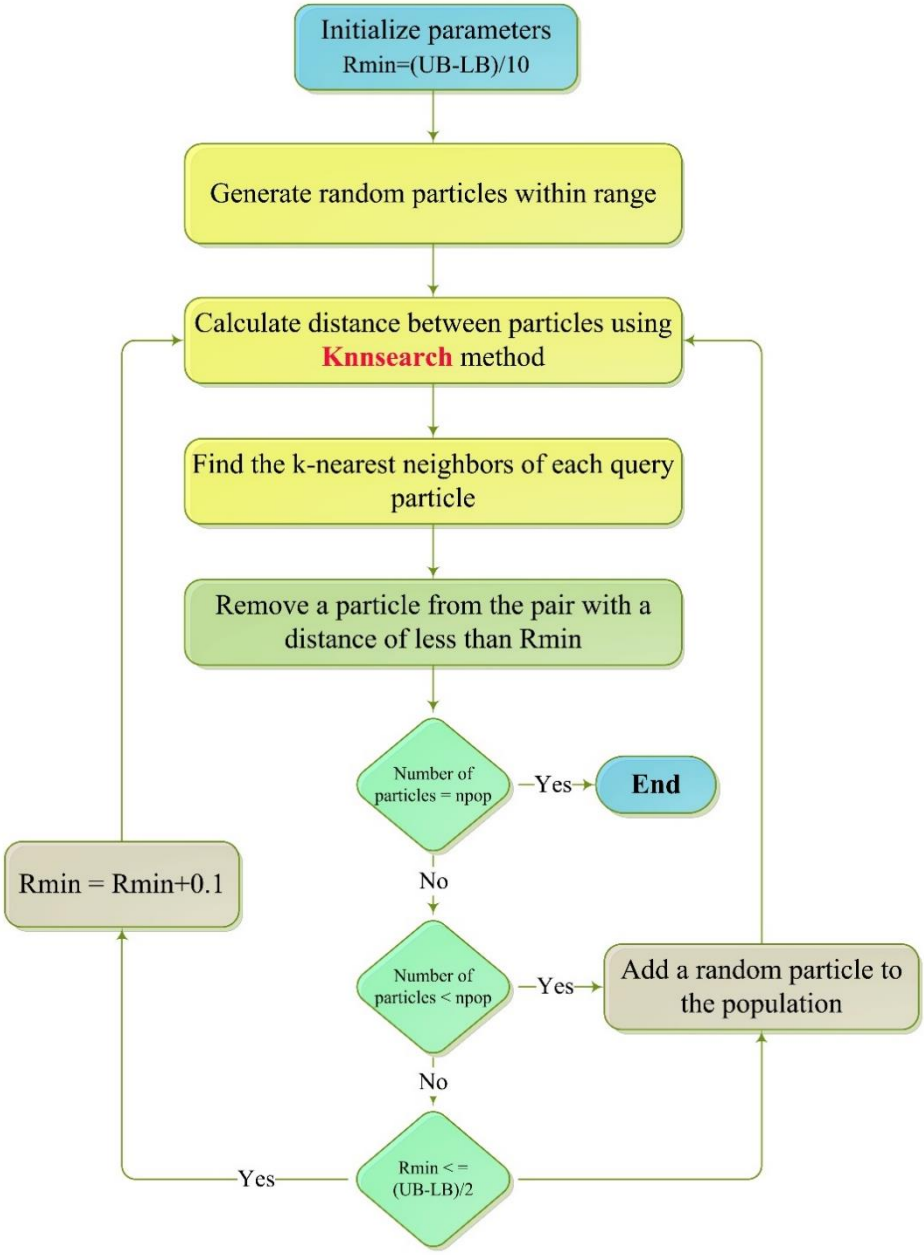


Fig. 4. Flowchart of the proposed PUD operator.

4.2. General steps of the hybrid PSO-GRG with PUD operator

In the proposed hybrid PSO-GRG algorithm, the GRG algorithm is summoned by satisfying the conditions of convergence. For this purpose, the best solution obtained by PSO is utilized as a starting point for the GRG during the local search process. Using the GRG gradient-based algorithm will lead to locally improve the optimal position in the vicinity of the starting point, as described in Section 3. After converging the GRG-based local search process, the termination condition(s) of the algorithm is checked. If the termination criteria are not passed, the solution obtained by the GRG algorithm is considered as a global best position (GBP) and, consequently, the PSO algorithm will continue the optimization process to achieve the new best position. As

shown in Fig. 5, this cycle continues until satisfying all convergence conditions. Therefore, the quality of the optimum solution is improved in each series cycle of the algorithm by utilizing the GBP obtained by PSO as a starting point for the GRG algorithm. Eventually, the solution obtained by the GRG algorithm is presented as the final solution of the proposed hybrid PSO-GRG. It should be mentioned that the maximum NFEs and/or the maximum number of iteration could be considered as the general termination condition(s) of the proposed hybrid algorithm. In this paper, the GRG algorithm is employed if the best solution of PSO does not improve after every 10 cycles.

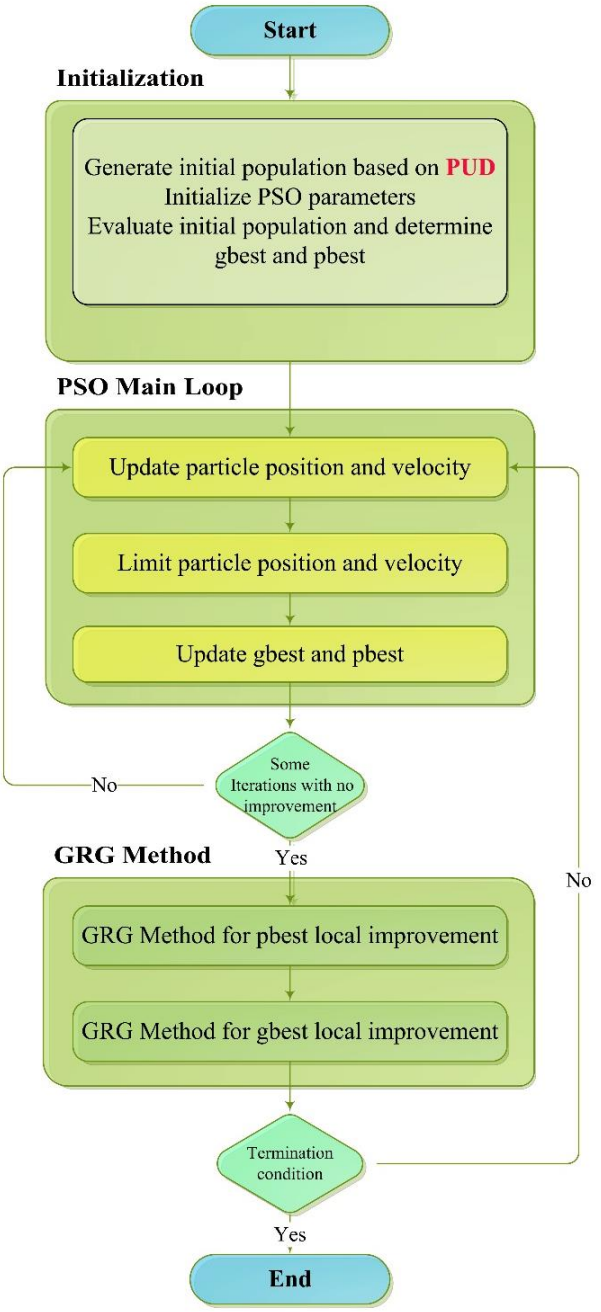


Fig. 5. The general flowchart of the proposed hybrid PGP algorithm.

The hybridization between PSO and GRG allows the proposed algorithm to accelerate throughout the early stages of the search process using the high exploration power of PSO whereas, in the later stages of optimization, promising solutions will possess a high probability to be exploited using the exploitation ability of the GRG. This prevents PSO-GRG from becoming trapped in local optima while maintaining a good compromise between exploration and exploitation. The general flowchart of the PSO-GRG with the PUD operator is depicted in Fig. 5.

4.3. The constraint handling approach

In this paper, a penalty-based constraint handling approach is considered to solve COPs. The search domain in the COPs includes feasible and infeasible spaces. For the feasible solutions, all the constraints are met. In contrast, in the infeasible space, at least one of the constraints is violated. Hence, the constraint functions can be taken into scrutiny through the penalty functions. This implies that constraints can be considered in the target function in one way or another. A penalty function can be defined as:

$$F(x) = f(x) + h(k)H(x), x \in S \subset R^n \tag{11}$$

where f(x) denotes the target (objective) function; $h(k)$ denotes the dynamic penalty value at iteration k; and $H(x)$ is a penalty factor, defined as:

$$H(x) = \sum_{i=1}^{m} \theta\,(q_i(x))\, q_i(x)^{\,\gamma\,(q_i(x))} \tag{12}$$

where $q_i(x) = \{0, g_i(x)\}, i = 1, \dots, m$. The function $q_i(x)$ is a relative violation function for the i-th constraint $g_i(x)$; $\theta(q_i(x))$ denotes the multi-segment assignment function; and $\gamma(q_i(x))$ denotes the power of the penalty function.

In this method, the initial penalty should be considered as the lowest possible value. The penalty value should also be increased in every iteration as the algorithm proceed [76]. Thus, the initial value selection and the updating strategy for the penalty coefficient are the main problems of the penalty function methods. If the considered penalty value is too small, the algorithm may generate a solution outside the feasible region. On the contrary, if the value is too large, approaching the boundary outside the feasible region might be arduous as well as the boundary might remain un-surveyed. Moreover, at least one of the constraints is usually active at the optimum solution. Therefore, searching the entire feasible zone, including the boundaries, is also momentous [77].

In this paper, the penalty parameters are selected based on the recommendations suggested in [78]. If $q_i(x) < 1$, then $\gamma\left(q_i(x)\right) = 1$, otherwise $\gamma\left(q_i(x)\right) = 2$. Moreover, if $q_i(x) < 0.001$, then $\theta\left(q_i(x)\right) = 10$, else if $0.001 < q_i(x) < 0.1$ then $\theta\left(q_i(x)\right) = 20$, else if $q_i(x) < 1$, then $\theta\left(q_i(x)\right) = 100$; otherwise $\theta\left(q_i(x)\right) = 300$, and $h(k)$ is set to $k\sqrt{k}$ where k is the current cycle number.

5. Results

The efficiency and applicability of the proposed PSO-GRG and PGP (PSO-GRG with PUD operator) algorithms in solving mathematical and engineering design optimization benchmark COPs are investigated in this section. For each test problem, the results of the proposed algorithms including the Best, Average, Worst, Standard deviation (Std.), Median, total NFEs, and the average rank based on these performance indices are presented in comparison to the results of the PSO, FA, CA, ABC, and CBO algorithms. The parameter settings for all compared algorithms are set based on the recommendations of the literature (Table 1). The population size is set to $20 \times d$ for all algorithms, where d is the dimension of each problem. The maximum number of iterations is set to 100 and each problem is solved 25 times using Matlab 2016a on the personal computer with Intel ® Core i7-7500 CPU @ 2.70 GHz.

Table 1
Parameter settings of all compared algorithms in solving COPs.

Algorithm	Parameter setting	References
PSO PSO-GRG PGP	Cognitive coefficient = 2, Social coefficient = 2, Start inertia weight = 0.6, Final inertia weight = 0.6.	[79]
FA	Gamma = 1, Beta = 2, Alpha = 0.2; Mutation coefficient damping ratio = 0.98, m = 2.	[80]
CA	Acceptance ratio = 0.35, alpha = 0.25, Beta = 0.5.	[81]
ABC	Number of bee = population size, Number of food = population size / 2, Limit = 50.	[82]
CBO	Coefficient of restitution = $1 - (\text{iter}/\text{maxIt})$.	[83]

5.1. Benchmark mathematical constrained test problems

In this section, a challenging mathematical problem with highly non-linear objective functions and constraints with various dimensions is assessed to demonstrate the efficacy of the proposed algorithm. For this purpose, Keane's bump problem that is known as a challenging multimodal COP with a highly bumpy surface is investigated [84,85]:

$$\text{Minimize } f(x) = -\left|\left\{\sum_{i=1}^{n} cos^4(x_i) - 2\prod_{i=1}^{n} cos^2(x_i)\right\} \Big/ \left(\sum_{i=1}^{n} i x_i^2\right)^{0.5}\right| \tag{13}$$

Subject to:

$$g_1(x): 0.75 - \prod_{i=1}^{n} x_i < 0;$$
$$g_2(x): \sum_{i=1}^{m} x_i - 7.5n < 0; \tag{14}$$

where $0 < x_i < 10$ (i $=$ 1. 2. …. n) are the optimization variables and n is the dimension of the problem. In Fig. 6, a 2-dimensional perspective view of Keane's bumpy function is presented.

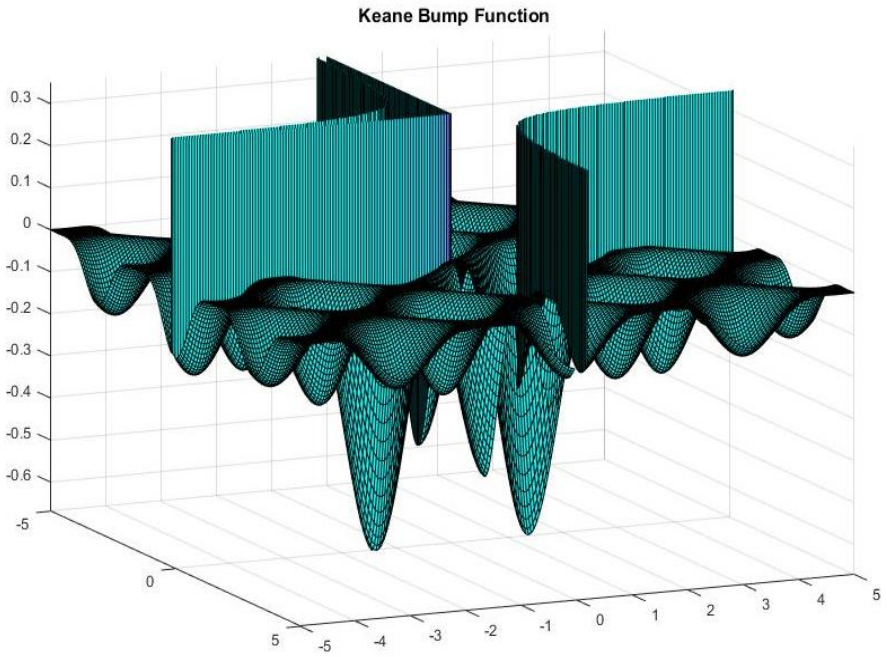


Fig. 6. A perspective view of Keane's bumpy function without the constraints**.**

Keane investigated this problem using a parallel GA with 12-bit binary encoding, crossover, inversion, mutation, niche forming, and a modified Fiacco-McCormick constraint penalty function [84]. For n=20, he obtained approximately the value of 0.76 after 20,000 NFEs. For n=50, the value close to 0.76 after 50,000 NFEs was obtained. Ghasemi et al also solved the problem with and without the use of rebirthing, for both n=20 and n=50 [86]. In their study, without the rebirthing technique, the algorithm was converged to an optimum value of 0.736 after 15,800 NFEs. However, by applying the rebirthing technique, an optimum solution of 0.796 was obtained with 31,800 NFEs. For the case n=50, the optimum solution obtained without rebirthing was 0.780 after 36,400 NFEs, while the algorithm converged to 0.820 by 41,000 NFEs after applying the rebirthing technique.

The results of the PSO, FA, CA, ABC, CBO, and the proposed PSO-GRG and PGP algorithms for the Keane's bumpy function in the case of 5, 10, 20-and 50-dimension are summarized in Tables 2-5, respectively. The presented results in Tables 2-5 conclude that the proposed PGP algorithm achieved significantly better values than the other compared algorithms, especially for the NFEs which are found out in the lowest value by the proposed algorithm. It is worth mentioning that, for all cases under consideration, the solutions obtained by the proposed PSO-GRG were enhanced after applying the PUD operator. As can be seen from Tables 2-5, for all dimensions except for n=50, the proposed PGP algorithm with PUD operator provided better solutions compared to standard PSO, PSO-GRG, and other compared algorithms in terms of the Best and Average of the results. It is also worth mentioning that, for n=50, more accurate optimum solutions are provided by the proposed PSO-GRG and the PGP algorithms. According to Table 5, the first constraint (g_1) is only active at both optimum solutions provided by the proposed algorithms. However, PGP converged after 22000 NFEs which is much lower than that of PSO-GRG (56000).

Table 2
Comparison results of Keane's bumpy problem (n=5).

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		-0.634434	-0.599817	-0.634445	-0.634242	-0.634130	-0.634448	**-0.634449**
Average		-0.626494	-0.400488	-0.529209	-0.549381	-0.548245	-0.625281	**-0.632458**
Worst		-0.581363	-0.287488	-0.376375	-0.466548	-0.401325	-0.490594	**-0.621936**
Std.		0.012689	0.087641	0.088487	0.047079	0.066905	0.029145	**0.004184**
Median		-0.634240	-0.405735	-0.555322	-0.548945	-0.556504	-0.634445	**-0.634446**
Best Design	x_1	3.074789	3.086110	3.076389	3.057021	3.061141	3.075973	3.075468
	x_2	2.994059	3.017713	2.991997	2.993556	3.008029	2.991598	2.992304
	x_3	1.473922	1.410819	1.475373	1.472613	1.477723	1.474809	1.475700
	x_4	0.235087	0.238800	0.235129	0.237445	0.233605	0.236236	0.236575
	x_5	0.235162	0.239277	0.234883	0.234876	0.236858	0.233935	0.233442
Constraints	g_1	-1.938E-04	-1.002E-03	-1.265E-06	-2.104E-03	-3.831E-03	-2.931E-06	-3.046E-7
	g_2	-0.786320	-0.786861	-0.786299	-0.786786	-0.786204	-0.786332	-0.786307
NFE [a]		n/a[b]	n/a[b]	5,600	n/a[b]	n/a[b]	5,600	**5,200**
Average Rank		2.67	6.17	4.83	4.33	4.67	2.50	**1.00**
Overall Rank		3	7	4	6	5	2	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-5}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-5} within 10,000 NFEs

Table 3
Comparison results of Keane's bumpy problem (n=10).

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		-0.744044	-0.558783	-0.740216	-0.572987	-0.674613	-0.747299	**-0.747305**
Average		-0.717396	-0.373637	-0.633815	-0.507654	-0.606945	-0.741269	**-0.741660**
Worst		-0.649883	-0.262882	-0.491727	-0.466982	-0.550462	**-0.693472**	**-0.693472**
Std.		0.024401	0.087521	0.076900	0.028318	0.036180	**0.011460**	0.015575
Median		-0.722268	-0.392195	-0.637958	-0.506425	-0.602725	-0.747286	**-0.747287**
Best Design	x_1	3.136318	9.356265	3.129749	3.330347	3.126558	3.120992	3.124911
	x_2	3.120114	3.109227	3.063582	3.245365	2.960576	3.069567	3.071144
	x_3	3.023238	3.071965	3.014805	3.129597	2.942394	3.016382	3.014750
	x_4	2.914836	0.281657	2.970186	3.056053	2.954436	2.956655	2.960695
	x_5	1.414281	0.303940	0.359903	3.069894	2.954242	1.467547	1.466807
	x_6	0.334365	2.990283	1.416733	0.482239	0.306420	0.367869	0.368655
	x_7	0.389125	0.292132	0.389952	0.529499	1.465520	0.359778	0.364679
	x_8	0.397362	0.281131	0.359083	1.657751	0.208071	0.358139	0.356782
	x_9	0.355535	0.282255	0.343383	0.561539	0.134308	0.358082	0.353976
	x_{10}	0.343091	1.416041	0.356475	0.010000	0.314868	0.352417	0.351564
Constraints	g_1	-2.488E-02	-1.234E-03	-4.766E-04	-5.720E-03	-2.015E-01	-1.568E-05	-1.412E-5
	g_2	-0.794290	-0.714868	-0.794615	-0.745703	-0.768435	-0.794301	-0.794214
NFE [a]		n/a[b]	n/a[b]	n/a[b]	n/a[b]	n/a[b]	7,600	**5,200**
Average Rank		3.00	6.33	4.33	5.17	4.50	1.67	**1.17**
Overall Rank		3	7	4	6	5	2	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-3}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-3} within 20,000 NFEs

Table 4
Comparison results of Keane's bumpy problem (n=20).

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		-0.71083	-0.42773	-0.66698	-0.49294	-0.53417	-0.803530	**-0.803619**
Average		-0.62331	-0.31443	-0.57056	-0.44184	-0.46866	-0.792628	**-0.792712**
Worst		-0.52547	-0.24313	-0.44121	-0.39734	-0.43106	**-0.780758**	-0.758546
Std.		0.058073	0.042407	0.055441	0.021851	0.023259	**0.006264**	0.012396
Median		-0.64165	-0.31714	-0.57051	-0.44491	-0.46493	-0.792547	**-0.792567**
Best Design	x_1	6.179856	7.895262	6.107420	9.413282	5.706031	3.161323	3.162461
	x_2	3.339417	3.179144	3.082757	2.960146	2.431078	3.121573	3.128331
	x_3	2.877750	6.229477	3.183959	3.089952	3.231847	3.100686	3.094792
	x_4	3.028898	6.210913	3.106224	3.139072	2.876008	3.063361	3.061451
	x_5	3.109965	0.130187	3.065760	3.113860	2.746013	3.024778	3.027929
	x_6	3.216482	6.161879	1.068054	0.151008	2.789593	2.986022	2.993826
	x_7	2.923089	0.116712	3.103356	3.006541	3.162986	2.975655	2.958669
	x_8	2.873630	0.181137	0.652909	3.055659	2.823960	2.928689	2.921842
	x_9	2.873586	0.221049	0.484927	3.086395	3.055117	0.492330	0.494825
	x_{10}	0.261387	0.191882	0.294239	0.010000	2.897448	0.478291	0.488357
	x_{11}	0.461674	3.009281	2.141949	0.010000	0.377123	0.473688	0.482317
	x_{12}	2.476116	0.202544	0.914048	3.003675	1.046887	0.471621	0.476645
	x_{13}	0.271598	5.997859	0.455738	3.178702	3.295561	0.469087	0.471296
	x_{14}	0.521309	0.178230	0.562195	3.228055	0.010000	0.467744	0.466231
	x_{15}	0.151406	3.001337	0.583698	3.399763	0.582466	0.460982	0.461420
	x_{16}	0.399679	0.146228	0.217040	3.175888	0.223557	0.460596	0.456837
	x_{17}	0.230483	3.128780	0.749454	0.010000	0.609969	0.451792	0.452459
	x_{18}	0.428238	3.056438	0.314096	1.504638	0.244804	0.451123	0.448267
	x_{19}	0.255280	0.132435	0.887907	0.937604	0.854372	0.451122	0.444247
	x_{20}	0.410572	3.063635	0.397180	0.551531	0.604213	0.451122	0.440382
Constraints	g_1	-0.34800	-0.03235	-0.19350	-0.19619	-0.92646	-4.288E-7	1.214E-10
	g_2	-0.75806	-0.65044	-0.79085	-0.66649	-0.73621	-0.800389	-0.800449
NFE [a]		n/a[b]	n/a[b]	n/a[b]	n/a[b]	n/a[b]	22,000	**16,000**
Average Rank		3.67	6.00	4.17	5.00	4.50	1.67	1.33
Overall Rank		3	7	4	6	5	2	**1**

[a] The required NFEs to find the solution with absolute error less than 10^{-3}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-3} within 40,000 NFEs

The ANOVA and MCT results for different dimensional Keane's bumpy function are summarized in Fig. 7. To illustrate the performance of the hybrid PGP over other compared algorithms, the convergence history of the best solution is also presented in this figure. The box-plot diagram, in the middle column, presents a box and whisker plot for the applied algorithm. On the right side of Fig. 7, the results of the MCT are also provided among the different optimizers where the red color lines signifies the methods which are statistically differ with the proposed PGP algorithm. The fast convergence rate and superior performance of both the proposed PSO-GRG and PGP algorithms compared to other algorithms are quite obvious from Fig. 7. It can be seen that the results of the PSO-GRG are very close to the PGP algorithm. However, for all dimensions of Keane's bumpy function, except for n=50, the PGP showed faster convergence speed compared to standard PSO, PSO-GRG, and other compared algorithms. These results verify the superior performance of both proposed algorithms compared to the standard PSO algorithm.

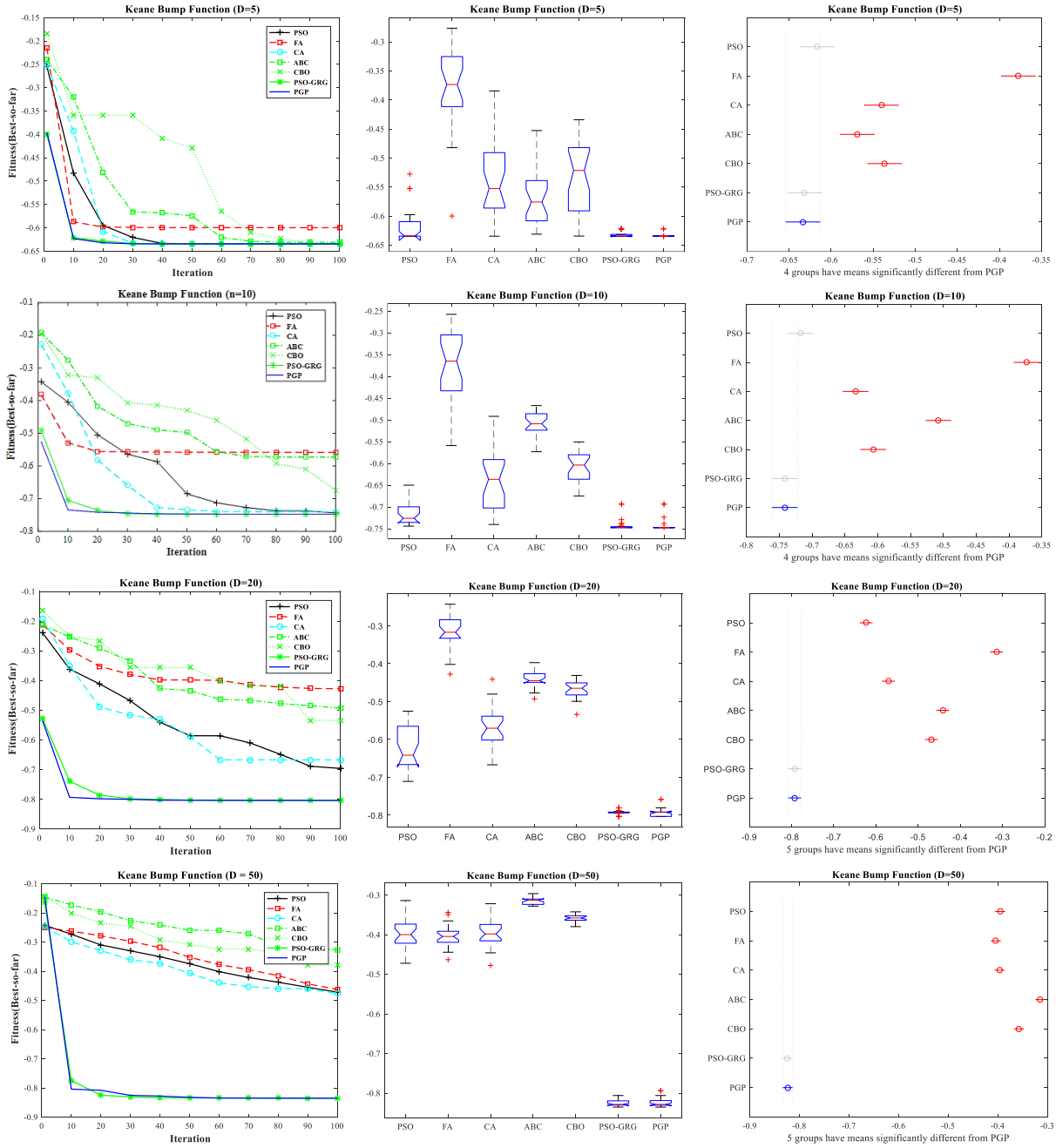


Fig. 7. Convergence history and box-plot for the 5-, 10-, 20- and 50-D Keane's bumpy function.

5.2 Constraint engineering problems

In this section, six Constraint Engineering Problems (CEPs), including pressure vessel, welded beam, tension/compression spring, speed reducer, tabular column, and three-bar truss design optimization problem having various objective functions, constraints, and various design variables are investigated to demonstrate the performance and efficiency of the proposed algorithm.

Table 5
Comparison results of Keane's bumpy problem (n=50).

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		-0.471779	-0.462759	-0.477349	-0.327489	-0.379232	-0.834949	**-0.835088**
Average		-0.395252	-0.403853	-0.396094	-0.313893	-0.357085	**-0.824259**	-0.823434
Worst		-0.313352	-0.343236	-0.321390	-0.295881	-0.341882	**-0.805795**	-0.793268
Std.		0.042339	0.027269	0.034852	0.008739	**0.008446**	0.009062	0.011403
Median		-0.399455	-0.404030	-0.397641	-0.312641	-0.357318	**-0.828226**	**-0.828226**
Best Design	x_1	9.410511	3.469326	10.000000	9.216507	10.000000	6.290523	6.285914
	x_2	10.000000	9.564449	6.075451	9.778087	9.644735	3.186494	3.188338
	x_3	2.938229	9.073511	5.390213	9.617806	3.089164	3.180744	3.138149
	x_4	0.167756	0.264134	10.000000	6.395946	0.010000	3.147263	3.127561
	x_5	6.228020	5.988497	9.977477	3.477556	3.513152	3.140566	3.117755
	x_6	9.998802	1.186314	9.521247	6.383476	5.982741	3.128632	3.083772
	x_7	9.815465	5.788175	6.404183	8.967747	3.797366	3.117164	3.081126
	x_8	3.209468	2.795344	3.012899	0.010000	3.553729	3.085441	3.078921
	x_9	2.203997	5.783436	2.556640	6.203629	3.991166	3.047790	3.070413
	x_{10}	3.369505	2.942758	0.238238	5.947903	2.764745	3.031983	3.053349
	x_{11}	2.277520	3.099589	3.136143	0.010000	0.010000	3.031983	3.053349
	x_{12}	2.823267	0.205758	2.585436	6.419981	0.191389	3.017884	3.035318
	x_{13}	0.043759	2.974324	0.264356	3.214985	4.510291	3.015682	3.019325
	x_{14}	3.139312	0.150017	0.175870	2.719449	3.333419	3.007139	3.001415
	x_{15}	0.409005	0.010000	5.756149	6.070885	1.863309	2.990378	3.001415
	x_{16}	3.038962	2.310010	3.101174	6.177382	2.159808	2.965040	2.981005
	x_{17}	3.078435	2.168791	3.518537	6.367681	3.292086	2.965040	2.968878
	x_{18}	0.546931	3.521509	2.989660	3.927616	1.852892	2.963173	2.948394
	x_{19}	3.055806	2.913267	2.964523	1.400079	3.311899	2.941978	2.925414
	x_{20}	3.363717	0.478454	3.060151	3.120178	2.680189	2.904756	2.925316
	x_{21}	0.598522	2.140694	0.575208	3.030786	3.045862	0.523359	0.481948
	x_{22}	3.006172	2.081622	0.119909	3.071928	3.331903	0.482412	0.478570
	x_{23}	0.476249	2.963819	0.075651	0.010000	0.297334	0.477589	0.477181
	x_{24}	0.520696	2.597928	3.378232	5.997941	3.018086	0.473354	0.475846
	x_{25}	3.043674	2.962616	0.340267	6.169653	0.332645	0.473354	0.475648
	x_{26}	3.262273	0.581859	3.214185	3.038837	3.577580	0.463703	0.475648
	x_{27}	0.506012	2.146101	0.120155	0.010000	2.289353	0.463703	0.473181
	x_{28}	0.316194	3.030326	0.310632	3.491023	1.326892	0.462681	0.473181
	x_{29}	0.015446	0.543079	3.227684	0.010000	2.478583	0.461987	0.469121
	x_{30}	0.397311	0.076902	1.198997	0.010000	2.696746	0.461987	0.468282
	x_{31}	3.016391	3.116546	0.153911	3.073662	0.129538	0.461813	0.468282
	x_{32}	2.905716	3.172568	0.129747	6.355025	3.369792	0.461813	0.468282
	x_{33}	3.063392	0.010000	3.428635	0.010000	0.010000	0.459576	0.467112
	x_{34}	0.397887	0.667877	3.182303	2.884593	1.860637	0.458884	0.466180
	x_{35}	2.333931	2.797400	0.487039	0.010000	2.477807	0.457712	0.461353
	x_{36}	0.088133	2.814992	1.957402	0.010000	3.226924	0.457712	0.458760
	x_{37}	0.149987	0.353585	0.200659	3.412273	0.687868	0.457507	0.458596
	x_{38}	0.030620	2.688724	0.387373	5.867905	3.089519	0.454784	0.449368
	x_{39}	2.746330	0.116995	0.303985	5.947218	3.038505	0.452126	0.448437
	x_{40}	3.370927	3.123347	0.880716	6.136220	0.117234	0.450870	0.448437
	x_{41}	0.682072	1.972640	0.439939	2.944232	2.053382	0.450784	0.448433
	x_{42}	2.948914	0.570532	3.268731	0.010000	1.007302	0.450784	0.448433
	x_{43}	2.924522	0.746806	0.055606	0.010000	0.517989	0.449387	0.448416
	x_{44}	0.372355	0.636965	1.326077	2.996635	3.307481	0.449387	0.448416
	x_{45}	2.865578	0.120737	0.131260	3.146681	1.111492	0.449387	0.448408
	x_{46}	2.210320	2.656277	0.216518	3.052102	2.196973	0.449387	0.448408
	x_{47}	0.426598	0.131851	0.104023	6.418209	0.875964	0.449387	0.447125
	x_{48}	0.010000	0.246309	3.227407	2.938445	0.369650	0.449387	0.447125
	x_{49}	0.166768	0.227149	0.110096	0.010000	0.306980	0.449387	0.447125
	x_{50}	0.019552	0.356385	0.301644	0.419811	0.386300	0.449387	0.446836
Constraints	g_1	-0.255234	-0.947413	-0.210178	-0.206746	-0.999996	-1.552E-04	-6.725E-04
	g_2	-0.674691	-0.705759	-0.670447	-0.504213	-0.674431	-0.792071	-0.792247
NFE [a]		n/a[b]	n/a[b]	n/a[b]	n/a[b]	n/a[b]	56,000	**22,000**
Average Rank		4.83	3.67	4.33	5.50	4.33	**1.67**	1.83
Overall Rank		5	3	4	6	4	**1**	2

[a] The required NFEs to find a solution with absolute error less than 10^{-3}

[b] The algorithm was not able to find a solution with absolute error less than 10^{-3} within 100,000 NFEs

5.2.1. Optimal design of the pressure vessel

This example was frequently applied to evaluate the performance of various optimization algorithms such as MFO [87], multiverse optimization [88], crow search algorithm (CSA) [89], thermal exchange optimization (TEO) [90], NM-PSO [91], DELC [92], Co-evolutionary DE (CDE) [93], and MDDE [94]. This problem includes two discrete and two continuous design variables, four unequal constraints, and aims to minimize the total cost of producing a cylindrical pressure vessel shown in Fig. 8. The optimization formulation for this problem is as follows [2]:

$$\text{Minimize: } f(T_s.T_h.R.L) = 0.6224T_sRL + 1.7781T_hR^2 + 3.1661T_s^2L + 19.84T_s^2R \tag{15}$$

Subjected to:

$$g_1 = -T_s + 0.0193R \le 0; \quad g_2 = -T_h + 0.0095R \le 0;$$
$$g_3 = -\pi R^2L - \frac{4}{3}\pi R^3 + 1.296.000 \le 0; \quad g_4 = L - 240 \le 0 \tag{16}$$

As illustrated in Fig. 8, the design variables include the thickness of the shell (T_s), the cylinder cap thickness (T_h), the radius of the cylinder (R), and the length of the shell (L). The thickness of the shell and warhead must be multiples of 0.026 in, and between the range of 1× 0.0625 to 99× 0.0625 in. The radius and length of the shell are also limited between 10 and 200 in.

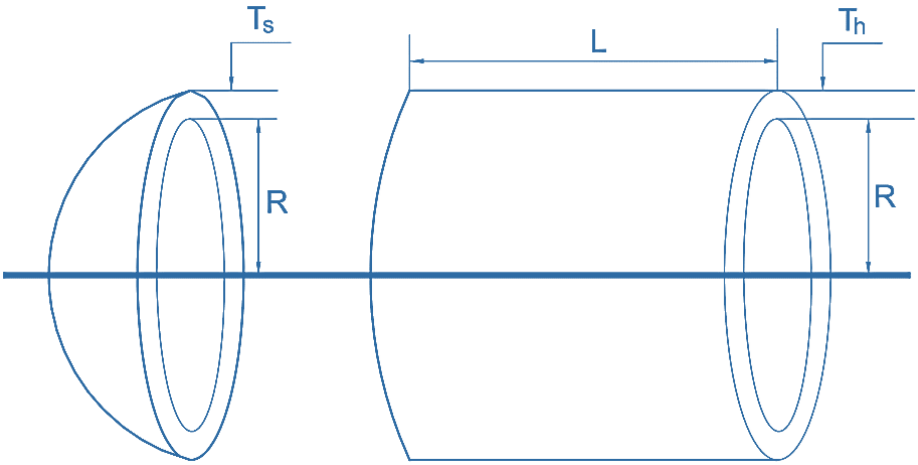


Fig. 8. The pressure vessel optimization problem and corresponding design variables.

A summary of the optimization results of the proposed PSO-GRG and the PGP algorithms for this problem is presented in Table 6. For comparison analysis results of the standard PSO, FA, CA, ABC, and CBO algorithms are also presented. The results presented in Table 6 demonstrate the superiority of the proposed PGP algorithm in the Best, Average, standard deviation, Worst, and NFE value over the other optimization algorithms. Remarkably, the PSO-GRG algorithm possesses the best performance in the Median value, followed by PGP, PSO, and ABC. It is worth mentioning that the PSO, FA, CA, ABC, and CBO algorithms were not able to find the solution with an absolute error of less than 10^{-1} within 8000 NFEs, while the proposed PSO-GRG and PGP achieved the optimum solution with the 6240 and 4240 NFEs, respectively.

Table 6
Statistical and comparison results of the pressure vessel problem.

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		6059.8265	6,060.2731	6,090.5273	6,083.9245	6,403.85485	6,059.71444	**6,059.71434**
Average		6417.1191	6,481.5781	6,390.3885	6299.0163	6,974.03672	6,369.47673	**6,157.81921**
Worst		7544.4925	7,333.4231	6,820.4169	6549.9455	7,544.49252	7,544.49252	**6,371.59763**
Std.		463.28956	440.76311	254.91880	133.73136	408.266446	454.834390	**115.695200**
Median		6235.2429	6,371.2812	6410.0885	6,302.6835	7,047.34813	**6,090.526838**	6,110.086867
Best Design	x_1	0.8125	0.8125	0.8125	0.8125	0.8125	0.8125	0.8125
	x_2	0.4375	0.4375	0.4375	0.4375	0.4375	0.4375	0.4375
	x_3	42.098005	42.097427	45.336787	41.955367	48.504412	42.098445	42.098446
	x_4	176.644556	176.667808	140.253890	178.705513	111.513835	176.636605	176.636597
Constraints	g_1	-0.00002	-1.37E-06	-1.37E-06	-0.00425	-0.03425	-8.80E-07	-3.1764E-10
	g_2	-0.03589	-3.59E-04	-3.59E-04	-0.03412	-0.05442	-0.0359	-0.03756
	g_3	-27.8861	-118.7687	-118.762	-127.654	-205.6	-0.05442	-0.00012
	g_4	-63.3459	-63.2535	-63.2527	-63.3212	-40	-63.3634	-63.3634
NFE [a]		n/a [b]	n/a [b]	n/a [b]	n/a [b]	n/a [b]	6,240	**4,240**
Average Rank		5.00	4.57	4.00	2.86	5.43	3.57	**1.14**
Overall Rank		6	5	3	2	7	3	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-1}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-1} within 8000 NFEs

5.2.2. Welded beam design problem

The optimal design of the welded beam, introduced by Rao [95], is investigated as another benchmark CEP to investigate the capability of different algorithms [75–79]. The design variables include the weld thickness (h), the weld length (l), the beam width (b), and the beam thickness (t) as to be visible in Fig. 9. The constraints for this problem consist of shear stress (τ), bending stress (σ), buckling pressure (P), and maximum end deflection (δ). Achieving a set of product variables to optimize the construction costs of the welded beam is the main objective of this example. The mathematical formula of the problem is given as:

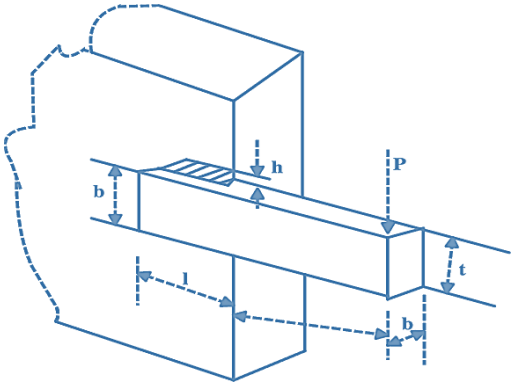


Fig. 9. Welded beam design problem and corresponding design variables

$$\text{Minimize: } f(h.L.b.t) = 1.1047h^2l + 0.04811bt(14.0 + l) \tag{17}$$

Subject to:

$$g_1 = \tau - 13600 \le 0;\ g_2 = \sigma - 30000 \le 0;\ g_3 = w - h \le 0;$$
$$g_4 = 0.1047h^2 + 0.04811tb(14 + l) - 0.5 \le 0;\ g_5 = 0.125 - h \le 0; \tag{18}$$
$$g_6 = \delta - 0.25 \le 0;\ g_7 = 6000 - P \le 0;$$

where:

$$\sigma = \frac{504000}{bt^2};\ Q = 6000\left(14 + \frac{l}{2}\right);\ D = \frac{1}{2}\sqrt{(l^2 + (h+t)^2}};\ \delta = \frac{65856}{30000bt^3}$$
$$J = \sqrt{2}hl\left[\frac{l^2}{6} + \frac{(h+t)^2}{2}\right];\alpha = \frac{6000}{\sqrt{2}hl};\ \beta = \frac{QD}{J};\ \tau = \sqrt{\alpha^2 + \frac{\alpha\beta l}{D} + \beta^2} \quad (19)$$
$$P = 0.61423 \times 10^6 \frac{tb^3}{6}\left(1 - \frac{t\sqrt{30/48}}{28}\right)$$

The upper and lower bound of design variables are:

$$0.1 \le [l.b] \le 10.0, and\ 0.1 \le [h.t] \le 2.0 \quad (20)$$

The comparison results of the PGP algorithm and other algorithms are presented in Table 7. As can be seen, the proposed PGP algorithm provided superior results in terms of the Best and Median of the optimum solution by performing 6800 NFEs. For this problem, the FA algorithm achieved the lowest value in Average, Worst, and standard deviation of the best solution. The second minimum value in the Worst and Average is obtained by PGP. Also, the PSO and PGP algorithms found the best value in standard deviation after the FA. Again, none of the other algorithms can provide a solution with an absolute error less than 10^{-5}, even after 8000 NFEs.

Table 7
Statistical and comparison results of the welded beam problem.

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		1.696147	1.696056	1.695351	1.790788	1.697897	1.695265	**1.695260**
Average		1.705324	**1.697902**	1.864491	2.080159	1.784404	1.715602	1.702906
Worst		1.794779	**1.700015**	2.508534	2.416066	2.102385	1.828331	1.793794
Std.		0.020529	**0.000866**	0.206230	0.155202	0.122863	0.041257	0.020762
Median		1.698942	1.697962	1.769431	2.051234	1.711707	1.696341	**1.695596**
Best Design	x1	0.205588	0.205825	0.205758	0.224663	0.205042	0.205730	0.205712
	x2	3.254807	3.253392	3.252786	3.116304	3.266899	3.253132	3.253447
	x3	9.042406	9.036001	9.036002	8.566833	9.042781	9.036566	9.036629
	x4	0.205715	0.205828	0.205758	0.229220	0.205828	0.205732	0.205731
Constraint	g_1	-2.786410	-6.803175	-0.010849	-190.329092	-11.888931	-0.009043	-0.056304
	g_2	-0.054237	-0.054067	-0.054000	-0.054263	-0.054361	-0.054000	-0.054001
	g_3	-36.249462	-10.180023	-0.013431	-40.235039	-55.193204	-0.007961	-0.166476
	g_4	-1.269E-04	-2.562E-06	-2.713E-8	-4.556E-03	-7.864E-04	-2.355E-06	-1.851E-5
	g_5	-1.212396	-8.279332	-2.169778	-2,007.408480	-11.278900	-0.163051	-0.035736
	g_6	-0.080588	-0.080825	-0.080758	-0.099663	-0.080042	-0.080730	-0.080712
	g_7	-3.409135	-3.409403	-3.410011	-3.327215	-3.407387	-3.410089	-3.410072
NFE [a]		n/a [b]	n/a [b]	n/a [b]	n/a [b]	n/a [b]	n/a [b]	**6,800**
Average Rank		3.17	2.00	5.17	5.83	4.67	3.00	**1.67**
Overall Rank		4	2	6	7	5	3	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-5}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-5} within 8000 NFEs

5.2.3. Optimal design of the tension/compression spring

Design optimization of the tension/compression spring shown in Fig. 10 is considered as another CEPs [96]. This is an optimization problem with the original purpose to minimize the weight of

compression/tension spring exposed to surge frequency, shear stress, and minimum deformation. This problem possesses three design variables including the wire diameter (d), mean coil diameter (w), and the number of active coils (N). The problem is expressed as follows:

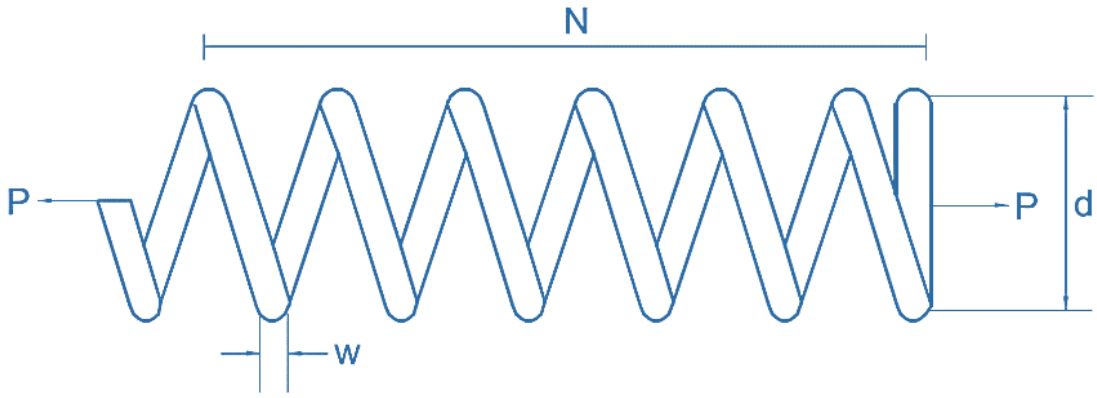


Fig. 10. Schematic view of the spring design problem and corresponding design variables.

$$\text{Minimize: } f(w,d,N) = (N+2)w^2 d \tag{21}$$

Subject to:

$$g1 = 1 - \frac{d^3 N}{71785w^4} \le 0; \quad g2 = \frac{d(4d-w)}{12566w^3(d-w)} + \frac{1}{5108w^2} - 1 \le 0;$$
$$g3 = 1 - \frac{140.45w}{d^2 N} \le 0; \quad g4 = \frac{2(w+d)}{3} - 1 \le 0 \tag{22}$$

where

$$2.0 \le N \le 15.0; \qquad 0.025 \le d \le 1.3; \quad 0.05 \le \mathrm{w} \le 2.0 \tag{23}$$

Belegundu [96] used eight different mathematical optimization methods to solve this problem. This example was investigated by Arora [97] employing a numerical optimization approach, Coello et al. [98] using a GA-based method, and also Wang [99] utilizing a Co-evolutionary PSO (CPSO). According to recent studies, researchers found the optimum results using Water Cycle Algorithm (WCA) [100], and the Charged System Search (CSS) [101].

The best results acquired by the PGP algorithm and those reported by other researchers have been compared in Table 8. Investigating the Best, Average, Worst, Median, standard deviation and NFE demonstrate that the PGP algorithm performance is more consistent than the other compared algorithms. Remarkably, the PGP and CA algorithm converged the same value for the best solution. Overall, the FA algorithm has possessed the best performance after the PGP algorithm. Moreover, both the proposed PSO-GRG and the PGP algorithms significantly enhanced the performance of the standard PSO, specifically in terms of the lower NFEs.

Table 8
Statistical and comparison results of the tension/compression spring problem.

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		0.012670	0.012669	**0.012665**	0.012699	0.012897	0.012666	**0.012665**
Average		0.013419	0.012957	0.013681	0.013204	0.013744	0.013659	**0.012832**
Worst		0.017783	0.013727	0.017752	0.014220	0.014832	0.017773	**0.013349**
Std.		0.001397	0.000337	0.001363	0.000394	0.000604	0.001518	**0.000167**
Median		0.012874	0.012811	0.013185	0.013099	0.013816	0.013002	**0.012720**
Best Design	x1	0.051216	0.051674	0.051603	0.052449	0.055333	0.051919	0.051659
	x2	0.345454	0.356324	0.354660	0.374894	0.450875	0.362271	0.355982
	x3	11.981789	11.315872	11.410636	10.313233	7.342640	10.970663	11.332314
Constraint	g_1	-0.000845	-2.45E-03	-1.65E-13	-0.000013	-0.051396	-5.39E-07	-7.89E-08
	g_2	-1.26E-05	-3.16E-03	-7.90E-14	-0.000021	-0.00134	-2.56E-08	-3.77E-08
	g_3	-4.051300	-4.055676	-4.053345	-1.061328	-4.145832	-4.05879	-4.05519
	g_4	-0.727090	-1.563377	-0.727864	-0.7222698	-0.698289	-1.126787	-1.09087
NFE [a]		4,320	4,080	3,360	n/a [b]	n/a [b]	2,640	**2,400**
Average Rank		5.00	2.67	4.33	4.33	5.83	4.50	**1**
Overall Rank		5	2	3	3	6	4	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-5}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-5} within 6000 NFEs

5.2.4. speed reducer

The weight minimization of the speed reducer, shown in Fig. 11, subject to constraints on bending stress of the gear teeth, surfaces stress, transverse deflections of the shafts, and stresses in the shafts is the purpose of this example [102]. The design variables are the face width (b), the module of teeth (m), number of teeth in the pinion (z), length of the first shaft between bearings (l_1), length of the second shaft between bearings (l_2), diameter of the first shaft (d_1), and diameter of the second shaft (d_2), respectively. All variables are continuous except the third one that is a discrete variable [103].

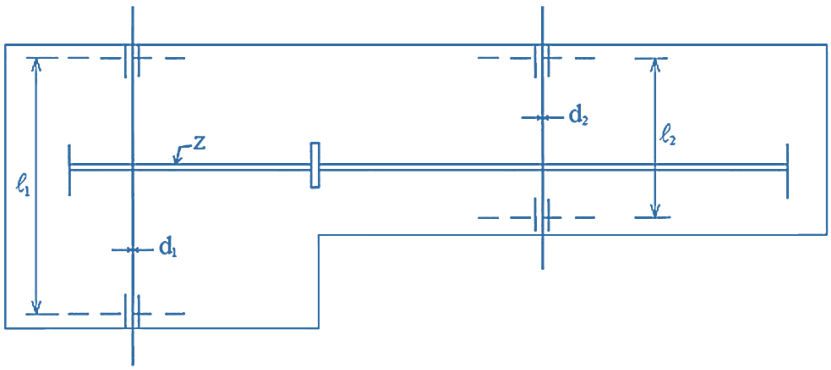


Fig. 11. The speed reducer design problem.

Minimize:

$$f(b,m,z,l_1,l_2,d_1,d_2) = 0.785bm^2(3.3333z^2 + 14.9334z - 43.0934) - 1.508y_1(d_1^2 + d_2^2) + 7.477(d_1^3 + d_2^3) + 0.7854(l_1d_1^2 + l_2d_2^2) \quad (24)$$

Subject to:

$$g_1 = \frac{27}{bm^2z} - 1 \le 0;\ g_2 = \frac{397.5}{bm^2z^2} - 1 \le 0;\ g_3 = \frac{1.93l_1^3}{mzd_1^4} - 1 \le 0;$$
$$g_4 = \frac{1.93l_2^3}{mzd_2^4} - 1 \le 0;\ g_5 = \frac{\sqrt{(745l_1/mz)^2 + 1.69 \times 10^6}}{110d_1^3} - 1 \le 0;$$
$$g_6 = \frac{\sqrt{(745l_2/mz)^2 + 157.5 \times 10^6}}{8d_2^3} - 1 \le 0;\ g_7 = \frac{mz}{40} - 1 \le 0;\ g_8 = \frac{5m}{b} - 1 \le 0; \quad (25)$$
$$g_9 = \frac{b}{12m} - 1 \le 0;\ g_{10} = \frac{1.5d_1 + 1.9}{d_1} - 1 \le 0;\ g_{11} = \frac{1.1d_2 + 1.9}{l_2} - 1 \le 0$$

where $2.6 \le b \le 3.6, 0.7 \le m \le 0.8, 17 \le z \le 28, 7.3 \le l_1 \le 8.3, 7.8 \le l_2 \le 8.3, 2.9 \le d_1 \le 3.9,\ 5.0 \le d_2 \le 5.5$

Table 9
Statistical and comparison results of the speed reducer problem.

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		2,895.33335	2,895.33899	2,895.333352	2,895.340173	2,895.355131	**2,895.333350**	**2,895.333350**
Average		2,895.347031	2,895.362097	2,895.425885	2,895.394523	2,895.394514	2,895.335864	**2,895.334607**
Worst		2,895.446645	2,895.431787	2,897.645068	2,895.483608	2,895.430139	**2,895.349061**	**2,895.349061**
Std.		0.037544	0.020329	0.462330	0.045243	0.020053	0.005879	**0.004350**
Median		2,895.333431	2,895.356897	2,895.333389	2,895.392151	2,895.395483	**2,895.333350**	**2,895.333350**
Best Design	x_1	3.500000	3.500013	3.500000	3.500012	3.500009	3.500000	3.500000
	x_2	0.700000	0.700000	0.700000	0.700000	0.700001	0.700000	0.700000
	x_3	17.000000	17.000000	17.000000	17.000000	17.000020	17.000000	17.000000
	x_4	7.300000	7.300000	7.300000	7.300000	7.300000	7.300000	7.300000
	x_5	7.800000	7.800000	7.800000	7.800000	7.800212	7.800000	7.800000
	x_6	2.900000	2.900000	2.900000	2.900000	2.900000	2.900000	2.900000
	x_7	5.286683	5.286684	5.286683	5.286686	5.286695	5.286683	5.286683
Constraint	g_1	-0.073915	-0.073919	-0.073915	-0.073919	-0.073920	-0.073915	-0.073915
	g_2	-0.197999	-0.198001	-0.197999	-0.198001	-0.198004	-0.197999	-0.197999
	g_3	-0.107955	-0.107955	-0.107955	-0.107955	-0.107956	-0.107955	-0.107955
	g_4	-0.901472	-0.901472	-0.901472	-0.901472	-0.901465	-0.901472	-0.901472
	g_5	-0.486358	-0.486358	-0.486358	-0.486358	-0.486358	-0.486358	-0.486358
	g_6	-9.738E-10	-4.450E-07	-9.255E-10	-1.664E-06	-6.595E-06	-1.878E-13	-7.605E-14
	g_7	-0.702500	-0.702500	-0.702500	-0.702500	-0.702499	-0.702500	-0.702500
	g_8	-1.025E-09	-3.704E-06	-1.023E-10	-3.570E-06	-1.622E-06	-3.201E-13	-1.044E-14
	g_9	-0.583333	-0.583332	-0.583333	-0.583332	-0.583333	-0.583333	-0.583333
	g_{10}	-0.143836	-0.143836	-0.143836	-0.143836	-0.143836	-0.143836	-0.143836
	g_{11}	-0.010852	-0.010852	-0.010852	-0.010852	-0.010878	-0.010852	-0.010852
NFE [a]		n/a [b]	n/a [b]	12,880	n/a [b]	n/a [b]	10,640	**9,660**
Average Rank		4.14	4.33	5.00	5.67	4.83	1.50	**1**
Overall Rank		3	4	6	7	5	2	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-5}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-5} within 14000 NFEs

The solutions obtained by the prosed PSO-GRG and PGP and other algorithms are presented in Table 9. For this problem, both the prosed PSO-GRG and PGP algorithms provided better results in terms of the best, worst, and median of the optimum solution. These results are obtained by 10,640 and 9,660 NFEs for the PSO-GRG and PGP algorithms, respectively. However, the reported values for the average, Std. and NFE indicate that the PGP algorithm is superior to all other algorithms. The PSO-GRG algorithm achieved the second rank for this problem. It should also be mentioned that, among other compared algorithms, only the CA (with 12,880 NFEs) was able to provide a solution with absolute error less than 10^{-5}.

5.2.5. Tabular column design

The minimization of the construction cost and materials in designing a uniform tabular section column shown in Fig. 12 with a length (L) of 250 cm to carry a compressive load of $P = 2500$ kgf is the objective of this CEP. The average diameter (d) of the column is limited between 2 and 14 cm, and thickness (t) of the tabular section lies in the range 0.2-0.8 cm. The material with yield stress (σ_y=500 kgf/cm^2), a modulus of elasticity (E=0.85×106 kgf/cm^2), and a density=0.0025 kgf/cm^2 is considered to make the desired column [104]. The optimization model for this example is formulated as follows:

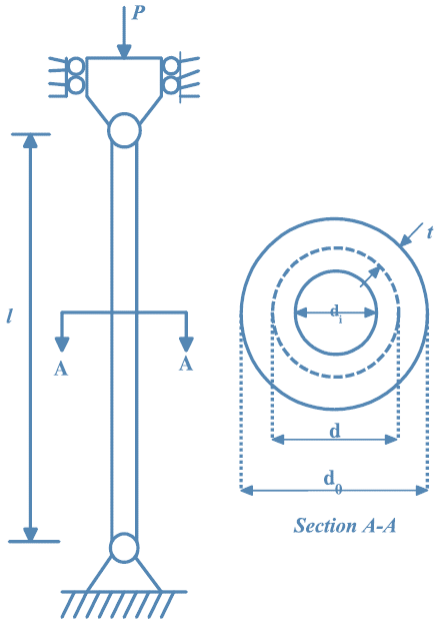


Fig. 12. Tabular column design problem.

$$\text{Minimize: } f(Y) = 9.82\,dt + 2d \tag{26}$$

Subject to:

$$g_2(Y) = \frac{P}{\pi dt\sigma_y} - 1 \le 0;\ g_2(Y) = \frac{8P}{\pi^3 Edt(d^2+t^2)} - 1 \le 0;\ g_3(Y) = \frac{2.0}{d} - 1 \le 0; \tag{27}$$

$$g_4(Y) = \frac{d}{14} - 1 \le 0;\ \ g_5(Y) = \frac{0.2}{t} - 1 \le 0;\ \ g_6(Y) = \frac{t}{0.8} - 1 \le 0$$

It should be mentioned that some of the previous studies by Hsu and Liu [105], and Rao [95] converged to infeasible solutions and unable to provide an accurate solution for this problem. The comparison analysis results between the proposed PSO-based hybrid algorithms and the other compared algorithms together with the statistical data are given in Table 10. These data indicated that the PGP algorithm has achieved the same values in best and median like the PSO-GRG and CBO algorithms, as well as the same value in average such as PSO-GRG. However, the minimum NFEs belong to the proposed PGP algorithm. Moreover, the PGP algorithm obtained a lower value in worst, Std., and NFE compared with the other algorithms. Generally, the PSO-GRG and the CBO achieved the second and third ranks with the 2840 and 2920 NFEs, respectively.

Table 10
Statistical and comparison results of the tabular column problem.

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		26.531337	26.531501	26.531330	26.541629	**26.531328**	**26.531328**	**26.531328**
Average		26.531440	26.532294	26.531542	26.633089	26.531384	**26.531328**	**26.531328**
Worst		26.532150	26.533504	26.532935	26.837602	26.532481	26.531332	**26.531329**
Std.		1.837E-04	5.087E-04	3.604E-04	7.738E-02	2.292E-04	9.524E-07	**2.304E-07**
Median		26.531375	26.532343	26.531417	26.618614	**26.531328**	**26.531328**	**26.531328**
Best Design	x_1	5.451161	5.451149	5.451157	5.451276	5.451156	5.451156	5.451156
	x_2	0.291965	0.291969	0.291965	0.292147	0.291965	0.291965	0.291965
Constraint	g_1	-1.467E-10	-1.203E-05	-1.229E-07	-6.433E-04	-4.895E-12	-1.077E-10	-8.017E-12
	g_2	-1.577E-06	-9.470E-06	-2.505E-07	-6.908E-04	-7.803E-10	-1.545E-11	-2.715E-10
NFE [a]		3,560	n/a [b]	3,520	n/a [b]	2,920	2,840	**2,680**
Average Rank		5.00	6.00	4.67	6.83	2.67	1.50	**1**
Overall Rank		5	6	4	7	3	2	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-5}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-5} within 4000 NFEs

5.2.6. Three-bar truss design problem

The volume minimization of the benchmark three-bar structure (Fig. 13) subjected to the stress constraints on the truss members is investigated as the final example to exhibit the applicability and efficiency of the proposed algorithm. This problem was formerly solved by some researchers using evolutionary computational technique [106], convexification strategies [107], dynamic stochastic selection differential evolution [108], and cuckoo search [109]. The mathematical formula for design optimization of the benchmark three-bar truss is defined as [104]:

$$\text{Minimize: } f(A_1, A_2) = (2\sqrt{2}A_1 + A_2) \times l \tag{28}$$

Subject to:

$$g_1 = \frac{\sqrt{2}A_1 + A_2}{\sqrt{2}\,A_1^2 + 2A_1A_2} P - \sigma \le 0;\ g_2 = \frac{A_2}{\sqrt{2}\,A_1^2 + 2A_1A_2} P - \sigma \le 0; \tag{29}$$

$$g_3 = \frac{1}{A_1 + \sqrt{2}A_2} P - \sigma \le 0;$$

where $l = 100$ cm, $P = 2\frac{\text{KN}}{\text{cm}^2}$, $\sigma = 2$ KN/cm^2, and $0 \leq A_1, A_2 \leq 1$.

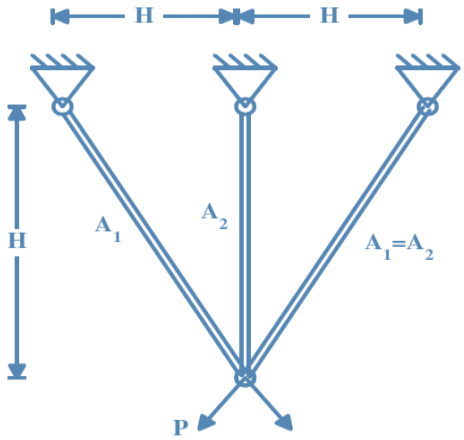


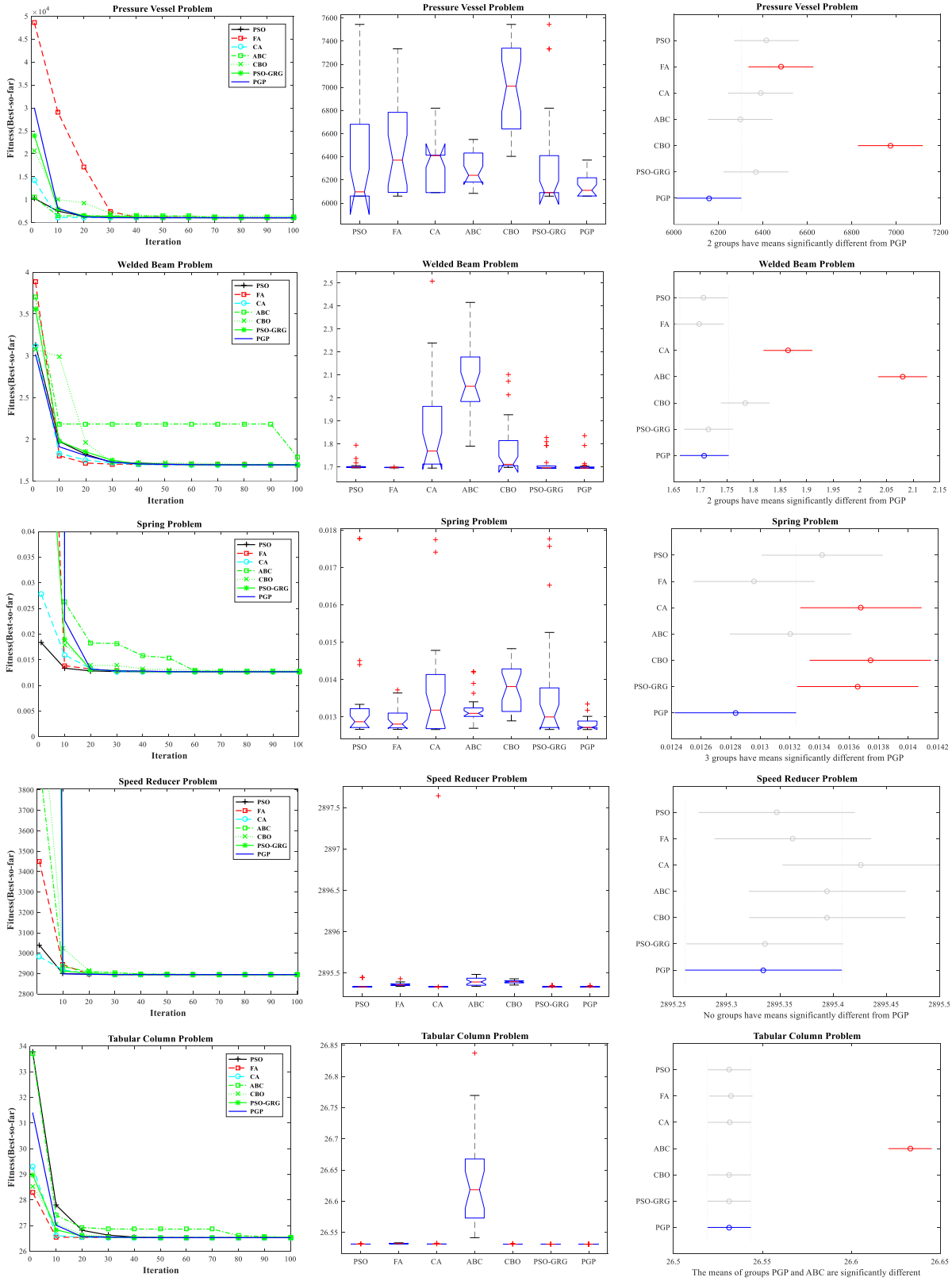
Fig. 13. The three-bar truss design problem.

A summary of the optimization results for this example is presented in Table 11. As can be seen, the PGP algorithm achieved a superior result in terms of the best value same as the CA algorithm. Besides, the minimum values in average, worst, Std., and Median of the optimum solution are obtained by the FA algorithm, followed by the PGP and PSO-GRG algorithms, respectively. Remarkably, the PGP algorithm obtained the best solution (with 1320 NFEs) and with much lower NFEs than those of PSO-GRG (3800), FA (4000), and other compared algorithms. Again, the statistical data provided in Table 11 and a comparison of the NFEs verify that both the proposed algorithms improved the accuracy and efficiency of the standard PSO algorithm.

Table 11
Statistical and comparison results for the benchmark three bar truss structure.

		PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Best		263.895897	263.895860	**263.895844**	263.906859	263.895925	263.895851	**263.895844**
Average		263.909291	**263.896017**	264.187533	264.065740	263.939009	263.902666	263.900698
Worst		263.975412	**263.896291**	267.339494	264.663483	264.445407	263.933167	263.921411
Std.		2.153E-02	**1.183E-04**	7.217E-01	1.622E-01	1.097E-01	9.381E-03	6.038E-03
Median		263.900801	**263.896018**	263.944274	264.058901	263.909091	263.899149	263.898151
Best Design	x_1	0.788828	0.788629	0.788704	0.790610	0.788343	0.788749	0.788657
	x_2	0.407817	0.408380	0.408167	0.402886	0.409189	0.408038	0.408299
Constraint	g_1	-2.786E-07	-1.125E-07	-2.414E-12	-6.330E-05	-2.752E-09	-2.470E-08	-1.312E-09
	g_2	-1.464592	-1.463952	-1.464194	-1.470245	-1.463033	-1.464341	-1.464043
	g_3	-0.753477	-0.754196	-0.753924	-0.747166	-0.755228	-0.753759	-0.754093
NFE [a]		n/a [b]	n/a [b]	2,920	n/a [b]	n/a [b]	3,800	**1,320**
Average Rank [c]		4.00	1.83	5.00	5.83	4.83	3	**1.67**
Overall Rank		4	2	6	7	5	3	**1**

[a] The required NFEs to find a solution with absolute error less than 10^{-5}
[b] The algorithm was not able to find a solution with absolute error less than 10^{-5} within 4000 NFEs



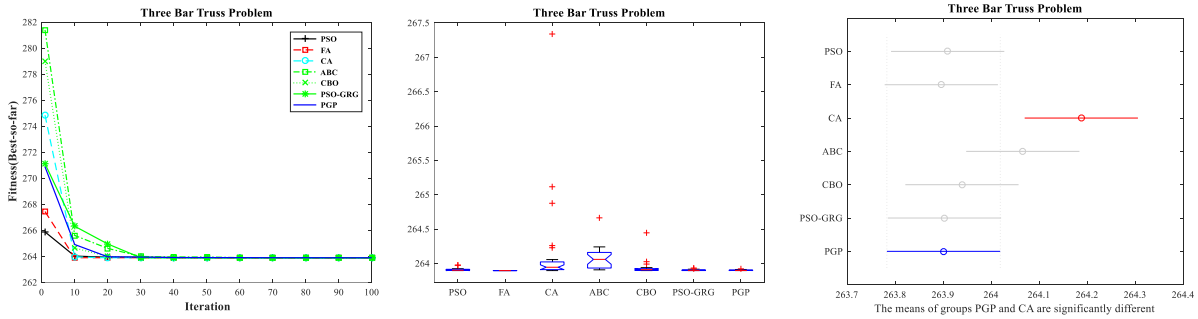


Fig. 14. Convergence history, ANOVA test, and MCT result for different CEPs.

The comparison of the convergence curves for the best solutions, box-plot, and the MCT analysis for all investigated CEPs are presented in Fig.14. The provided MCT results denote the methods which are statistically differ than the presented PGP algorithm. Results of Fig.14 imply that the PGP algorithm is superior to the other compared algorithms in a majority of CEPs and show comparable or significantly better performance against other mentioned algorithms.

5.3. Statistical test results

Two non-parametric statistical tests (NPST) were used for meticulous performance comparison of the proposed PGP, PSO-GRG, and other five optimization algorithms. The consistency and overall proficiency of the developed PGP algorithm are investigated using the best values obtained from 25 runs, ANOVA test, and rank function. The boxes from Figs. 7 and 14 have three lines to show the 1st, 2nd, and 3rd quartiles. The whiskers have lines extending vertically from boxes to demonstrate the grade of the rest of the information. Figs. 7 and 14, also show the MCT results for all employed algorithms. In these figures, the red color lines denote the methods that are statistically differ than the PGP algorithm. Furthermore, all mentioned algorithms were ranked based on the best, average, worst, standard deviation, median, and NFE values for each problem. Finally, the average and overall ranks were presented in Tables 2-11.

Table 12
Comparison of algorithms and Final ranking.

Problem	PSO	FA	CA	ABC	CBO	PSO-GRG	PGP
Keane's bumpy problem (n=5)	3	7	4	6	5	2	1
Keane's bumpy problem (n=10)	3	7	4	6	5	2	1
Keane's bumpy problem (n=20)	3	7	4	6	5	2	1
Keane's bumpy problem (n=50)	5	3	4	6	4	1	2
Pressure vessel problem	6	5	4	2	7	3	1
Welded beam problem	4	2	6	7	5	3	1
Tension/Compression spring problem	5	2	3	3	6	4	1
Speed reducer problem	3	4	6	7	5	2	1
Tabular column problem	5	6	4	7	3	2	1
Three bar truss problem	4	2	6	7	5	3	1
Total	41	45	45	57	50	24	11
Average rank	4.1	4.5	4.5	5.7	5.0	2.4	1.1
Overall rank	3	4	4	6	5	2	1

To decide on one of the utilized algorithms as the best optimizer, the ranking of each algorithm is calculated from Tables 2-11, and a summary of the ranking results for all mathematical and CEPs are listed in Table 12. Results of Table 12 show that the PGP algorithm has better overall performance compared to standard PSO and other compared algorithms. As can be seen, the hybrid PSO-GRG and the standard PSO algorithm rated second and third, respectively. FA and CA jointly in 4^{th} and CBO and ABC stand the 5^{th} and 6^{th} rank, respectively.

6. Conclusions

In this study, a hybrid PSO-GRG algorithm with a purely uniform distributed initial swarm is proposed to enhance the convergence speed and robustness of the standard PSO. In the proposed hybrid PSO-GRG, a fast approximation of the optimal solution is first provided by probing the entire search space during some selective cycles using PSO as a global search engine. Then, the accuracy and quality of the optimum solution are further enhanced by local search around the current best solution using the GRG algorithm as a secondary local search engine of the optimizer, keeping a good compromise between accuracy and efficiency. The k-NN-based PUD operator was also applied for generating the initial swarm to cover the entire search space more effectively. The advantage of employing the PUD operator in the proposed hybridized PSO algorithm is that the particles located in dense subspaces are removed from the initial swarm and replaced with new particles having a larger distance than a certain radius from each other. Therefore, the search agents are scattered within the entire search space with equal density and thus enhance the exploration power of the optimizer. Hybridizing PUD-based PSO with the GRG algorithm provides the opportunity to keep a balance between the exploration and exploitation ability of the optimizer. The performance of the presented algorithms with and without PUD operator, namely the PSO-GRG and the PGP algorithms, were compared with five other well-known optimizers on solving some mathematical and engineering COPs that comprise highly nonlinear, non-convex, and non-differentiable functions having discrete and continuous variables. Results demonstrated that the developed PGP optimizer with the PUD operator remarkably improved the accuracy, efficiency, and convergence speed of the standard PSO. ANOVA test, MCT, and the ranking results reveal that the developed algorithm has robust and accurate performance and is efficient in terms of NFEs and computational cost. In future works, the proposed PGP algorithm may be used for solving large-scale real-world optimization problems which require significant computational efforts efficiently with an acceptable degree of accuracy for the solutions. In this regard, applying the PGP algorithm to solve more practical and complex CEPs comprising truss design, frame design, damage detection, and performance-based design optimization is the main scope for future researches. The capability of the proposed algorithm can also be enhanced for solving binary and multi-objective optimization problems.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Authors Contribution Statement

H. Varaee, N. Safaeian Hamzehkolaei, and M. Safari contributed to the conceptualization, theoretical framework, design and implementation of the research, analysis of the results, and to the writing of original draft; H. Varaee and N. Safaeian Hamzehkolaei: Writing – review & editing; N. Safaeian Hamzehkolaei was in charge of overall direction and planning.

References

[1] Lin YC. Mixed-integer constrained optimization based on Memetic Algorithm. J Appl Res Technol 2013;11:242–50. doi:10.1016/S1665-6423(13)71534-7.

[2] Varaee H, Ghasemi MR. Engineering optimization based on ideal gas molecular movement algorithm. Eng Comput 2017;33:71–93. doi:10.1007/s00366-016-0457-y.

[3] Barbosa TM, Bragada JA, Reis VM, Marinho DA, Carvalho C, Silva AJ. Energetics and biomechanics as determining factors of swimming performance: Updating the state of the art. J Sci Med Sport 2010;13:262–9. doi:10.1016/j.jsams.2009.01.003.

[4] Javidy B, Hatamlou A, Mirjalili S. Ions motion algorithm for solving optimization problems. Appl Soft Comput J 2015;32:72–9. doi:10.1016/j.asoc.2015.03.035.

[5] Molina D, Poyatos J, Ser J Del, García S, Hussain A, Herrera F. Comprehensive Taxonomies of Nature- and Bio-inspired Optimization: Inspiration Versus Algorithmic Behavior, Critical Analysis Recommendations. Cognit Comput 2020;12:897–939. doi:10.1007/s12559-020-09730-8.

[6] Ahmadi-Nedushan B, Varaee H. Minimum cost design of concrete slabs using particle swarm optimization with time varying acceleration coefficients. World Appl Sci J 2011;13:2484–94.

[7] Liu L, Yang S, Wang D. Particle swarm optimization with composite particles in dynamic environments. IEEE Trans Syst Man, Cybern Part B Cybern 2010;40:1634–48. doi:10.1109/TSMCB.2010.2043527.

[8] Dorigo M, Maniezzo V, Colorni A. Ant system: Optimization by a colony of cooperating agents. IEEE Trans Syst Man, Cybern Part B Cybern 1996;26:29–41. doi:10.1109/3477.484436.

[9] Karaboga D, Basturk B. Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics), vol. 4529 LNAI, Citeseer; 2007, p. 789–98. doi:10.1007/978-3-540-72950-1_77.

[10] Yang X-S. Firefly algorithm, stochastic test functions and design optimization. Int J Bio-Inspired Comput 2 2010;2:78–84. doi:10.1504/IJBIC.2010.032124.

[11] Krishnanand KN, Ghose D. Glowworm swarm optimisation: a new method for optimising multi-modal functions. Int J Comput Intell Stud 2009;1:93. doi:10.1504/ijcistudies.2009.515637.

[12] Breitung K. The geometry of limit state function graphs and subset simulation: Counterexamples. Reliab Eng Syst Saf 2019;182:98–106. doi:10.1016/j.ress.2018.10.008.

[13] Yazdani M, Jolai F. Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm. J Comput Des Eng 2016;3:24–36. doi:10.1016/j.jcde.2015.06.003.

[14] Mirjalili SM, Mirjalili SM, Lewis A. Grey Wolf Optimizer. Adv Eng Softw 2014;69:46–61. doi:10.1016/j.advengsoft.2013.12.007.

[15] Wang GG, Deb S, Cui Z. Monarch butterfly optimization. Neural Comput Appl 2019;31:1995–2014. doi:10.1007/s00521-015-1923-y.

[16] Gandomi AH, Alavi AH. Krill herd: A new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 2012;17:4831–45. doi:10.1016/j.cnsns.2012.05.010.

[17] Wang GG, Deb S, Coelho LDS. Elephant Herding Optimization. Proc - 2015 3rd Int Symp Comput Bus Intell ISCBI 2015, 2016, p. 1–5. doi:10.1109/ISCBI.2015.8.

[18] Rajabioun R. Cuckoo optimization algorithm. Appl Soft Comput J 2011;11:5508–18. doi:10.1016/j.asoc.2011.05.008.

[19] Guo L, Wang GG, Gandomi AH, Alavi AH, Duan H. A new improved krill herd algorithm for global numerical optimization. Neurocomputing 2014;138:392–402. doi:10.1016/j.neucom.2014.01.023.

[20] Wang GG, Deb S, Gao XZ, Dos Santos Coelho L. A new metaheuristic optimisation algorithm motivated by elephant herding behaviour. Int J Bio-Inspired Comput, vol. 8, 2016, p. 394–409. doi:10.1504/IJBIC.2016.081335.

[21] Ghasemi MR, Varaee H. A fast multi-objective optimization using an efficient ideal gas molecular movement algorithm. Eng Comput 2017;33:477–96. doi:10.1007/s00366-016-0485-7.

[22] Ghasemi MR, Ghiasi R, Varaee H. Probability-Based Damage Detection of Structures Using Surrogate Model and Enhanced Ideal Gas Molecular Movement Algorithm. Adv Struct Multidiscip Optim, vol. 4, 2018, p. 1657–74. doi:10.1007/978-3-319-67988-4_124.

[23] Ghasemi MR, Varaee H. Damping vibration-based IGMM optimization algorithm: fast and significant. Soft Comput 2019;23:451–81. doi:10.1007/s00500-017-2804-3.

[24] Ghasemi MR, Ghiasi R, Varaee H. Probability-Based Damage Detection of Structures Using Surrogate Model and Enhanced Ideal Gas Molecular Movement Algorithm. Adv Struct Multidiscip Optim, vol. 4, 2018, p. 1657–74. doi:10.1007/978-3-319-67988-4_124.

[25] Ghasemi MR, Varaee H. Enhanced IGMM optimization algorithm based on vibration for numerical and engineering problems. Eng Comput 2018;34:91–116. doi:10.1007/s00366-017-0523-0.

[26] Ghasemi MR, Varaee H. Modified Ideal Gas Molecular Movement Algorithm Based on Quantum Behavior. In: Schumacher A, Vietor T, Fiebig S, Bletzinger K-U, Maute K, editors. Adv Struct Multidiscip Optim, Cham: Springer International Publishing; 2018, p. 1997–2010. doi:10.1007/978-3-319-67988-4_148.

[27] Wang GG, Guo L, Duan H, Wang H. A new improved firefly algorithm for global numerical optimization. J Comput Theor Nanosci 2014;11:477–85. doi:10.1166/jctn.2014.3383.

[28] Wang GG, Guo L, Gandomi AH, Hao GS, Wang H. Chaotic Krill Herd algorithm. Inf Sci (Ny) 2014;274:17–34. doi:10.1016/j.ins.2014.02.123.

[29] Zamani H, Nadimi-Shahraki M-H. Feature selection based on whale optimization algorithm for diseases diagnosis. Int J Comput Sci Inf Secur 2016;14:1243.

[30] Banaie-Dezfouli M, Nadimi-Shahraki MH, Beheshti Z. R-GWO: Representative-based grey wolf optimizer for solving engineering problems. Appl Soft Comput 2021;106:107328.

[31] Zamani H, Nadimi-Shahraki MH, Gandomi AH. CCSA: conscious neighborhood-based crow search algorithm for solving global optimization problems. Appl Soft Comput 2019;85:105583.

[32] Shalchi Tousi M, Ghazavi M, Laali S. Optimizing Reinforced Concrete Cantilever Retaining Walls Using Gases Brownian Motion Algorithm (GBMOA). J Soft Comput Civ Eng 2021;5:1–18.

[33] Nenavath H, Jatoth RK. Hybrid SCA–TLBO: a novel optimization algorithm for global optimization and visual tracking. Neural Comput Appl 2019;31:5497–526. doi:10.1007/s00521-018-3376-6.

[34] Kumar P. A Modified Genetic Algorithm in C ++ for Optimization of Steel Truss Structures. J Soft Comput Civ Eng 2021;1:95–108.

[35] Shobeiri V, Ahmadi-Nedushan B. TOPOLOGY OPTIMIZATION OF PRETENSIONED CONCRETE BEAMS CONSIDERING MATERIAL NONLINEARITY. vol. 9. 2019.

[36] Ghasemi MR, Dizangian B. SIZE, SHAPE AND TOPOLOGY OPTIMIZATION OF COMPOSITE STEEL BOX GIRDERS USING PSO METHOD. ASIAN J Civ Eng (BUILDING Hous 2010;11:699–715.

[37] Ferdowsi A, Hoseini S, Farzin S, Faramarzpour M, Mousavi S. Shape optimization of gravity dams using a nature-inspired approach. J Soft Comput Civ Eng 2020;4:56–69. doi:10.22115/scce.2020.224492.1196.

[38] Ghasemi MR, Ghiasi R, Varaee H. Probability-based damage detection using kriging surrogates and enhanced ideal gas molecular movement algorithm. World Congr Struct Multidiscip Optim 2017;11:1657–74. doi:https://doi.org/10.1007/978-3-319-67988-4_124.

[39] Ghasemi MR, Ghiasi R, Varaee H. Probability-based damage detection of structures using model updating with enhanced ideal gas molecular movement algorithm. 12th World Congr Struct Multidiscip Optim 2017;11:1657–74.

[40] Ghasemi MR, Ghiasi R, Varaee H. Probability-Based Damage Detection of Structures Using Surrogate Model and Enhanced Ideal Gas Molecular Movement Algorithm. World Congr Struct Multidiscip Optim, Springer International Publishing; 2017, p. 1657–74. doi:https://doi.org/10.1007/978-3-319-67988-4_124.

[41] Fattahi F, Gholizadeh S. Seismic fragility assessment of optimally designed steel moment frames. Eng Struct 2019;179:37–51. doi:10.1016/j.engstruct.2018.10.075.

[42] Kaveh A, Talatahari S. Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. Comput Struct 2009;87:267–83. doi:10.1016/j.compstruc.2009.01.003.

[43] Kaveh A, Mahdavi VR. A hybrid CBO-PSO algorithm for optimal design of truss structures with dynamic constraints. Appl Soft Comput J 2015;34:260–73. doi:10.1016/j.asoc.2015.05.010.

[44] Rezaee Manesh M, Ghasemi SH, Rezaee Manesh M. Dual Target Optimization of Two-Dimensional Truss Using Cost Efficiency and Structural Reliability Sufficiency. J Soft Comput Civ Eng 2020;4:98–111. doi:10.22115/scce.2020.244833.1252.

[45] Heidari A, Raeisi J. Optimum design of structures against earthquake by simulated annealing using wavelet transform. J Soft Comput Civ Eng 2018;2:23–33.

[46] Kaveh A, Maniat M. Damage detection based on MCSS and PSO using modal data. Smart Struct Syst 2015;15:1253–70. doi:10.12989/sss.2015.15.5.1253.

[47] Law SS, Li J, Ding Y. Structural response reconstruction with transmissibility concept in frequency domain. Mech Syst Signal Process 2011;25:952–68. doi:10.1016/j.ymssp.2010.10.001.

[48] Luh GC, Lin CY, Lin YS. A binary particle swarm optimization for continuum structural topology optimization. Appl Soft Comput J, vol. 11, 2011, p. 2833–44. doi:10.1016/j.asoc.2010.11.013.

[49] Javidrad F, Nazari M, Javidrad HR. An Innovative Optimized Design for Laminated Composites in terms of a Proposed Bi-Objective Technique. J Soft Comput Civ Eng 2020;4:1–28.

[50] Chakri A, Rabia XY, Mohamed K. Reliability-based design optimization using the directional bat algorithm. Neural Comput Appl 2017. doi:10.1007/s00521-016-2797-3.

[51] Safaeian Hamzehkolaei N, Miri M, Rashki M. Reliability-based design optimization of rotating FGM cylindrical shells with temperature-dependent probabilistic frequency constraints. Aerosp Sci Technol 2017;68:223–39. doi:10.1016/j.ast.2017.05.004.

[52] Safaeian Hamzehkolaei N, Miri M, Rashki M. An enhanced simulation-based design method coupled with meta-heuristic search algorithm for accurate reliability-based design optimization. Eng Comput 2016;32:477–95. doi:10.1007/s00366-015-0427-9.

[53] Petrović M, Vuković N, Mitić M, Miljković Z. Integration of process planning and scheduling using chaotic particle swarm optimization algorithm. Expert Syst Appl 2016;64:569–88. doi:10.1016/j.eswa.2016.08.019.

[54] Paiva FAP, Silva CRM, Leite IVO, Marcone MHF, Costa JAF. Modified bat algorithm with cauchy mutation and elite opposition-based learning. 2017 IEEE Lat Am Conf Comput Intell LA-CCI 2017 - Proc, vol. 2017- Novem, IEEE; 2018, p. 1–6. doi:10.1109/LA-CCI.2017.8285715.

[55] Safaeian Hamzehkolaei N, Miri M, Rashki M. An improved binary bat flexible sampling algorithm for reliability-based design optimization of truss structures with discrete-continuous variables. Eng Comput 2018;35:641–71. doi:10.1108/EC-06-2016-0207.

[56] Mozafari M, Tafazzoli S, Jolai F. A new IPSO-SA approach for cardinality constrained portfolio optimization. Int J Ind Eng Comput 2011;2:249–62. doi:10.5267/j.ijiec.2011.01.004.

[57] Ahmadi M, Mojallali H. Chaotic invasive weed optimization algorithm with application to parameter estimation of chaotic systems. Chaos, Solitons and Fractals 2012;45:1108–20. doi:10.1016/j.chaos.2012.05.010.

[58] Kaveh A, Bakhshpoori T, Afshari E. Hybrid PSO and SSO algorithm for truss layout and size optimization considering dynamic constraints. Struct Eng Mech 2015;54:453–74. doi:10.12989/sem.2015.54.3.453.

[59] Kotinis M. Improving a multi-objective differential evolution optimizer using fuzzy adaptation and K-medoids clustering. Soft Comput 2014;18:757–71. doi:10.1007/s00500-013-1086-7.

[60] Gupta S, Deep K. Hybrid sine cosine artificial bee colony algorithm for global optimization and image segmentation. Neural Comput Appl 2020;32:9521–43. doi:10.1007/s00521-019-04465-6.

[61] Yildizdan G, Baykan ÖK. A new hybrid BA_ABC algorithm for global optimization problems. Mathematics 2020;8:1–36. doi:10.3390/math8101749.

[62] Yue S, Zhang H. A hybrid grasshopper optimization algorithm with bat algorithm for global optimization. Multimed Tools Appl 2021;80:3863–84. doi:10.1007/s11042-020-09876-5.

[63] Yue ZH, Zhang S, Xiao WD. A novel hybrid algorithm based on grey wolf optimizer and fireworks algorithm. Sensors (Switzerland) 2020;20:1–17. doi:10.3390/s20072147.

[64] Khoshahval F, Zolfaghari A, Minuchehr H, Abbasi MR. A new hybrid method for multi-objective fuel management optimization using parallel PSO-SA. Prog Nucl Energy 2014;76:112–21. doi:10.1016/j.pnucene.2014.05.014.

[65] Mesloub S, Mansour A. Hybrid PSO and GA for global maximization. Int J Open Probl Comput Sci Math 2009;2:597–608. doi:1998-6262.

[66] Jayaprakasam S, Rahim SKA, Leow CY. PSOGSA-Explore: A new hybrid metaheuristic approach for beampattern optimization in collaborative beamforming. Appl Soft Comput J 2015;30:229–37. doi:10.1016/j.asoc.2015.01.024.

[67] Jia P, Duan S, Yan J. An Enhanced Quantum-Behaved Particle Swarm Optimization Based on a Novel Computing Way of Local Attractor 2015:633–49. doi:10.3390/info6040633.

[68] Pluhacek M, Senkerik R, Davendra D. Chaos particle swarm optimization with Eensemble of chaotic systems. Swarm Evol Comput 2015;25:29–35. doi:10.1016/j.swevo.2015.10.008.

[69] Shahzad M, Zahid S, Farooq M. A Hybrid GA-PSO Fuzzy System for User Identification on Smart Phones Categories and Subject Descriptors n.d.

[70] Lee H, Chen S, Kang H-Y. A Study of Generalized Reduced Gradient Method with Different Search Directions. A Study Gen Reduc Gradient Method with Differ Search Dir 2004;1:25–38.

[71] Arora JS. More on Numerical Methods for Constrained Optimum Design. Introd to Optim Des, 2004, p. 379–412. doi:10.1016/b978-012064155-0/50011-2.

[72] Rosen JB. The gradient projection method for nonlinear programming. Part I. Linear constraints. J Soc Ind Appl Math 1960;8:181–217.

[73] Lasdon LS, Fox RL, Ratner MW. NONLINEAR OPTIMIZATION USING THE GENERALIZED REDUCED GRADIENT METHOD. Rev Fr Autom Inf Rech Oper 1974;8:73–103. doi:10.1051/ro/197408V300731.

[74] Gabriele GA, Ragsdell KM. The generalized reduced gradient method: A reliable tool for optimal design. J Eng Ind 1977:394–400.

[75] Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. Am Stat 1992;46:175–85. doi:10.1080/00031305.1992.10475879.

[76] Kulkarni O, Kulkarni N, Kulkarni AJ, Kakandikar G. Constrained cohort intelligence using static and dynamic penalty function approach for mechanical components design. Int J Parallel, Emergent Distrib Syst 2018;33:570–88. doi:10.1080/17445760.2016.1242728.

[77] Grandgirard J, Poinsot D, Krespi L, Nénon JP, Cortesero AM, Miettinen K, et al. Numerical comparison of some penalty-based constraint handling techniques in genetic algorithms. J Glob Optim 2003;27:427–46. doi:10.1023/A.

[78] Parsopoulos KE, Vrahatis MN, others. Particle swarm optimization method for constrained optimization problems. Intell Technol Appl New Trends Intell Technol 2002;76:214–20.

[79] Daniel IM. Self-adapting control parameters in particle swarm optimization. University of British Columbia, 2019.

[80] Wang H, Cui Z, Sun H, Rahnamayan S, Yang X, Wang H. Randomly attracted firefly algorithm with neighborhood search and dynamic parameter adjustment mechanism. Soft Comput 2016. doi:10.1007/s00500-016-2116-z.

[81] Fattahi H, Babanouri N. Predicting tensile strength of rocks from physical properties based on support vector regression optimized by cultural algorithm. J Min Environ 2017;8:467–74. doi:10.22044/jme.2016.824.

[82] Sharma TK, Pant M, Singh VP. Adaptive Bee Colony in an Artificial Bee Colony for Solving. arxiv, 2012. doi:1211.0957.

[83] Kaveh a., Mahdavi VRR. Colliding Bodies Optimization method for optimum design of truss structures with continuous variables. Adv Eng Softw 2014;70:1–12. doi:10.1016/j.advengsoft.2014.01.002.

[84] Keane AJ. Experiences with optimizers in structural design. Conf Adapt Comput Eng Des Control 1994;94:14–27.

[85] Mishra SK. Minimization of Keane's Bump Function by the Repulsive Particle Swarm and the Differential Evolution Methods. SSRN Electron J 2011. doi:10.2139/ssrn.983836.

[86] Ghasemi MR, Hinton E, Wood RD. Optimization of trusses using genetic algorithms for discrete and continuous variables. vol. 16. 1999. doi:10.1108/02644409910266403.

[87] Sapre S, Mini S. Opposition-based moth flame optimization with Cauchy mutation and evolutionary boundary constraint handling for global optimization. Soft Comput 2019;23:6023–41. doi:10.1007/s00500-018-3586-y.

[88] Mirjalili S, Mirjalili SM, Hatamlou A. Multi-Verse Optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 2016;27:495–513. doi:10.1007/s00521-015-1870-7.

[89] Askarzadeh A. A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. Comput Struct 2016;169:1–12. doi:10.1016/j.compstruc.2016.03.001.

[90] Kaveh A, Dadras A. A novel meta-heuristic optimization algorithm: Thermal exchange optimization. Adv Eng Softw 2017;110:69–84. doi:10.1016/j.advengsoft.2017.03.014.

[91] Zahara E, Kao Y-TT. Hybrid Nelder-Mead simplex search and particle swarm optimization for constrained engineering design problems. Expert Syst Appl 2009;36:3880–6. doi:10.1016/j.eswa.2008.02.039.

[92] Wang L, Li LP. An effective differential evolution with level comparison for constrained engineering design. Struct Multidiscip Optim 2010;41:947–63. doi:10.1007/s00158-009-0454-5.

[93] Huang F zhuo, Wang L, He Q. An effective co-evolutionary differential evolution for constrained optimization. Appl Math Comput 2007;186:340–56. doi:10.1016/j.amc.2006.07.105.

[94] Mezura-Montes E, Coello CACC, Velazquez-Reyes J, Munoz-Davila L. Multiple trial vectors in differential evolution for engineering design. Eng Optim 2007;39:567–89. doi:10.1080/03052150701364022.

[95] Rao SS. Engineering Optimization: Theory and Practice: Fourth Edition. John Wiley & Sons; 2009. doi:10.1002/9780470549124.

[96] Belegundu AD, Arora JS. A study of mathematical programmingmethods for structural optimization. Part II: Numerical results. Int J Numer Methods Eng 1985;21:1601–23. doi:10.1002/nme.1620210905.

[97] Arora J. Introduction to optimum design. Academic Press; 2004.

[98] Coello Coello CA, Montes EM. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. Adv Eng Informatics 2002;16:193–203. doi:10.1016/S1474-0346(02)00011-3.

[99] Zhou Y, Liu L. An effective chaotic cultural-based particle swarm optimization for constrained engineering design problems. Appl Mech Mater, vol. 20–23, Elsevier; 2010, p. 64–9. doi:10.4028/www.scientific.net/AMM.20-23.64.

[100] Eskandar H, Sadollah A, Bahreininejad A, Hamdi M. Water cycle algorithm - A novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 2012;110–111:151–66. doi:10.1016/j.compstruc.2012.07.010.

[101] Kaveh A, Talatahari S. A novel heuristic optimization method: charged system search. Acta Mech 2010;213:267–89. doi:10.1007/s00707-009-0270-4.

[102] Ben Guedria N. Improved accelerated PSO algorithm for mechanical engineering optimization problems. Appl Soft Comput J 2016;40:455–67. doi:10.1016/j.asoc.2015.10.048.

[103] Cagnina LC, Esquivel SC, Coello CAC. Solving engineering optimization problems with the simple constrained particle swarm optimizer. Bioinspired Optim Methods their Appl - Proc 3rd Int Conf Bioinspired Optim Methods their Appl BIOMA 2008, vol. 32, 2008, p. 107–20.

[104] Garg H. A hybrid GSA-GA algorithm for constrained optimization problems. Inf Sci (Ny) 2019;478:499–523. doi:10.1016/j.ins.2018.11.041.

[105] Hsu YL, Liu TC. Developing a fuzzy proportional-derivative controller optimization engine for engineering design optimization problems. Eng Optim 2007;39:679–700. doi:10.1080/03052150701252664.

[106] Raj KH, Sharma RS, Mishra GS, Dua A, Patvardhan C. An evolutionary computational technique for constrained optimisation in engineering design. J Inst Eng Mech Eng Div 2005;86:121–8.

[107] Tsai JFA. Global optimization of nonlinear fractional programming problems in engineering design. Eng Optim 2005;37:399–409. doi:10.1080/03052150500066737.

[108] Zhang M, Luo W, Wang X. Differential evolution with dynamic stochastic selection for constrained optimization. Inf Sci (Ny) 2008;178:3043–74. doi:10.1016/j.ins.2008.02.014.

[109] Gandomi AH, Yang XS, Alavi AH. Erratum: Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems (Engineering with Computers DOI:10.1007/s00366-011-0241-y). Eng Comput 2013;29:245. doi:10.1007/s00366-012-0308-4.