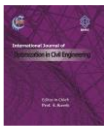
INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2026; 16(1):1-29



PROBABILISTIC GUIDE-SELECTION PARTICLE SWARM OPTIMIZATION

M. Ravan[1], H. Rahami[2*, †], and M. R. Shokoohfar[1]
[1]*School of Civil Engineering, College of Engineering, University of Tehran, Tehran, Iran*
[2]*School of Engineering Science, College of Engineering, University of Tehran, Tehran, Iran*

[DOI: 10.22068/ijoce.2026.16.1.660]

ABSTRACT

Optimization is a key tool for solving complex engineering problems. This research introduces a novel particle swarm optimization algorithm in which all particles have a probability of being selected as guide particles, while the likelihood of each particle influencing others is determined proportionally to its performance. In other words, unlike the classical PSO algorithm where only the best particle is chosen as the fixed guide in each iteration, every particle can independently select its own guide based on the performance of other particles. This approach appears to prevent premature convergence of particles and enhance the exploration capability of the algorithm. Additionally, a parameter has been defined and investigated in this algorithm to adjust the ratio of exploration to exploitation power, which can be initialized according to the complexity type of the problem. The performance of the proposed algorithm was first evaluated using a set of benchmark mathematical functions, which confirmed the high accuracy of the algorithm in finding optimal solutions. Then, several truss design problems were examined as real structural case studies, and the obtained results indicate that the proposed algorithm exhibits suitable and acceptable performance compared with other well-known algorithms.

Keywords: Optimization; Structural optimization; Computational efficiency; Particle swarm optimization; Metaheuristic Search Algorithms.

Received: 13 November 2025; Accepted: 3 January 2026

1. INTRODUCTION

Optimization represents a key tool for addressing complex engineering problems. In the

*Corresponding author: School of Engineering Science, College of Engineering, University of Tehran, Tehran, Iran
†E-mail address: hrahami@ut.ac.ir (H. Rahami)

current era, where limited resources and increasing environmental requirements have gained prominence, optimization techniques have become essential for structural engineers. In structural design, special attention must be given to numerous factors such as extensive constraints, a large number of variables, construction limitations, and design regulations. Consequently, structural optimization is considered one of the most challenging optimization problems.

Classical optimization methods in structural problems are computationally expensive due to the need for numerical gradient calculations and their inability to escape local optima. Therefore, gradient-free metaheuristic algorithms with superior global search capabilities have become widely used in structural optimization.

Holland [1] introduced the genetic algorithm in the 1970s by modeling Darwin's natural evolution process and genetic mechanisms such as selection, crossover, and mutation, which guides a population of random solutions toward the global optimum across generations.

Rajeev and Krishnamoorthy [2], along with Adeli and Kumar [3], were among the first to apply genetic algorithms to structural optimization; they considered discrete element sizes as design variables and minimized structural weights while satisfying design constraints. Additionally, Ghasemi and Yousefi [4] addressed reliability-based optimization of steel structures using genetic algorithms. Overall, the application of genetic algorithms for truss optimization has consistently attracted researchers' attention.

Kennedy and Eberhart [5] introduced the particle swarm optimization (PSO) algorithm by modeling the social behavior of birds and fish. This algorithm moves particles in the search space based on their individual and collective best experiences to converge to the global optimum. Due to its computational simplicity and fast convergence speed, PSO has been extensively applied in structural optimization [6].

Dorigo et al. [7] introduced the ant colony optimization (ACO) algorithm, inspired by ants' behavior in finding the shortest path to food. The ant colony optimization algorithm is a metaheuristic that simulates real ants' behavior in discovering the shortest path to food sources. When ants search for food, they deposit pheromones that attract other ants to follow their paths.

Structural problems have been repeatedly evaluated and compared using various other optimization algorithms [8-14]. Structural optimization is employed for diverse objectives, such as reducing structural weight, construction costs, execution time, analysis duration, and others.

As mentioned earlier, the particle swarm optimization algorithm is one of the most widely used algorithms for solving and optimizing structural problems. In the PSO algorithm, particles are initially randomly distributed throughout the search space, after which they begin moving to find the optimal solution.

The velocity of each particle at each iteration is determined according to its previous inertia (V_i^t), movement toward its personal best experience (xlb_i^t), and movement toward the global best experience (xg^t) as per Eq. (1). Finally, the new position of particles (x_i^{t+1}) is calculated as the algebraic sum of the current particle position (x_i^t) and its velocity according (V_i^{t+1}) to Eq. (2). This process continues until the algorithm termination conditions are met and the optimal solution is obtained.

[DOI: 10.22068/ijoce.2026.16.1.660]

$$V_i^{t+1} = wV_i^t + C_1 r_1 (xlb_i^t - x_i^t) + C_2 r_2 (xg^t - x_i^t) \quad (1)$$

$$x_i^{t+1} = V_i^{t+1} + x_i^t \quad (2)$$

The velocity vector magnitude is limited to the range $[-V_{max}. V_{max}]$ to reduce the likelihood of particles exiting the search space. The value of $V_{max}.$ is typically calculated as $k \cdot X_{max}$, where $0{,}01 \leq k \leq 1$.

r_1 and r_2 in Eq. (1) are random numbers between 0 and 1 that, along with the two acceleration coefficients C_1 and C_2, control the distance each particle travels per iteration. These values are typically assumed to be 2 [5], although in some problems, different values for C_1 and C_2 are considered to improve performance [15].

The value of w, known as the inertia weight, is defined to control the particle's velocity influence from the previous movement. Increasing this coefficient enhances the algorithm's global exploration capability in the search space, while decreasing it improves the local exploration capability. This coefficient typically varies linearly from 0.9 to 0.4 throughout the analysis [16,17]. In general, w is determined by the following equation:

$$w(t) = w_{final} + \frac{Iter_{max} - Iter}{Iter_{max}} \times (w_{initial} - w_{final}) \quad (3)$$

where $w_{initial}$ and w_{final} represent the initial and final inertia weights, respectively. Additionally, $Iter_{max}$ denotes the maximum number of iterations specified for the problem, and Iter represents the current iteration number.

Since the introduction of the PSO algorithm, due to its simplicity and high effectiveness, it has enjoyed considerable popularity, and researchers have proposed numerous enhanced versions up to the present day.

Eberhart and Shi [18] considered the inertia weight as a fixed constant throughout the entire optimization process. They also demonstrated that the inertia weight can be selected randomly at each iteration, in addition to functions dependent on t (where t represents the iteration number in the search) [19]. Van den Bergh and Engelbrecht [20] also proposed a relationship between the inertia weight and coefficients C_1 and C_2, expressed as follows:

$$w > \frac{1}{2}(c_1 + c_2) - 1 \quad (4)$$

Researchers have employed particle relocation strategies when particles become too close to each other to enhance diversity, prevent entrapment in local optima, and avoid premature convergence. Additionally, they have addressed this issue by incorporating multiple collision-avoiding mechanisms [21].

Poli et al. [22] demonstrated that using local best selection instead of global best selection improves results. In this approach, particles select their guide particle from the best experiences of neighboring particles, which reduces the likelihood of premature convergence and enhances the algorithm's exploration capability.

In this approach, various strategies such as niche technology and topology structures (including star, pyramid, ring, and von Neumann topologies) have been used to define the neighborhood region of particles [23,24].

Lovbjerg et al. [25] defined the particle swarm algorithm in a multi-population form and

[DOI: 10.22068/ijoce.2026.16.1.660]

increased population diversity by allowing breeding operations within subpopulations or between subpopulations.

Van den Bergh and Engelbrecht [26] introduced the cooperative particle swarm optimizer (CPSO-H), in which particles search each problem dimension separately in a one-dimensional space, and the search results in each dimension are finally combined.

The CLPSO algorithm maintains population diversity and reduces the risk of falling into local optima by employing a comprehensive learning strategy instead of merely following the best local and global positions, which improves its performance in complex, high-dimensional optimization problems [27,28].

Achieving a proper balance between exploration and exploitation has always been one of the key goals in the development and extension of optimization algorithms. In classical particle swarm–based algorithms, a single particle is usually used as the group leader, and all particles follow the best particle as a guide and leader, which causes particles to converge toward a single point, resulting in premature convergence and entrapment in local optima. In the algorithm developed in this study, each particle freely selects its own leader. It should be noted that in this approach, the probability of selecting better particles as leaders is higher.

The results show that the exploration capability of this algorithm is higher than that of classical particle swarm algorithms, and it provides better solutions. Different functions, due to their specific complexities, require different levels of exploration and exploitation; in some functions, due to a large number of variables or numerous local optima, a higher exploration capability is needed, while in others a higher exploitation capability is required. In all optimization algorithms, achieving an appropriate balance between exploration and exploitation remains one of the key challenges. In the algorithm proposed in this study, a parameter is defined to adjust the exploration and exploitation strengths, and it is shown to have a significant effect on tuning these two characteristics.

In this study, after describing the operating mechanism of the developed algorithm in Section 2, Section 3 addresses the tuning of the balancing parameter between the exploration and exploitation properties of the algorithm. Then, in Section 4, the performance of the algorithm is validated by solving several mathematical problems, and finally, in Sections 5 and 6, structural engineering problems are introduced and their results are discussed and analyzed, respectively.

[DOI: 10.22068/ijoce.2026.16.1.660]

2. DEVELOPED ALGORITHM

Particle swarm optimization with leader selection is a swarm-based algorithm developed by taking inspiration from the movement of birds in nature. In the standard PSO algorithm, when two or more particles reach relatively good solutions, the remaining particles only select the best particle as a guide and show no tendency toward other good solutions. In the proposed algorithm, unlike conventional PSO and similar algorithms, all particles simultaneously possess the potential for leadership, and each particle can freely choose a different guide particle. However, the probability of following better particles is higher, and as the search progresses, particles become greedier, increasing the likelihood that each particle follows the best particles.

In this approach, a parameter is introduced to adjust the probability of selecting the best

particle in the final iteration, which in turn controls the exploration and exploitation strengths of the algorithm. Fig. 1 illustrates the behavior of the developed algorithm.

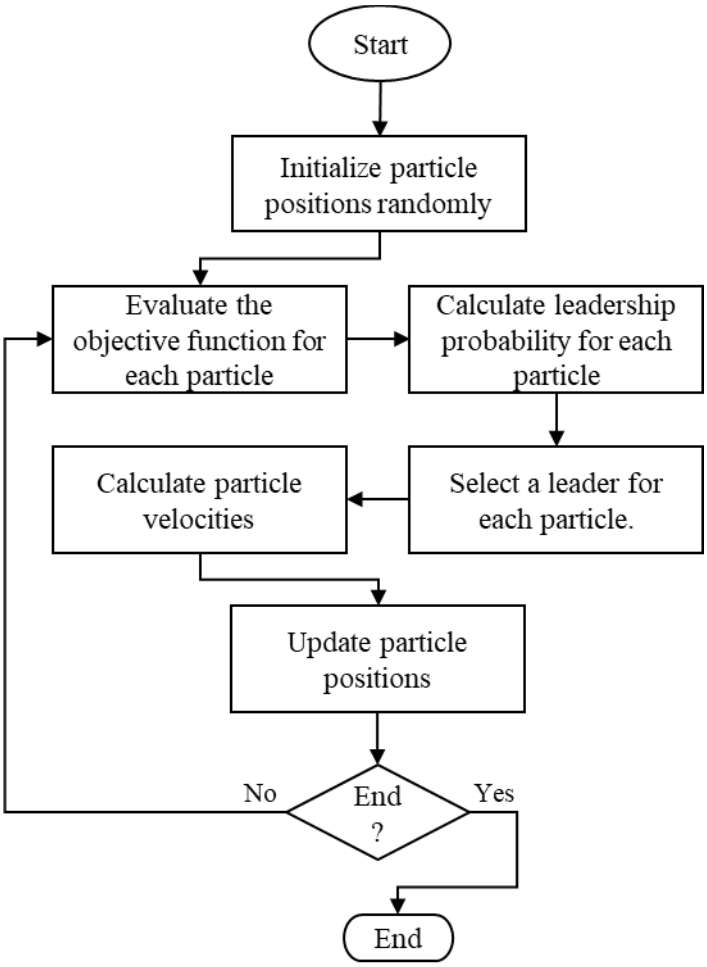


Figure 1: Flowchart of the developed algorithm.

The operation of the proposed algorithm is described in detail in the following.

2.1. Initialization

In most metaheuristic algorithms, the optimization process starts by defining the algorithmic parameters and randomly generating an initial position for each particle within the feasible search space. In an n-variable optimization problem, if each variable is considered as one dimension, the particles are therefore distributed in an n-dimensional space. The problem is then evaluated for all distributed particles, and the objective function value is computed for each particle.

2.2. Leader selection for each particle

In order to select a leader for each particle, the particles are first ranked according to the

[DOI: 10.22068/ijoce.2026.16.1.660]

value of their objective function, based on their personal best experience. The best particle is assigned rank 1, while the worst particle is assigned rank N_P (population size).
The selection probability ratio of each particle ($SPR(x_i)$) is computed according to Eq. (5)

$$0 \le SPR(x_i) = \left(1 - SP + SP \times \frac{N_P - RPop_i}{N_P - 1}\right)^{(\Omega I^{(It-1)})} \le 1 \quad (5)$$

In this context, $RPop_i$ represents the rank of the i^{th} particle, and SP denotes the selection pressure, which is set to 0.99 in this study. Moreover, ΩI is defined as a convergence coefficient used to establish a balance between the exploration and exploitation capabilities of the algorithm. As indicated by Eq. (5), an increase in $\Omega I^{(It-1)}$ enhances the likelihood of selecting the best particles as leader particles, whereas a decrease in its value makes the selection probabilities among particles more uniform, allowing the algorithm to exhibit a more stochastic search behavior. The value of ΩI, is computed according to Eq. (7) such that, at the final iteration of the optimization process, the probability ratio of selecting the second-best particle, n_2, to the best particle, n_1, is equals to α. An increase in the coefficient α results in more stochastic particle movements and enhances the exploration ability, whereas a decrease in α causes the particles to follow the best particle more closely, thereby increasing the exploitation capability of the algorithm.

$$n_2 = \left(1 - SP + SP \times \frac{N_P - 2}{N_P - 1}\right) . n_1 = 1 \quad (6)$$

$$\Omega I = \sqrt[(MaxIt-1)]{\log_{(\frac{n_2}{n_1})} \alpha} \quad . 0 \le \alpha \le 1 \quad (7)$$

Finally, each particle randomly selects another particle as its leader based on the calculated selection probability for each particle, as defined in Eq. (8).

$$F(x_i) = \frac{SPR(x_i)}{\sum SPR(x_i)} \times 100 \quad (8)$$

2.3. Particle motion

In particle swarm-based algorithms, particles collaboratively search for and explore the optimal solution. In this algorithm, after each particle selects its leader, it begins to move by following the behavioral pattern of that leader. The direction and velocity of each particle at any given moment or iteration are determined by the sum of three vectors, as expressed in Eq. (9) and Eq. (10). These three motion components include the inertia of the previous velocity V_i^t, the movement toward the particle's own best personal experience xlb_i^t, and the movement toward its selected leader xlb_{Pops}^t.

$$V_i^{t+1} = wV_i^t + C_1 r_1 (xlb_i^t - x_i^t) + C_2 r_2 (xlb_{Pops}^t - x_i^t) \quad (9)$$

$$X^{t+1} = V^{t+1} + X^t \quad (10)$$

[DOI: 10.22068/ijoce.2026.16.1.660]

3. BALANCING THE EXPLORATION AND EXPLOITATION CAPABILITIES OF THE ALGORITHM

In all optimization algorithms, one of the key challenges is achieving an appropriate balance between exploration and exploitation. In the algorithm developed in this study, the coefficient α (the ratio of the probability of selecting the best particle in the group to the probability of selecting the second-best particle in the group in the final optimization iteration) is defined as an effective parameter for adjusting the exploration and exploitation strengths in the algorithm, which has a significant impact on the particles' dispersion behavior and consequently the algorithm's performance in solving complex problems. As mentioned in the previous section, increasing the coefficient α leads to increased exploration capability and decreased exploitation capability in the algorithm, while decreasing the coefficient α results in increased exploitation capability and decreased exploration capability in the algorithm.

To calibrate the value of the coefficient α, in this section, several different objective functions are defined according to Table 1, and the algorithm's performance has been investigated with different values of the parameter α.

Table 1: Benchmark mathematical functions

NO.	Name	Formula	Range
1	Ackley	$f = -20 \times \exp\left(-0{,}2 \times \sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i{}^2}\right) - \exp\left(-0{,}2 \times \sqrt{\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi x_i)}\right) + 20 + \exp(1)$	[-15, 30]
2	Rosen Brock	$f = \sum_{i=1}^{D-1} (100(x_{i+1} - x_i{}^2)^2 + (x_i - 1)^2)$	[-5, 10]
3	Sphere	$f = \sum x_i{}^2$	[-5.12, 5.12]
4	Sum of Different Powers	$f = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	[-1, 1]
5	Zakharov	$f = \sum_{i=1}^{D} x_i{}^2 + \left(\sum_{i=1}^{D} 0.5 i x_i\right)^2 + \left(\sum_{i=1}^{D} 0.5 i x_i\right)^4$	[-5, 10]

3.1. Particle density analysis

Premature convergence occurs when the particles approach each other too early. As previously discussed, since in the developed algorithm all particles have the opportunity to be selected as guide particles, the probability of premature convergence is reduced. In this section, the distribution of particles in the search space is examined for different values of α to demonstrate how the particles are dispersed throughout the optimization process.

[DOI: 10.22068/ijoce.2026.16.1.660]

To examine the distribution of particles in the search space, the overall standard deviation of the particles across the problem dimensions is employed. This measure is more precisely defined, in accordance with Eq. (11), as the square root of the sum of squares of the particle variances in each dimension.

$$\sigma_{total} = \sqrt{\sum_{j=1}^{D} \left(\frac{1}{NP} \sum_{i=1}^{N} \left(x_{i,j} - \bar{x}_j\right)^2\right)} \tag{11}$$

In this equation, $x_{i,j}$ represents the position of the i^{th} particle in the j^{th} dimension, $\bar{x}_j$ denotes the mean position of the particles in the j^{th} dimension, NP is the particle population size, and D is the problem dimensionality. The value of σ_{total} indicates the overall dispersion of particles in the search space and serves as a metric for evaluating the algorithm's behavior.

Figs. 2–6 illustrate the particle dispersion and the process of obtaining optimal solutions throughout the optimization process for different values of the coefficient α in the various benchmark problems presented.

As evident from the graphs in Figs. 2–6, the rapid decrease in the standard deviation value or in other words, the early proximity of particles to each other leads to premature convergence in the algorithm, thereby increasing the likelihood that particles fail to locate the global optimum.

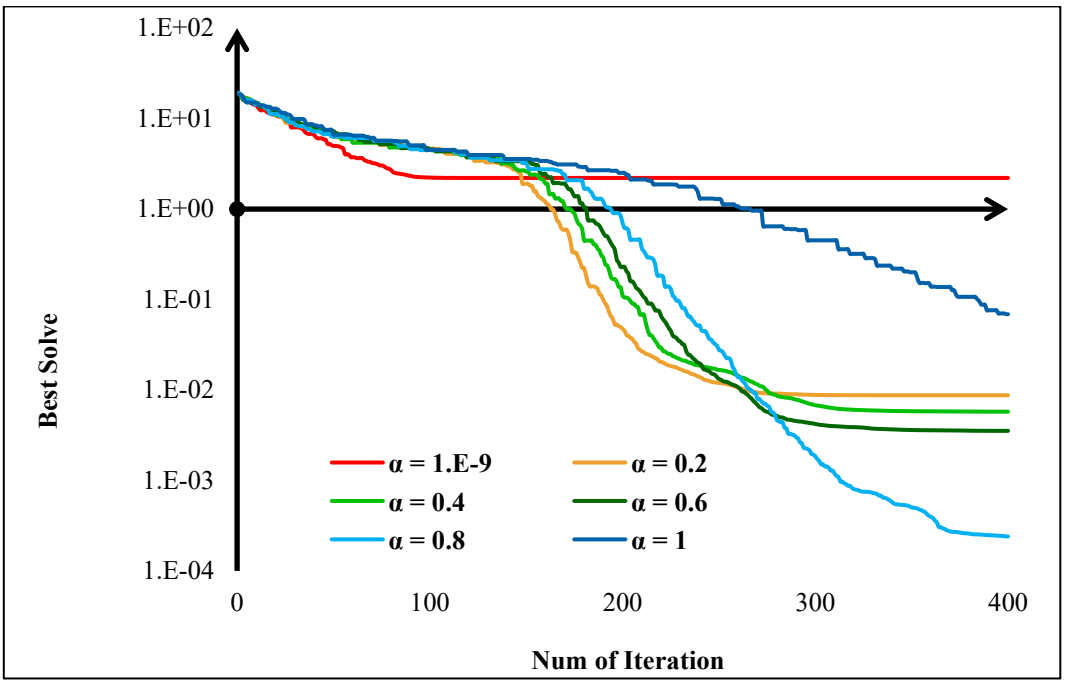
If the coefficient α is very close to zero, the algorithm exhibits the greatest similarity to the classical PSO algorithm, in which case, as shown in Figs. 2–6, the probability of premature convergence is high. Furthermore, these graphs indicate that when the coefficient α equals one, the algorithm's exploration capability is enhanced, and particles remain dispersed; however, the exploitation power decreases, potentially preventing the identification of the optimal solution. This also clearly demonstrates the direct proportionality of the coefficient α to the exploration capability of the algorithm and its inverse proportionality to the exploitation capability.

3.2 Sensitivity Analysis and Selection of the α Parameter

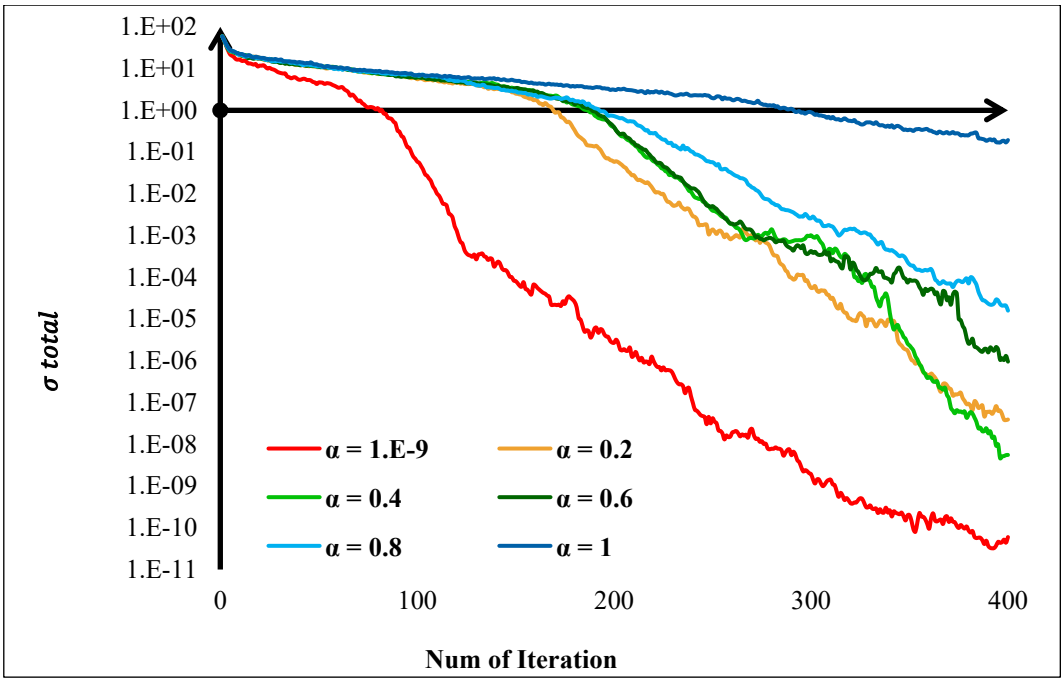
To apply the proposed algorithm in this study to various problems, it is necessary to investigate the sensitivity of the introduced parameter α with respect to the number of iterations and the number of particles in the algorithm. Fig. 7 illustrates the optimal value of α for three different problems with respect to the population size and the maximum number of iterations in the algorithm.

The optimization process was conducted for various maximum numbers of iterations MaxIt = [50, 60, …, 600] and for α values ranging from 0 to 1, identifying the best α for each MaxIt. This process was repeated 500 times, and the average of the best α values calculated for each MaxIt was determined. The same procedure was repeated for a fixed MaxIt and varying population sizes NumPop = [20, 30, …, 100], with the results presented in Fig. 7.

[DOI: 10.22068/ijoce.2026.16.1.660]



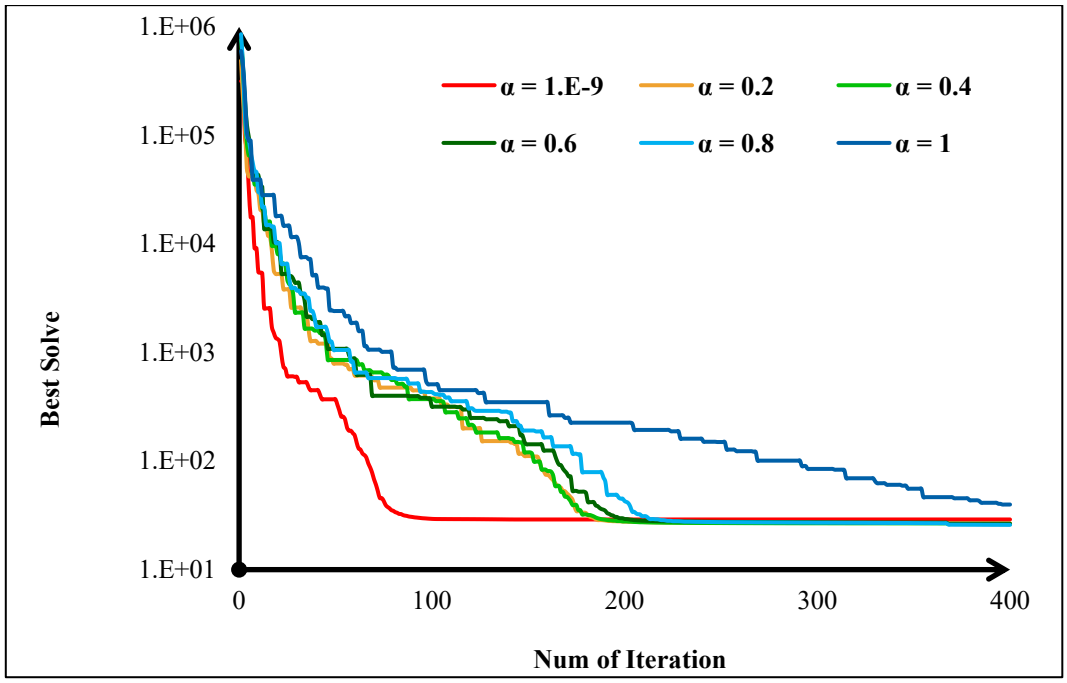
a) Convergence graphs of the best solution for different values of α.



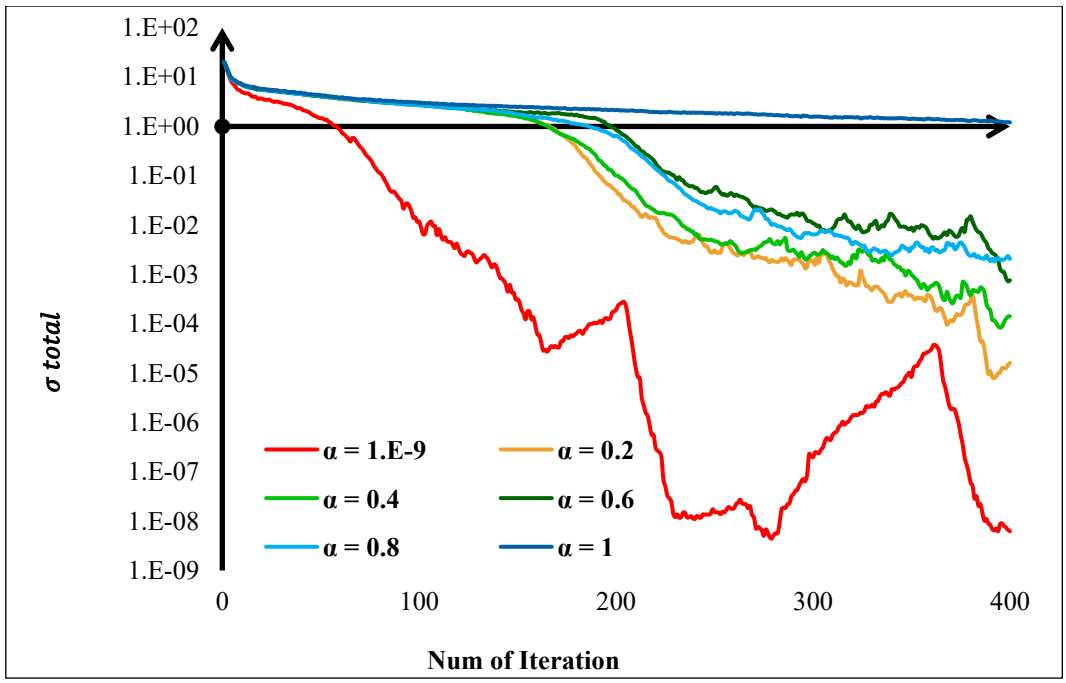
b) The process of particle dispersion variation throughout the optimization process for different values of α.

Figure 2: The optimization process of the Ackley function using 400 iterations, 40 particles, and 30 variables.

[DOI: 10.22068/ijoce.2026.16.1.660]



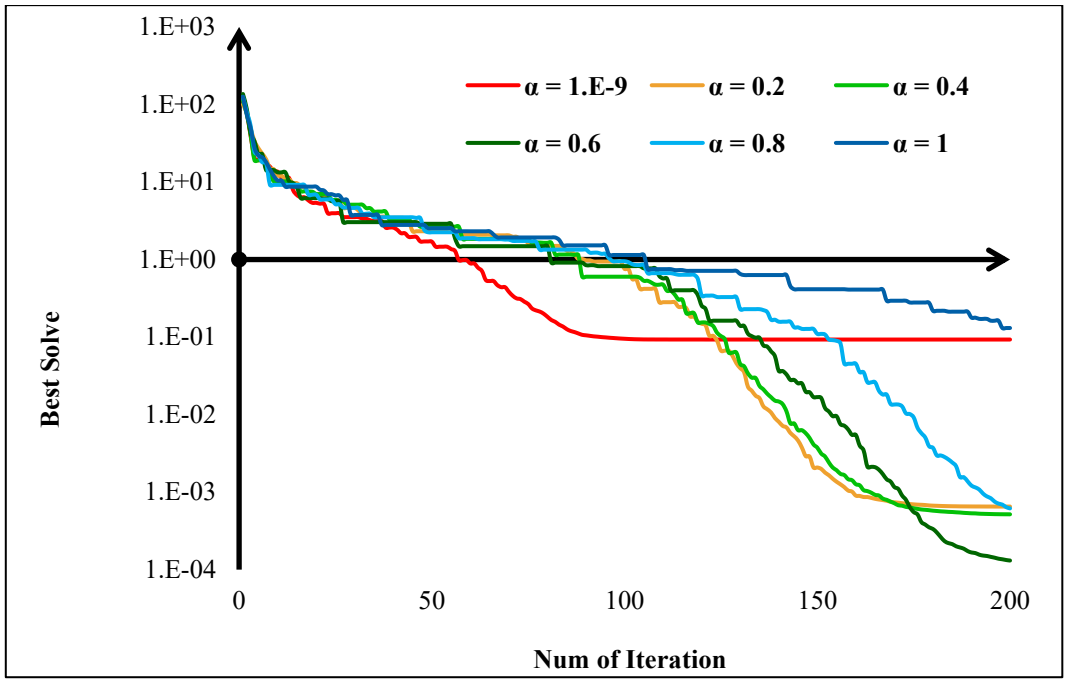
a) Convergence graphs of the best solution for different values of α.



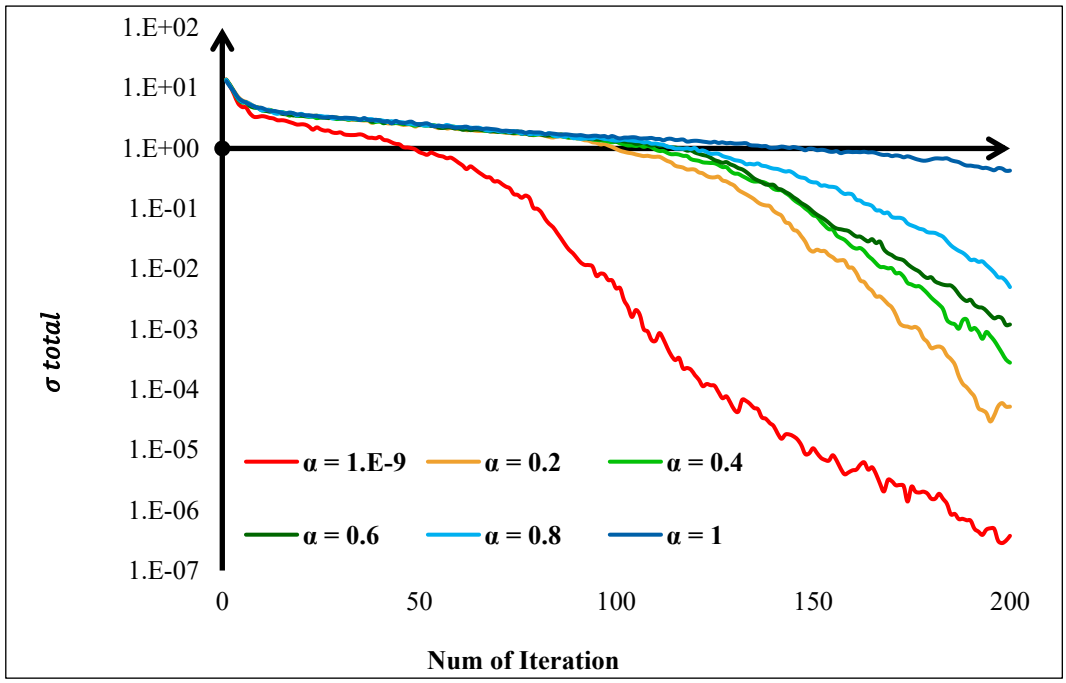
b) The process of particle dispersion variation throughout the optimization process for different values of α.

Figure 3: The optimization process of the Rosenbrock function using 400 iterations, 200 particles, and 30 variables.

[DOI: 10.22068/ijoce.2026.16.1.660]



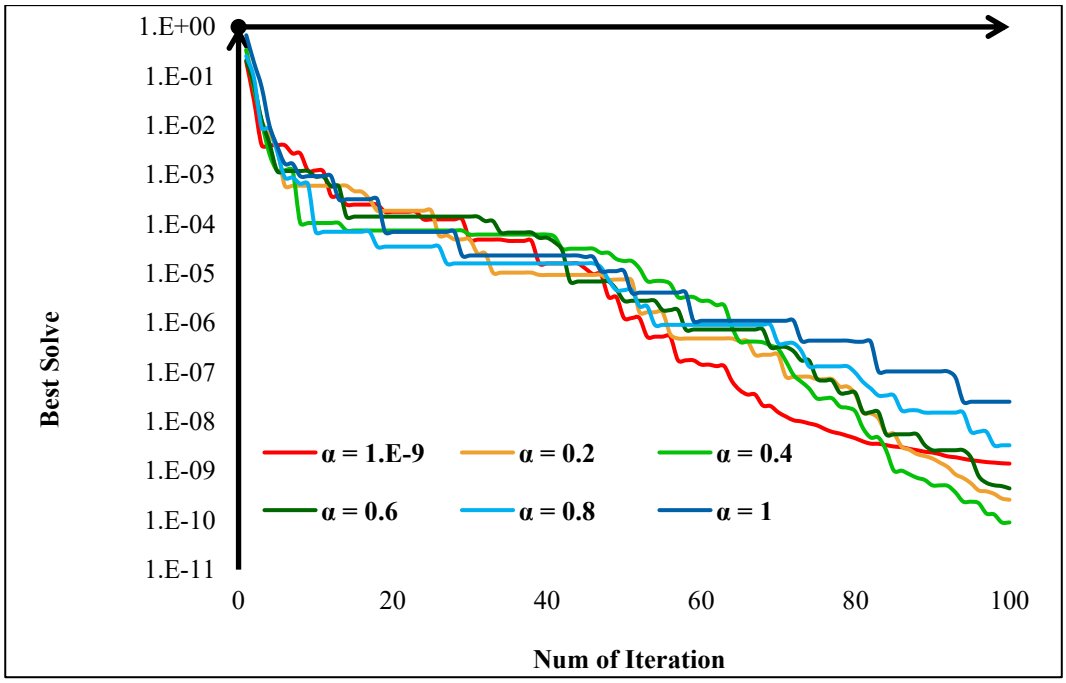
a) Convergence graphs of the best solution for different values of α.



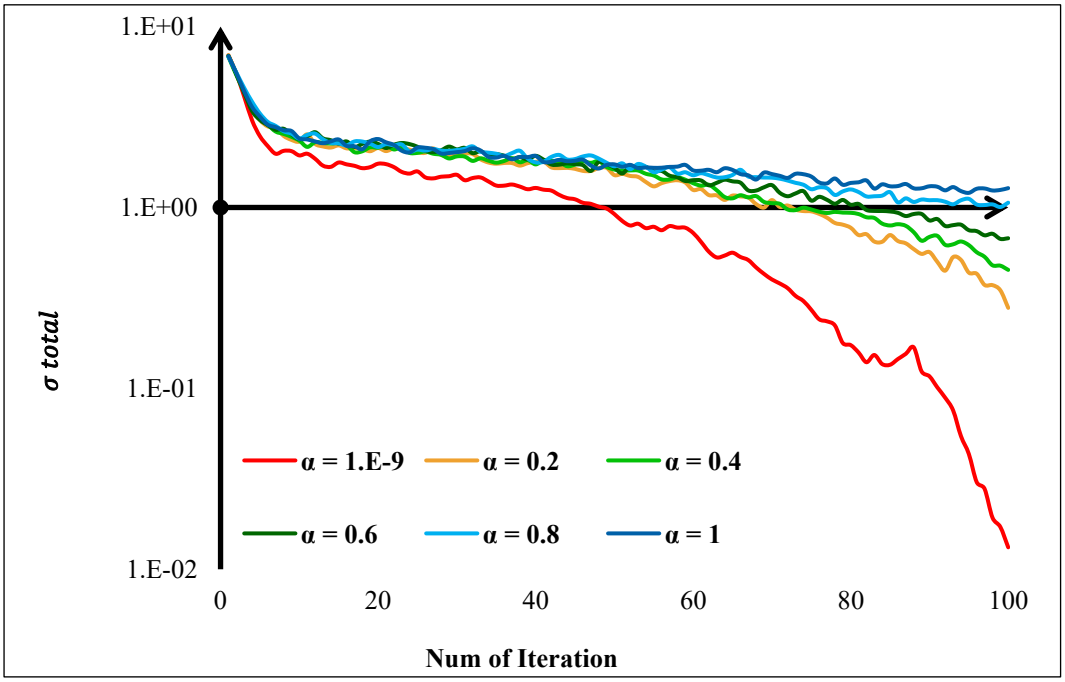
b) The process of particle dispersion variation throughout the optimization process for different values of α.

Figure 4: The optimization process of the Sphere function using 400 iterations, 40 particles, and 30 variables.

[DOI: 10.22068/ijoce.2026.16.1.660]



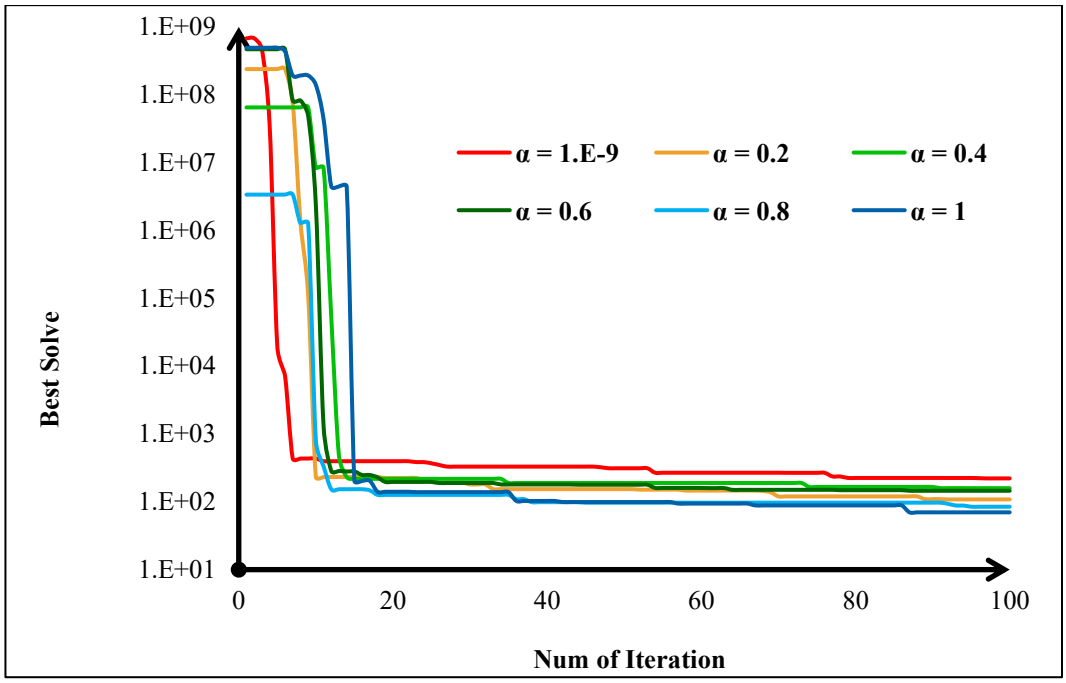
a) Convergence graphs of the best solution for different values of α.



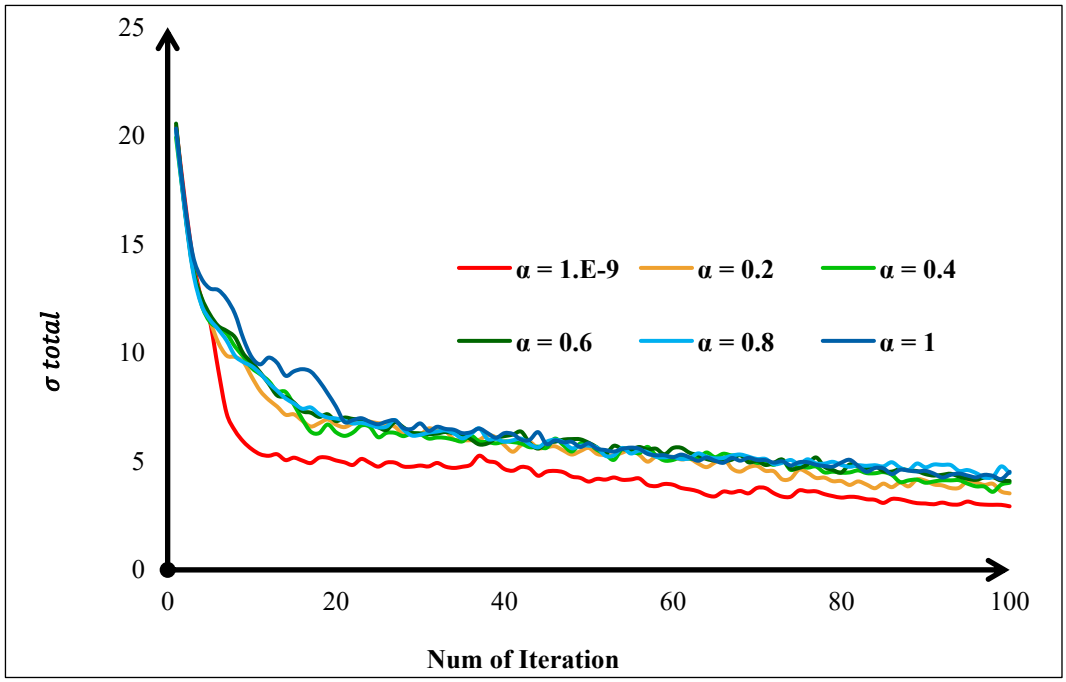
b) The process of particle dispersion variation throughout the optimization process for different values of α.

Figure 5: The optimization process of the SDP function using 100 iterations, 20 particles, and 200 variables.

[DOI: 10.22068/ijoce.2026.16.1.660]



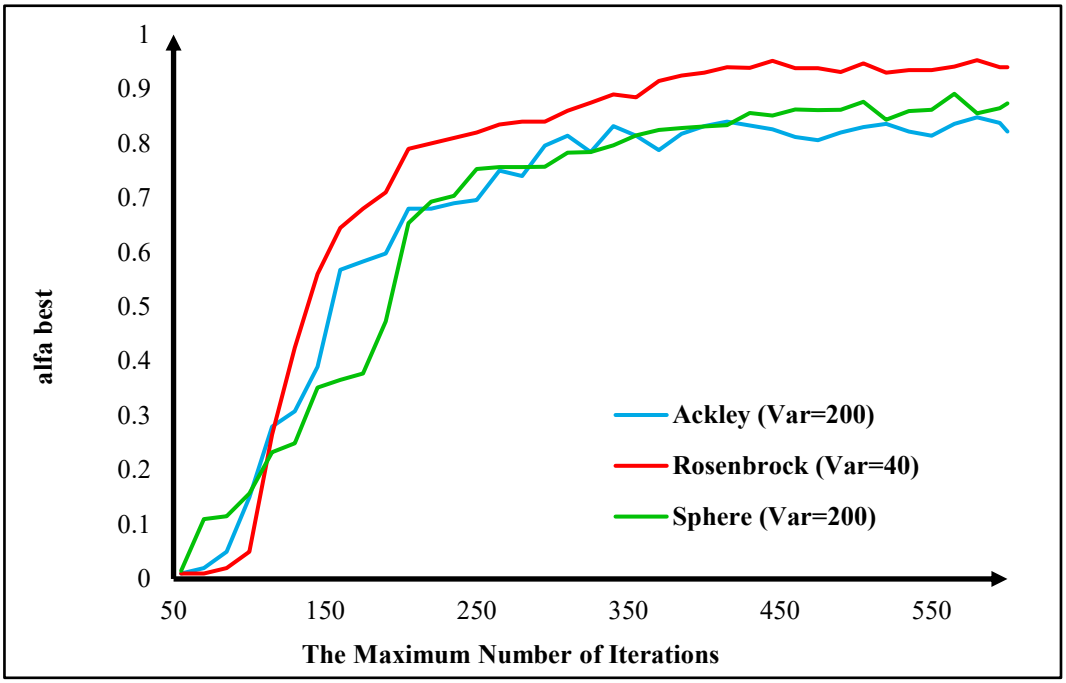
a) Convergence graphs of the best solution for different values of α.



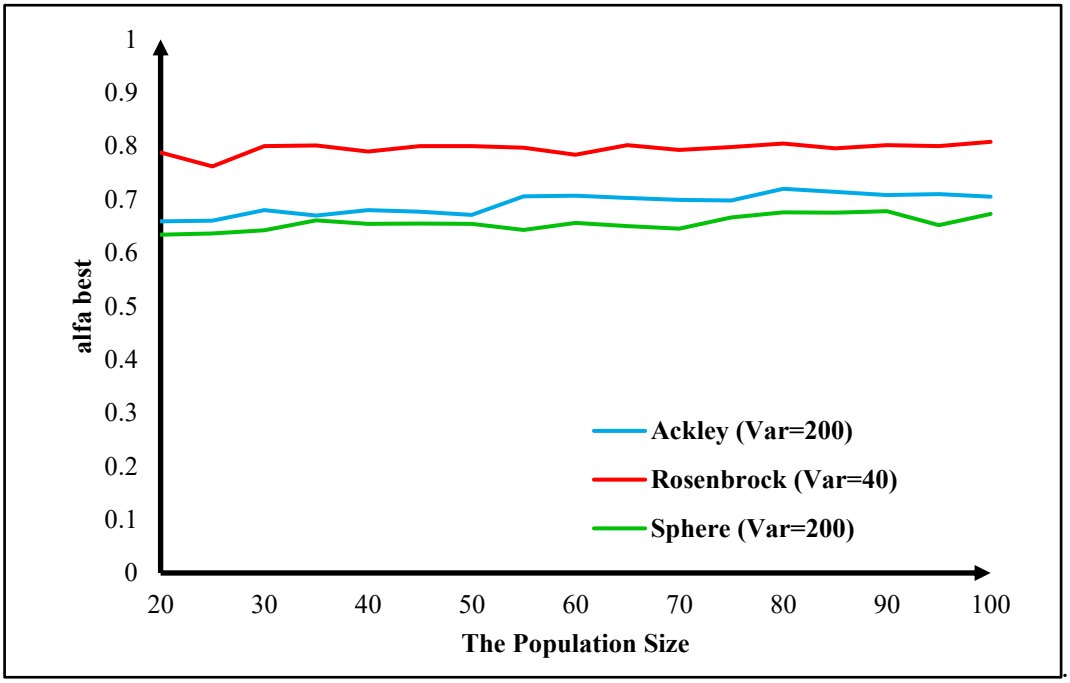
b) The process of particle dispersion variation throughout the optimization process for different values of α.

Figure 6: The optimization process of the Zakharov function using 100 iterations, 20 particles, and 200 variables.

[DOI: 10.22068/ijoce.2026.16.1.660]



a) The optimal value of α for different problems with respect to various numbers of iterations, using a fixed particle population size of 40.



b) The optimal value of α for different problems with respect to various population sizes, using a fixed maximum number of iterations of 200.

Figure 7: Calculation of the optimal α value for different problems with respect to the maximum number of iterations and the population size.

[DOI: 10.22068/ijoce.2026.16.1.660]

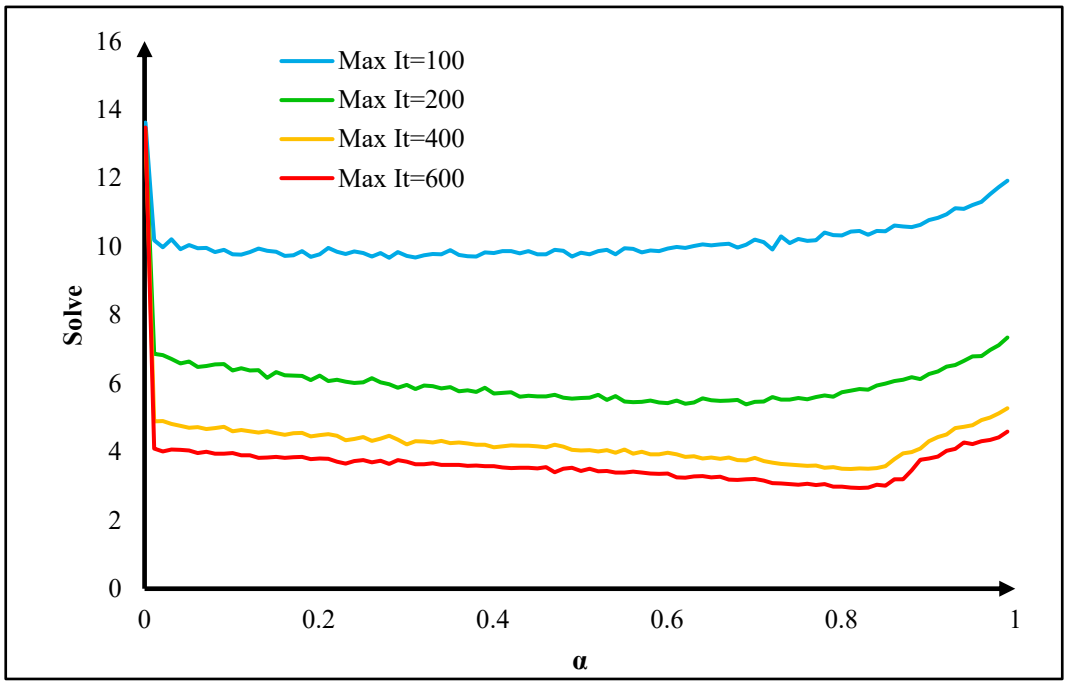
Figure 7 illustrates that the sensitivity of the α value to MaxIt is greater than its sensitivity to NumPop. As shown in this figure, at lower MaxIt values, the algorithm lacks the opportunity to locate the global optimum; hence, a smaller α value results in better solutions by facilitating rapid solution extraction. Conversely, at higher MaxIt values, while the algorithm has sufficient time for extraction, larger α values yield superior performance by preventing premature convergence and enhancing exploration capability.

Figure 8 illustrates the optimal solutions obtained for various problems with respect to α values ranging from 0 to 1, for different numbers of iterations.

Fig. 8 illustrates the relationship between the α value and the maximum number of iterations in the algorithm. This figure reveals that excessively high α values are detrimental, causing the algorithm to fail in extracting the optimal solution. Accordingly, based on Fig. 8 and to prevent such failure, α was set to 0.2 in this study.

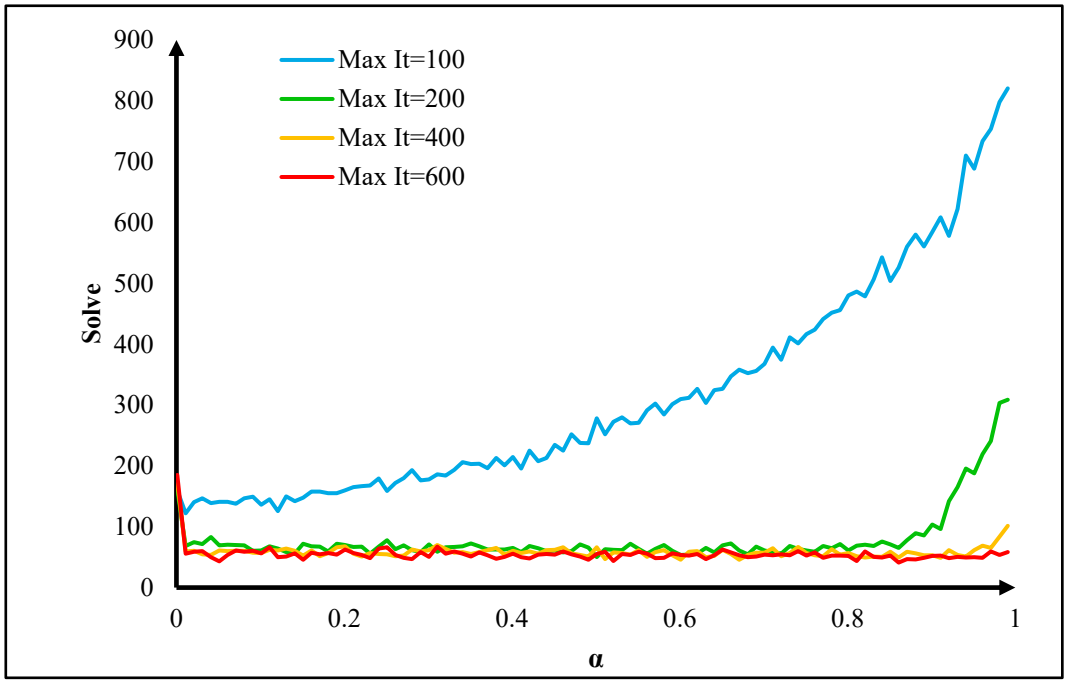
4. VALIDATION OF THE DEVELOPED ALGORITHM

To validate the performance of the developed algorithm, its results must be compared with those of other optimization algorithms. Accordingly, the problems introduced in Table 1 are compared with similar algorithms presented in Table 2.

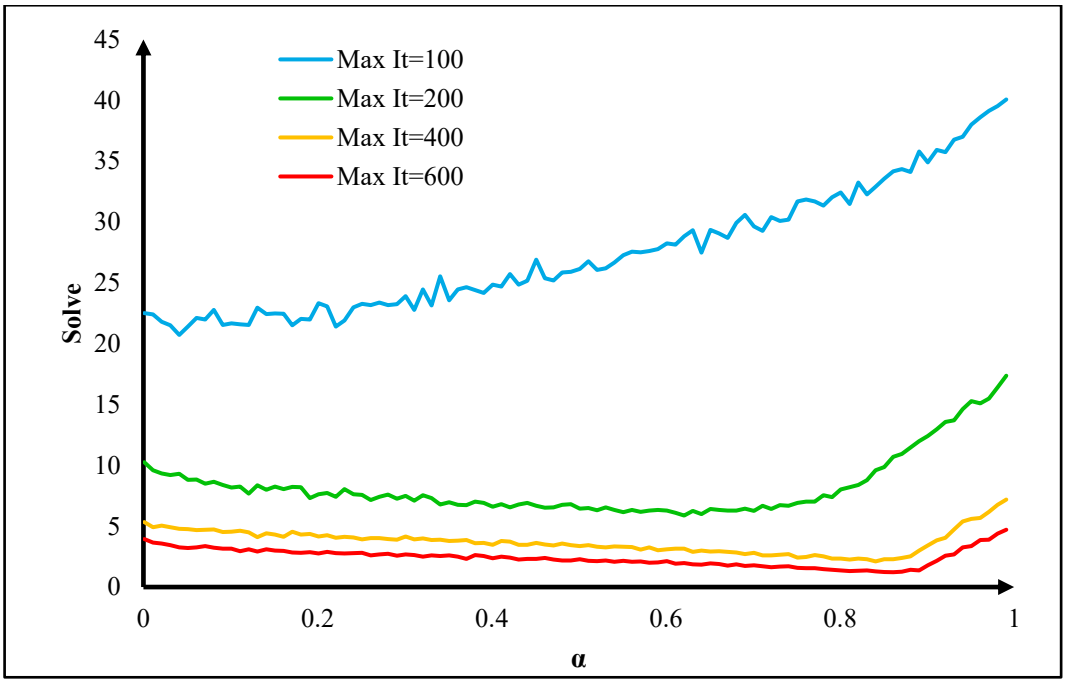


a) Optimal Solution of the 200-Dimensional Ackley Function with Respect to Various α Values Using Different Maximum Numbers of Iterations and 40 Particles.

[DOI: 10.22068/ijoce.2026.16.1.660]



b) Optimal Solution of the 40-Dimensional Rosenbrock Function with Respect to Various α Values Using Different Maximum Numbers of Iterations and 40 Particles.



c) Optimal Solution of the 200-Dimensional Sphere Function with Respect to Various α Values Using Different Maximum Numbers of Iterations and 40 Particles.

Figure 8: Sensitivity of optimal solutions for various problems with respect to α.

[DOI: 10.22068/ijoce.2026.16.1.660]

Table 2: Parameter Settings of Benchmark Optimization Algorithms

Methods	Parameter settings
SA [29]	Initial temperature = 1.0, Cooling factor = 0.95, Stopping temperature = 1e-10
CSO [30]	Discovery probability = 0.25
PSO [5]	Maximum velocity = 0.6, Inertia weight = 0.78, Acceleration constants = 1.2
DFO [31]	Separation weight = 0.1, Alignment weight = 0.1, Cohesion weight = 0.7, Food factor = 1, Enemy factor = 1, Initial inertia weight = 0.9
BSO [32,33]	Loudness = 0.5, Pulse rate = 0.5
LSFA, SFA [34]	Attractiveness = 1.0, Randomization parameter = 0.25, Absorption coefficient =1.0, SA – Initial temperature = 1.0, Cooling factor = 0.95, Stopping temperature = 1e-10
FA, ODFA, CFA [35-37]	Attractiveness = 1.0, Randomization parameter = 0.25, Absorption coefficient =1.0
RFA, SRFA, ESRFA [38]	Attractiveness = 1.0, Randomization parameter = 0.25, Absorption coefficient =1.0, Repulsive force factor = 0.6, Repulsive immune threshold = 0.1, Neighbour number, threshold = 0.5 (for SRFA and ESRFA)
ALO [39]	does not require any operation parameters
Present study	Maximum velocity = 0.1, Inertia weight = 0.99^(It-1), Acceleration constants = 1.5, alfa=0.2

For optimization of the problems using various algorithms, 20 particles, 100 iterations, and 200 design variables were employed. The results from 30 independent runs are reported in Table 3.

Table 3: Rrsult of Benchmark Function Optimization

Functions		Ackley	Rosenbrock	Sphere	Sum of Different Powers	Zakharov
ESRFA	mean	2.930E-04	1.980E+02	8.920E-07	6.300E-12	1.730E-04
	min	2.490E-04	1.980E+02	5.460E-07	5.920E-14	1.050E-04
	max	4.040E-04	1.990E+02	1.350E-06	2.250E-11	3.290E-04
	std	3.550E-05	7.460E-02	2.180E-07	5.350E-12	4.170E-05
SRFA	mean	5.230E-03	1.990E+02	1.740E-05	4.320E-12	7.720E-03
	min	4.140E-03	1.990E+02	9.750E-06	1.310E-13	5.060E-03
	max	6.050E-03	1.990E+02	2.290E-05	1.820E-11	1.020E-02
	std	4.730E-04	4.210E-02	2.760E-06	4.530E-12	1.360E-03
RFA	mean	1.490E-02	1.990E+02	1.460E-04	8.100E-12	5.910E-02
	min	1.290E-02	1.990E+02	1.010E-04	1.230E-13	4.420E-02
	max	1.680E-02	1.990E+02	1.930E-04	3.840E-11	7.080E-02
	std	9.620E-04	5.090E-02	2.180E-05	1.020E-11	7.070E-03
ODFA	mean	1.730E+01	1.790E+06	7.940E+02	6.890E-02	2.570E+03
	min	1.570E+01	4.560E+05	3.800E+02	2.850E-03	1.080E+03
	max	1.860E+01	3.250E+06	1.070E+03	1.920E-01	3.240E+03
	std	7.780E-01	8.250E+05	1.830E+02	5.110E-02	4.740E+02
SFA	mean	1.450E+01	1.250E+05	2.180E+01	1.790E-06	1.770E+03
	min	1.380E+01	4.810E+04	1.380E+01	3.120E-07	1.210E+03
	max	1.540E+01	2.210E+05	3.290E+01	5.810E-06	2.120E+03
	std	4.430E-01	4.560E+04	5.420E+00	1.450E-06	2.650E+02
LSFA	mean	1.590E+01	9.280E+04	3.150E+00	6.990E-07	1.460E+03
	min	1.540E+01	5.430E+04	9.420E-01	6.500E-09	1.300E+03

[DOI: 10.22068/ijoce.2026.16.1.660]

	max	1.620E+01	1.550E+05	5.400E+00	3.200E-06	1.640E+03
	std	1.870E-01	2.520E+04	1.240E+00	8.320E-07	8.750E+01
CFA	mean	1.990E+01	1.460E+07	1.300E+03	1.010E+01	5.140E+03
	min	1.960E+01	1.200E+07	1.240E+03	7.720E+00	4.920E+03
	max	2.000E+01	1.630E+07	1.380E+03	1.100E+01	5.420E+03
	std	1.030E-01	9.440E+05	3.710E+01	1.150E+00	1.200E+02
FA	mean	1.560E+01	6.150E+05	1.320E+02	1.830E-05	2.490E+03
	min	1.450E+01	3.610E+05	9.750E+01	1.390E-07	2.270E+03
	max	1.660E+01	1.080E+06	1.690E+02	1.110E-04	2.790E+03
	std	4.830E-01	1.530E+05	2.000E+01	2.150E-05	1.230E+02
SA	mean	7.590E+03	8.510E+02	9.110E+02	8.120E+02	8.690E+02
	min	6.410E+03	5.130E+02	5.210E+02	5.140E+02	5.280E+02
	max	1.120E+04	1.300E+03	1.890E+03	1.680E+03	1.610E+03
	std	9.580E+02	2.340E+02	3.600E+02	2.610E+02	2.840E+02
BSO	mean	1.880E+01	5.430E+06	5.140E+02	2.020E-02	3.940E+03
	min	1.800E+01	2.510E+06	3.100E+02	1.970E-05	3.440E+03
	max	1.960E+01	1.090E+07	8.680E+02	2.900E-01	4.690E+03
	std	4.120E-01	1.900E+06	1.470E+02	6.180E-02	3.320E+02
CSO	mean	1.790E+01	1.660E+06	2.090E+02	1.820E-03	3.400E+03
	min	1.760E+01	1.090E+06	1.510E+02	1.600E-05	3.150E+03
	max	1.840E+01	2.370E+06	3.030E+02	1.120E-02	3.690E+03
	std	2.000E-01	3.060E+05	3.200E+01	2.340E-03	1.410E+02
PSO	mean	1.760E+01	1.080E+06	2.020E+02	1.030E-04	2.840E+03
	min	1.680E+01	6.300E+05	1.640E+02	9.600E-06	2.600E+03
	max	1.830E+01	1.340E+06	2.620E+02	6.060E-04	3.140E+03
	std	3.140E-01	1.880E+05	2.190E+01	1.070E-04	1.450E+02
DFO	mean	1.750E+01	2.300E+06	2.370E+02	7.520E-03	3.320E+03
	min	1.270E+01	7.200E+05	1.270E+02	5.260E-05	2.410E+03
	max	1.910E+01	4.780E+06	4.060E+02	7.820E-02	4.080E+03
	std	1.380E+00	1.150E+06	7.910E+01	1.760E-02	4.410E+02
ALO	mean	1.900E+01	1.780E+07	1.570E+03	2.010E+00	5.000E+03
	min	1.900E+01	1.630E+07	1.460E+03	6.510E-01	5.000E+03
	max	1.900E+01	1.790E+07	1.640E+03	3.090E+00	5.000E+03
	std	1.450E-14	3.600E+05	4.570E+01	4.760E-01	0.000E+00
Present study	mean	8.440E+00	6.442E+04	1.894E+01	4.080E-10	1.924E+03
	min	7.912E+00	4.445E+04	1.894E+01	6.365E-12	1.397E+03
	max	8.884E+00	8.712E+04	2.889E+01	4.697E-09	2.314E+03
	std	2.925E-01	1.038E+04	2.360E+00	9.000E-10	2.158E+02

Table. 3 demonstrates that the optimized solutions obtained by this method outperform many comparable algorithms and exhibit superior performance. Furthermore, the low standard deviation of the responses in the developed algorithm indicates a reduced likelihood of particles becoming trapped in local optima.

5. MODELING AND OPTIMIZATION OF STRUCTURES

One of the applications of optimization algorithms is the determination of truss elements to minimize structural weight while satisfying all design requirements and constraints. This section introduces and optimizes three different trusses. It should be noted that optimizing real structures can effectively validate the performance of the developed algorithm.

[DOI: 10.22068/ijoce.2026.16.1.660]

To minimize truss weight, the cross-sectional area of each element is considered as the design variable, and the objective function is computed according to Eq. (12).

$$\text{ObjFunc} = w_T \times (1 + 10^4 \times P) \tag{12}$$

In this equation, w_T represents the total structural weight and P denotes the penalty function.

5.1. 72-Bar Tower Truss

The objective of this problem is to minimize the weight of a 72-bar tower truss shown in Fig. 9, with its members grouped into 16 sets for optimization as depicted in Fig. 9. In this structure, member stresses must not exceed ± 25 ksi and nodal displacements must remain within $0{,}25$ in. The design variables consist of the cross-sectional areas of these 16 groups, with the minimum cross-sectional area set to $0{,}1\ \text{in}^2$, modulus of elasticity to 10^7psi, and material density to $0{,}1\ \frac{\text{ib}}{\text{in}^3}$.

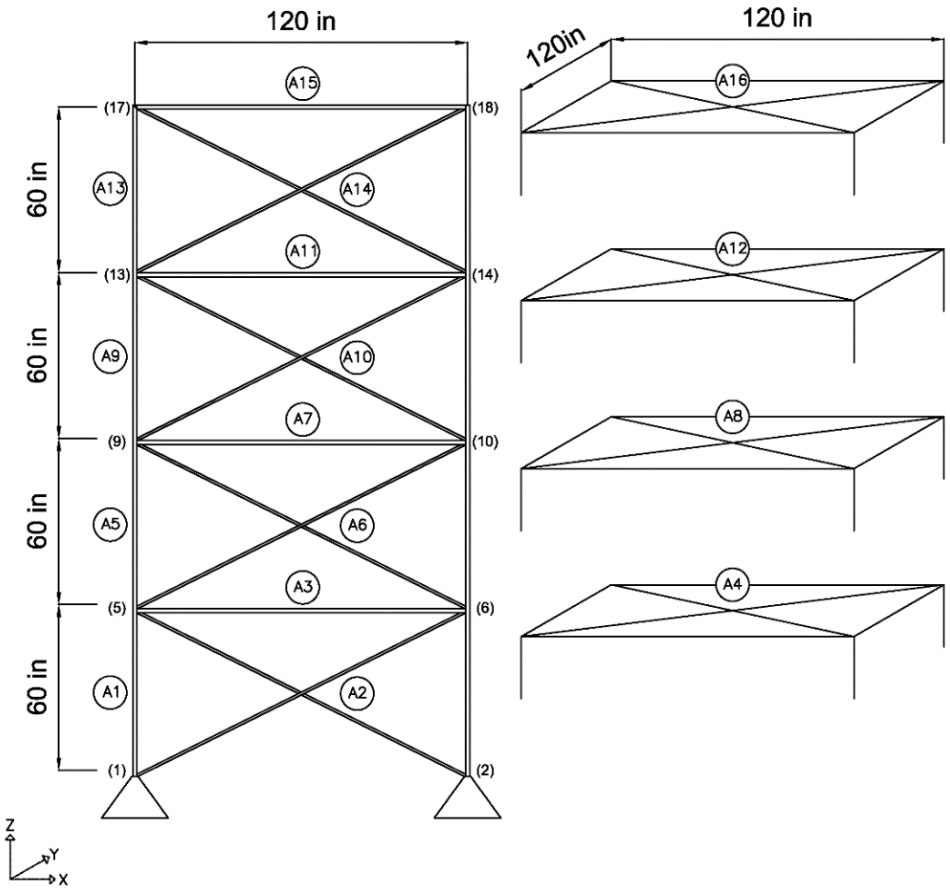


Figure 9: 72-bar space truss

[DOI: 10.22068/ijoce.2026.16.1.660]

Table 4: Loading conditions for the 72-bar space truss

	Node	Force (kips)		
		Px	Py	Pz
Case 1	17	5	5	-5
Case 2	17	0	0	-5
	18	0	0	-5
	19	0	0	-5
	20	0	0	-5

5.2. 200-Bar Tower Truss

The objective of this problem is to minimize the weight of a 200-bar tower truss shown in Fig. 10, with members grouped into 29 sets for optimization as per Table 6. Loading conditions are specified in Table 5. Member stresses must not exceed ± 10 ksi, and design variables comprise the cross-sectional areas of these 29 groups. The minimum cross-sectional area is $0.1\ \text{in}^2$, modulus of elasticity is 10^7 psi, and material density is $0.283\ \frac{\text{ib}}{\text{in}^3}$.

Table 5: Load cases for the planar 200-bar truss

Case	Force (kips)	Direction	Node
1	1000	X	1, 6, 15, 20, 29, 34, 43, 48, 57, 62, 71
2	10000	Y	1-6, 8, 10, 12, 14-20, 22, 24, 26, 28-34, 36, 38, 40, 42-48, 50, 52, 54, 56-62, 64, 66, 68, 70-75
3	Load cases 1 and 2 are acting simultaneously		

Table 6: Member Grouping of the 200-Bar Truss

Group	Members	Group	Members	Group	Members
A1	1, 2, 3, 4	A11	44, 45, 47, 48, 50, 51, 53, 54, 65, 66, 68, 69, 71, 72, 74, 75	A21	120, 121, 123, 124, 126
A2	5, 8, 11, 14, 17	A12	77, 78, 79, 80	A22	153, 154, 155, 156
A3	19, 20, 21, 22, 23, 24	A13	81, 84, 87, 90, 93	A23	157, 160, 163, 166, 169
A4	18, 25, 56, 63, 94, 101, 132, 139, 170, 177	A14	95, 96, 97, 98, 99, 100	A24	171, 172, 173, 174, 175
A5	26, 29, 32, 35, 38	A15	102,105,108,111,114	A25	178, 181, 184, 187, 190
A6	6, 7, 9, 10, 12, 13, 15, 16, 27, 28, 30, 31, 33, 34, 36, 37	A16	82, 83, 85, 86, 88, 89, 91, 92, 103, 104, 106, 107, 109, 110, 112, 113	A26	158, 159, 161, 162, 164
A7	39, 40, 41, 42	A17	115,116,117,118	A27	191, 192, 193, 194
A8	43, 46, 49, 52, 55	A18	119,122,125,128,131	A28	195, 197, 198, 200
A9	57, 58, 59, 60, 61, 62	A19	133, 134, 135, 136, 137, 138	A29	196, 199
A10	64, 67, 70, 73, 76	A20	140, 143, 146, 149, 152		

[DOI: 10.22068/ijoce.2026.16.1.660]

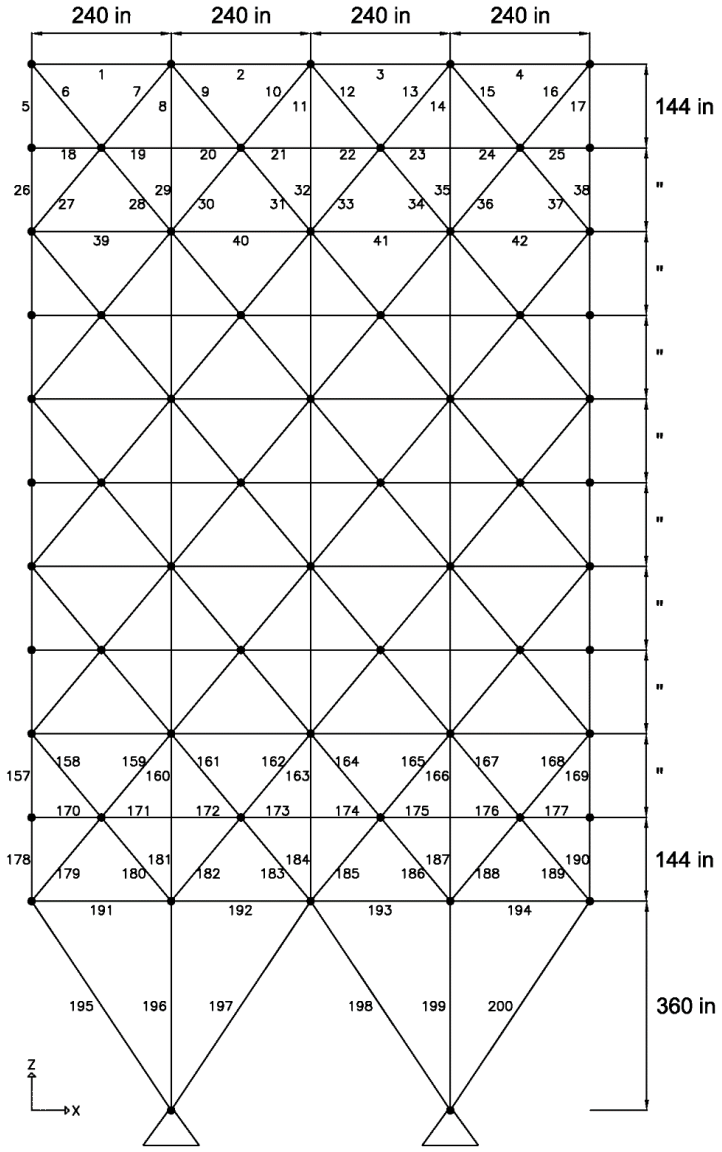


Figure 10: Schematic of the planar 200-bar truss structure.

5.3. 26-Story Truss

The objective of this problem is to optimize the steel weight used in the 26-story truss tower shown in Fig. 11. This tower consists of 942 elements and 244 nodes, which, due to geometric symmetry, can be grouped into 59 sets as depicted in Fig. 11.

The density and modulus of elasticity of the truss material are $0{,}1\ \mathrm{lb/in^3}$ and $10^7\ \mathrm{psi}$, respectively. The stress constraint for all truss members is ± 25 ksi, and the four top nodes are not permitted to displace more than $0{,}25$ in (approximately $1/250$ of the total tower height) in any direction. Additionally, the minimum and maximum cross-sectional areas of members are $0{,}1\ \mathrm{in^2}$ and $15{,}0\ \mathrm{in^2}$, respectively, with tower loading specified in Table 7.

[DOI: 10.22068/ijoce.2026.16.1.660]

Table 7: Load cases for the 942-bar space truss

Case number	Load (Ib)	Direction	Nodes
1	-3000	Z	each node in the first section
2	-6000	Z	at each node in the second section
3	-9000	Z	at each node in the third section
4	1000	X	at each node on the right side
5	1500	X	at each node on the left side
6	1000	Y	All nodes of the tower

6. RESULTS AND DISCUSSION

This section reports the results of the optimization problems addressed in Section 5 and compares them with outcomes from other optimization algorithms.

6.1. Results of the 72-Member Truss

For the optimization of the 72-bar truss defined in Section 5.1, 320 particles and 320 iterations were employed. Table 8 presents the computed results using the developed algorithm in this study compared with other metaheuristic algorithms.

Table 8: Comparison of Results for the 72-Bar Truss

Variables	Optimal cross-sectional areas (in^2)								
	CSP [40]	SAHS [41]	HBB–BC [42]	TLBO [43]	FPA [44]	RO [14]	ECM [45]	MNSV [46]	Present study
A1	0.15618	0.168	0.1566	0.1565	0.1575	0.157583	0.1564	0.1565	0.156683
A2	0.54022	0.584	0.5421	0.5429	0.5329	0.52222	0.544	0.5458	0.542321
A3	0.42229	0.433	0.4132	0.4081	0.4089	0.435582	0.4106	0.4104	0.413197
A4	0.57941	0.52	0.5756	0.5733	0.5731	0.597158	0.5624	0.5699	0.567273
A5	0.50674	0.485	0.5178	0.5317	0.4971	0.572989	0.5298	0.5233	0.52495
A6	0.51651	0.501	0.5214	0.5134	0.5089	0.549872	0.5172	0.5172	0.517529
A7	0.10752	0.1	0.1	0.1	0.1	0.100445	0.1	0.1	0.10025
A8	0.1	0.1	0.1007	0.1	0.1	0.100102	0.1	0.1	0.100526
A9	1.26757	1.271	1.2582	1.2711	1.2993	1.252233	1.2467	1.268	1.283595
A10	0.5099	0.509	0.5035	0.5151	0.5246	0.503347	0.5128	0.5115	0.508971
A11	0.1	0.1	0.1	0.1	0.1001	0.100176	0.1	0.1	0.1
A12	0.1	0.1	0.1	0.1	0.1	0.100151	0.1	0.1	0.100178
A13	1.94459	1.86	1.9042	1.8807	1.8758	1.83649	1.9298	1.8858	1.8997
A14	0.5026	0.521	0.5162	0.5142	0.516	0.502096	0.509	0.5124	0.510842
A15	0.1	0.1	0.1	0.1	0.1	0.100007	0.1	0.1	0.1
A16	0.1	0.1	0.1	0.1	0.1	0.10039	0.1	0.1	0.100154
Best weight (lb)	379.97	380.62	379.66	379.632	379.095	380.458	379.65	379.61	379.682
Average weight (lb)	381.56	382.42	381.85	379.759	379.534	382.5538	380.29	379.62	379.7095
Std Dev (lb)	1.803	1.38	1.201	0.149	0.272	1.2211	0.52	0	0.022

[DOI: 10.22068/ijoce.2026.16.1.660]

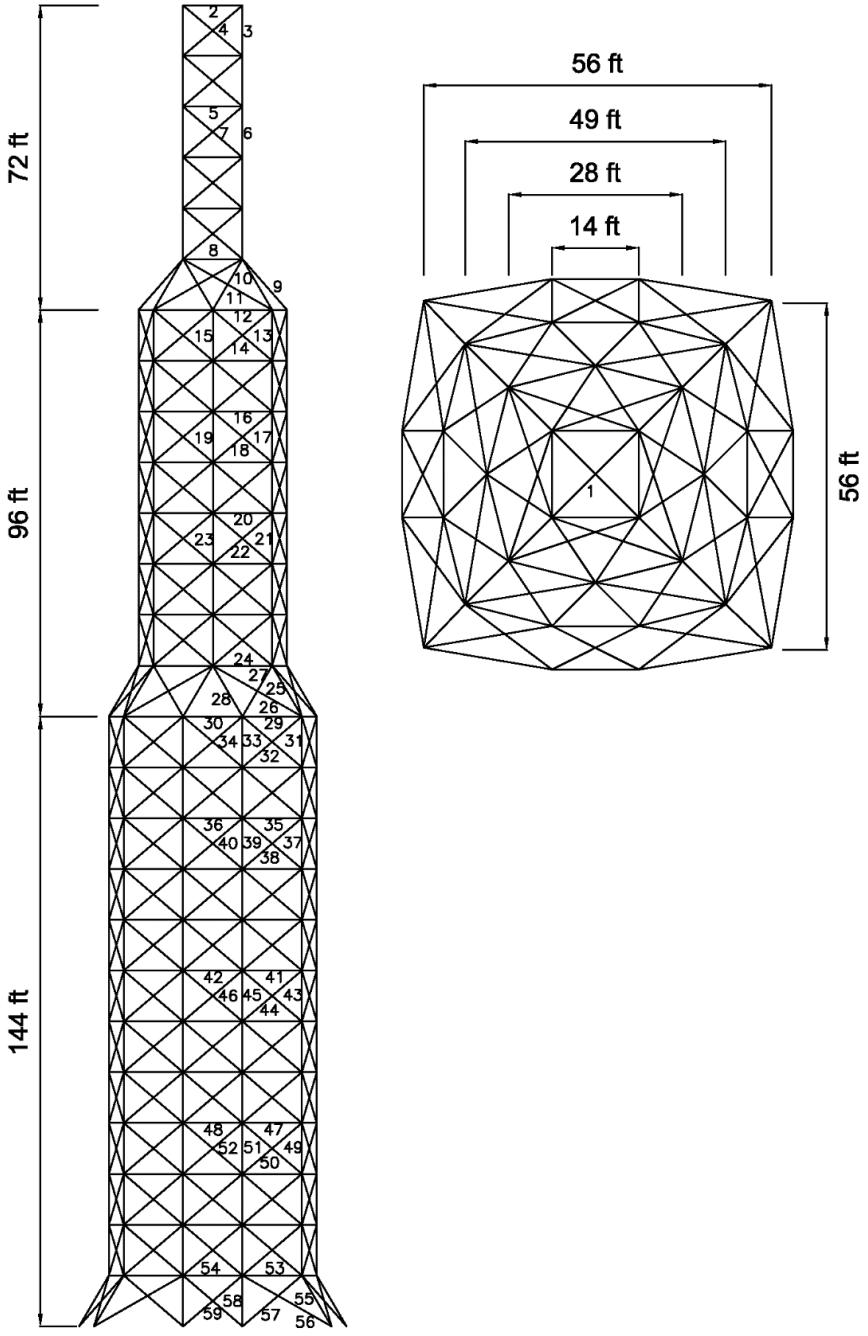


Figure 11: Schematic of the 942-bar space tower

Table 8 indicates that the optimal weight of 379,682 lb, computed using the developed optimization algorithm in this study, demonstrates its good and acceptable performance compared to other metaheuristic algorithms. Furthermore, Table 8 shows that the standard deviation of the solutions is very low, indicating a reduced probability of particles converging to local optima.

[DOI: 10.22068/ijoce.2026.16.1.660]

6.2. Results of the 200-Member Truss

For optimization of the 200-member truss defined in Section 5.2, a population of 600 particles and 600 iterations were employed. Table 9 presents the computed results for this problem using the developed algorithm in this study, compared to other metaheuristic algorithms.

Table 9: Comparison of Results for the 200-Bar Truss

Variables	**Optimal cross-sectional areas (in^2)**							
	WEO [47]	WSA [48]	CSP [40]	SAHS [41]	FPA [44]	ECM [45]	MNSV [46]	Present study
A1	0.1144	0.144758	0.148	0.154	0.1425	0.1471	0.1484	0.1442
A2	0.9443	0.943058	0.946	0.941	0.9637	0.94	0.9445	0.9404
A3	0.131	0.101225	0.101	0.1	0.1005	0.1	0.1	0.1012
A4	0.1016	0.100001	0.101	0.1	0.1	0.1	0.1	0.1
A5	2.0353	1.943059	1.9461	1.942	1.9514	1.94	1.9445	1.9404
A6	0.3126	0.296271	0.2979	0.301	0.2957	0.2965	0.298	0.2957
A7	0.1679	0.103267	0.101	0.1	0.1156	0.1	0.1	0.102
A8	3.1541	3.114355	3.1072	3.108	3.1133	3.1049	3.1227	3.1077
A9	0.1003	0.102462	0.101	0.1	0.1006	0.1	0.1	0.1021
A10	4.1005	4.114354	4.1062	4.106	4.11	4.1049	4.1227	4.1079
A11	0.435	0.400374	0.4049	0.409	0.4165	0.4038	0.399	0.4013
A12	0.1148	0.113995	0.1944	0.191	0.1843	0.1907	0.1	0.1527
A13	5.3823	5.388609	5.4299	5.428	5.4567	5.4298	5.3934	5.3964
A14	0.1607	0.100012	0.101	0.1	0.1	0.1006	0.1	0.1015
A15	6.4152	6.388601	6.4299	6.427	6.4559	6.4298	6.3934	6.3964
A16	0.5629	0.533194	0.5755	0.581	0.58	0.574	0.5264	0.5561
A17	0.401	0.394526	0.1349	0.151	0.1547	0.1333	0.4351	0.3016
A18	7.9735	7.941942	7.9747	7.973	8.0132	7.9745	7.9503	7.9503
A19	0.1092	0.100949	0.101	0.1	0.1	0.1	0.1	0.1009
A20	9.0155	8.94192	8.9747	8.974	9.0135	8.9745	8.9503	8.9503
A21	0.8628	0.834785	0.70648	0.719	0.7391	0.7064	0.859	0.7995
A22	0.222	0.151136	0.4225	0.422	0.787	0.434	0.1499	0.2725
A23	11.0254	10.940045	10.8685	10.892	11.1795	10.8791	10.9973	10.9355
A24	0.1397	0.100028	0.101	0.1	0.1462	0.1	0.1	0.101
A25	12.034	11.940047	11.8684	11.887	12.1799	11.8791	11.9973	11.9355
A26	1.0043	0.89727	1.035999	1.04	1.3424	1.0453	0.9124	0.9705
A27	6.5762	6.848813	6.6859	6.646	5.4844	6.63	6.6633	6.7147
A28	10.7265	10.884812	10.8111	10.804	10.1372	10.7828	10.8067	10.8049
A29	13.9666	13.749529	13.84649	13.87	14.5262	13.8692	13.8231	13.8352
Best weight (lb)	25674.83	25453.77	25467.9	25491.9	25521.81	25448.89	25446.67	25466.74
Average weight (lb)	26613.45	25455.67	25547.6	25610.2	25543.51	25531.69	25459.84	25507.35
Std Dev (lb)	702.8	2.337	135.09	141.85	18.13	42.16	10.64	20.63

According to Table 9, the best weight obtained using the algorithm proposed in this study is 25,466.74 lb, which demonstrates an acceptable performance compared to other algorithms. Moreover, the average calculated weight is 25,507.35 lb, indicating the reliability of the solutions generated by this algorithm in each optimization run.

[DOI: 10.22068/ijoce.2026.16.1.660]

6.3. Results of the 26-Story Truss

For optimization of the 942-member truss defined in Section 5.3, a population of 1200 particles and 1200 iterations were employed. Table 10 presents the computed results for this problem using the developed algorithm in this study, compared to other metaheuristic algorithms.

Table 10: Comparison of Results for the 26-Story Truss

Variables	**Optimal cross-sectional areas (in^2)**					
	CS [49]	JA [50]	IGWO [51]	ECM [45]	MNSV [46]	Present study
A1	1	1.045258	4.2489	1.0001	1.0014	1.0001
A2	1	1.00163	1.7702	1	1.0002	1.0001
A3	3.01	3.549999	1.5892	3.1663	3.1478	3.383
A4	1.75	1.92459	1.5235	1.8157	1.8196	1.8993
A5	1	1.000032	1.0265	1	1.0001	1
A6	14.27	15.337079	15.3979	14.4764	14.7104	14.6937
A7	2.93	3.108905	2.8825	3.0067	3.0264	3.0123
A8	1	6.589077	6.9912	7.036	6.7849	7.4347
A9	1	16.569661	11.2039	16.3761	17.2403	17.8889
A10	9.38	2.553777	2.7262	2.3792	2.8082	3.0145
A11	4.43	6.433946	8.1921	6.4381	5.8952	5.7641
A12	4.54	5.812166	6.2178	5.6027	5.5516	5.4979
A13	16.41	15.836882	16.5585	15.1181	15.3543	15.4353
A14	2.33	2.196943	2.3668	2.1249	2.15	2.2026
A15	7.51	4.324553	4.1519	4.0989	4.17	4.4819
A16	1	1.000047	1.237	1	1	1
A17	22.47	21.973772	22.3006	21.6802	21.4785	21.646
A18	2.7	2.674909	2.9996	2.5974	2.5845	2.6126
A19	13.58	8.722646	7.7559	7.8701	7.8867	8.1916
A20	1	1.000032	1.1283	1	1.0001	1
A21	28.93	29.898613	28.2646	27.7533	28.5209	28.2999
A22	3.23	3.249223	3.1924	3.1353	3.1416	3.1337
A23	23.87	16.995624	16.3965	15.8434	15.8106	15.7933
A24	41.67	25.510407	22.6095	26.3174	24.1869	23.93
A25	36.02	37.634066	40.0759	40.695	35.4787	35.4809
A26	6.41	1.220731	5.3549	1.15	1.8118	2.4213
A27	23.79	11.944077	9.2695	11.65	11.3197	10.9132
A28	28.39	16.515003	15.0911	16.0757	15.661	15.7564
A29	19.38	14.822892	14.0704	13.9046	13.9358	13.9262
A30	20.31	15.983565	15.1962	14.526	14.9356	15.271
A31	31.41	38.514252	37.149	35.385	36.9306	36.8498
A32	2.57	3.323571	3.1643	3.2074	3.1978	3.1795
A33	4.18	3.189674	3.4414	2.5329	2.5905	2.7757
A34	3.33	2.82237	2.2813	2.5066	2.5597	2.5626
A35	1	1.001323	1.0166	1	1	1.0003
A36	1	1.002606	1.4089	1	1	1.0001
A37	47.11	59.530117	59.6649	57.2845	57.395	57.3638
A38	2.35	3.250054	3.3173	3.1633	3.1552	3.1259
A39	3.79	2.068093	2.0249	2.1464	2.1302	2.1067
A40	3.3	3.084539	2.3953	2.8503	2.9218	3.0097
A41	1	1.000717	1.0554	1	1	1.0005
A42	1	1.239938	1.2294	1.0006	1.0119	1.0004
A43	63.33	79.891179	79.5798	76.7265	77.3501	76.021
A44	3.21	3.299488	3.2875	3.1547	3.1623	3.1459
A45	4.86	1.964128	1.9028	2.0641	2.0481	2.0226

[DOI: 10.22068/ijoce.2026.16.1.660]

A46	2.22	3.489718	3.246	3.2799	3.2504	3.3111
A47	1	1.000032	1.0277	1.0001	1	1.0004
A48	1	1.000032	1.0898	1	1	1.0003
A49	76.93	97.181471	93.8836	91.1669	93.2945	93.0759
A50	3.54	3.322281	3.0634	3.2346	3.2094	3.2441
A51	3.91	1.002997	1.7246	1	1	1.0007
A52	2.25	3.651629	3.9313	3.6002	3.6288	3.719
A53	11.44	7.226228	8.1063	6.5839	7.0099	7.679
A54	11.64	4.544599	9.8391	3.7855	4.1181	4.658
A55	36.94	41.411074	42.7529	41.7263	39.4509	39.824
A56	1	1.002207	1.1219	1	1	1.0005
A57	48.1	64.803517	63.0179	63.4174	61.3775	59.6052
A58	5.88	2.525618	2.6542	2.3264	2.5653	3.0479
A59	1	1.000054	1.6685	1	1.0013	1.0004
Best weight (lb)	134120	137344.36	136311.13	131984.4	131881.59	131940.98
Average weight (lb)	135244.7	N/A	137453.67	135768.12	133705.17	132213.54
Std Dev (lb)	1497.06	N/A	673.8566	2289.15	1075.02	358.0921

According to Table 10, the comparison of the results obtained from the developed algorithm with other optimization methods, such as CS, JA, IGWO, and ECM, indicates that the proposed algorithm demonstrates superior performance in terms of accuracy and convergence rate. In fact, the algorithm successfully reduced the structural weight to 131,940.98 lb, which is the smallest value recorded among the mentioned methods. Moreover, the low standard deviation obtained reveals the high stability of the method in producing repeatable results. This finding confirms the high efficiency of the proposed algorithm in solving complex structural optimization problems.

7. CONCLUSION

This study develops a particle swarm optimization algorithm incorporating a probabilistic guide selection mechanism, inspired by the collective behavior of birds and their flocking movements. In most particle swarm-based metaheuristic algorithms, the best particle in the swarm is utilized as the leader for all other members, which reduces the exploration capability of the algorithm. In the algorithm proposed in this study, all particles have the opportunity to be selected as leaders, and each particle independently chooses its own leader. In this algorithm, the probability of each particle being selected is calculated based on its performance. Additionally, to achieve better control over the exploration and exploitation properties of the algorithm, a parameter denoted as α is introduced and evaluated. In this study, the performance of the developed algorithm was validated, following the tuning of the equilibrium parameter α, using a set of mathematical benchmark problems. Subsequently, the algorithm was applied to structural engineering problems, and the obtained results were analyzed and compared. If the value of parameter α is very close to zero, the algorithm's performance will most closely resemble that of the classical PSO algorithm, which exhibits very high exploitation capability. Conversely, if the value of parameter α equals one, the algorithm will operate nearly as a random search. Thus, as previously stated, parameter α has a direct relationship with the exploration power of the algorithm and an inverse relationship with its exploitation power. The particle density variation process during the optimization, as

[DOI: 10.22068/ijoce.2026.16.1.660]

illustrated in Fig. 2 - 6, demonstrates that parameter α controls particle cominvergence throughout the optimization process and prevents premature convergence.

In this study, three trusses comprising 72, 200, and 942 members—with 16, 29, and 56 design variables, respectively—were defined under all structural constraints. Their weights were subsequently optimized using the developed algorithm, and the optimization results were compared with those obtained from other metaheuristic optimization algorithms. The comparison of results indicates a lower probability of the algorithm getting trapped in local optima. Additionally, the lower standard deviation of the solutions demonstrates the higher reliability of the algorithm in finding the optimal solution.

Overall, the obtained results demonstrate that the developed algorithm outperforms the classical particle swarm optimization algorithm in preventing premature convergence. Furthermore, it incorporates a tunable parameter to balance exploration and exploitation, which can be calibrated for various problems.

[DOI: 10.22068/ijoce.2026.16.1.660]

REFERENCES

1. Holland JH. *Adaptation in Natural and Artificial Systems*. MIT Press; 1992.
2. Rajeev S, Krishnamoorthy CS. Discrete optimization of structures using genetic algorithms. *J Struct Eng*. 1992;**118**(5):1233–50.
3. Adeli H, Kumar S. Distributed genetic algorithm for structural optimization. *J Aerosp Eng*. 1995;**8**(3):156–63.
4. Ghasemi MR, Yousefi M. Reliability-based optimization of steel frame structures using modified genetic algorithm. *Asian J Civ Eng*. 2011;**12**:449–75.
5. Kennedy J, Eberhart R. Particle swarm optimization. In: *Proc ICNN'95 Int Conf Neural Netw*. 1995;**4**:1942–48.
6. Gomes HM. Truss optimization with dynamic constraints using a particle swarm algorithm. *Expert Syst Appl*. 2011;**38**(1):957–68.
7. Dorigo M, Maniezzo V, Colorni A. Ant system: optimization by a colony of cooperating agents. *IEEE Trans Syst Man Cybern B*. 1996;**26**(1):29–41.
8. Ravan M, Ghiami Azad AR. Optimization of cable pre-tensioning force in cable-stayed bridges by simplifying the structural model with a case study of Lali bridge. *Sharif J Civ Eng*. 2023;**39**(2):125–35.
9. Ravan M, Tamimi A, Ghiami Azad AR. A novel simplified approach for calculating optimal cable pre tensioning forces in symmetrical cable stayed bridges. *Struct Multidiscip Optim*. 2026.
10. PayamiFar T, Sojoudizadeh R, Azizian H, Rahimi L. Enhanced prairie dog metaheuristic optimization algorithm for engineering optimization problems. *Int J Optim Civ Eng*. 2025.
11. Kamgar R, Rahmani F. Geometrical and material optimization of functionally graded doubly-curved shells using the grey wolf optimization algorithm. *Int J Optim Civ Eng*. 2025.
12. Fahimi Farzam M, Salehi M. Optimization-based simplification of a high-rise benchmark steel building for dynamic analysis. *Int J Optim Civ Eng*. 2025.

13. Asaad Samani A, Hoseini Vaez SR, Hosseini P. Optimal design of unprotected steel moment frames under fire condition. *Int J Optim Civ Eng*. 2025.
14. Kaveh A, Khayatazad M. Ray optimization for size and shape optimization of truss structures. *Comput Struct*. 2013;**117**:82–94.
15. Suganthan PN. Particle swarm optimiser with neighbourhood operator. In: *Proc 1999 Congr Evol Comput (CEC99)*. 1999:1958–62.
16. Clerc M. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: *Proc 1999 Congr Evol Comput (CEC99)*. 1999:1951–57.
17. Clerc M, Kennedy J. The particle swarm – explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans Evol Comput*. 2002;**6**(1):58–73.
18. Eberhart RC, Shi Y. Comparing inertia weights and constriction factors in particle swarm optimization. In: *Proc 2000 Congr Evol Comput (CEC2000)*. 2000:84–88.
19. Eberhart RC, Shi Y. Tracking and optimizing dynamic systems with particle swarms. In: *Proc 2001 Congr Evol Comput (CEC2001)*. 2001:94–100.
20. Van den Bergh F, Engelbrecht AP. Particle swarm weight initialization in multi-layer perceptron artificial neural networks. In: *Development and Practice of Artificial Intelligence Techniques*. 1999.
21. Lovbjerg M, Krink T. Extending particle swarm optimisers with self-organized criticality. In: *Proc 2002 Congr Evol Comput (CEC'02)*. 2002:1588–93.
22. Poli R, Kennedy J, Blackwell T. Particle swarm optimization. *Swarm Intell*. 2007;**1**(1):33–57.
23. Brits R, Engelbrecht AP, van den Bergh F. Locating multiple optima using particle swarm optimization. *Appl Math Comput*. 2007;**189**(2):1859–83.
24. Parrott D, Li X. Locating and tracking multiple dynamic optima by a particle swarm model using speciation. *IEEE Trans Evol Comput*. 2006;**10**(4):440–58.
25. Lovbjerg M, Rasmussen TK, Krink T. Hybrid particle swarm optimiser with breeding and subpopulations. 2001.
26. Van den Bergh F, Engelbrecht AP. A cooperative approach to particle swarm optimization. *IEEE Trans Evol Comput*. 2004;**8**(3):225–39.
27. Liang JJ, Qin AK, Suganthan PN, Baskar S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans Evol Comput*. 2006;**10**(3):281–95.
28. Cao Y, Zhang H, Li W, Zhou M, Zhang Y, Chaovalitwongse WA. Comprehensive learning particle swarm optimization algorithm with local search for multimodal functions. *IEEE Trans Evol Comput*. 2019;**23**(4):718–31.
29. Bertsimas D, Tsitsiklis J. Simulated annealing. *Stat Sci*. 1993;**8**(1).
30. Selvakumar AI, Thanushkodi K. Optimization using civilized swarm: solution to economic dispatch with multiple minima. *Electr Power Syst Res*. 2009;**79**(1):8–16.
31. Larson J, Menickelly M, Wild SM. Derivative-free optimization methods. *Acta Numer*. 2019;**28**:287–404.
32. Yang X-S. A new metaheuristic bat-inspired algorithm. 2010:65–74.
33. Yang X, Hossein Gandomi A. Bat algorithm: a novel approach for global engineering optimization. *Eng Comput*. 2012;**29**(5):464–83.
34. Alweshah M, Abdullah S. Hybridizing firefly algorithms with a probabilistic neural

[DOI: 10.22068/ijoce.2026.16.1.660]

network for solving classification problems. *Appl Soft Comput*. 2015;**35**:513–24.
35. Yang X-S. Firefly algorithms for multimodal optimization. 2009:169–78.
36. Verma OP, Aggarwal D, Patodi T. Opposition and dimensional based modified firefly algorithm. *Expert Syst Appl*. 2016;**44**:168–76.
37. Kazem A, Sharifi E, Hussain FK, Saberi M, Hussain OK. Support vector regression with chaos-based firefly algorithm for stock market price forecasting. *Appl Soft Comput*. 2013;**13**(2):947–58.
38. Pandit D, Zhang L, Chattopadhyay S, Lim CP, Liu C. A scattering and repulsive swarm intelligence algorithm for solving global optimization problems. *Knowl Based Syst*. 2018;**156**:12–42.
39. Mirjalili S. The ant lion optimizer. *Adv Eng Softw*. 2015;**83**:80–98.
40. Kaveh A, Sheikholeslami R, Talatahari S, Keshvari-Ilkhichi M. Chaotic swarming of particles: a new method for size optimization of truss structures. *Adv Eng Softw*. 2014;**67**:136–47.
41. Degertekin SO. Improved harmony search algorithms for sizing optimization of truss structures. *Comput Struct*. 2012;**92–93**:229–41.
42. Kaveh A, Talatahari S. Size optimization of space trusses using Big Bang–Big Crunch algorithm. *Comput Struct*. 2009;**87**(17–18):1129–40.
43. Camp CV, Farshchin M. Design of space trusses using modified teaching–learning based optimization. *Eng Struct*. 2014;**62–63**:87–97.
44. Bekdaş G, Nigdeli SM, Yang X-S. Sizing optimization of truss structures using flower pollination algorithm. *Appl Soft Comput*. 2015;**37**:322–31.
45. Pouriyanezhad E, Rahami H, Mirhosseini SM. Truss optimization using eigenvectors of the covariance matrix. *Eng Comput*. 2021;**37**(3):2207–24.
46. Shojaei I, Rahami H. A heuristic swarm-based optimization method using multi-variate normal distributions with self-adaptive variance matrices. *Structures*. 2022;**36**:372–91.
47. Kaveh A, Bakhshpoori T. A new metaheuristic for continuous structural optimization: water evaporation optimization. *Struct Multidiscip Optim*. 2016;**54**(1):23–43.
48. Adil B, Cengiz B. Optimal design of truss structures using weighted superposition attraction algorithm. *Eng Comput*. 2020;**36**(3):965–79.
49. Gandomi AH, Talatahari S, Yang X-S, Deb S. Design optimization of truss structures using cuckoo search algorithm. *Struct Des Tall Spec Build*. 2013;**22**(17):1330–49.
50. Degertekin SO, Lamberti L, Ugur IB. Sizing, layout and topology design optimization of truss structures using the Jaya algorithm. *Appl Soft Comput*. 2018;**70**:903–28.
51. Kaveh A, Zakian P. Improved GWO algorithm for optimal design of truss structures. *Eng Comput*. 2018;**34**(4):685–707.

[DOI: 10.22068/ijoce.2026.16.1.660]