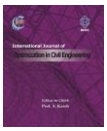
INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2024; 14(2):189-210



AN ENHANCED GENETIC ALGORITHM BASED ON THE INTRODUCTION OF FIXED STATION GROUPS AND A NEW VARIABLE MULTI-PARENT CROSSOVER TECHNIQUE

M. A. Roudak[1*, †], M. A Shayanfar[2], M. Farahani[1], S. Badiezadeh[1], and R. Ardalan[1]
[1]*Department of Civil Engineering, Faculty of Engineering, Alzahra University, Tehran, Iran*
[2]*School of Civil Engineering, Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Tehran, Iran*

[DOI: 10.22068/ijoce.2024.14.2.582]

ABSTRACT

Genetic algorithm is a robust meta-heuristic algorithm inspired by the theory of natural selection to solve various optimization problems. This study presents a method with the purpose of promoting the exploration and exploitation of genetic algorithm. Improvement in exploration ability is made by adjusting the initial population and adding a group of fixed stations. This modification increases the diversity among the solution population, which enables the algorithm to escape from local optimum and to converge to the global optimum even in fewer generations. On the other hand, to enhance the exploitation ability, increasing the number of selected parents is suggested and a corresponding crossover technique has been presented. In the proposed technique, the number of parents to generate offspring is variable during the process and it could be potentially more than two. The effectiveness of the modifications in the proposed method has been verified by examining several benchmark functions and engineering design problems.

Keywords: Genetic algorithm; meta-heuristic optimization; fixed station groups; variable multi-parent crossover.

Received: 17 January 2024; Accepted: 17 March 2024

1. INTRODUCTION

Gradient-based methods have been commonplace among researchers for solving

[*]Corresponding author: 1Department of Civil Engineering, Faculty of Engineering, Alzahra University, Tehran, Iran
[†]E-mail address: a.roudak@alzahra.ac.ir (M. A. Roudak)

optimization problems [1-3]. These methods enable a search of the solution space near a specific point, where gradient information of the objective function is available [4, 5]. These algorithms are probable to provide an optimal solution if the method is well-implemented [6]. However, the majority of these mathematical optimization techniques require the calculation of the gradients of objective function and constraints. This might be applicable in simple small-scale problems. Having said that, certain problems involve discontinuous constraint functions, and therefore their gradients do not exist. Additionally, in some cases the constraint functions could be complicated which can make the computation of their gradients difficult [7, 8]. By and large, obtaining gradient information for the objective function may incur significant costs, or in some instances it might be even unattainable [6].

Hence, due to the above-mentioned computational downside of the mathematical algorithms, the meta-heuristic approach was introduced. The emergence of this approach has enabled efficient exploration of the entire search space to discover optimal solutions [7]. Meta-heuristics are unconventional optimization approaches that are applicable without the need for gradient information. Furthermore, they operate without the need for an explicit relationship between the objective function and constraints. Fundamentally, meta-heuristic algorithms are strategies inspired by natural, social, biological, or physical principles that have found many applications in various domains like engineering, economics, mathematics, and other areas of science [8]. Most meta-heuristic algorithms imitate the intelligence exhibited by natural phenomena in order to direct the search process. The underlying premise of all meta-heuristic strategies is to approach the optimal answer as closely as feasible, rather than achieving the exact ultimate solution. Nevertheless, the procedure does not ensure that the solution produced at the end is the optimal solution [9]. In recent years, this feature has spurred numerous scholars to devise novel algorithms or enhance existing methods [10-14].

Evolutionary algorithms represent a new category of meta-heuristic techniques. One of the most successful methods among these evolutionary algorithms is Genetic Algorithm (GA), originally developed by Holland [15] and later revised by De Jong [16] and Goldberg [17]. Due to its characteristics, this approach has been proven more effective than previous algorithms in solving various optimization problems [18]. GA operates simultaneously with a population of design points. Therefore, it leads to a more diverse and easier exploration of design space. Moreover, mutation, selection, and crossover parts are employed by GA to explore the solution space. These are randomized operators, used instead of deterministic operators. It is important to note that with a few adjustments, GA can deal with more various optimization problems [19-22]. As it appears, the significant attraction toward genetic algorithms or other meta-heuristic algorithms, originates from their efficacy in the complex problems that are challenging for conventional methods. In this regard, Gandomi and Alavi [23] employed a multi-gene genetic programming (MGGP) approach to address various engineering problems in material, structural, geotechnical, and earthquake engineering. In another study, Gandomi, Alavi [24] utilized gene expression programming (GEP) to predict the shear strength of slender reinforced concrete (RC) beams. Yazdani, Khatibinia [25] suggested a modified discrete gravitational search algorithm for probabilistic performance-based optimization design of complex structures subjected to earthquake. Degertekin, Tutar [26] applied school-based optimization for performance-based optimum seismic design of steel frames. Rao and Pawar [27] utilized the Rao algorithms for the optimum design of

[DOI: 10.22068/ijoce.2024.14.2.582]

mechanical system components. Hassan, Kamel [28] made modifications to the second Rao algorithm for solving the optimal power flow problem. Kaveh and Zaerreza [29] enhanced meta-heuristic algorithms for optimization of the structures with deterministic and probabilistic constraints. Recently, Kaveh has reviewed 25 meta- heuristic algorithms [30].

However, since most engineering problems involve optimization within complicated constraints, simple genetic algorithms in many cases are not able to produce successful applications. Nevertheless, their performance can be potentially enhanced. This is possible by using tailored approaches, which is a topic of current debate. In this regard recently, Zheng, Zhong [31] and Sun, Shen [32] employed reinforced hybrid GA. Chowdhury and Hovda [33] and Ishaque, Johar [34] applied fuzzy logic in combination with GA.

This research aims to introduce a new method based on GA, to avoid being stuck in local optima by increasing exploration ability, and to increase the efficiency by introducing a new version of crossover. The former is provided by adding a group of fixed points or stations to the population of design variables. Moreover, to exploit the population more efficiently, multi-parent selection is employed. An effective crossover approach is implemented to form a multi-parent combination.

[DOI: 10.22068/ijoce.2024.14.2.582]

2. GENETIC ALGORITHM

Genetic algorithm is a meta-heuristic algorithm that operates based on the theory of natural selection. Genetic algorithm consists of five main stages: initialization, fitness evaluation, parent selection which is performed based on a defined fitness function, crossover which combines parents' genes to produce offspring, and mutation which is designed to make random changes in some individuals. In the following, the steps of GA, as well as how they are applied in the proposed algorithm are explained briefly.

2.1 Initialization

In the first step, a set of individuals are needed to start the algorithm. These individuals are randomly generated to form the first generation, which is named the initial population. Each member of the population is considered a possible solution.

2.2 Fitness evaluation

The fitness of an individual is evaluated by a fitness function. The function determines how close an individual is to the solution. The individuals with high fitness are selected as parents to form the next generations. Therefore, applying an appropriate fitness function is important in proper parent selection.

In this study, the fitness function assigns a larger value to individuals with the lower objective function, as it is for minimization problems. As the objective function value increases, the assigned fitness decreases uniformly until the individual with the highest objective function value receives the lowest fitness. This uniform fitness function makes the selection of parents not dependent on the value of the objective function, i.e. just the orders matter not the values. This causes the probability of selection to decrease or increase within equal intervals.

2.3 Selection

In this stage, a number of individuals are selected as parents. Selection is performed based on the fitness function. Different types of parent selections are available be used in genetic algorithm. Roulette Wheel Selection, Tournament Selection, Rank Selection, etc. are some of the common selection techniques.

Roulette Wheel Selection is the employed method for parent selection in this study. In this method, the entire population is examined and individuals with higher fitness are selected as parents with greater opportunity.

2.4 Crossover

Crossover is a technique that decides how parents are combined to produce offspring. In fact, crossover determines the percentage of each parent's contribution to generate new children. Based on the application, different types of crossover techniques could be implemented. In this study, a new crossover technique has proposed based on Whole Arithmetic crossover.

Whole Arithmetic crossover is a method in which every child is made of a linear combination of the parents' genes. In fact, offspring inherit a specific percentage of genes from each parent, as follows:

$$\begin{aligned} \mathbf{CH}_1 &= \boldsymbol{\alpha}.\times \mathbf{P}_1 + (\mathbf{1} - \boldsymbol{\alpha}).\times \mathbf{P}_2 \\ \mathbf{CH}_2 &= (\mathbf{1} - \boldsymbol{\alpha}).\times \mathbf{P}_1 + \boldsymbol{\alpha}.\times \mathbf{P}_2 \end{aligned} \tag{1}$$

where $\mathbf{P_1}$ and $\mathbf{P_2}$ are parents, $\mathbf{1}$ denotes a *d*-dimensional vector with unit elements, and $\boldsymbol{\alpha}$ contains *d* components of random numbers(*d* represents the dimension of the problem).. These components could be constant or change randomly to induce more diversity. The operator "×" denotes the element-by-element multiplication.

2.5 Mutation

After the production of each generation, a small number of offspring are mutated to maintain the diversity of solutions. In this study, the number and position of the mutant offspring are randomly selected. Then, the selected individuals are added a specific normally distributed random value.

3. PROPOSED METHO

In this paper, an effective method is presented to modify the genetic algorithm by increasing the exploration ability. The method is based on adding some fixed individuals in specific regions in the search space. On the other hand, a multi-parent selection approach is adopted to exploit individuals effectively. The number of selected parents ranges randomly between 1 and 5 in each step. Eiben *et al*. [34] demonstrated that selecting more than two parents, depending on the employed crossover method, can reduce the number of required

[DOI: 10.22068/ijoce.2024.14.2.582]

generations.

In Section 3.1, a variable multi-parent compatible crossover is defined, and Section 3.2 presents a technique to enhance exploration ability by adding fixed station groups.

3.1 Variable multi-parent crossover

The proposed method introduces a multi-parent compatible crossover technique based on Whole Arithmetic crossover. As previously mentioned, the number of parents is randomly selected between 1 and 5. At each step, the number of produced children matches the number of selected parents. A random vector $\boldsymbol{\alpha_i}$, including *d* random numbers, determines the inheritance percentage of each parent to make every child. Then, the set of all random vectors $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \ldots, \boldsymbol{\alpha}_n]$ is circularly shifted to create the subsequent offspring differently. Fig. 1 illustrates the variation in $\boldsymbol{\alpha}$ after the creation of each child.

$$\begin{aligned}
\mathbf{CH}_1 &= (\boldsymbol{\alpha}_1.\times \mathbf{P}_1 + \boldsymbol{\alpha}_2.\times \mathbf{P}_2 + \mathrm{L} + \boldsymbol{\alpha}_{\mathrm{n}-1}.\times \mathbf{P}_{\mathrm{n}-1} + \boldsymbol{\alpha}_{\mathrm{n}}.\times \mathbf{P}_{\mathrm{n}})./\boldsymbol{\lambda} \\
\mathbf{CH}_2 &= (\boldsymbol{\alpha}_{\mathrm{n}}.\times \mathbf{P}_1 + \boldsymbol{\alpha}_1.\times \mathbf{P}_2 + \mathrm{L} + \boldsymbol{\alpha}_{\mathrm{n}-2}.\times \mathbf{P}_{\mathrm{n}-1} + \boldsymbol{\alpha}_{\mathrm{n}-1}\mathbf{P}_{\mathrm{n}})./\boldsymbol{\lambda} \\
&\vdots \\
\mathbf{CH}_n &= (\boldsymbol{\alpha}_2.\times \mathbf{P}_1 + \boldsymbol{\alpha}_3.\times \mathbf{P}_2 + \mathrm{L} + \boldsymbol{\alpha}_{\mathrm{n}}.\times \mathbf{P}_{\mathrm{n}-1} + \ \boldsymbol{\alpha}_1.\times \mathbf{P}_{\mathrm{n}})./\boldsymbol{\lambda}
\end{aligned} \tag{2}$$

In the above equation, *n* is the number of parents in every selection step, and the operator "./" denotes the element-by-element division. The vector $\boldsymbol{\lambda}$ is introduced to control the summation of genes' percentage and is obtained by:

$$\boldsymbol{\lambda} = \sum_{k=1}^{n} \boldsymbol{\alpha}_{\mathbf{k}} \tag{3}$$

As it could be observed from Eq. (2), the number of parents *n* does not have to be necessarily 2 and could possess any arbitrary value.

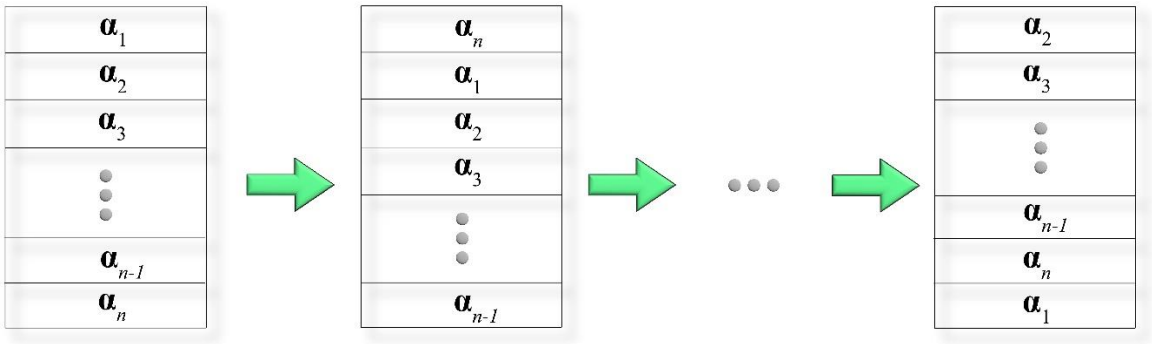


Figure 1. The variation of $\boldsymbol{\alpha}$ and its circular shift

3.2 Fixed station groups

One of the challenges in Genetic Algorithm or other optimization methods is that it might converge to local optima instead of the global optimum solution. To address this problem,

[DOI: 10.22068/ijoce.2024.14.2.582]

the proposed method presents an effective approach by increasing exploration ability. In this method, some fixed individuals are added to the population. The location of fixed stations is determined such that it could cover all the search space constantly. This ensures that the algorithm does not focus on a specific region.

In the proposed approach, first, the center of each variable range is obtained. The associated point, located at the center of search space, is denoted by $\mathbf{X_C}$. This is the first fixed station. Then, other fixed stations are selected at a specific distance from $\mathbf{X_C}$. Thus, the set of all fixed stations is obtained as follows:

$$\mathbf{X_C} = \frac{1}{2}(\mathbf{X_{max}} + \mathbf{X_{min}})$$

$$\mathbf{X_j^k} = \left[x_{1C}, x_{2C}, \mathrm{K}, x_{(i-1)C}, x_{iC} - S(k)r_i, x_{(i+1)C}, \mathrm{K}, x_{dC}\right]^T k = 1{:}\,FSG; i = 1{:}\,d; j = i \tag{4}$$

$$\mathbf{X_j^k} = \left[x_{1C}, x_{2C}, \mathrm{K}, x_{(i-1)C}, x_{iC} + S(k)r_i, x_{(i+1)C}, \mathrm{K}, x_{dC}\right]^T k = 1{:}\,FSG; i = 1{:}\,d; j = d + i$$

where $\mathbf{X_j^k}$ denotes *j*th point in the *k*th group of fixed stations and *FSG* is the number of fixed station groups. Accordingly, there is a total of $2(FSG)d+1$ fixed stations. $\mathbf{X_{max}}$ and $\mathbf{X_{min}}$ are the vectors including upper and lower bounds of all variables, respectively. The scalar r_i is the *i*th component of the vector $\boldsymbol{r}$

$$\mathbf{r} = \frac{1}{2}(\mathbf{X_{max}} - \mathbf{X_{min}}) \tag{5}$$

and the value $S(k)\epsilon[0,1]$ is proportional to the distance of the *k*th group of fixed stations from the center point $\mathbf{X_C}$. $S(k)$ is defined as

$$S(k) = S_0 \frac{k}{FSG} \qquad k = 1{:}\,FSG \tag{6}$$

The constant S_0 is associated with the distance of the furthest group of fixed stations from the center point $\mathbf{X_C}$. For instance, for a two-dimensional case (d=2), if the number of fixed station groups is 3 (FSG=3), and S_0 is set to 0.75, then 13 stations are obtained from Eq. (4). These stations are illustrated in Fig. 2. As observed in the figure, the third (furthest) group of stations are specified by S_0. Then its distance from $\mathbf{X_C}$ is equally divided to locate other groups of fixed stations. These 13 stations relatively cover the search space. Inaddition, since they are among the potential parents forever (of course with their weights), they decrease the probability of ignoring a region. This results in much smaller probability for local traps.

[DOI: 10.22068/ijoce.2024.14.2.582]

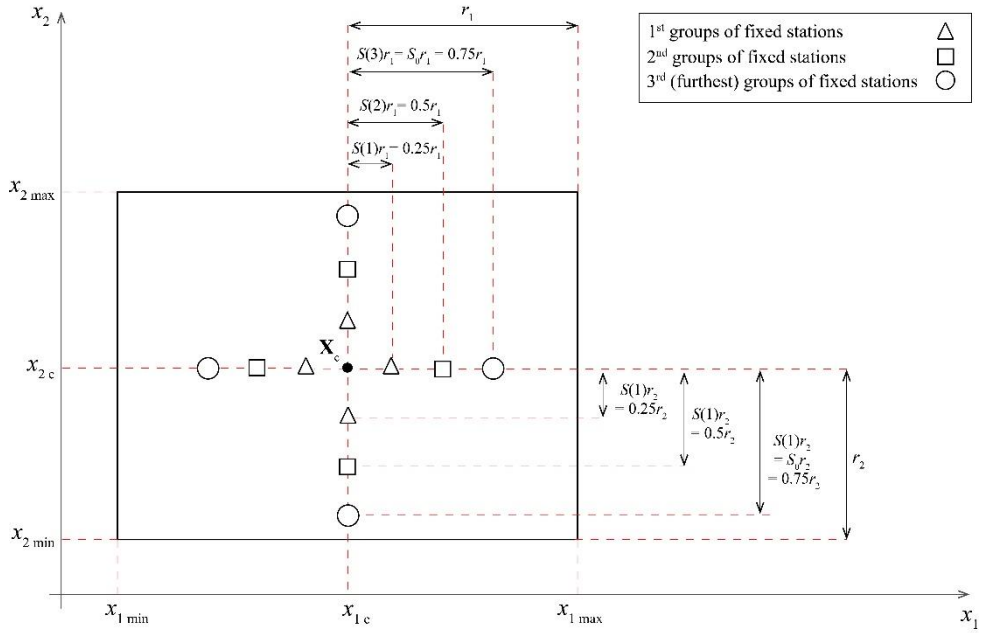


Figure 2. Distribution of fixed stations in search space for d=2, FSG=3, and S0=0.75

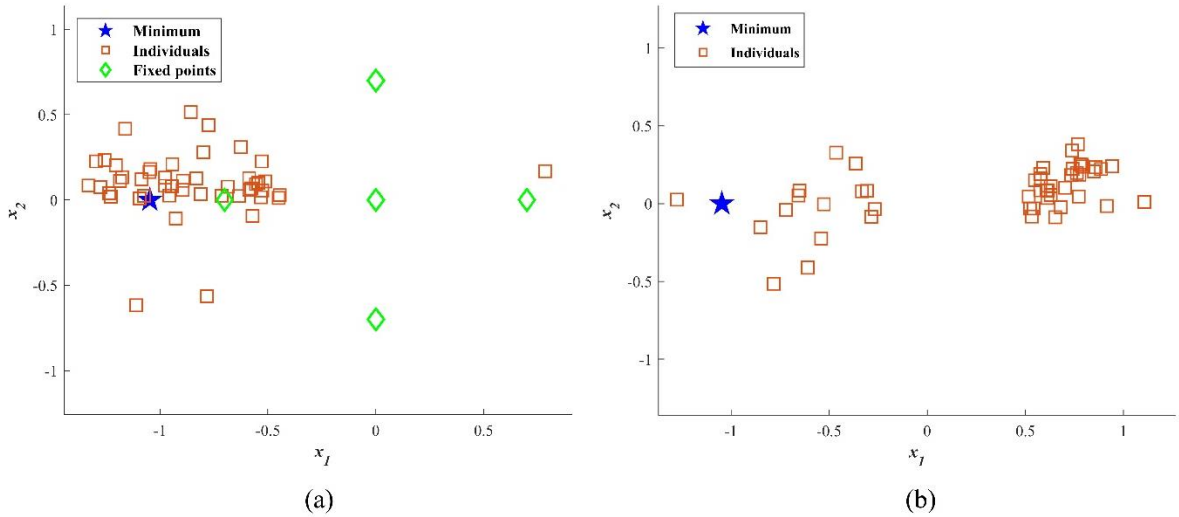


Figure 3. The population after three generations (a) at presence of fixed station groups (b) without fixed station groups

The presence of fixed stations causes the search space to be divided into several subspaces. The algorithm considers all these sections to generate the individuals. As a result, the population could often be concentrated more quickly in a subspace in which the optimum point is located. Just to clarify, an illustrative example of such a performance is displayed in Fig. 3. The figure indicates the population after three generations for a two-dimensional function in the presence of fixed stations and without them. As shown in the figure, fixed individuals have caused the population to concentrate around the optimum

[DOI: 10.22068/ijoce.2024.14.2.582]

within just three generations. While without them, the algorithm has stuck in the local minimum.

4. RESULTS

To evaluate the effectiveness of the proposed method, several diverse numerical examples sourced from the literature have been analyzed. These examples include a set of 13 benchmark functions. Additionally, the efficiency of the proposed method has been validated through the examination of 3 distinct engineering constrained optimization problems, taken from previous studies.

4.1 Benchmark problems

In this section, the optimization process is conducted for 13 benchmark functions extracted from Ref. [36]. The specifications of these functions, including their mathematical formula, variable ranges, and respective global minimum values are presented in Table 1. Additionally, visual representations of bivariate functions and their corresponding contour lines are illustrated in Figs. 4-12.

The results of each function are detailed in Tables 2-14. These tables present a comparison between the proposed modified GA (mGA) and GA. The tables compare the number of generations needed to converge to the solution.

Additionally, the impact of the population size of each generation and the number of fixed station groups has been examined. In each function, the results for population sizes of 10, 20, 50, and 100 are compared in the rows of tables. The columns demonstrate the results for different values of fixed station groups. $FSG = 0$ is considered to assess the effect of the variable multi-parent selection and multi-crossover technique. In other columns, the number of generations required for FSG =1, 5, and 10 are listed. All data in the tables are the mean values of 10 independent runs.

Analysis of the results demonstrates a substantial enhancement in the algorithm by employing multiple parents alongside the presented crossover method. Especially when the population size per generation is small, this technique leads to faster convergence to the solution compared to GA. In fact, employing the variable multi-parent crossover technique involving more than two parents, enables reaching the global minimum even with a notably reduced population size. This is shown in illustrative figures and tables.

As shown in the tables, adding fixed stations, which improves the exploration ability, could even facilitate reaching the global minimum in fewer generations. In fact, the inclusion of fixed stations prevents being stuck in local minima and enhances the efficiency of reaching solution. The optimal number of fixed station groups is observed typically between 1 and 5, as indicated by the analysis results.

[DOI: 10.22068/ijoce.2024.14.2.582]

Table1. Benchmark problems

Function name	Range	Function	Global minimum
Aluffi-Pentiny	$\mathbf{X}\in[-10, 10]^2$	$f_{cost}(\mathbf{X})=\frac{1}{4}x_1^4-\frac{1}{2}x_1^2+\frac{1}{10}x_1+\frac{1}{2}x_2^2$	−0.352386
Bohachevsky 1	$\mathbf{X}\in[-100, 100]^2$	$f_{cost}(\mathbf{X})=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1)-\frac{4}{10}\cos(4\pi x_2)+\frac{7}{10}$	0.0
Bohachevsky 2	$\mathbf{X}\in[-50, 50]^2$	$f_{cost}(\mathbf{X})=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2)+\frac{3}{10}$	0.0
Camel	$\mathbf{X}\in[-5, 5]^2$	$f_{cost}(\mathbf{X})=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	−1.0316
Cb3	$\mathbf{X}\in[-5, 5]^2$	$f_{cost}(\mathbf{X})=2x_1^2-1.05x_1^5+\frac{1}{6}x_1^6+x_1x_2+x_2^2$	0.0
Cosine mixture	$n = 4$, $\mathbf{X}\in[-1, 1]^n$	$f_{cost}(\mathbf{X})=\sum_{i=1}^{n}x_i^2-\frac{1}{10}\sum_{i=1}^{n}\cos(5\pi x_i)$	−0.4
DeJoung	$\mathbf{X}\in[-5.12, 5.12]^2$	$f_{cost}(\mathbf{X})=x_1^2+x_2^2+x_3^2$	0
Exponential	$n = 2,4,8$, $\mathbf{X}\in[-1, 1]^n$	$f_{cost}(\mathbf{X})=-\exp\left(-0.5\sum_{i=1}^{n}x_i^2\right)$	−1
Goldstein and price	$\mathbf{X}\in[-2, 2]^2$	$f_{cost}(\mathbf{X})=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	3.0
Griewank	$\mathbf{X}\in[-100, 100]^2$	$f_{cost}(\mathbf{X})=1+\frac{1}{200}\sum_{i=1}^{2}x_i^2-\prod_{i=1}^{2}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	0.0
Rastrigin	$\mathbf{X}\in[-1, 1]^n$	$f_{cost}(\mathbf{X})=\sum_{i=1}^{2}(x_i^2-\cos(18x_i))$	−2.0
Rosenbrock	$n = 2$, $\mathbf{X}\in[-30, 30]^n$	$f_{cost}(\mathbf{X})=\sum_{i=1}^{n-1}100(x_{i+1}-x_i^2)^2+(x_i-1)^2$	0.0
Bukin	$-15\le x_1\le -5$ $-3\le x_2\le 3$	$f_{cost}(\mathbf{X})=100\sqrt{\lvert x_2-0.01x_1^2\rvert}+0.01\lvert x_1+10\rvert$	0.0

[DOI: 10.22068/ijoce.2024.14.2.582]

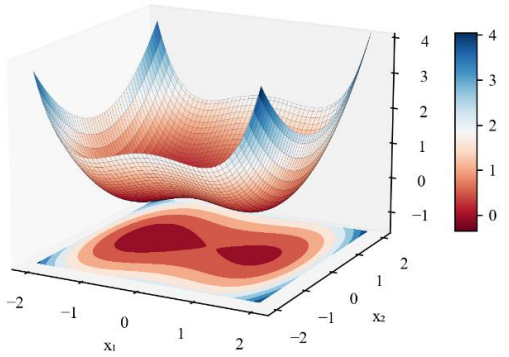


Figure 4. The visual of "Aluffi-Pentiny" and its contour lines

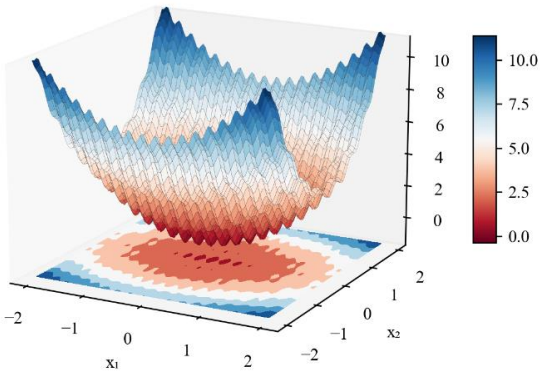


Figure 5. The visual of "Bohachevsky 1" and its contour lines

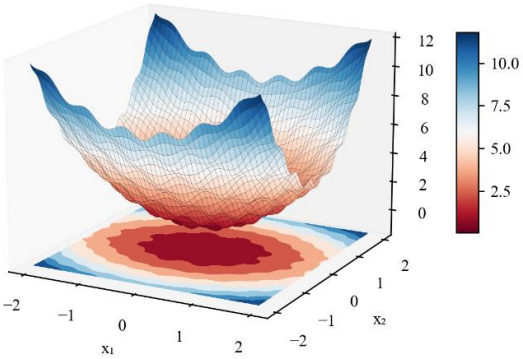


Figure 6. The visual of "Bohachevsky 2" and its contour lines

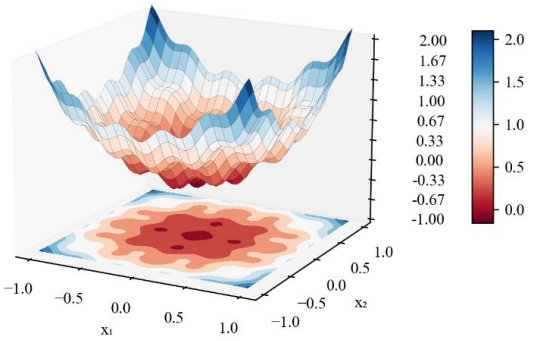


Figure 7. The visual of "Cosine mixture" and its contour lines

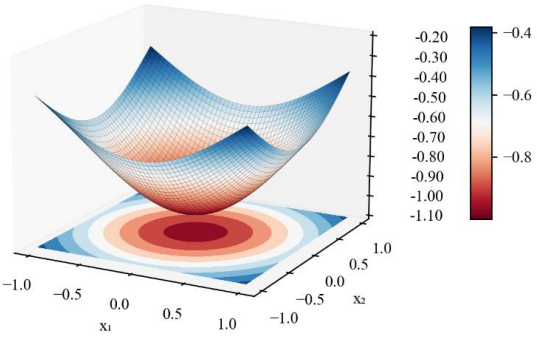


Figure 8. The visual of " Exponential " and its contour lines

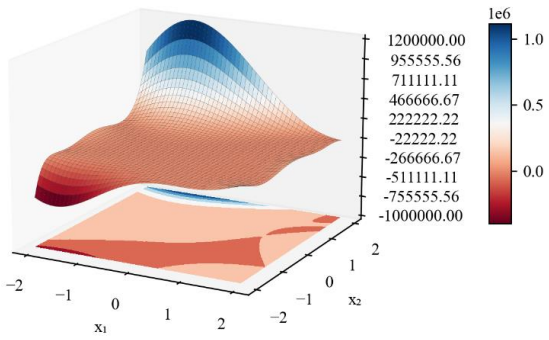


Figure 9. The visual of "Goldstein and price" and its contour lines

[DOI: 10.22068/ijoce.2024.14.2.582]

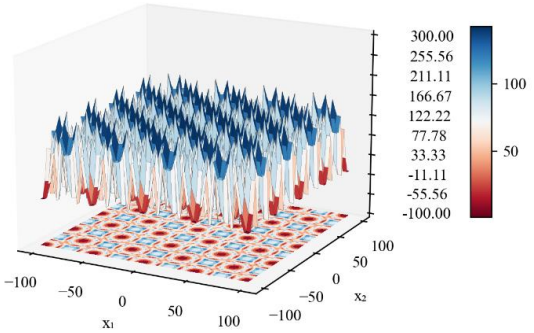


Figure 10. The visual of "Griewank" and its contour lines

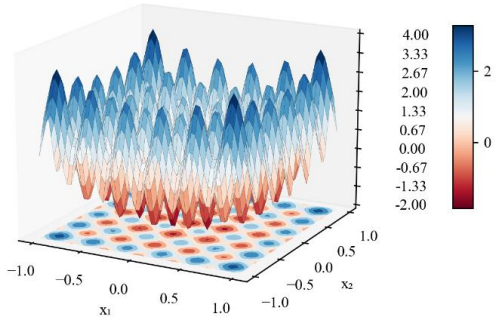


Figure 11. The visual of " Rastrigin " and its contour lines

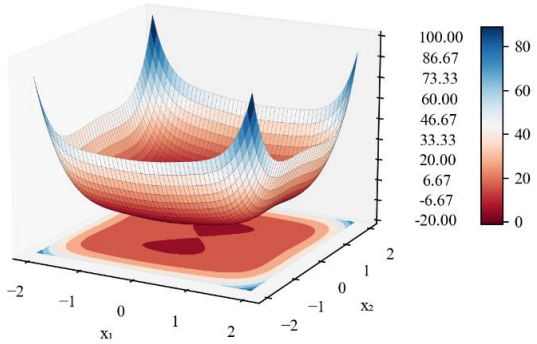


Figure 12. The visual of "Camel" and its contour lines

Table 2. The number of generations in Aluffi-Pentiny

Pop. size	GA	Proposed method			
		FSG=0	*FSG*=1	*FSG*=5	*FSG*=10
10	233	80.4	71	44.4	61.8
20	117	51	44	40	94
50	20	36.8	31	28	30.9
100	16	15.8	13.5	14.6	21

Table 3. The number of generations in Bohachevsky 1

Pop. size	GA	Proposed method			
		FSG=0	*FSG*=1	*FSG*=5	*FSG*=10
10	>50000	34.4	10.2	12.3	17.3
20	125	27.3	11.4	13.1	24.5
50	19	18.3	11.1	10.7	12.9
100	16	12.7	6.4	10.1	11.5

[DOI: 10.22068/ijoce.2024.14.2.582]

Table 4. The number of generations in Bohachevsky 2

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	>50000	27.4	3	19.8	23
20	55	25.7	9.3	13.2	15.1
50	17	19.2	8.4	14.4	14
100	15	11.9	5.3	9.3	9.2

Table 5. The number of generations in Camel

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	106	120.5	163.8	162.1	505.1
20	32	62.3	44.6	87.1	112.8
50	13	39.7	12.1	20.7	32.2
100	12	11.3	9.8	10.5	15

Table 6. The number of generations in Cb3

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	438	19.3	8.1	15.7	19
20	17	13	6.8	17.6	13.1
50	12	9.4	6.3	7.6	7.6
100	10	7.5	6.4	7	6.3

Table 7. The number of generations in Cosine mixture

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	430	17.7	9.6	21.9	28.5
20	297	18.8	11.4	12.2	26
50	43	15.5	7.3	9	10.9
100	38	12.3	7.7	7.2	7.7

Table 8. The number of generations in DeJoung

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	307	15.9	8.3	12.6	19.1
20	207	14.3	5.3	10.1	18.3
50	17	12	5.7	8.4	6.8
100	14	8.9	7.1	7.2	7.7

Table 9. The number of generations in Exponential

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	140	12.5	8.7	16.3	18.7
20	12	10.8	7	10.5	11.6
50	9	8.6	5.7	5.2	7.9
100	7	6.9	6.5	5.1	5.5

[DOI: 10.22068/ijoce.2024.14.2.582]

Table10. The number of generations in Goldstein

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	2047	94	108	70	1
20	995	254	89	33	1
50	17	20	71	20	1
100	12	10	20	14	1

Table11. The number of generations in Griewank

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	203	30	7	11	26
20	90	22	4.9	11.1	14.9
50	29	16	5	11	11.5
100	13	12.7	7.1	7.4	10.2

Table12. The number of generations in Rastrigin

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	1723	18	5.3	19.6	20.7
20	313	13.9	4.3	11.9	17.9
50	17	10.2	6.6	8.1	10.9
100	16	9.5	6.1	7.9	8.8

Table13. The number of generations in Rosenbrock

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	>50000	>50000	25408	8706	6203
20	>50000	>50000	13528	10458	3272
50	9139	12307	9951	2275	3509
100	2475	8152	7952	1884	846s

Table14. The number of generations in Bukin

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	82	75	45	165	745
20	78	71	57	73	211
50	62	203	55	149	371
100	59	102	33	82	221

4.2 Engineering design problems

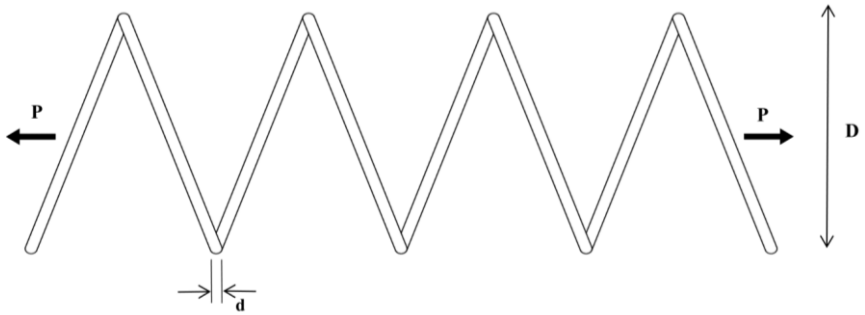
Three engineering design problems in the category of constrained optimization are examined to demonstrate the performance of the proposed algorithm. To manage constraints, a penalty method is adopted in these examples.

[DOI: 10.22068/ijoce.2024.14.2.582]

4.2.1 A tension/compression spring design problem

The objective is to minimize the weight of a tension/compression spring shown in Fig.13 (described in [37] and [38]) with constraints on shear stress, surge frequency, and deflection. The design variables include the mean coil diameter ($D=x_1$), the wire diameter ($d=x_2$), and the number of active coils (x_3). The problem can be defined by the following cost function

$$f_{cost}(\mathbf{X})=(x_3+2)\,x_2\,x_1^2 \tag{7}$$



Figue 13. The tension/compression spring problem

The constraints are expressed as

$$\begin{aligned} g_1(\mathbf{X}) &= 1-\frac{x_2^3\,x_3}{71785\,x_1^4}\le 0 \\ g_2(\mathbf{X}) &= \frac{4\,x_2^2-x_1x_2}{12566\,(x_2\,x_1^3-x_1^4)}+\frac{1}{5108\,x_1^2}-1\le 0 \\ g_3(\mathbf{X}) &= 1-\frac{140.45\,x_1}{x_2^2\,x_3}\le 0 \\ g_4(\mathbf{X}) &= \frac{x_1+x_2}{1.5}-1\le 00 \end{aligned} \tag{8}$$

and the boundaries of design variables are: $0.05\le x_1\le 2$, $0.25\le x_2\le 1.3$, and $2\le x_3\le 15$.

Table 15 compares the results obtained from the proposed method with GA. These results clearly show that by selecting multiple parents and applying multi-crossover, the proposed method converges to the solution within fewer iterations. As shown, by adding fixed stations, the number of required generations is significantly reduced compared to GA. It can be mentioned that the optimal number of fixed station groups in this problem is equal to 1.

[DOI: 10.22068/ijoce.2024.14.2.582]

Fig 14 compares the number of iterations required to converge to the global minimum, between the proposed method and GA. This graph is plotted for *FSG* = 1, which is the optimal value. This figure shows that the presence of fixed stations, which increases the exploration ability, causes the first generation to be closer to the global minimum, and consequently the number of required generations for the rest is reduced.

Table15. The number of generations in the spring design problem

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	5750	10649	4838	7536	6973
20	4624	2348.2	2153	1933	2594
50	3618	2044.2	1550	2536	2874
100	2485	1587	1383	2096	2154.4

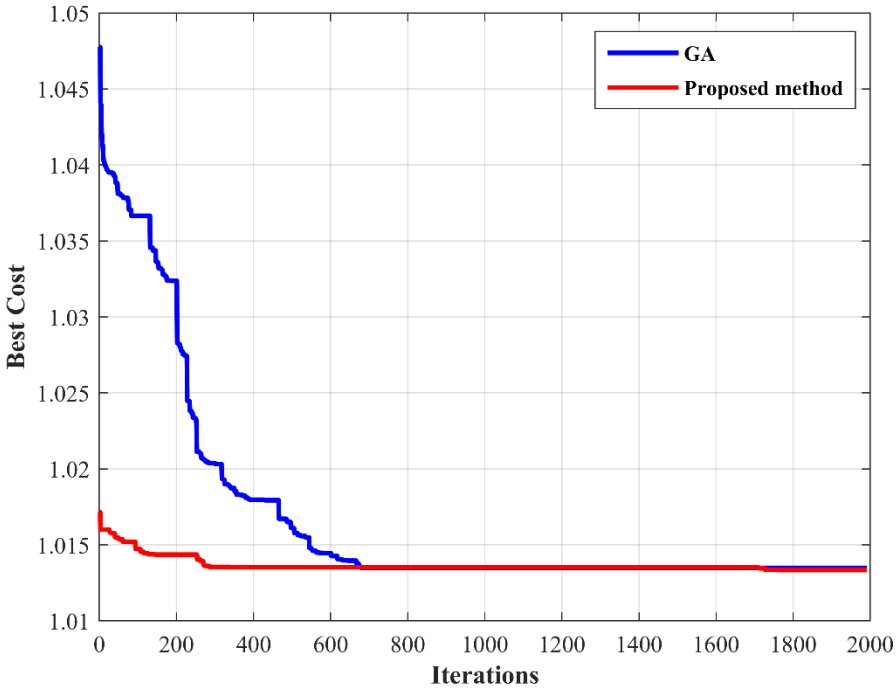


Figure 14. The comparison of required number of generations in spring design problem (*FSG* = 1)

[DOI: 10.22068/ijoce.2024.14.2.582]

4.2.2 A pressure vessel design problem

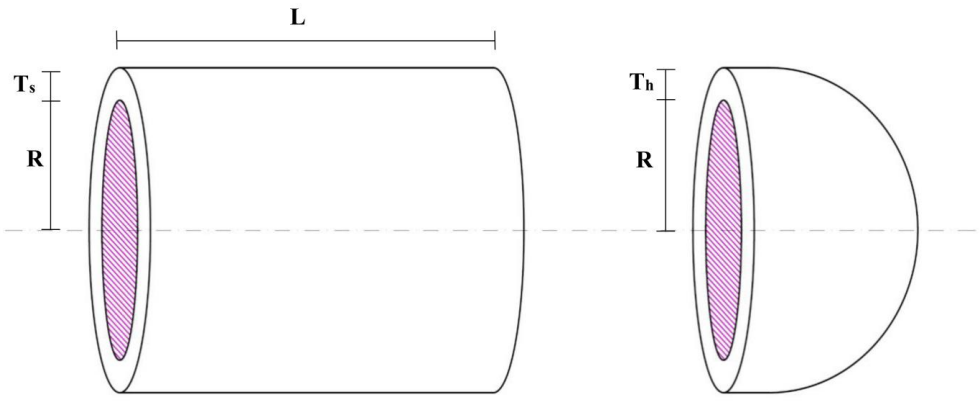
The cylindrical vessel shown in Fig. 15 is capped at both ends by hemispherical heads [39]. The objective in this problem is to minimize the total cost of the welding, material, and forming. This function is formulated by:

$$f_{cost}(\mathbf{X}) = 0.6224\, x_1 x_3 x_4 + 1.7781\, x_2 x_3^2 + 3.1661\, x_1^2 x_4 + 19.84\, x_1^2 x_3 \tag{9}$$

where x_1 represents the shell thickness (T_s), x_2 signifies the head thickness (T_h), x_3 denotes the inner radius (R), and x_4 indicates the length of the cylindrical section excluding the head (L). T_s and T_h are integer multiples of 0.0625 inches. These are the available thickness of rolled steel plates. R and L, on the other hand, are continuous variables. Constraints of the problem are:

$$\begin{aligned} g_1(\mathbf{X}) &= -x_1 + 0.0193\, x_3 \le 0 \\ g_2(\mathbf{X}) &= -x_2 + 0.00954\, x_3 \le 0 \\ g_3(\mathbf{X}) &= -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0 \\ g_4(\mathbf{X}) &= x_4 - 240 \le 00 \end{aligned} \tag{10}$$

and variable bounds are $0 \le x_1 \le 99$, $0 \le x_2 \le 99$, $10 \le x_3 \le 200$, and $10 \le x_4 \le 200$.



Figue 15. The pressure vessel design problem

Table 16 indicates the number of iterations required for convergence to the global minimum in the proposed method and GA. As it can be seen in the table, by using multiple

[DOI: 10.22068/ijoce.2024.14.2.582]

parents, fewer generations are needed to reach the solution. Furthermore, adding fixed stations to the population increases the efficiency of the algorithm significantly. The best convergence occurs at *FSG*=1. In Fig 16, the graph of the number of iterations required to converge to the solution is presented for *FSG*=1. As displayed in the figure, in this specific case GA is trapped in a local minimum. While in the proposed method, in the first generations, there are closer generated points to the global minimum due to the existence of fixed stations. This results in convergence with fewer iterations and escaping local traps.

Table16. The number of generations in the spring design problem

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	10440	6839	3051	>20000	>20000
20	5380	5037	782	1943	5827
50	2036	1741.5	566	739	1294
100	1038	865	543	609	616

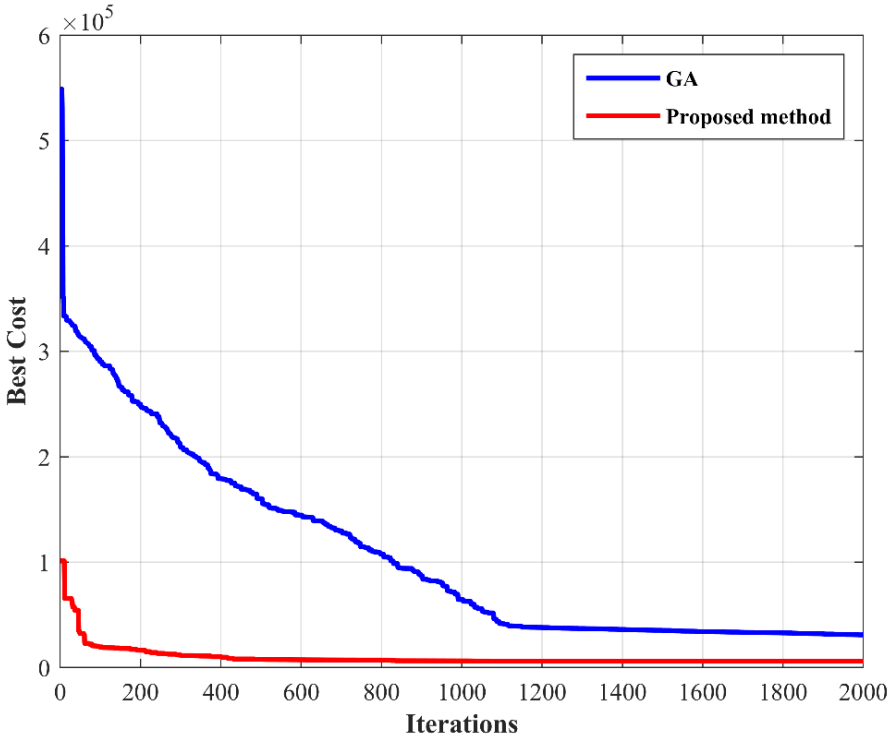


Figure 16. The comparison of required number of generations in spring design problem (*FSG* = 1)

[DOI: 10.22068/ijoce.2024.14.2.582]

4.2.3 A cantilever beam

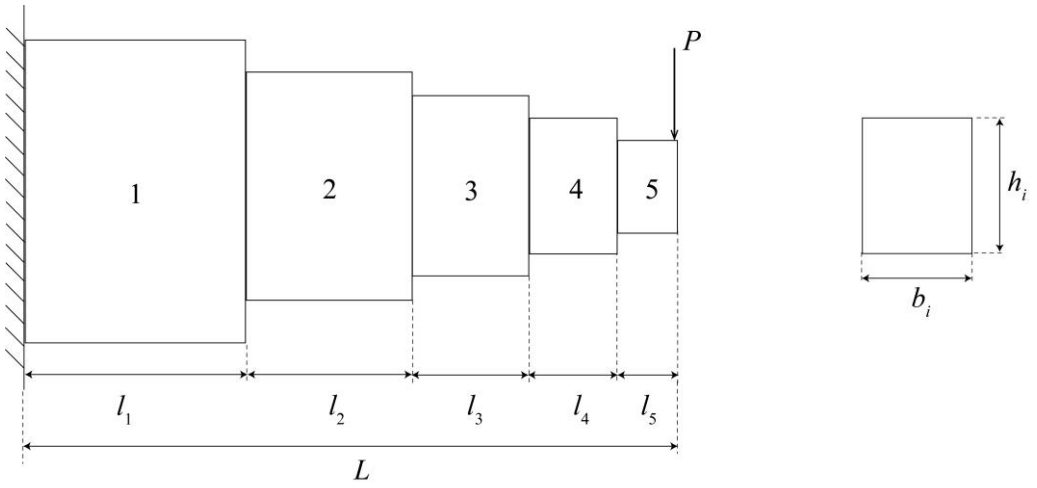
This problem aims to minimize the weight of a cantilever beam composed of five hollow square blocks. The first block has rigid support, and a vertical load is applied on the fifth block, as indicated in Fig. 17. The design variables x_1, x_2, x_3, x_4, x_5 determine the dimensions of the cross-section of the cubes, respectively. The cost function of this problem is defined as

$$f_{\text{cost}}(\mathbf{X}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5) \tag{11}$$

The constraint is:

$$g_1(\mathbf{X}) = \frac{61}{x_1^3} + \frac{27}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0 \tag{12}$$

and ranges of design variables are $0.01 \leq x_{1,2,3,4,5} \leq 100$.



Figue 17. The cantilever beam problem

Table 17 compares the number of iterations required to reach the global optimum between GA and the proposed method. According to the table, the proposed method achieves convergence with fewer iterations, especially at *FSG*=1 which results in the best convergence. This observation is further illustrated in Fig. 18, which shows the number of required iterations for convergence at *FSG*=1. The efficiency of the proposed method is clearly indicated in this figure.

[DOI: 10.22068/ijoce.2024.14.2.582]

Table17. The number of generations in the cantilever beam problem

Pop. size	GA	Proposed method			
		FSG=0	FSG=1	FSG=5	FSG=10
10	36937	29873	25241	26946	33308
20	18804	14507	10373	17420	21304
50	5396	4562	3677	4458	5927
100	3452	1995	1110	1916	2108

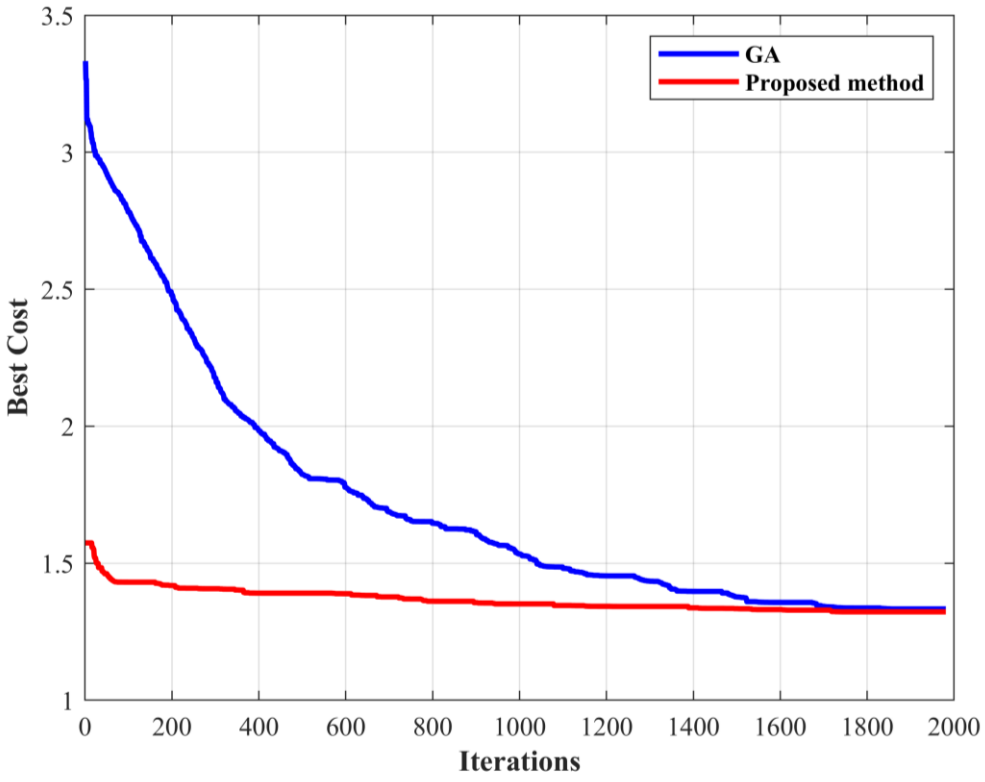


Figure 18. The comparison of required number of generations in the cantilever beam problem (*FSG* = 1)

5. CONCLUSIONS

In this paper, a method is presented to improve the genetic algorithm by increasing the exploration ability. Adding fixed stations to the population ensures more coverage of the entire search space. This technique helps to escape from local minima, especially in multi-modal functions. Furthermore, due to thoroughly exploring the entire search space, it

[DOI: 10.22068/ijoce.2024.14.2.582]

efficiently decreases the required iterations to reach the global minimum. This occurs by producing the first generations closer to the solution. In addition, the proposed method improves exploitation ability by increasing the number of selected parents and generating offspring based on a variable multi-parent compatible crossover. This approach decreases the number of required generations to reach the global minimum. The effectiveness of the proposed method has been validated by using several benchmark functions and engineering constrained optimization problems. Therefore, the presented method leads to solving optimization problems more efficiently by improving genetic algorithm.

REFERENCES

1. Soh CK, Yang J. Fuzzy controlled genetic algorithm search for shape optimization, *J Comput Civ Eng*, 1996; **10**: 143-50.
2. Roudak MA, Karamloo M, Shayanfar MA. A numerical optimization approach for structural reliability analysis using the control parameters in the generalized HLRF method, *Asian J Civ Eng,* 2022; **23**: 1321-42.
3. Chen G, Liang Y, Li S, Xu Z. A novel gradient descent optimizer based on fractional order scheduler and its application in deep neural networks, *Appl Math Model,* 2024; **128**: 26-57.
4. Kirsch U. *Structural optimization: fundamentals and applications*. Springer Science & Business Media; 2012.
5. Roudak MA, Karamloo M, Shayanfar MA. An Iterative Two-step Lagrangian-based Method for Evaluation of Structural Reliability Index, *Period Polytech Civ Eng,* 2022; **66**: 1207-19.
6. Kaveh A. *Applications of metaheuristic optimization algorithms in civil engineering*. Springer; 2017.
7. Saka MP, Hasançebi O, Geem ZW. Metaheuristics in structural optimization and discussions on harmony search algorithm, *Swarm Evol Comput,* 2016; **28**: 88-97.
8. Gen M, Cheng R. *Genetic algorithms and engineering optimization*. John Wiley & Sons; 1999.
9. 9. Kashani AR, Camp CV, Rostamian M, Azizi K, Gandomi AH. Population-based optimization in structural engineering: a review, *Artif Intell Rev,* 2022: 1-108.
10. Abdel-Basset M, Abdel-Fatah L, Sangaiah AK. *Metaheuristic algorithms: A comprehensive review, Computational intelligence for multimedia big data on the cloud with engineering applications,* 2018: 185-231.
11. Dokeroglu T, Sevinc E, Kucukyilmaz T, Cosar A. A survey on new generation metaheuristic algorithms, *Comput Indust Eng,* 2019; **137**: 106040.
12. Abualigah L, Yousri D, Abd Elaziz M, Ewees AA, Al-Qaness MA, Gandomi AH. Aquila optimizer: a novel meta-heuristic optimization algorithm, *Comput Indust Eng,* 2021; **157**: 107250.
13. Yang Y, Chen H, Heidari AA, Gandomi AH. Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts, *Exp SysAppl,* 2021; **177**: 114864.
14. Kashani AR, Chiong R, Dhakal S, Gandomi AH. Investigating bound handling schemes

[DOI: 10.22068/ijoce.2024.14.2.582]

and parameter settings for the interior search algorithm to solve truss problems, *Eng Report*, 2021; **3**: e12405.
15. Holland JH. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*: MIT press; 1992.
16. De Jong KA. *An analysis of the behavior of a class of genetic adaptive systems*: University of Michigan; 1975.
17. Goldberg DE. Cenetic Algorithms in Search, *Optimization, Machine Learning*, 1989.
18. Davis L. Handbook of Genetic Algorithms, Van Nostrand Reinhold, *遺伝的アルゴリズムハンドブック*, 1994.
19. Lagaros ND, Papadrakakis M, Kokossalakis G. Structural optimization using evolutionary algorithms, *Comput struct,*, 2002; **80**: 571-89.
20. Papadrakakis M, Lagaros ND, Thierauf G, Cai J. Advanced solution methods in structural optimization based on evolution strategies, *Eng Comput,* 1998; **15**: 12-34.
21. Adeli H, Cheng N-T. Concurrent genetic algorithms for optimization of large structures, *J Aerosp Eng*, 1994; **7**: 276-96.
22. Sohail A. Genetic algorithms in the fields of artificial intelligence and data sciences, *Ann Data Sci*, 2023; **10**: 1007-18.
23. Gandomi AH, Alavi AH. A new multi-gene genetic programming approach to nonlinear system modeling. Part I: materials and structural engineering problems, *Neu Comput Appl*, 2012; **21**: 171-87.
24. Gandomi AH, Alavi AH, Gandomi M, Kazemi S. Formulation of shear strength of slender RC beams using gene expression programming, part II: With shear reinforcement, *Measurement*, 2017; **95**: 367-76.
25. Yazdani H, Khatibinia M, Gharehbaghi S, Hatami K. Probabilistic Performance-Based Optimum Seismic Design of RC Structures Considering Soil–Structure Interaction Effects, *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 2017; **3**: G4016004.
26. Degertekin SO, Tutar H, Lamberti L. School-based optimization for performance-based optimum seismic design of steel frames, *Eng Comput*, 2021; **37**: 3283-97.
27. Rao RV, Pawar RB. Constrained design optimization of selected mechanical system components using Rao algorithms, *Appl Soft Comput*, 2020; **89**: 106141.
28. Hassan MH, Kamel S, Selim A, Khurshaid T, Domínguez-García JL. A Modified Rao-2 Algorithm for Optimal Power Flow Incorporating Renewable Energy Sources. Mathematics2021.
29. Kaveh A, Zaerreza A. Enhanced Rao Algorithms for Optimization of the Structures Considering the Deterministic, Probabilistic Constraints, *Period Polytech Civ Eng,* 2022; **66**: 694-709.
30. Kaveh A. *Advances in metaheuristic algorithms for optimal design of structures, 3rd edition*. Switzerland: Springer; 2021.
31. Zheng J, Zhong J, Chen M, He K. A reinforced hybrid genetic algorithm for the traveling salesman problem, *Comput Oper Res,* 2023; **157**: 106249.
32. Sun X, Shen W, Vogel-Heuser B. A hybrid genetic algorithm for distributed hybrid blocking flowshop scheduling problem, *J Manufactur Sys,* 2023; **71**: 390-405.
33. Chowdhury D, Hovda S. A hybrid fuzzy logic/genetic algorithm model based on experimental data for estimation of cuttings concentration during drilling, *Geoenergy Sci*

[DOI: 10.22068/ijoce.2024.14.2.582]

and Eng, 2023; **231**: 212387.
34. Ishaque M, Johar MGM, Khatibi A, Yamin M. A novel hybrid technique using fuzzy logic, neural networks and genetic algorithm for intrusion detection system, *Measurement: Sensors*, 2023; **30**: 100933.
35. Eiben P-ER AE, Ruttkay Zs. Genetic algorithms with multi-parent recombination. Proceedings of International Conference on Parallel Problem Solving from Nature, 2005.
36. Tsoulos IG. Modifications of real code genetic algorithm for global optimization, *Applied Mathematics and Computation*, 2008.
37. Belegundu AD. A study of mathematical programming methods for structural optimization. Ph.D. thesis, *Department of Civil and Environmental Engineering, University of Iowa, Iowa, USA*, 1982.
38. Arora JS. *Introduction to Optimum Design*. McGraw-Hill, New York; 1989.
39. Sandgren E. *Nonlinear Integer and Discrete Programming in Mechanical Design*. ASME 1988 Design Technology Conferences1988. p. 95-105.

[DOI: 10.22068/ijoce.2024.14.2.582]