Frequency-constrained optimization of large-scale cyclically symmetric domes using improved hybrid growth optimizer

Ali Kaveh*, Kiarash Biabani Hamedani, Seyed Milad Hosseini

School of Civil Engineering, Iran University of Science and Technology, Tehran, Iran

Abstract

This paper presents an efficient method for frequency-constrained optimization of large-scale cyclically symmetric domes. The approach integrates the improved hybrid growth optimizer (IHGO) algorithm with an eigenvalue decomposition method. IHGO incorporates the exploration mechanism of the improved arithmetic optimization algorithm (IAOA) into its learning phase, along with algorithm-specific modifications. While these modifications are general and problem-independent, their effectiveness in broader structural optimization tasks remains unexplored. To enhance computational efficiency, a decomposition-based method performs free vibration analysis. This method partitions the eigenvalue problem into smaller, decoupled sub-eigenproblems through block-diagonalization of structural matrices, significantly reducing CPU time and memory requirements compared to the standard method (which solves the full eigenvalue problem without decomposition). The performance of IHGO is demonstrated via optimization of two large-scale domes, comparing results against the original growth optimizer (GO) and literature-best solutions. These comparisons highlight the outstanding computational efficiency and accuracy of IHGO. The results confirm the robustness and computational advantages of IHGO, establishing it as a powerful tool for large-scale structural optimization under natural frequency constraints.

Keywords: Metaheuristics; Improved hybrid growth optimizer; Dome structures; Frequency constraints; Free vibration analysis; Block-diagonalization.

1. Introduction

* Corresponding author at: School of Civil Engineering, Iran University of Science and Technology (IUST), Narmak, Tehran, P.O. Box 16846-13114, Iran.
E-mail addresses: alikaveh@iust.ac.ir (A. Kaveh), kiarash_biabani@alumni.iust.ac.ir (K. Biabani Hamedani), hosseini_milad@alumni.iust.ac.ir (S.M. Hosseini).

Natural frequencies, particularly the fundamental frequency and corresponding mode shapes, are critical indicators of a structure's dynamic behavior [1]. Determining the fundamental frequency is essential for preventing resonance due to external excitations, making frequency constraints a key consideration in structural design, especially in aerospace engineering [2]. For instance, to avoid flutter instability, frequency constraints are applied to main structural components and control surfaces in aircraft design [2]. Incorporating such constraints into structural optimization enables better control over dynamic characteristics, allowing engineers to achieve the desired performance more effectively [3].

Over the past few decades, significant research has been dedicated to optimizing structures under natural frequency constraints. Early studies primarily employed mathematical programming [4] and optimality criteria methods [5], but these techniques struggle with highly non-linear, non-convex frequency constraints [6,7]. A major drawback is their sensitivity to the initial search point, often leading to local optima entrapment [6]. Moreover, these methods require gradient-based sensitivity analyses of objective functions and constraints [8], which are computationally expensive and complex [6]. Given these challenges, metaheuristic algorithms have emerged as viable alternatives due to their independence from gradient information and ability to explore complex search spaces effectively [9]. Various metaheuristics have been applied to frequency-constrained structural optimization. Pioneering efforts in this area include studies by Lingyun et al. [6,7] and Gholizadeh et al. [10]. Gomes [11] utilized particle swarm optimization (PSO) for truss sizing and geometry optimization with frequency constraints. Harmony search (HS) and the firefly algorithm (FA) have also been employed for similar problems [12]. Additionally, Kaveh and Zolghadr [13] introduced CSS-BBBC, a hybrid of the big bang-big crunch (BB-BC) and charged system search (CSS) algorithms, for optimizing structures with frequency constraints. However, most of these studies focused on small- to medium-scale problems. In recent years, research has shifted towards large-scale frequency-constrained structural optimization. For instance, Kaveh and Ilchi Ghazaan [14] developed a cascade variant of the enhanced colliding bodies optimization (ECBO) algorithm for optimizing large-scale domes. Several other studies have also contributed to this field [15–23].

Metaheuristic algorithms often encounter high computational costs, primarily due to the numerous evaluations of the objective function [24]. This challenge becomes more significant as the number of design variables increases. In structural optimization, particularly for large-scale structures, a substantial portion of CPU time is consumed by structural analyses. When solving frequency-constrained structural optimization problems using metaheuristic algorithms, multiple free vibration analyses must be performed. In the finite element method (FEM), free vibration analysis results in a large generalized eigenvalue problem, with a size equal to the degrees of freedom of the structure

[25]. Solving this eigenvalue problem provides the natural frequencies and mode shapes of the structure. However, the standard method (i.e., solving the full eigenvalue problem without decomposition) is computationally intensive in both processing time and memory usage [26], mainly due to the large structural matrices involved. The computational resources required heavily depend on the size of these matrices, which scale with the degrees of freedom. As a result, standard eigenvalue solution methods may not be feasible for large-scale frequency-constrained structural optimization problems. This makes it crucial to develop and use efficient computational methods for solving eigenvalue problems. Several efficient techniques have been proposed for the free vibration analysis of regular and symmetric structures [27,28]. Generally, the idea behind these methods is to exploit the inherent properties of structural matrices associated with such structures to reduce computational time and memory requirements. For example, Kaveh and his colleagues [29–31] introduced a computationally efficient method for the free vibration analysis of structures with cyclic symmetry. Their approach employs graph theory to block-diagonalize structural matrices, partitioning the original free vibration eigenvalue problem of a cyclically symmetric structure into smaller, decoupled sub-eigenproblems, each associated with a substructure. This significantly reduces computational time and memory usage.

The growth optimizer (GO) is a recently developed metaheuristic inspired by human learning and reflection strategies [32]. Its search process consists of two main phases: learning, which involves cooperative search, and reflection, which enhances global convergence. GO has demonstrated competitive performance in benchmark and real-world optimization problems [32,33]. However, studies have identified limitations in operator refinement and parameter tuning [34], as well as challenges related to exploration and local optima entrapment [35]. To overcome these drawbacks, Kaveh and Biabani Hamedani [36] introduced the improved hybrid growth optimizer (IHGO) for discrete structural optimization. Their results showed that IHGO significantly outperforms GO in accuracy and efficiency. IHGO incorporates four key improvements: (a) enhanced global search by integrating the exploration mechanism of the improved arithmetic optimization algorithm (IAOA) into the learning phase, (b) a corrected replacement phase that preserves the best solution found so far during the search process, (c) an adaptive hierarchical population structure to reduce sensitivity to population size, and (d) a refined reflection phase to improve exploitation of promising search regions. These modifications are general and independent of specific optimization problems. However, the performance of IHGO in solving other types of structural optimization problems has not been studied yet.

Motivated by the potential applications of IHGO, this study integrates the IHGO algorithm with an eigenvalue decomposition method for the first time to optimize large-scale cyclically symmetric structures under frequency constraints. Initially, further modifications to IHGO were expected to be necessary for this type of structural optimization. However, contrary to expectations, preliminary experiments revealed that IHGO, in its current form, effectively handles frequency-constrained structural optimization problems. This can be attributed to the general nature of the modifications introduced in IHGO. The efficiency and robustness of the proposed approach are demonstrated through two large-scale cyclically symmetric dome optimization problems. The performance of IHGO is evaluated by comparing its results with those obtained using the original GO and other algorithms in the literature. To enhance computational efficiency, free vibration analyses during the optimization process are performed using a block-diagonalization method, which significantly reduces CPU time and memory requirements compared to the standard method.

The remainder of this paper is structured as follows: Section 2 presents the mathematical formulation of the sizing optimization problem for truss structures with frequency constraints. The free vibration analysis of cyclically symmetric structures is expressed in Section 3. A brief description of the original GO algorithm is provided in Section 4, followed by a detailed presentation of the IHGO algorithm in Section 5. Section 6 evaluates the performance of IHGO through two large-scale numerical examples and demonstrates the efficiency and accuracy of the proposed solution method. Finally, Section 7 concludes the paper.

2. Sizing optimization of truss structures with frequency constraints

In truss sizing optimization under multiple frequency constraints, the goal is to determine the optimal member cross-sections that minimize total structural weight while satisfying all specified frequency constraints [37]. In practical applications, structural members are typically grouped, with a single sizing variable assigned to all members within each group. This approach ensures that members in the same group will have identical cross-sections, significantly reducing the number of design variables [38]. The problem is mathematically formulated as follows [39]:

Find $\{A\} = \{A_1, A_2, \ldots, A_d\}$ (1)

to minimize $M\left(\{A\}\right) = \sum_{i=1}^{d} A_i \sum_{j=1}^{m_i} L_{ij}\rho_{ij}$ (2)

subject to $\begin{cases} \omega_n \le \omega_n^{\max} \text{ for some natural frequencies } n \\ \omega_k \ge \omega_k^{\min} \text{ for some natural frequencies } k \end{cases}$ (3)

where $Lb_i \le A_i \le Ub_i;\ i = 1, 2, \ldots, d$ (4)

where $\{A\}$ is the vector of design variables; d denotes the number of member groups; A_i is the cross-sectional area of the members in the i-th member group; $M(\{A\})$ is the objective function to be minimized with respect to $\{A\}$; L_{ij} and ρ_{ij} are the length and material density of the j-th member in the i-th group, respectively; m_i is the number of members of the i-th group; ω_n and ω_k are respectively the n-th and k-th natural frequencies of the truss; $\omega_k^{\min}$ is the lower limit for ω_k; $\omega_n^{\max}$ is the upper limit for ω_n; Ub_i and Lb_i are respectively the upper and lower limits for the cross-sectional area of the members in the i-th member group.

The optimization problem outlined above is a constrained minimization problem with inequality constraints. In the literature, various methods and techniques have been developed to address such constraints in optimization problems, the most common of which is the penalty function method. In this study, a dynamic multiplicative penalty function is employed, defined as follows [40]:

$$f_{penalty}(\{A\}) = (1+\varepsilon_1 \nu)^{\varepsilon_2} \tag{5}$$

where $f_{penalty}(\{A\})$ is the penalty function; ε_1 and ε_2 are parameters determining the extent of penalization for infeasible solutions; ν is the penalty term which regards the constraint violations, and it is defined as:

$$\nu = \sum_{k=1}^{nf_{\min}} \max\left\{0, \frac{\omega_k^{\min} - \omega_k}{\omega_k^{\min}}\right\} + \sum_{n=1}^{nf_{\max}} \max\left\{0, \frac{\omega_n - \omega_n^{\max}}{\omega_n^{\max}}\right\} \tag{6}$$

where $nf_{\min}$ and $nf_{\max}$ are respectively the number of lower- and upper-bounded frequency constraints.

Therefore, the optimization problem can be reformulated using the penalty function method as follows:

$$\text{Find} \quad \{A\} = \{A_1, A_2, \ldots, A_d\} \tag{7}$$

$$\text{to minimize} \quad P(\{A\}) = M(\{A\})\, f_{penalty}(\{A\}) \tag{8}$$

$$\text{where} \quad Lb_i \le A_i \le Ub_i;\ i = 1, 2, \ldots, d \tag{9}$$

where $P(\{A\})$ represents the penalized objective function to be minimized with respect to $\{A\}$.

For all experiments conducted in this study, ε_1 is kept constant, while ε_2 is specified as a linearly increasing function of the number of objective function evaluations, consistent with prior studies [16].

3. Efficient free vibration analysis of cyclically symmetric structures

The governing free vibration equation of motion for an undamped structural system can be expressed as a generalized eigenvalue problem in the following form [41]:

$$K\phi_i = \omega_i^2 M\phi_i \tag{10}$$

Here, M and K represent the mass and stiffness matrices of the structure, respectively. These matrices are known. The problem is to determine the N natural frequencies ω_i ($i = 1, 2, \ldots, N$) and the corresponding N mode shapes ϕ_i, where N denotes the total number of degrees of freedom of the structure.

The above generalized eigenvalue problem can be interpreted as a system of homogeneous algebraic equations with N unknowns ϕ_i. Therefore, we rewrite Eq. (10) as follows:

$$\Omega_i = \left(K - \omega_i^2 M\right)\phi_i = 0 \tag{11}$$

The above system of equations has a nontrivial solution if and only if the determinant of the coefficient matrix $\left(K - \omega_i^2 M\right)$ vanishes:

$$\det\left(K - \omega_i^2 M\right) = 0 \tag{12}$$

where $\det\left(K - \omega_i^2 M\right)$ denotes the determinant of $K - \omega_i^2 M$.

The standard method solves this global eigenvalue problem directly at full scale (order N, where N is the total number of degrees of freedom) without exploiting cyclic symmetry. In contrast, Kaveh and his colleagues [29–31] introduced the block-diagonalization method which exploits structural symmetry to decouple the problem into smaller sub-eigenproblems. Specifically, by block-diagonalizing the matrix $K - \omega_i^2 M$, the frequency equation is partitioned into independent sub-eigenproblems, each corresponding to a substructure. This approach provides considerable advantages in computational efficiency and memory usage.

4. Original growth optimizer (GO)

The details of the original GO are explained in subsequent subsections for minimization problems. It should be noted here that we found some slight differences between the MATLAB source code available from https://github.com/tsingke/Growth-Optimizer and the pseudocode provided by Zhang et al. [32]. Here, the original GO is implemented as described by Zhang et al. [32].

4.1. Initialization

The GO algorithm begins the optimization process with a set of randomly generated solutions, referred to as the initial population of individuals, as represented in Eq. (13):

$$X_{i,j} = Lb_i + rand\left(Ub_i - Lb_i\right);\ i = 1,2,\ldots,N \text{ and } j = 1,2,\ldots,d \tag{13}$$

Here, N stands for the population size; d denotes the number of design variables; $rand$ refers to a uniformly distributed random number between 0 and 1; $X_{i,j}$ indicates the value of the j-th component of the i-th individual.

4.2. Hierarchical population structure

In the original GO, the population is hierarchically structured into three distinct levels using the parameter P_1. Following the suggestion of Zhang et al. [32], P_1 is set to 5. Let GR_i represent the penalized objective function value of the solution represented by the i-th individual, where lower GR indicates better fitness. At the start of each iteration (prior to the learning phase), individuals are sorted in ascending order of GR values. The resulting hierarchy comprises: (a) an upper level containing the leader (best solution at rank 1) and elites (the next $P_1 - 1$ best solutions occupying ranks 2 through P_1); (b) a middle level spanning solutions ranked $P_1 + 1$ to $N - P_1$; and (c) a bottom level consisting of the P_1 worst-performing solutions at ranks at $N - P_1 + 1$ to N.

4.3. Learning phase

Four distinct gaps are defined to contribute to the learning process of each individual i. These gaps are mathematically modelled as shown below:

$$\overrightarrow{Gap}_1 = X_{best} - X_{better};\ \overrightarrow{Gap}_2 = X_{best} - X_{worse};\ \overrightarrow{Gap}_3 = X_{better} - X_{worse};\ \overrightarrow{Gap}_4 = X_{L_1} - X_{L_2} \tag{14}$$

where X_{best} stands for the leader of the population; X_{better} represents a randomly chosen individual from the other $P_1 - 1$ individuals of the upper level (i.e., the elites); X_{worse} denotes a randomly chosen individual from the bottom level (i.e., the P_1 worst-performing individuals in the population); X_{L_1} and X_{L_2} represent two randomly selected individuals different from each other as well as from X_i (i.e., $L_1 \neq L_2 \neq i$).

The difference between the impacts of different gaps is described by a parameter called learning factor (LF):

$$LF_k = \frac{\left\|\overrightarrow{Gap}_k\right\|}{\sum_{k-1}^{4}\left\|\overrightarrow{Gap}_k\right\|};\ k = 1,2,3,4 \tag{15}$$

Here, $\left\|\overrightarrow{Gap_k}\right\|$ denotes the Euclidean norm of the gap k and LF_k is the Euclidean norm of gap k divided by the total sum of the Euclidean norms of all the gaps, and it varies from 0 to 1.

Different individuals exhibit different levels of willingness to learn. This is defined mathematically as:

$$SF_i = \frac{GR_i}{GR_{\max}};\ k = 1, 2, \ldots, N \tag{16}$$

where GR_i represents the penalized objective function value of individual i; $GR_{\max}$ refers to the penalized objective function value of the worst-performing individual within the population; SF_i signifies the self-perception factor of individual i.

The amount of knowledge gained by individual i from the gap k, denoted by $\overrightarrow{KA_k}$, is determined as follows:

$$\overrightarrow{KA_k} = SF_i LF_k \overrightarrow{Gap_k};\ k = 1, 2, 3, 4 \tag{17}$$

where $\overrightarrow{KA_k}$ denotes the amount of knowledge acquired by individual i from gap k.

With the definitions above, we can now formulate the learning process for individual i as:

$$X_i^{It+1} = X_i^{It} + \overrightarrow{KA_1} + \overrightarrow{KA_2} + \overrightarrow{KA_3} + \overrightarrow{KA_4} \tag{18}$$

where X_i^{It} and X_i^{It+1} stand respectively for the current position and the new position of individual i and It denotes the current iteration number. Here, X_i^{It+1} updates the current position X_i^{It} by accumulating knowledge from the four gaps ($\overrightarrow{KA_1}$ to $\overrightarrow{KA_4}$, defined in Eq. (17)). Each $\overrightarrow{KA_k}$ term is scaled by the self-perception factor SF_i (Eq. (16)) and learning factor LF_k (Eq. (15)).

4.4. Reflection phase

The mathematical expression of the reflection phase of each individual i is defined as:

$$X_{i,j}^{It+1} = \begin{cases} \begin{cases} Lb_j + r_4\left(Ub_j - Lb_j\right); & \text{if } r_3 < AF \\ X_{i,j}^{It} + r_5\left(R_j - X_{i,j}^{It}\right); & \text{otherwise} \end{cases}; & \text{if } r_2 < P_3 \\ X_{i,j}^{It}; & \text{otherwise} \end{cases} \tag{19}$$

$$AF = 0.01 + 0.99\left(1 - FEs/MaxFEs\right) \tag{20}$$

where $X_{i,j}^{It}$ and $X_{i,j}^{It+1}$ denote respectively the values of the j-th component of the current position and the new position of individual i; r_2, r_3, r_4, and r_5 denote four random numbers uniformly distributed between 0 and 1; P_3 is a parameter controlling the reflection probability, typically set to

0.3; R denotes an individual chosen randomly from the best $P_1 + 1$ individuals in the population; AF is a function of the current number of objective function evaluations, as defined by Eq. (20).

4.5. Boundary constraints of the search space

During the search process, some components of the new position of an individual may violate the boundary constraints of the search space. To handle this, Eq. (21) is used to guarantee that the new position remains within the search space boundary:

$$X_{i,j}^{It+1} = \begin{cases} Lb_j; \text{ if } X_{i,j}^{It+1} < Lb_j \\ Ub_j; \text{ if } X_{i,j}^{It+1} > Ub_j \end{cases} \tag{21}$$

4.6. Replacement phase

During the replacement phase of the original GO, if the penalized objective function value of the solution corresponding to the new position of an individual is lower than that corresponding to the current position, the individual transitions to the new position. Otherwise, the individual is likely to remain in their current position. Indeed, even if the new position is not better, there is a small probability P_2 that the individual will move to it. Zhang et al. [32] claim the condition $ind(i) \neq ind(1)$ prevents the best solution (rank 1) from being replaced by inferior candidates. However, as will be demonstrated in Subsection 5.2, this formulation does not actually preserve the best solution during the search process. The replacement phase of the original GO is formulated as follows:

$$X_i^{It+1} = \begin{cases} X_i^{It+1}; & \text{if } P\left(\left\{X_i^{It+1}\right\}\right) < P\left(\left\{X_i^{It}\right\}\right) \\ \begin{cases} X_i^{It+1}; \text{ if } r_1 < P_2 \text{ \&\& } ind(i) \neq ind(1) \\ X_i^{It}; \qquad \text{otherwise} \end{cases} & ; \text{ otherwise} \end{cases} \tag{22}$$

where r_1 denotes a random number uniformly distributed between 0 and 1; $ind(i)$ is the rank of individual i when the individuals are arranged in ascending order based on their penalized objective function values; $P\left(\left\{X_i^{It}\right\}\right)$ and $P\left(\left\{X_i^{It+1}\right\}\right)$ are respectively the penalized objective function values corresponding to X_i^{It} and X_i^{It+1}; P_2 is a parameter determining whether a newly generated solution is retained even if it is not better than the current solution. Zhang et al. [32] suggested that the parameter P_2 be set to 0.001. This setting gives a 0.1% chance for inferior solutions to survive, promoting population diversity. The replacement strategy is implemented not only following the learning phase but also following the reflection phase. The pseudocode and flowchart of the original GO are given in Figure 1 and Figure 2, respectively.

5. Improved hybrid GO (IHGO)

In our recent study, we introduced an improved hybrid variant of GO, named IHGO, for discrete sizing optimization problems of skeletal structures [36]. The results revealed that IHGO successfully addresses the limitations of the original GO and delivers significant improvements in both solution accuracy and computational cost through four key modifications introduced by Kaveh and Biabani Hamedani [36]: (a) hybridization with the exploration mechanism of IAOA (Section 5.1) to avoid ineffective search and enhance exploration; (b) correction of the replacement phase to preserve the best solution (Section 5.2), (c) implementation of a population-size-independent hierarchy via $P_1 = round\left(N/4\right)$ (Section 5.3) to reduce performance dependency on population size; (d) refining the reflection phase with re-ranking and revised AF (Section 5.4) to boost exploitation capabilities.

5.1. Hybrid learning phase

In IHGO, to reinforce the exploration ability, Kaveh and Biabani Hamedani [36] incorporated the exploration mechanism of a recently developed improved arithmetic optimization algorithm (IAOA) into the learning phase, as outlined below. In the learning phase of IHGO, if $\sum_{k=1}^{4} \overrightarrow{Gap}_k = 0$ (indicating stagnant search), the exploration mechanism of IAOA is activated to diversify solutions (Eq. (23)). Otherwise, the learning phase of the original GO is performed. The mathematical formula of the exploration mechanism of IAOA is as below [42]:

$$X_{i,j}^{It+1} = \begin{cases} X_{i,j}^{It} \div \left(1+0.5rand\right)\overline{MOP}\left(-1\right)^{randi\left(\left[1,2\right]\right)}; \text{ if } r_2 > 0.5 \\ X_{i,j}^{It}\left(1+0.5rand\right)\overline{MOP}\left(-1\right)^{randi\left(\left[1,2\right]\right)}; \quad \text{otherwise} \end{cases} \tag{23}$$

In the above equation, $randi\left(\left[1,2\right]\right)$ generates a pseudorandom integer scalar between 1 and 2; r_2 and $rand$ is two random number uniformly distributed between 0 and 1; $\overline{MOP}$ represents the following function [42]:

$$\overline{MOP}\left(FEs\right) = \left(1 - FEs/MaxFEs\right)^{randi\left(\left[1,2\right]\right)} \tag{24}$$

5.2. Corrected replacement phase

As stated by Zhang et al. [32], the condition $r_1 < P_2$ && $ind\left(i\right) \neq ind\left(1\right)$ is employed during the replacement phase of GO to achieve two objectives: (a) to give any inferior solution a 0.1% chance of surviving to the next iteration, and (b) to ensure the preservation of the best solution found so far during the replacement phase. The former is fully ensured by the condition $r_1 < P$. The condition $ind(i) \neq ind(1)$, however, is not correctly defined and the latter objective does not follow from it. Indeed, instead of preserving the best solution found so far for the next iteration, the condition $ind(i) \neq ind(1)$ ensures that the best solution found so far cannot be replaced by X_i^{It+1} during the replacement phase. As a result, Eq. (22) could still lead to the loss of the best solution found so far during the search process. In IHGO, to address the weakness described above, the replacement phase was reformulated as follows [36]:

$$X_i^{It+1} = \begin{cases} X_i^{It+1}; & \text{if } P\left(\left\{X_i^{It+1}\right\}\right) < P\left(\left\{X_i^{It}\right\}\right) \\ \begin{cases} X_i^{It+1}; & \text{if } r_1 < P_2 \text{ \&\& } i \neq ind(1) \\ X_i^{It}; & \text{otherwise} \end{cases}; & \text{otherwise} \end{cases} \tag{25}$$

The key correction $i \neq ind(1)$ (replacing $ind(i) \neq ind(1)$ in the original GO) ensures the global best solution is never replaced by inferior candidates, preserving it throughout the optimization process, as is desired.

5.3. Adaptive hierarchical population structure

In GO, as suggested by Zhang et al. [32], the parameter P_1 is fixed at 5, regardless of the population size. As a result, both the bottom and upper levels of the hierarchical population structure always contain 5 individuals, while the size of the middle level is significantly influenced by the population size. This imbalance can affect the distribution of individuals within the hierarchical population structure, potentially leading to performance degradation in GO. To address this, IHGO defines P_1 as a function of the population size [36]:

$$P_1 = round\left(N/4\right) \tag{26}$$

where N represents the population size, and $round$ is a MATLAB function that rounds its input to the nearest integer. Consequently, in IHGO, both the bottom and upper levels of the hierarchical population structure are set to one-quarter of the population size, while the middle level consists of half. This design makes IHGO less sensitive to population size variations than GO.

5.4. Refined reflection phase

In the original GO, solutions are ranked based on their penalized objective function values at the start of each iteration, before the learning phase. This ranking impacts both the learning and reflection phases, but since solutions are updated during the learning phase, the initial rankings may become invalid. Therefore, rankings should be updated after the learning phase to avoid performance degradation in the reflection phase. If rankings are not updated after the learning phase, R, chosen randomly from the top P_1+1 individuals in the population, might not actually belong to the best P_1+1 individuals. IHGO addresses this by re-ranking solutions before the reflection phase to ensure the best P_1+1 individuals are accurately identified [36]. Another difference between the original GO and IHGO is how R is selected: in IHGO, R is chosen randomly from the leader and elites, while in the original GO, it is selected from the best P_1+1 individuals [32]. Additionally, IHGO modifies the function AF, which controls the rate of re-initializations. In the original GO, AF starts from $0.01+0.99\left(1-FEs/MaxFEs\right)\approx 1$ at $FEs=1$ and decreases linearly until reaching 0.01 at $FEs=MaxFEs$. This results in a high number of random re-initializations during the reflection phase, which can lead to inefficiencies like low computational efficiency and slow convergence rate. In IHGO, to resolve this deficiency, AF is reformulated as given in Eq. (27) [36]:

$$AF=0.01+0.09\left(1-FEs/MaxFEs\right) \tag{27}$$

From the above equation, it can clearly be seen that AF starts from $0.01+0.09\left(1-FEs/MaxFEs\right)\approx 0.1$ at $FEs=1$ and decreases linearly until reaching 0.01 at $FEs=MaxFEs$. The updated AF (Eq. (27)) reduces random re-initializations by narrowing its range from [0.01, 0.1] (vs. [0.01, 1] in GO), thus accelerating convergence.

5.5. The IHGO algorithm

In their recent research, Kaveh and Biabani Hamedani [36] incorporated the modifications described in the previous four subsections into the original GO and proposed the IHGO algorithm. Figure 3 and Figure 4 show respectively the pseudocode and flowchart of IHGO, both as described in [36].

6. Results and discussions

This section evaluates the effectiveness of IHGO for sizing optimization of two large-scale domes under frequency constraints: a 600-bar dome with 25 and a 1410-bar dome with 47 design variables. Table 1 summarizes material properties, frequency constraints, and cross-sectional area limits. The results found by IHGO are compared with GO and the best-known literature results. Each algorithm runs independently 20 times with different initial populations. Statistical metrics include the minimum (best weight, also referred to as the optimal weight) and corresponding design variable values (best or optimal design), the maximum (worst weight), the mean weight, and the standard deviation (SD) of the optimized weights over 20 runs. Additionally, the minimum number of objective function evaluations required to obtain the optimal design ($MinFEs$) and the maximum number of objective function evaluations ($MaxFEs$) are provided. The finite element method (FEM) is used for free vibration analyses. Using the FEM code developed for this study, the feasibility of reported designs is verified, and the percentage of constraint violations (CV) is presented. Following Kaveh and Biabani Hamedani [36], population sizes of 20 for GO and 30 for IHGO are used. Other GO parameters are fixed at $P_1 = 5$, $P_2 = 0.001$, and $P_3 = 0.3$, following Zhang et al. [32], while IHGO uses $P_2 = 0.001$ and $P_3 = 0.3$. The termination criterion for both algorithms is the maximum number of objective function evaluations ($MaxFEs$), set to 20000 and 30000 for the first and second design examples, respectively. The CPU time and memory requirements of the block-diagonalization method are compared with the standard method. The algorithms and FEM models are programmed in MATLAB R2016b, and all experiments are performed on an ASUS notebook PC with the following specifications: Intel(R) Core (TM) i5-7200U CPU @ 2.50 GHz 2.71 GHz processor, 8.00 GB RAM, and Windows 10 Enterprise 64-bit Operating System. Note that the results obtained by this study are shown in bold.

6.1. The 600-bar single-layer dome

The first design example involves the sizing optimization of a 600-bar single-layer dome structure illustrated in Figure 5. This dome is cyclically symmetric, consisting of 24 identical substructures. Each substructure includes 9 nodes and 25 members. The Cartesian coordinates of the nodes of the substructure can be found in [15]. The connectivity details are listed in the first column of Table 3. The optimization involves 25 design variables, representing the cross-sectional areas of the members of the substructure. Non-structural masses equal to 100 kg are attached to the free nodes. This optimization problem has been extensively addressed by researchers using a range of metaheuristic algorithms, including both basic and advanced ones [15–22].

The entire structure has a total of 576 degrees of freedom. Using the standard method, each free vibration analysis required during the optimization process involves solving an eigenvalue problem of order 576 to determine the natural frequencies. In contrast, the block-diagonalization method simplifies this process by breaking it into 24 eigenvalue problems, each of order 24, resulting in a substantial reduction in computational time and memory usage. With the block-diagonalization method, a single free vibration analysis takes only 0.0055 seconds, whereas the standard method requires 0.0360 seconds, making the block-diagonalization method approximately 6.49 times faster. Over 20 runs of IHGO, the block-diagonalization method completed the task in 2650 seconds (about

44 minutes), compared to an estimated 15359 seconds (about 256 minutes) for the standard method, which is roughly 5.80 times slower. A comparative analysis of the computational time and memory requirements for both methods is presented in Table 2, with a graphical comparison of computational times shown in Figure 6.

Table 3 presents the optimal designs obtained for the 600-bar dome using the original GO and IHGO, while Table 4 compares their performance against eight state-of-the-art metaheuristic algorithms from the literature. Compared to the original GO, IHGO demonstrates significantly superior performance in both solution accuracy (0.44% lighter best weight, 1.27% lighter mean weight, 3.29% lighter worst weight, and 96.03% reduction in standard deviation) and computational efficiency (0.94% fewer objective function evaluations), validating the effectiveness of its modifications. IHGO achieves the lightest optimal weight (6057.87 kg) among all benchmarked algorithms in Table 4, with weight reductions of 0.01% relative to CGFA [16] (6058.49 kg), 0.03% relative to ED [18] (6059.70 kg), and 0.08% relative to FAFBI [19] (6062.85 kg). Furthermore, IHGO demonstrates superior robustness, achieving the lowest mean weight (6060.99 kg), the lowest worst weight (6063.71 kg), and the lowest standard deviation (1.65 kg) among all benchmarked algorithms. Remarkably, the worst weight found by IHGO is surprisingly lower than the mean weights of the other algorithms. Regarding computational cost, while some algorithms required fewer structural analyses than IHGO (19726) to reach their optimal designs, all produced heavier structures than the 6057.87 kg achieved by IHGO. Specifically, PFJA [17] (8580), IGWO [20] (≤15000), ED [18] (18089), CGFA [16] (≤10000), and FAFBI [19] (≤15000) yielded weight increases of 0.01% to 4.55% relative to that achieved by IHGO. This demonstrates the superior efficiency-accuracy balance of IHGO: it delivers lighter structures despite moderately higher computational demands. Table 4 also provides the first and third natural frequencies of the optimal designs obtained using different metaheuristics, showing no violations of frequency constraints. To verify these designs, the first and third natural frequencies calculated using the FEM model developed in this study are also presented (see rows ω_1^* and ω_3^* in Table 4), rounded to four decimal places. The FEM-derived frequencies closely match those in the literature, with minor differences and negligible constraint violations (see footnotes and CV values in Table 4) likely due to rounding errors in continuous design variables. Figure 7 depicts the optimized weights obtained over 20 runs of GO and IHGO. It is clear that IHGO, in contrast to GO, consistently converged to high-quality solutions, confirming its high robustness. Figure 8 plots the convergence curves of the average results found by GO and IHGO over 20 runs, where IHGO demonstrates a significantly faster convergence rate.

6.2. The 1410-bar double-layer dome

The second design example involves the sizing optimization of a 1410-bar single-layer dome illustrated in Figure 9. The dome is a cyclically symmetric structure, consisting of 30 identical substructures. Each substructure includes 13 nodes and 47 members. The Cartesian coordinates of the nodes of the substructure can be found in [15]. The connectivity details are provided in the first column of Table 6. The optimization involves 47 design variables, representing the cross-sectional areas of the members of the substructure. Non-structural masses equal to 100 kg are attached to the free nodes. This optimization problem has been extensively addressed by researchers using a range of metaheuristic algorithms, including both basic and advanced versions [15,17–23].

The entire structure has a total of 1080 degrees of freedom. Using the standard method, each free vibration analysis during the optimization process requires solving an eigenvalue problem involving a square matrix of order 1080 to determine the natural frequencies. Alternatively, the block-diagonalization method divides the problem into 30 smaller eigenvalue problems, each involving a 36 × 36 matrix. This approach results in substantial savings in both computational time and memory. Specifically, the block-diagonalization method takes just 0.0149 seconds for a typical free vibration analysis, compared to 0.1808 seconds with the standard method, making it approximately 12.13 times faster. For 20 runs of IHGO, the block-diagonalization method required 9474 seconds (about 158 minutes), while the standard method was estimated to take 108084 seconds (about 1801 minutes), which is about 11.41 times longer. A detailed comparison of the computational time and memory usage for both methods is presented in Table 5, and the computational time comparison is shown in Figure 10.

The optimization results for the 1410-bar dome are presented in Tables 6 and 7. Table 6 lists the optimal designs obtained by GO and IHGO, while Table 7 compares their performance with other metaheuristics from the literature. Compared to the original GO algorithm, IHGO achieves markedly better performance in both solution accuracy (5.23% lighter best weight, 9.32% reduction in mean weight, 12.60% lower worst weight, and 92.48% decreased standard deviation) and computational efficiency (2.23% fewer objective function evaluations), demonstrating the efficacy of its algorithmic modifications. Across all evaluated metaheuristics in Table 7, IHGO records the lightest optimal weight (10248.13 kg), outperforming ED [18] (10257.30 kg) by 0.09%, FAFBI [19] (10257.84 kg) by 0.09%, and Rao-2 [22] (10281.06 kg) by 0.32%. IHGO also exhibits superior robustness, achieving the lowest mean (10276.16 kg) and worst (10351.57 kg) weights. In terms of computational cost, IHGO required 28965 structural analyses to obtain its optimal design. Although algorithms such as PFJA [17] (16900), ISMA [15] (20000), IGWO [20] (≤15000), and FAFBI [19] (≤20000) consumed fewer structural analyses to reach their optimal designs, their resulting structures were

0.09% to 0.99% heavier than that achieved by IHGO. Table 7 also provides the first and third natural frequencies of the optimal designs obtained using different metaheuristics, confirming that all satisfy frequency constraints. FEM simulation results further validate these findings (see rows ω_1^* and ω_3^* in Table 7), although a minor frequency constraint violation was observed in the Chaotic WSA [21] design, likely due to rounding of continuous variables. Such rounding errors also explain slight differences between FEM results and values reported in the literature. Figure 11 illustrates the optimized weights over 20 independent runs, showing that IHGO consistently converged to superior solutions, while GO often produced suboptimal results. Figure 12 depicts the convergence curves, demonstrating that IHGO achieves faster convergence compared to GO.

7. Conclusion

This paper has presented the successful application of the recently developed improved hybrid growth optimizer (IHGO), combined with an exact decomposition-based method, for the free vibration analysis and design optimization of cyclically symmetric domes under frequency constraints. Originally developed for discrete sizing optimization of skeletal structures, IHGO has already demonstrated competitive performance. The key modifications introduced in IHGO include the incorporation of the exploration mechanism of the improved arithmetic optimization algorithm (IAOA) into its learning phase, as well as several algorithm-specific adjustments. These modifications are general in nature, making IHGO adaptable to a wide range of structural optimization problems, though its application to other tasks has yet to be explored.

This study has focused on demonstrating the effectiveness and applicability of IHGO in solving large-scale frequency-constrained structural optimization problems. Both the original growth optimizer (GO) and IHGO were applied to the sizing optimization of two large-scale cyclically symmetric structures: a 600-bar and a 1410-bar dome. These two structures were selected to highlight the scalability of the algorithms in frequency-constrained structural optimization. The results were compared against each other and with the best-known literature results, marking the first time these algorithms have been used in such an application. For a fair comparison, results from basic and advanced algorithms were evaluated separately.

The numerical results have shown that IHGO significantly outperforms the original GO in both accuracy and efficiency, validating the effectiveness of its modifications. Additionally, IHGO has demonstrated highly competitive or superior performance, especially regarding solution accuracy and robustness, compared to other basic and advanced algorithms in the literature. These findings suggest that IHGO is a robust and effective optimization tool for frequency-constrained truss optimization problems.

To conduct the free vibration analyses required during optimization, an exact graph-theoretic decomposition method was utilized. By block-diagonalizing the mass and stiffness matrices of cyclically symmetric structures, this method broke down the free vibration eigenvalue problem of the entire structure into smaller, independent eigenvalue problems corresponding to substructures. The results showed substantial reductions in CPU time and memory requirements compared to the standard method. Specifically, using the standard method, optimizing the 600-bar and 1410-bar domes required approximately 256 minutes and 1801 minutes, respectively. In contrast, the block-diagonalization method reduced these times to just 45 minutes and 158 minutes, making the process 5.80 and 11.41 times faster, respectively.

Declaration of Competing Interest

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

Acknowledgements

This work is based upon research funded by Iran National Science Foundation (INSF) under project No. 4024911.

Data availability

All MATLAB source codes related to this study, including implementations of the original GO, the proposed IHGO, the eigenvalue decomposition method for cyclically symmetric structures, and optimization case studies for both benchmarked domes, are freely available at https://github.com/K-BiabaniHamedani/IHGO.

References

1. Miura, H. and Schmit Jr, L.A. “Second order approximation of natural frequency constraints in structural synthesis”, *Int. J. Numer. Methods Eng.*, **13**(2), pp. 337–351 (1978). https://doi.org/10.1002/nme.1620130209
2. Turner, M.J. “Design of minimum mass structures with specified natural frequencies”, *AIAA J.*, **5**(3), pp. 406–412 (1967). https://doi.org/10.2514/3.3994
3. Grandhi, R.V. “Structural optimization with frequency constraints - A review”, *AIAA J.*, **31**(12), pp. 2296–2303 (1993). https://doi.org/10.2514/3.11928

4. Woo, T.H. "Space frame optimization subject to frequency constraints", *AIAA J.*, **25**(10), pp. 1396–1404 (1987). https://doi.org/10.2514/3.9795
5. Kiusalaas, J. and Shaw, R.C. "An algorithm for optimal structural design with frequency constraints", *Int. J. Numer. Methods Eng.*, **13**(2), pp. 283–295 (1978). https://doi.org/10.1002/nme.1620130206
6. Lingyun, W., Mei, Z., Guangming, W., et al. "Truss optimization on shape and sizing with frequency constraints based on genetic algorithm", *Comput. Mech.*, **35**, pp. 361–368 (2005). https://doi.org/10.1007/s00466-004-0623-8
7. Wei, L., Tang, T., Xie, X., et al. "Truss optimization on shape and sizing with frequency constraints based on parallel genetic algorithm", *Struct. Multidiscip. Optim.*, **43**, pp. 665–682 (2011). https://doi.org/10.1007/s00158-010-0600-0
8. Song, J.D., Yang, B.S., Choi, B.G., et al. "Optimum design of short journal bearings by enhanced artificial life optimization algorithm", *Tribol. Int.*, **38**(4), pp. 403–412 (2005). https://doi.org/10.1016/j.triboint.2003.10.008
9. Ontiveros-Pérez, S.P., Miguel, L.F., and Riera, J.D. "Reliability-based optimum design of passive friction dampers in buildings in seismic regions", *Eng. Struct.*, **190**, pp. 276–284 (2019). https://doi.org/10.1016/j.engstruct.2019.04.021
10. Gholizadeh, S., Salajegheh, E., and Torkzadeh, P. "Structural optimization with frequency constraints by genetic algorithm using wavelet radial basis function neural network", *J. Sound Vib.*, **312**(1–2), pp. 316–331 (2008). https://doi.org/10.1016/j.jsv.2007.10.050
11. Gomes, H.M. "Truss optimization with dynamic constraints using a particle swarm algorithm", *Expert Syst. Appl.*, **38**(1), pp. 957–968 (2011). https://doi.org/10.1016/j.eswa.2010.07.086
12. Miguel, L.F. and Miguel, L.F. "Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms", *Expert Syst. Appl.*, **39**(10), pp. 9458–9467 (2012). https://doi.org/10.1016/j.eswa.2012.02.113
13. Kaveh, A. and Zolghadr, A. "Truss optimization with natural frequency constraints using a hybridized CSS-BBBC algorithm with trap recognition capability", *Comput. Struct.*, **102**, pp. 14–27 (2012). https://doi.org/10.1016/j.compstruc.2012.03.016
14. Kaveh, A. and Ilchi Ghazaan, M. "Optimal design of dome truss structures with dynamic frequency constraints", *Struct. Multidiscip. Optim.*, **53**, pp. 605–621 (2016). https://doi.org/10.1007/s00158-015-1357-2
15. Kaveh, A., Biabani Hamedani, K., and Kamalinejad, M. "Improved slime mould algorithm with elitist strategy and its application to structural optimization with natural frequency constraints", *Comput. Struct.*, **264**, pp. 106760 (2022). https://doi.org/10.1016/j.compstruc.2022.106760

16. Kaveh, A. and Javadi, S.M. “Chaos-based firefly algorithms for optimization of cyclically large-size braced steel domes with multiple frequency constraints”, *Comput. Struct.*, **214**, pp. 28–39 (2019). https://doi.org/10.1016/j.compstruc.2019.01.006

17. Degertekin, S.O., Bayar, G.Y., and Lamberti, L. “Parameter free Jaya algorithm for truss sizing-layout optimization under natural frequency constraints”, *Comput. Struct.*, **245**, pp. 106461 (2021). https://doi.org/10.1016/j.compstruc.2020.106461

18. Truong, D.N. and Chou, J.S. “Metaheuristic algorithm inspired by enterprise development for global optimization and structural engineering problems with frequency constraints”, *Eng. Struct.*, **318**, pp. 118679 (2024). https://doi.org/10.1016/j.engstruct.2024.118679

19. Truong, D.N. and Chou, J.S. “Fuzzy adaptive forensic-based investigation algorithm for optimizing frequency-constrained structural dome design”, *Math. Comput. Simul.*, **210**, pp. 473–531 (2023). https://doi.org/10.1016/j.matcom.2023.03.007

20. Abbasi, M. and Zakian, P. “Optimal design of truss domes with frequency constraints using seven metaheuristic algorithms incorporating a comprehensive statistical assessment”, *Mech. Adv. Mater. Struct.*, **31**(30), pp. 12533–12559 (2024). https://doi.org/10.1080/15376494.2024.2325662

21. Kaveh, A., Amirsoleimani, P., Dadras Eslamlou, A., et al. “Frequency-constrained optimization of large-scale dome-shaped trusses using chaotic water strider algorithm”, *Structures*, **32**, pp. 1604–1618 (2021). https://doi.org/10.1016/j.istruc.2021.03.033

22. Dede, T., Atmaca, B., Grzywinski, M., et al. “Optimal design of dome structures with recently developed algorithm: Rao series”, *Structures*, **42**, pp. 65–79 (2022). https://doi.org/10.1016/j.istruc.2022.06.010

23. Van, T.H., Tangaramvong, S., and Gao, W. “Chaotic heterogeneous comprehensive learning PSO method for size and shape optimization of structures”, *Eng. Appl. Artif. Intell.*, **126**, p. 107014 (2023). https://doi.org/10.1016/j.engappai.2023.107014

24. Ekinci, Y.L., Balkaya, Ç., Göktürkler, G., et al. “Model parameter estimations from residual gravity anomalies due to simple-shaped sources using differential evolution algorithm”, *J. Appl. Geophys.*, **129**, pp. 133–147 (2016). https://doi.org/10.1016/j.jappgeo.2016.03.040

25. Yin, J., XU, L., Wang, H., et al. “Accurate and fast three-dimensional free vibration analysis of large complex structures using the finite element method”, *Comput. Struct.*, **221**, pp. 142–156 (2019). https://doi.org/10.1016/j.compstruc.2019.06.002

26. Xiao, J., Zhou, H., Zhang, C., et al. “Solving large-scale finite element nonlinear eigenvalue problems by resolvent sampling based Rayleigh-Ritz method”, *Comput. Mech.*, **59**, pp. 317–334 (2017). https://doi.org/10.1007/s00466-016-1353-4

27. Dong, B. and Parker, R.G. “Sector-model subspace iteration for vibration of multi-stage, cyclically symmetric systems”, *J. Sound Vib.*, **544**, p. 117378 (2023). https://doi.org/10.1016/j.jsv.2022.117378

28. Zingoni, A. “On group-theoretic eigenvalue vibration analysis of structural systems with C6v symmetry”, *J. Sound Vib.*, **589**, p. 118608 (2024). https://doi.org/10.1016/j.jsv.2024.118608

29. Koohestani, K. and Kaveh, A. “Efficient buckling and free vibration analysis of cyclically repeated space truss structures”, *Finite Elem. Anal. Des.*, **46**(10), pp. 943–948 (2010). https://doi.org/10.1016/j.finel.2010.06.009

30. Kaveh, A. and Nemati, F. “Eigensolution of rotationally repetitive space structures using a canonical form”, *Int. J. Numer. Methods Biomed. Eng.*, **26**(12), pp. 1781–1796 (2010). https://doi.org/10.1002/cnm.1265

31. Kaveh, A. and Rahami, H. “Block circulant matrices and applications in free vibration analysis of cyclically repetitive structures”, *Acta Mech.*, **217**(1), pp. 51–62 (2011). https://doi.org/10.1007/s00707-010-0382-x

32. Zhang, Q., Gao, H., Zhan, Z.H., et al. “Growth Optimizer: A powerful metaheuristic algorithm for solving continuous and discrete global optimization problems”, *Knowl.-Based Syst.*, **261**, p. 110206 (2023). https://doi.org/10.1016/j.knosys.2022.110206

33. Aribia, H.B., El-Rifaie, A.M., Tolba, M.A., et al., “Growth optimizer for parameter identification of solar photovoltaic cells and modules”, *Sustainability*, **15**(10), p. 7896 (2023). https://doi.org/10.3390/su15107896

34. Gao, H., Zhang, Q., Bu, X., et al. “Quadruple parameter adaptation growth optimizer with integrated distribution, confrontation, and balance features for optimization”, *Expert Syst. Appl.*, **235**, p. 121218 (2024). https://doi.org/10.1016/j.eswa.2023.121218

35. Ozkaya, B. “Enhanced growth optimizer algorithm with dynamic fitness-distance balance method for solution of security-constrained optimal power flow problem in the presence of stochastic wind and solar energy”, *Appl. Energy*, **368**, p. 123499 (2024). https://doi.org/10.1016/j.apenergy.2024.123499

36. Kaveh, A. and Biabani Hamedani, K. “A hybridization of growth optimizer and improved arithmetic optimization algorithm and its application to discrete structural optimization”, *Comput. Struct.*, **303**, p. 107496 (2024). https://doi.org/10.1016/j.compstruc.2024.107496

37. Nguyen-Van, S., Nguyen, K.T., Luong, V.H., et al. “A novel hybrid differential evolution and symbiotic organisms search algorithm for size and shape optimization of truss structures under multiple frequency constraints”, *Expert Syst. Appl.*, **184**, p. 115534 (2021) https://doi.org/10.1016/j.eswa.2021.115534

38. Cucuzza, R., Aloisio, A., Rad, M.M., et al. "Constructability-based design approach for steel structures: From truss beams to real-world inspired industrial buildings", *Autom. Constr.*, **166**, p. 105630 (2024). https://doi.org/10.1016/j.autcon.2024.105630

39. Tang, H., Huynh, T.N., and Lee, J. "A novel adaptive 3-stage hybrid teaching-based differential evolution algorithm for frequency-constrained truss designs", *Structures*, **38**, pp. 934–948 (2022). https://doi.org/10.1016/j.istruc.2022.02.035

40. Kaveh, A. and Talatahari, S. "Optimum design of skeletal structures using imperialist competitive algorithm", *Comput. Struct.*, **88**(21–22), pp. 1220–1229 (2010). https://doi.org/10.1016/j.compstruc.2010.06.011

41. Fan, W. and Qiao, P. "A strain energy-based damage severity correction factor method for damage identification in plate-type structures", *Mech. Syst. Signal Process.*, **28**, pp. 660–678 (2012). https://doi.org/10.1016/j.ymssp.2011.11.010

42. Kaveh, A. and Biabani Hamedani, K. "Improved arithmetic optimization algorithm and its application to discrete structural optimization", *Structures*, **35**, pp. 748–764 (2022). https://doi.org/10.1016/j.istruc.2021.11.012

Authors' Biographies

Ali Kaveh is a distinguished professor of structural engineering at Iran University of Science and Technology. He obtained his PhD from Imperial College, London in 1974. He has been teaching at IUST, TU Wien, Sharif University and some other universities in Iran. He is a fellow of the Iranian Academy of Sciences, fellow of The World Academy of Sciences (TWAS), fellow of the European Academy of Sciences and Arts, and member of 8 scientific societies, founder and editor of more than 5 scientific journals and advisory member of many journals and conferences. He has published 800 scientific journal papers, 160 conference papers, 23 books in Farsi and 18 books in English published by Springer, Wiley and Research Studies Press.

Kiarash Biabani Hamedani received his BSc in civil engineering from Bu-Ali Sina University, his MSc in structural engineering from Shahrood University of Technology (SUT), and his PhD in structural engineering from Iran University of Science and Technology (IUST) under the supervision of Prof. Ali Kaveh. He is currently a postdoctoral fellow at the School of Civil Engineering, IUST, funded by the Iran National Science Foundation (INSF). He has co-authored a book published by Springer and several research papers in leading international journals. His research interests include metaheuristic algorithms, structural optimization, and the application of artificial intelligence in structural engineering.

Seyed Milad Hosseini received his BSc from the University of Zanjan and his MSc from Iran University of Science and Technology (IUST). Since completing his master’s, he has been collaborating on research projects in the fields of structural health monitoring (SHM) and structural optimization under the supervision of Prof. Ali Kaveh. His research interests include metaheuristic algorithms, evolutionary optimization, structural damage detection, and the application of artificial intelligence in civil engineering.

Figure captions

Figure 1. Pseudocode of the original GO [32].

Figure 2. Flowchart of the original GO.

Figure 3. Pseudocode of the IHGO algorithm [36].

Figure 4. Flowchart of the IHGO algorithm.

Figure 5. A 600-bar dome: (a) three-dimensional view, (b) top view, and (c) a typical substructure.

Figure 6. Graphical comparison of computational times (600-bar dome problem).

Figure 7. Optimized weights obtained over 20 runs of GO and IHGO for the 600-bar dome.

Figure 8. Convergence curves for GO and IHGO (600-bar dome problem).

Figure 9. A 1410-bar dome: (a) three-dimensional view, (b) top view, and (c) a typical substructure.

Figure 10. Graphical comparison of computational times (1410-bar dome problem).

Figure 11. Optimized weights obtained over 20 runs of GO and IHGO for the 1410-bar dome.

Figure 12. Convergence curves for GO and IHGO (1410-bar dome problem).

Table captions

Table 1. Cross-sectional area limits, material properties, and frequency constraints for the design examples.

Table 2. Comparative analysis of computational time and memory requirements (600-bar dome problem).

Table 3. Optimal designs (cm^2) of GO and IHGO for the 600-bar dome.

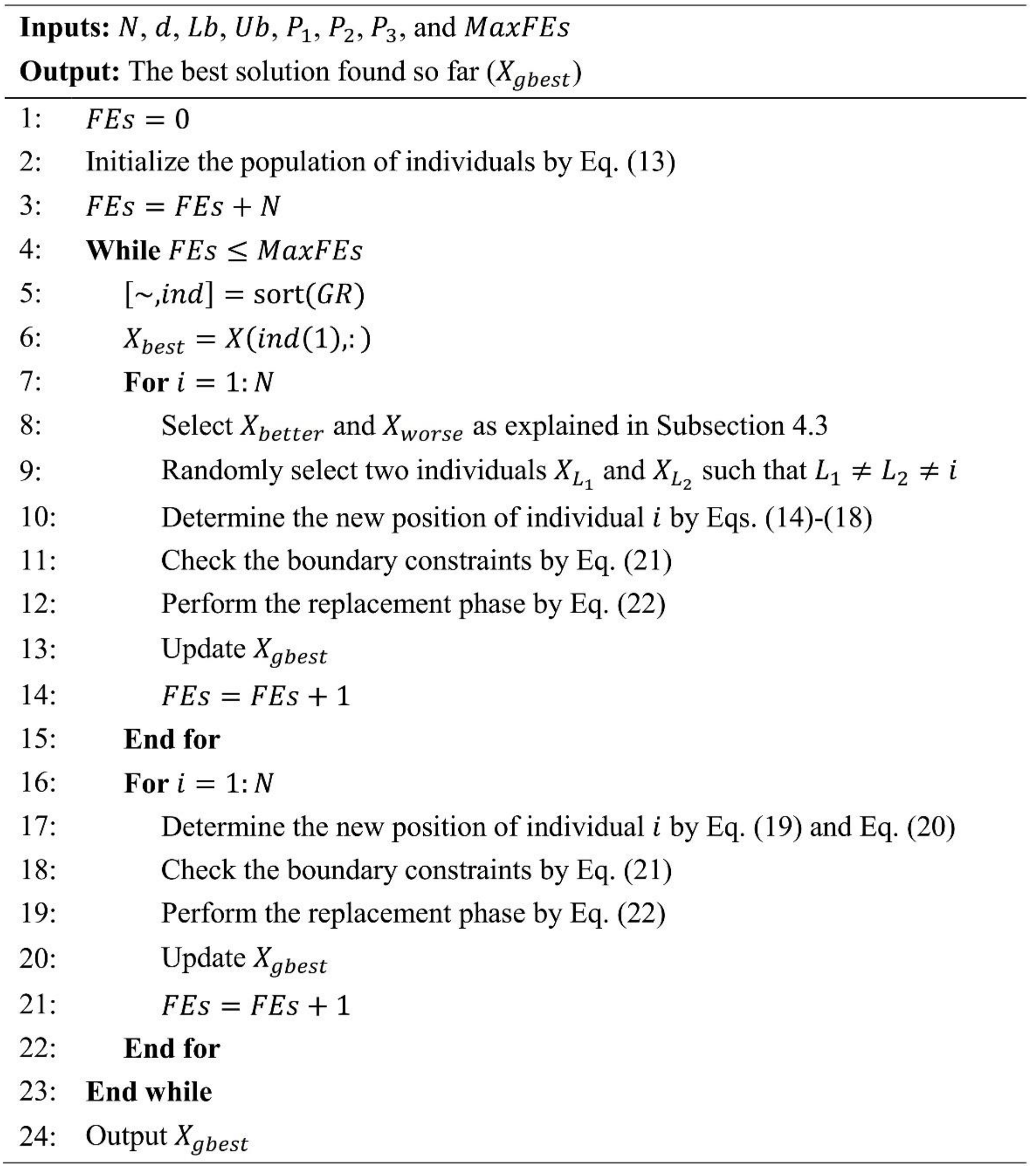
Table 4. Optimal design results for the 600-bar dome achieved by various metaheuristics.

Table 5. Comparative analysis of computational time and memory requirements (1410-bar dome problem).

Table 6. Optimal designs (cm^2) of GO and IHGO for the 1410-bar dome.

Table 7. Optimal design results for the 1410-bar dome achieved by various metaheuristics.

Figures



```
Inputs: $N$, $d$, $Lb$, $Ub$, $P_1$, $P_2$, $P_3$, and $MaxFEs$
Output: The best solution found so far ($X_{gbest}$)

$FEs = 0$
Initialize the population of individuals by Eq. (13)
$FEs = FEs + N$
While $FEs \leq MaxFEs$
    $[\sim, ind] = \mathrm{sort}(GR)$
    $X_{best} = X(ind(1),:)$
    For $i = 1:N$
        Select $X_{better}$ and $X_{worse}$ as explained in Subsection 4.3
        Randomly select two individuals $X_{L_1}$ and $X_{L_2}$ such that $L_1 \neq L_2 \neq i$
        Determine the new position of individual $i$ by Eqs. (14)-(18)
        Check the boundary constraints by Eq. (21)
        Perform the replacement phase by Eq. (22)
        Update $X_{gbest}$
        $FEs = FEs + 1$
    End for
    For $i = 1:N$
        Determine the new position of individual $i$ by Eq. (19) and Eq. (20)
        Check the boundary constraints by Eq. (21)
        Perform the replacement phase by Eq. (22)
        Update $X_{gbest}$
        $FEs = FEs + 1$
    End for
End while
Output $X_{gbest}$
```

Figure 1

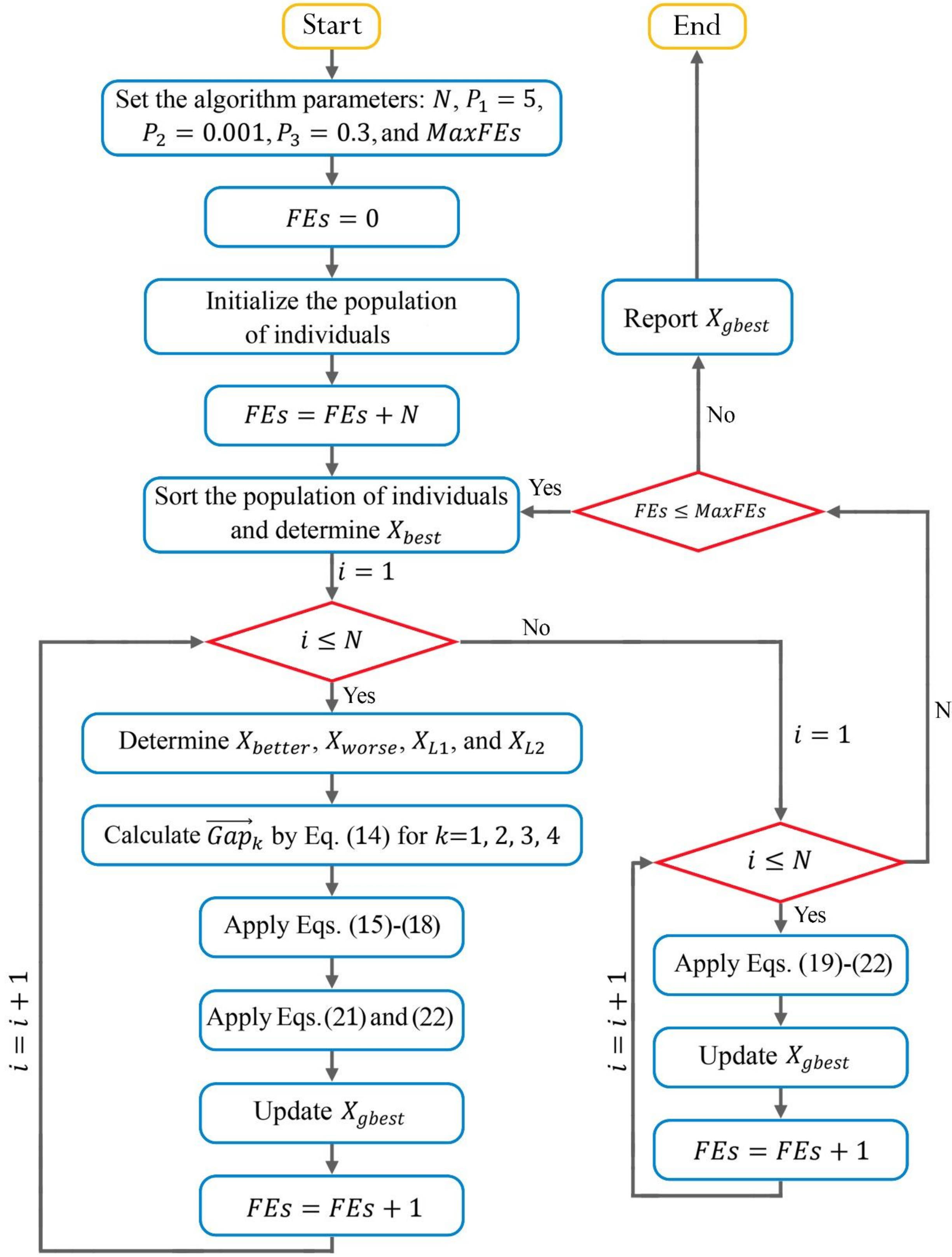

Start
Set the algorithm parameters: N, $P_1 = 5$, $P_2 = 0.001$, $P_3 = 0.3$, and $MaxFEs$
$FEs = 0$
Initialize the population of individuals
$FEs = FEs + N$
Sort the population of individuals and determine X_{best}
$i = 1$
$i \leq N$
Yes
Determine X_{better}, X_{worse}, X_{L1}, and X_{L2}
Calculate $\overrightarrow{Gap_k}$ by Eq. (14) for k=1, 2, 3, 4
Apply Eqs. (15)-(18)
Apply Eqs.(21) and (22)
Update X_{gbest}
$FEs = FEs + 1$
$i = i + 1$
No
$i = 1$
$i \leq N$
Yes
Apply Eqs. (19)-(22)
Update X_{gbest}
$FEs = FEs + 1$
$i = i + 1$
No
$FEs \leq MaxFEs$
Yes
No
Report X_{gbest}
End

Figure 2

Inputs: N, d, Lb, Ub, $P_1 = round(N/4)$, P_2, P_3, and $MaxFEs$

Output: The best solution found so far (X_{gbest})

```
$FEs = 0$
Initialize the population of individuals by Eq. (13)
$FEs = FEs + N$
While $FEs \leq MaxFEs$
    $[\sim, ind] = \text{sort}(GR)$
    $X_{best} = X(ind(1), :)$
    For $i = 1{:}N$
        Select $X_{better}$ and $X_{worse}$ as explained in Subsection 4.3
        Randomly select two individuals $X_{L_1}$ and $X_{L_2}$ such that $L_1 \neq L_2 \neq i$
        Calculate $\overrightarrow{Gap_k}$ for $k$=1, 2, 3, 4 by Eq. (14)
        If $\sum_{k=1}^{4} \overrightarrow{Gap_k} = 0$
            Determine the new position of individual $i$ by Eq. (23) and Eq. (24)
        Else
            Determine the new position of individual $i$ by Eqs. (15)-(18)
        End if
        Check the boundary constraints by Eq. (21)
        Perform the replacement phase by Eq. (25)
        Update $X_{gbest}$
        $FEs = FEs + 1$
    End for
    $[\sim, ind] = \text{sort}(GR)$
    For $i = 1{:}N$
        Determine the new position of individual $i$ by Eq. (19) and Eq. (27)
        Check the boundary constraints by Eq. (21)
        Perform the replacement phase by Eq. (25)
        Update $X_{gbest}$
        $FEs = FEs + 1$
    End for
End while
Output $X_{gbest}$
```

Figure 3

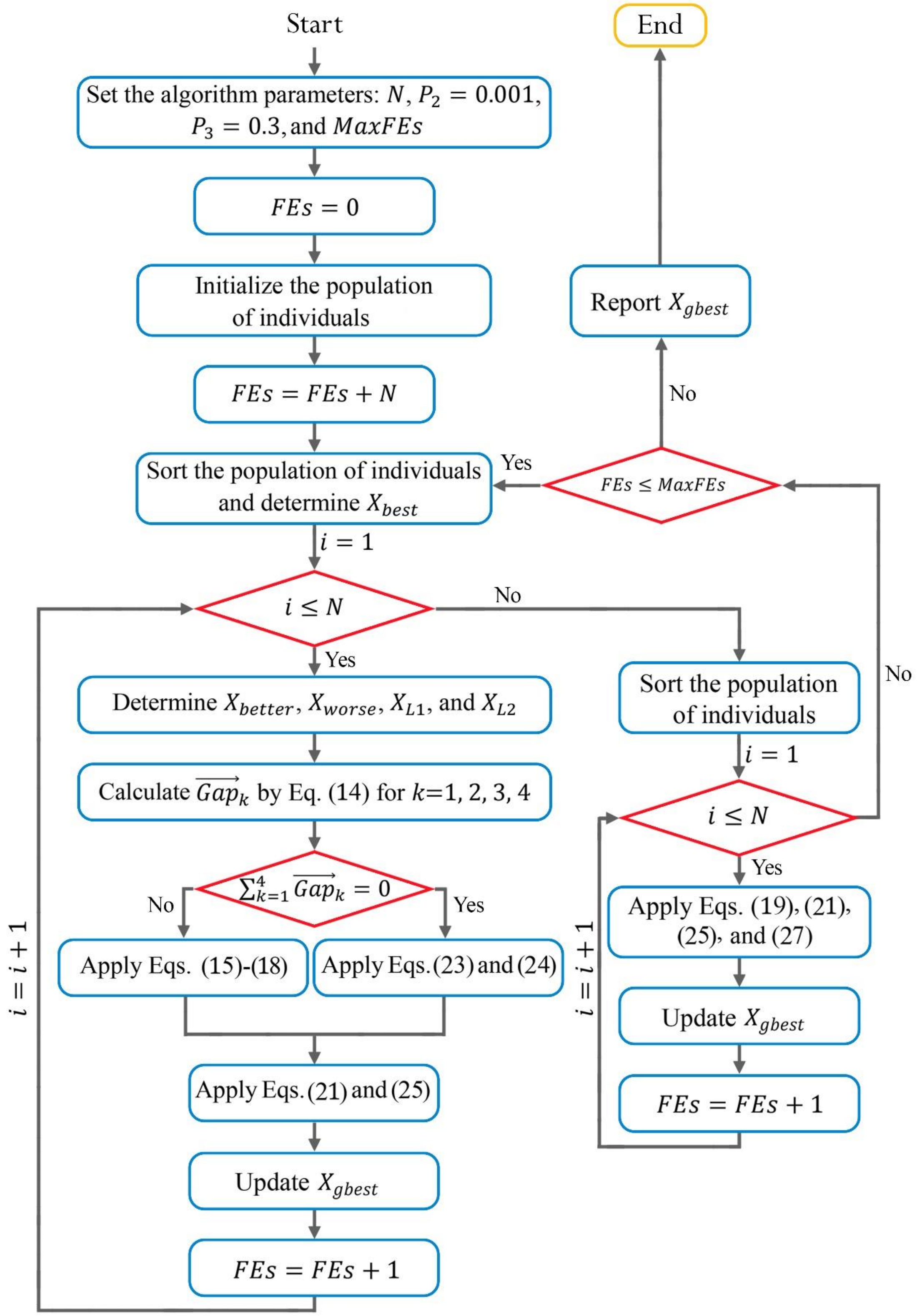
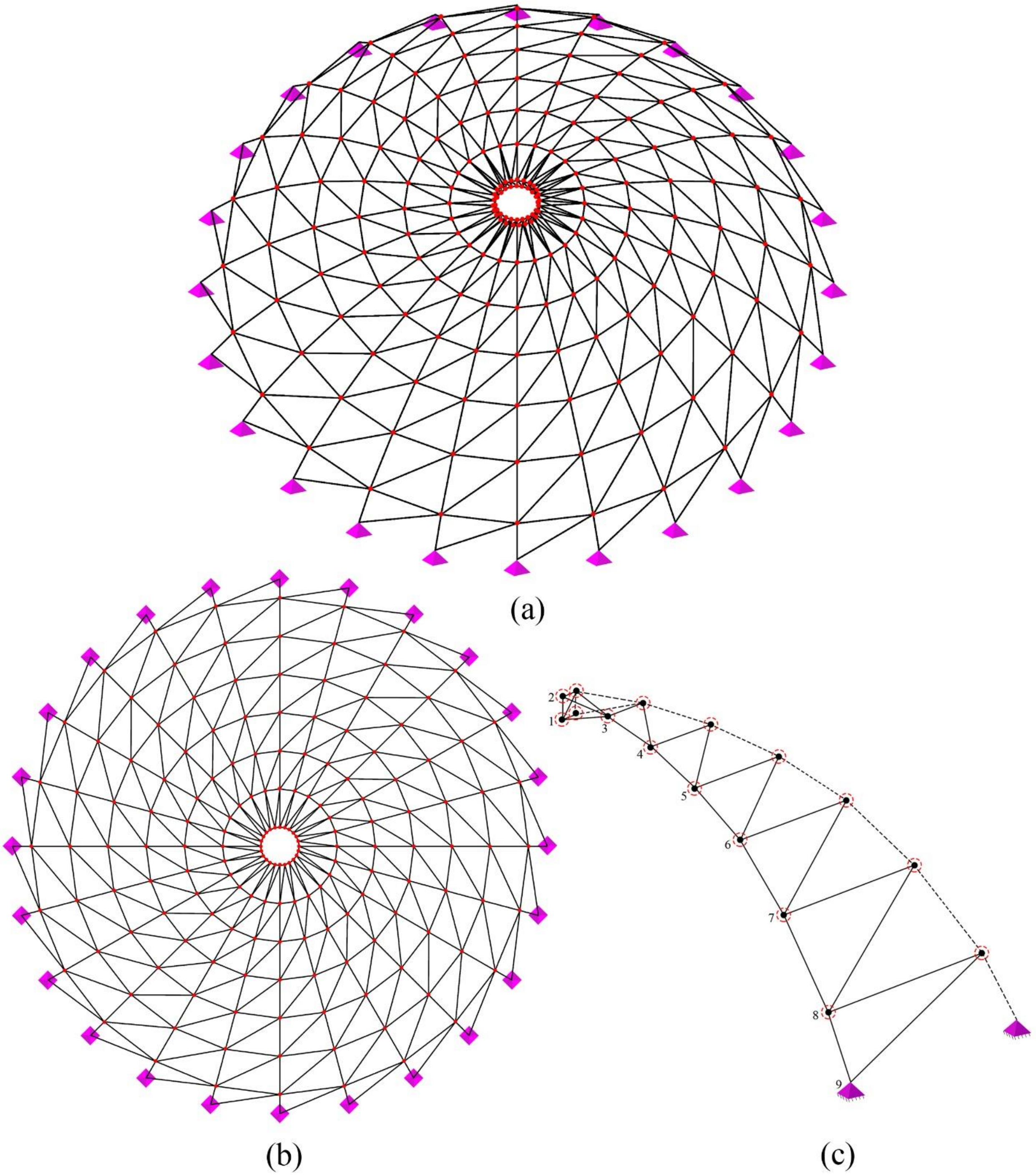

Start
Set the algorithm parameters: N, $P_2 = 0.001$, $P_3 = 0.3$, and $MaxFEs$
$FEs = 0$
Initialize the population of individuals
$FEs = FEs + N$
Sort the population of individuals and determine X_{best}
$i = 1$
$i \le N$
Yes
Determine X_{better}, X_{worse}, X_{L1}, and X_{L2}
Calculate $\overrightarrow{Gap_k}$ by Eq. (14) for k=1, 2, 3, 4
$\sum_{k=1}^{4} \overrightarrow{Gap_k} = 0$
No
Yes
Apply Eqs. (15)-(18)
Apply Eqs. (23) and (24)
Apply Eqs. (21) and (25)
Update X_{gbest}
$FEs = FEs + 1$
$i = i + 1$
No
Sort the population of individuals
$i = 1$
$i \le N$
Yes
Apply Eqs. (19), (21), (25), and (27)
Update X_{gbest}
$FEs = FEs + 1$
$i = i + 1$
No
$FEs \le MaxFEs$
Yes
No
Report X_{gbest}
End

Figure 4



(a)

(b)

(c)

Figure 5

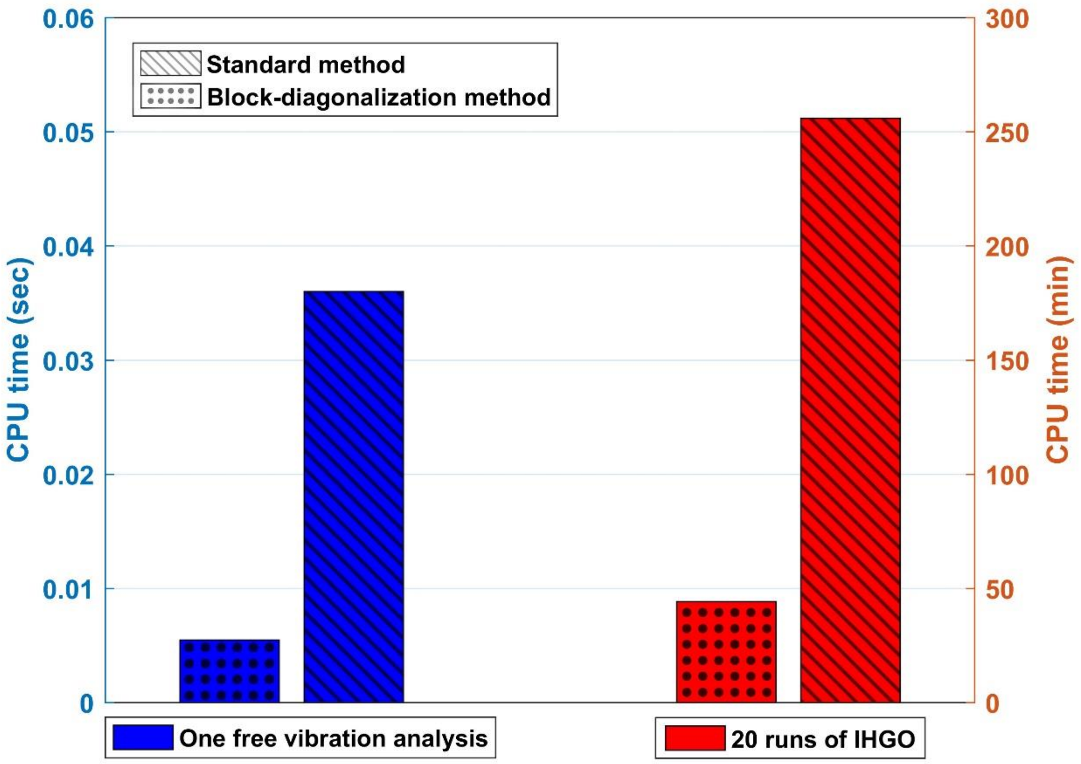


Figure 6

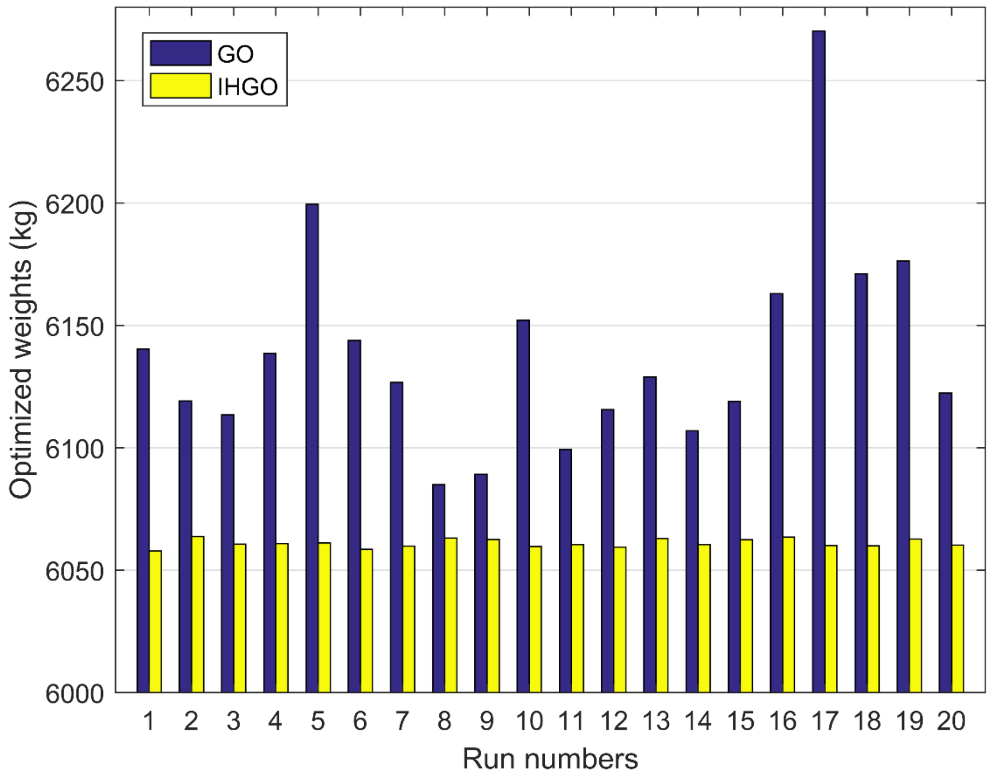


Figure 7

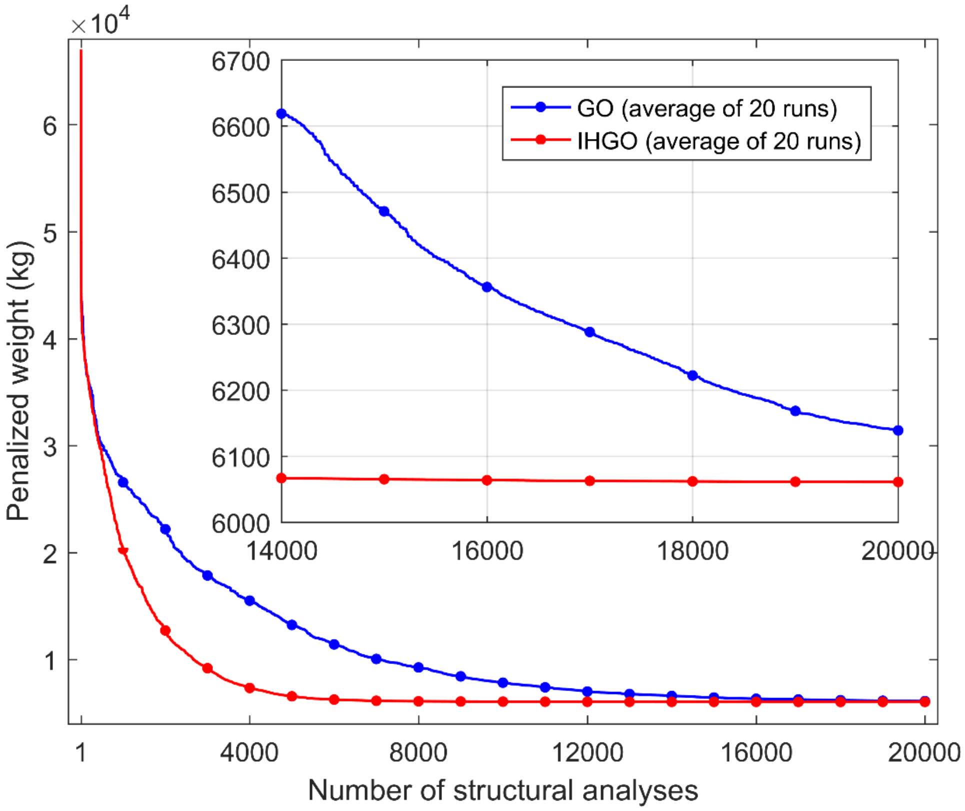

×10^4
6
5
4
3
2
1
Penalized weight (kg)
1
4000
8000
12000
16000
20000
Number of structural analyses
6700
6600
6500
6400
6300
6200
6100
6000
14000
16000
18000
20000
GO (average of 20 runs)
IHGO (average of 20 runs)

Figure 8

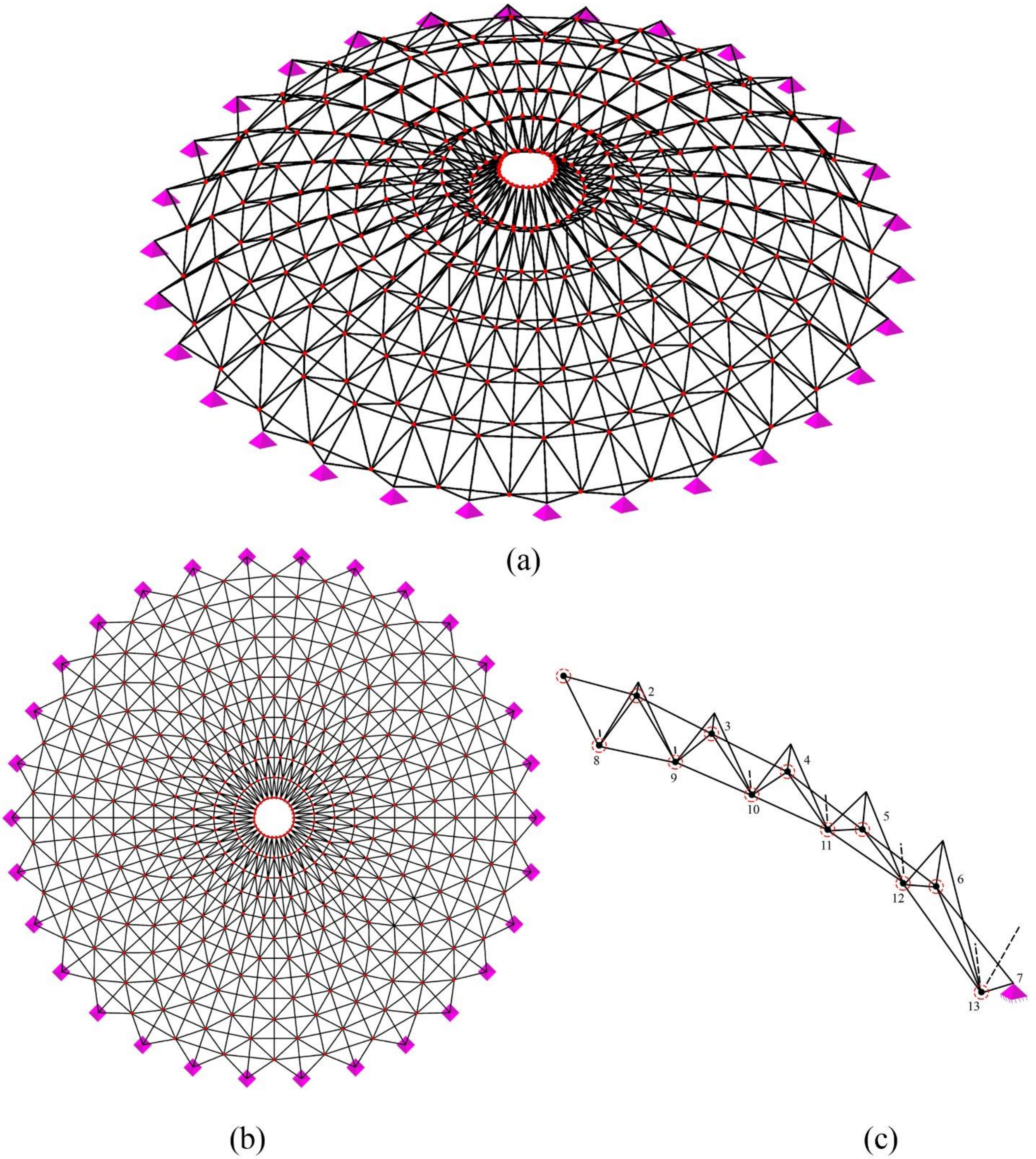


Figure 9

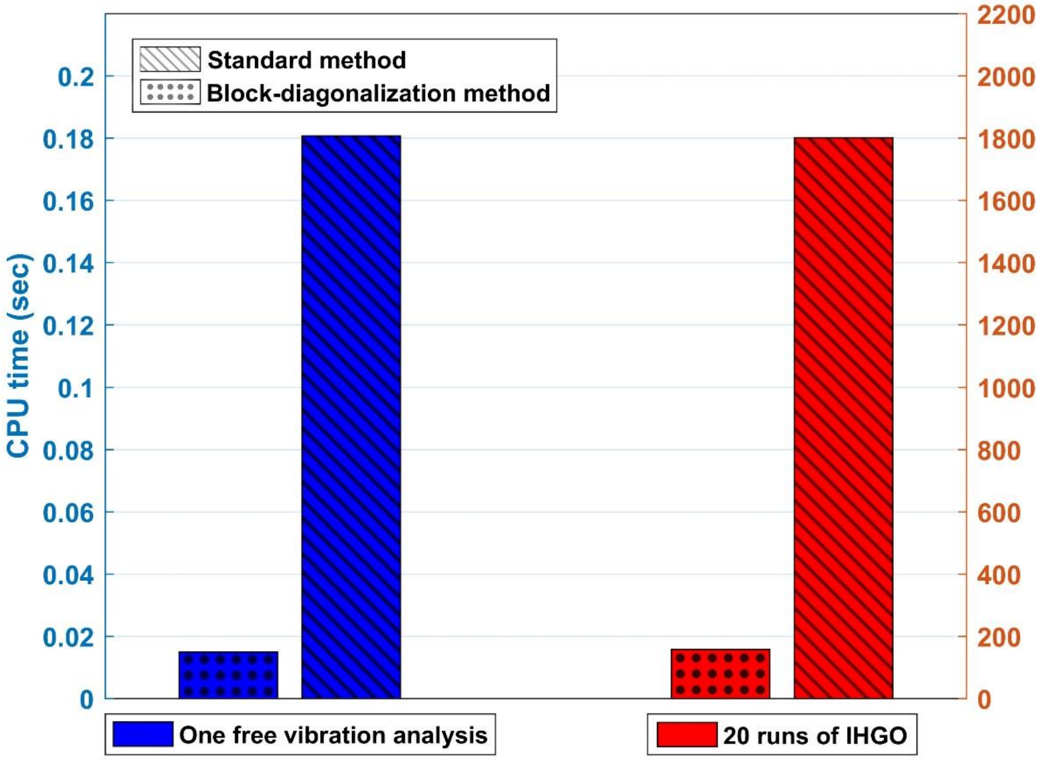


Figure 10

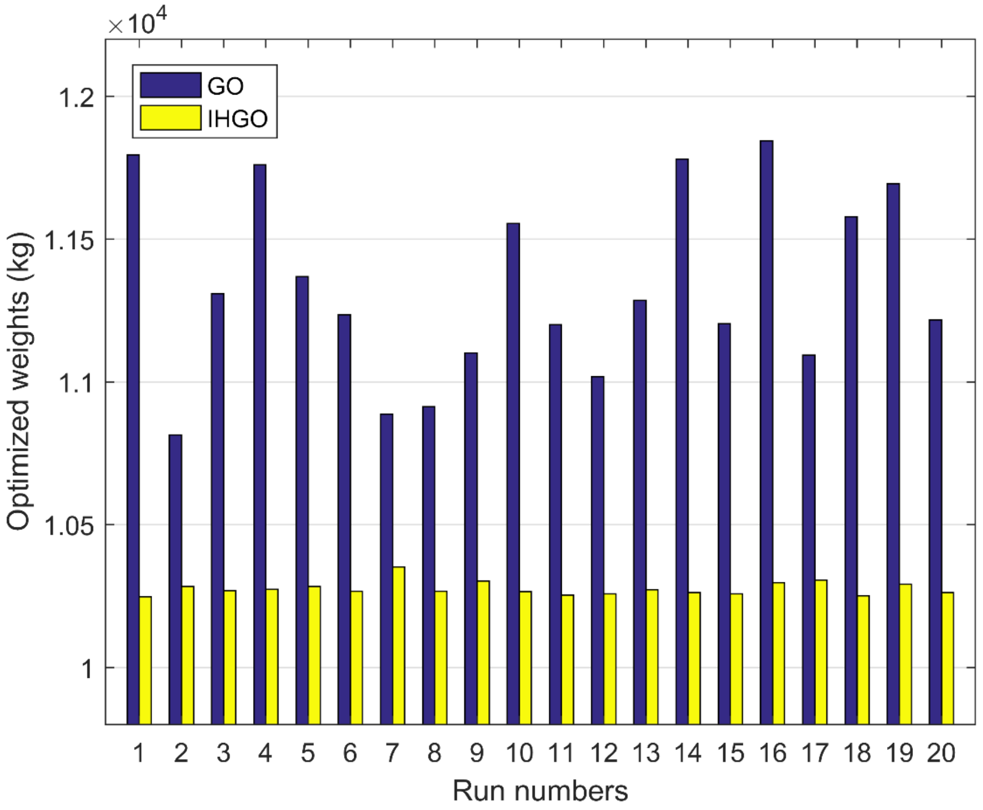
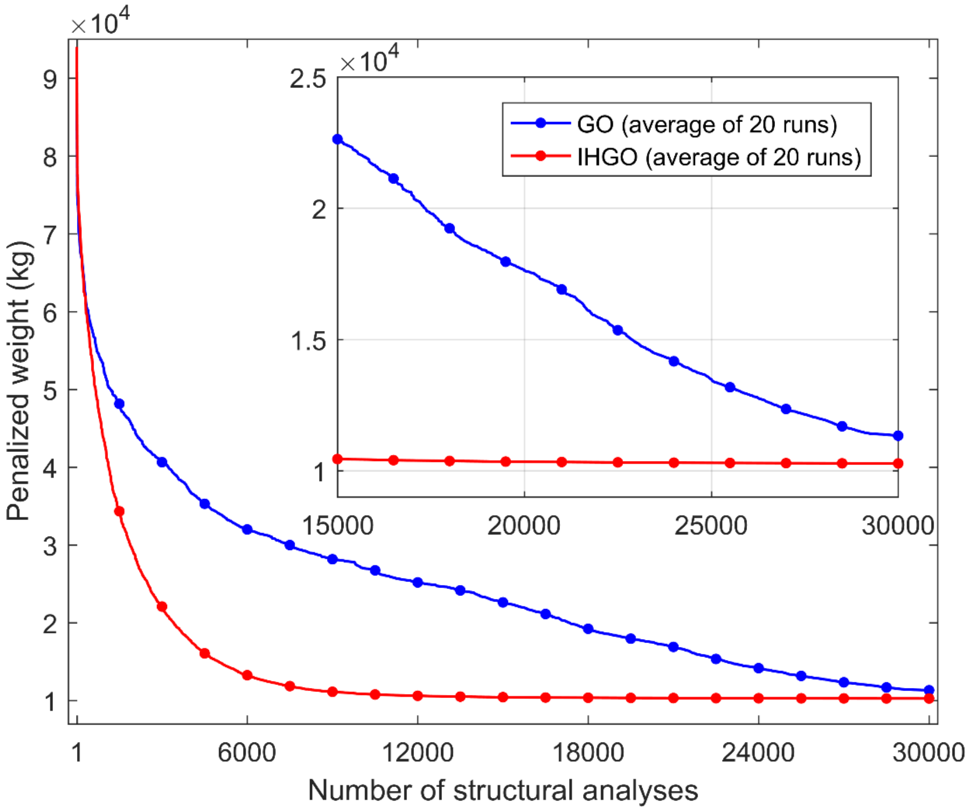


Figure 11



×10^4
9
8
7
6
5
4
3
2
1
Penalized weight (kg)
1
6000
12000
18000
24000
30000
Number of structural analyses
×10^4
2.5
2
1.5
1
15000
20000
25000
30000
GO (average of 20 runs)
IHGO (average of 20 runs)

Figure 12

Tables

Table 1

	600-bar dome	1410-bar dome
Material density (kg/cm^3)	0.00785	0.00785
Modulus of elasticity (N/cm^2)	2×10^7	2×10^7
Frequency constraints (Hz)	$\omega_1 \geq 5$ and $\omega_3 \geq 7$	$\omega_1 \geq 7$ and $\omega_3 \geq 9$
Cross-sectional area limits (cm^2)	$1 \leq A_i \leq 100;\ i = 1, 2, \ldots, 600$	$1 \leq A_i \leq 100;\ i = 1, 2, \ldots, 1410$

Table 2

	Memory requirements	Computational time	
		One free vibration analysis	20 runs of IHGO
Standard method	One square matrix of order 576	0.0360 s	15359 s (about 256 min)
Block-diagonalization method	24 square matrices of order 24	0.0055 s	2650 s (about 45 min)
Time ratio		6.49	5.80

Table 3

Design variables (end nodes)	Present study	
	GO	IHGO
A_1 (1-2)	**1.0701**	**1.4230**
A_2 (1-3)	**1.4391**	**1.3897**
A_3 (1-10)	**4.5604**	**5.1983**
A_4 (1-11)	**1.3484**	**1.3373**
A_5 (2-3)	**17.3835**	**17.0320**
A_6 (2-11)	**34.7019**	**37.6076**
A_7 (3-4)	**12.5073**	**12.7716**
A_8 (3-11)	**15.4886**	**15.4432**
A_9 (3-12)	**10.9628**	**11.3029**
A_{10} (4-5)	**9.0424**	**9.2889**
A_{11} (4-12)	**8.1699**	**8.3500**
A_{12} (4-13)	**8.9705**	**8.9979**
A_{13} (5-6)	**7.0833**	**7.1894**
A_{14} (5-13)	**5.4654**	**5.1356**
A_{15} (5-14)	**7.2635**	**6.7008**
A_{16} (6-7)	**5.5207**	**5.1627**
A_{17} (6-14)	**3.5405**	**3.5791**

A_{18} (6-15)	**8.5409**	**7.7180**
A_{19} (7-8)	**4.0977**	**4.2542**
A_{20} (7-15)	**2.1640**	**2.1782**
A_{21} (7-16)	**4.4742**	**4.6957**
A_{22} (8-9)	**4.0223**	**3.5841**
A_{23} (8-16)	**2.1292**	**1.8304**
A_{24} (8-17)	**4.4535**	**4.7384**
A_{25} (9-17)	**1.4143**	**1.6435**

Table 4

	Kaveh and Javadi [16]	Degertekin et al. [17]	Kaveh et al. [21]	Kaveh et al. [15]	Truong and Chou [19]	Dede et al. [22]	Abbasi and Zakian [20]	Truong and Chou [18]	Present study	
	CGFA (2019)	PFJA (2021)	Chaotic WSA (2021)	ISMA (2022)	FAFBI (2023)	Rao-2 (2022)	IGWO (2024)	ED (2024)	GO	IHGO
Best weight (kg)	6058.49	6333.251	6064.04	6068.34	6062.8485	6066.3123	6067.8756	6059.698	**6084.92**	**6057.87**
MinFEs	N/A	8580	N/A	20000	N/A	44220	N/A	18089	**19914**	**19726**
Mean weight (kg)	6076.67	6380.31	6081.23	6083.93	6075.82	6070.1723	6087.4038	6094.943	**6139.03**	**6060.99**
Worst weight (kg)	N/A	N/A	N/A	6095.41	N/A	N/A	6111.5616	N/A	**6270.27**	**6063.71**
SD (kg)	22.42	47.396	8.29	7.36	6.40	1.7832	11.9003	8.5870	**41.60**	**1.65**
MaxFEs	10000	25000	30000	20000	15000	45000	15000	20000	**20000**	**20000**
CV (%)	**0.0004**	**0**	**0.0005**	**0**	**0.0002**	**0**	**0**	**0.0003**	**0**	**0**
Number of runs	20	20	20	25	30	10	30	30	**20**	**20**
ω_1 (Hz)	5.000	5.0011	5.0005	5.0003	5.0000	5.0019	5.0006	5.0000	**5.0020**	**5.0000**
ω_3 (Hz)	7.000	7.0000	7.0000	7.0002	7.0000	7.0002	7.0000	7.0000	**7.0000**	**7.0000**
ω_1^* (Hz)	**5.0000**[1]	**5.0100**	**5.0005**	**5.0003**	**5.0000**[2]	**5.0019**	**5.0006**	**5.0000**[3]	**5.0020**	**5.0000**
ω_3^* (Hz)	**7.0000**	**7.0210**	**7.0000**[4]	**7.0002**	**7.0000**[5]	**7.0002**	**7.0000**	**7.0000**[6]	**7.0000**	**7.0000**

[1] 4.9999805440

[2] 4.9999904937

[3] 4.9999934348

[4] 6.9999656937

[5] 6.9999984292

[6] 6.9999907360

Table 5

	Memory requirements	Computational time	
		One free vibration analysis	20 runs of IHGO
Standard method	One square matrix of order 1080	0.1808 s	108084 s (about 1801 min)
Block-diagonalization method	30 square matrices of order 36	0.0149 s	9474 s (about 158 min)
Time ratio		12.13	11.41

Table 6

Design variables (end nodes)	Present study	
	GO	IHGO
A_1 (1-2)	**6.7062**	**6.1957**
A_2 (1-8)	**12.5717**	**4.9792**
A_3 (1-14)	**27.2736**	**28.9106**
A_4 (2-3)	**9.7815**	**8.7155**
A_5 (2-8)	**7.4770**	**5.3079**
A_6 (2-9)	**1.5483**	**1.1257**
A_7 (2-15)	**15.5878**	**15.7831**
A_8 (3-4)	**9.5492**	**8.8416**
A_9 (3-9)	**1.3643**	**2.0510**
A_{10} (3-10)	**2.9609**	**2.7904**
A_{11} (3-16)	**7.8200**	**10.1139**
A_{12} (4-5)	**10.3349**	**9.9897**
A_{13} (4-10)	**2.6390**	**2.0898**
A_{14} (4-11)	**5.0446**	**4.9902**
A_{15} (4-17)	**15.7627**	**16.1773**
A_{16} (5-6)	**11.7099**	**8.2430**
A_{17} (5-11)	**5.0095**	**3.3566**
A_{18} (5-12)	**5.5726**	**6.1709**
A_{19} (5-18)	**9.9867**	**12.3210**
A_{20} (6-7)	**12.0003**	**13.3990**
A_{21} (6-12)	**5.9915**	**5.0459**
A_{22} (6-13)	**9.1682**	**7.4871**
A_{23} (6-19)	**1.7542**	**1.0004**
A_{24} (7-13)	**3.8653**	**4.6944**
A_{25} (8-9)	**4.0597**	**2.8845**

A_{26} (8-14)	**6.3957**	**4.9705**
A_{27} (8-15)	**7.9448**	**5.6938**
A_{28} (8-21)	**7.2963**	**11.3754**
A_{29} (9-10)	**3.6399**	**3.7371**
A_{30} (9-15)	**1.0009**	**1.5501**
A_{31} (9-16)	**1.3369**	**2.1109**
A_{32} (9-22)	**4.7585**	**4.6317**
A_{33} (10-11)	**5.7677**	**5.3102**
A_{34} (10-16)	**2.3466**	**2.8785**
A_{35} (10-17)	**1.6140**	**2.0992**
A_{36} (10-23)	**1.6779**	**3.2080**
A_{37} (11-12)	**7.7679**	**7.6168**
A_{38} (11-17)	**5.8565**	**5.1908**
A_{39} (11-18)	**3.0653**	**3.0738**
A_{40} (11-24)	**1.5759**	**1.0237**
A_{41} (12-13)	**7.4482**	**6.8026**
A_{42} (12-18)	**6.5613**	**6.0202**
A_{43} (12-19)	**4.5542**	**5.0093**
A_{44} (12-25)	**1.8282**	**1.0000**
A_{45} (13-19)	**5.1004**	**7.3985**
A_{46} (13-20)	**3.9122**	**4.5707**
A_{47} (13-26)	**1.1472**	**1.0010**

Table 7

	Degertekin et al. [17]	Kaveh et al. [21]	Kaveh et al. [15]	Dede et al. [22]	Van TH et al. [23]	Truong and Chou [19]	Abbasi and Zakian [20]	Truong and Chou [18]	Present study	
	PEJA (2021)	Chaotic WSA (2021)	ISMA (2022)	Rao-2 (2022)	C-HCLPSO (2023)	FAFBI (2023)	IGWO (2024)	ED (2024)	GO	IHGO
Best weight (kg)	10326.296	10318.99	10309.41	10281.06	10541.33	10257.84	10349.6353	10257.30	**10813.94**	**10248.13**
MinFEs	16900	N/A	20000	44850	N/A	N/A	N/A	40737	**29625**	**28965**
Mean weight (kg)	10399.828	10521.67	10556.67	10289.22	10626.13	10280.59	10455.9838	10283.52	**11332.70**	**10276.16**
Worst weight (kg)	N/A	N/A	10825.00	N/A	N/A	N/A	10571.2224	N/A	**11843.96**	**10351.57**
SD (kg)	75.441	122.1458	130.92	3.06	79.56	13.70	62.6789	5.7429	**317.20**	**23.84**
MaxFEs	25000	30000	20000	45000	40000	20000	15000	45000	**30000**	**30000**

CV (%)	**0**	**0.0034**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Number of runs	20	20	25	10	30	30	30	N/A	**20**	**20**
ω_1 (Hz)	7.0009	7.0000	7.0001	7.0040	7.000	7.0000	7.0002	7.0000	**7.0041**	**7.0001**
ω_3 (Hz)	9.0001	9.0021	9.0002	9.0013	9.000	9.0000	9.0001	9.0000	**9.0050**	**9.0001**
ω_1^* (Hz)	**7.0125**	**6.9998**[7]	**7.0001**	**7.0040**	**7.0467**	**7.0000**	**7.0002**	**7.0000**	**7.0041**	**7.0001**
ω_3^* (Hz)	**9.0083**	**9.0021**	**9.0002**	**9.0013**	**9.0012**	**9.0000**	**9.0001**	**9.0000**	**9.0050**	**9.0001**

[7] 6.9997617653