Swarm and Evolutionary Computation 38 (2018) 109–119

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

# Accelerated multi-gravitational search algorithm for size optimization of truss structures

Mohsen Khatibinia[a,*], Hessam Yazdani[b]

[a] Department of Civil Engineering, University of Birjand, Birjand, Iran
[b] Department of Civil and Environmental Engineering, Howard University, Washington, DC 20059, USA

ARTICLE INFO

*Keywords:*
Size optimization
Truss structures
Gravitational search algorithm
Simplex crossover
Breeder genetic algorithm

ABSTRACT

Weak local exploitation capability of the gravitational search algorithm (GSA) and its slow convergence rate in final iterations have been demonstrated in the literature. This paper presents a modified GSA denoted here as the accelerated multi-gravitational search algorithm (AMGSA) that exhibits an improved convergence rate. In AMGSA, the simplex crossover (SPX) and the operator mutation of the breeder genetic algorithm (BGA) are incorporated with the multi-gravitational search algorithm (MGSA) to achieve an algorithm with a good exploration-exploitation balance. MGSA is adopted to prevent stagnation of the search into a local optimum (i.e. to improve the exploration capability), while the SPX and the BGA mutation operator are used to bias the search toward promising areas of the search space (i.e. to promote local exploitation). The performance of AMGSA is evaluated using several benchmark truss optimization examples. Results indicate that AMGSA not only exhibits an improved balance between the exploration and exploitation schemes but also shows competitive promise in effectively and efficiently solving large-scale optimization problems as it requires a significantly lower number of structural analyses compared to other algorithms that it is checked against.

## 1. Introduction

Over the course of billions of years of evolutionary history, nature has heuristically (i.e. experimentally, especially by trial and error) developed a diverse and remarkably-ingenious set of dynamic and robust strategies that have endowed creatures and organisms with optimum yet sustainably-resilient adaptability to their in-flux environments. The last few decades saw significant research efforts that were devoted to understanding these strategies and exploring their applicability to solve complex engineering problems (e.g. see [1]). These efforts coupled with those driven by other inspirations (e.g. musical improvisation [2]) led to the development of several optimization techniques knows as *optimization metaheuristics* that in turn made previously-unimaginable advances in structural optimization possible. The prefix *meta*-, meaning more developed/higher level, indicates that, in contrast to heuristic techniques, metaheuristics are problem-independent and versatile, making them suitable for various kinds of problems. In addition, compared to now-traditional gradient-based optimization techniques, the stochastic mechanism of metaheuristics allows them to effectively explore and exploit a vast search space enclosed by highly-nonlinear and discontinuous constraints without requiring gradient information and explicit formulations for the objective function and constraints [3]. It is not within the scope of the present study to detail the history of metaheuristics. However, a brief review of the most prominent metaheuristic optimization techniques is given below, and the interested reader is referred to, e.g., Sorensen et al. [4] for a portrait of the evolution of optimization metaheuristics.

The phenomenal growth in the field of optimization metaheuristics has been taking place in a series of stages with divisions usually spaced out over several years. While the first use of heuristics by humans can be traced back to prehistoric times when they employed the strategies learned from determining the trajectory of a stone to hit a bear toward hitting a mammoth with a spear, the modern application of heuristics took place in the late 1950s and early 1960s when the advent of computer enabled researchers to develop and use algorithms to study the phenomenon of natural evolution. One of the first methods to receive recognition was the so-called evolution strategy, where the better of a solution and its mutated version is used as the parent for the next round of mutation [5]. Another method was evolutionary programming, where solutions are represented as finite-state machines that change from one state to another in response to a mutation operator [6].

These early algorithms lacked the concepts of population and crossover until 1975 when the recognition of their importance by

* Corresponding author.
*E-mail address:* m.khatibinia@birjand.ac.ir (M. Khatibinia).

http://dx.doi.org/10.1016/j.swevo.2017.07.001
Received 15 December 2016; Received in revised form 2 June 2017; Accepted 1 July 2017
Available online 04 July 2017

John Holland in his seminal work [7] followed by a book by his student David Goldberg [8] sparked tremendous research activities in the field of evolutionary algorithms. Inspired by the Darwinian "survival of the fittest" concept, Holland [7] proposed the genetic algorithm (GA) where a population of candidate solutions each represented by an array of bits is randomly generated and iteratively improves through the repetitive application of a series of stochastic operators (i.e. mutation, crossover, inversion, transposon, and selection).

The 1980s saw the development of two of the most popular optimization algorithms. First, Kirkpatrick et al. [9] developed the simulated annealing (SA) optimization algorithm by analogizing the optimization process to annealing—the controlled heating and cooling process used in metallurgy and glass production to remove stresses from the material. In SA, solutions are stochastically changed in each step so as to iteratively lead the system toward states of lower energy. This process is repeated until the system achieves a sufficiently-low energy, or until a given computational budget is exhausted. Later, Glover [10] proposed tabu search, where a memory structure retaining rules and previously-visited solutions is used to refine the search space from poor solutions, thus gradually isolating the optimum.

In the 1990s, swarm intelligence—collective behavior of a system capable of accomplishing difficult tasks in dynamic and varied environments without any central coordination, external guidance, or control—inspired the development of two other prominent metaheuristics; the ant colony (ACO) and particle swarm optimization (PSO) algorithms. ACO simulates the foraging behavior of ants and makes an analogy between finding the shortest path from nest to food and minimizing an objective function [11]. PSO, in contrast, adopts the synchronization mechanism of the migration of a flock [12]. It uses a swarm of candidate solutions (dubbed particles) and moves it toward optimum solutions by sharing improvements discovered by all particles. The 1990s also saw the development of one of the most powerful evolutionary computing algorithms known as the differential evolution [13]. Many more metaheuristic algorithms have recently been proposed and used in structural optimization (e.g. harmony search [14]). Among them, the gravitational search algorithm (GSA) has received considerable attention due to its strength in the exploration of the search space [15]. GSA is a stochastic and population-based optimization algorithm that mimics Newton's law of universal gravitation and laws of motion. In GSA, a population of bodies with masses proportional to their fitness value is randomly generated and iteratively moved throughout the search space in order to find the optimum region. The interactions among the bodies (potential solutions) are governed by Newton's law of universal gravitation.

The global search capability of GSA in its original, extended, or hybrid forms and its merit in solving nonlinear optimization problems have been demonstrated in numerous studies (e.g. see [16–19]). Some other studies, however, have shown the weak local exploitation capability of original GSA and its slow convergence rate in final iterations (e.g. see [20,21]). This recognition led to a wave of studies focused on improving the efficiency of GSA through hybridizing it with features of other metaheuristics (e.g. the crossover operator of GA). Further details are given in Section 3.1, and the interested reader is encouraged to consult Khatibinia et al. [22] and Khatibinia and Khosravi [23] for additional information.

Pursuant to recent efforts devoted to improving GSA, an amended version of the multi-gravitational search algorithm (MGSA [24]) is proposed in this study. In the proposed algorithm, referred here to as accelerated MGSA (AMGSA), an exploration-exploitation balance is achieved through incorporating MGSA with the simplex crossover (SPX [25]) and the mutation operator of the breeder genetic algorithm (BGA [26]). MGSA is adopted in order to prevent the stagnation of the search into a local optimum (i.e. to improve the exploration capability), while the SPX and the BGA mutation operator are used to bias the search toward promising areas of the search space (i.e. to promote local exploitation). In addition, the feasibility-based rule proposed by Deb [27] is used to direct solutions to feasible regions of the search space. Several benchmark truss optimization problems are solved to demonstrate the efficiency of AMGSA. These include 10-, 18-, and 200-bar planar trusses in addition to a 72-bar space truss.

Formulation of a truss optimization problem is given in Section 2 followed by a detailed description of the proposed technique and its constituents (Sections 3–6). Finally, the performance of the technique is demonstrated in Section 7.

## 2. Formulation of the structural optimization problem

Structural optimization in general and truss sizing optimization in particular fall within the context of nonlinear programming. In truss sizing optimization, the objective function is to minimize the weight of a truss subject to stress and displacement constraints while taking the cross-sectional area of members as design variables. These problems can be expressed in a canonical form as:

$$\begin{aligned}
&\text{find} && \mathbf{A} = \{A_1,\ A_2,\ .\ .\ .\ .\ ,\ A_{ng}\}^T \\
&\text{min} && W(\mathbf{A}) = \textstyle\sum_{i=1}^{ng} A_i \sum_{j=1}^{nm\ i} \rho_j L_j \\
&\text{s.t.} && \sigma_{mn} \le \sigma_{all}\ ;\quad m = 1,\ 2,\ \dots,\ ne\ ,\quad n = 1,\ 2,\ \dots,\ nl \\
& && \Delta_{kn} \le \Delta_{all}\ ;\quad k = 1,\ 2,\ \dots,\ nd\ ,\quad n = 1,\ 2,\ \dots,\ nl \\
& && \mathbf{A}^L \le \mathbf{A} \le \mathbf{A}^U
\end{aligned} \tag{1}$$

where $\mathbf{A}$ represents the vector of design variables; $W$ is the weight of the structure; $nm_i$ is the number of members included in the $i$th group; $\rho_j$ and $L_j$ are the material density and the length of the $j$th member belonging to the group $i$, respectively; $ng$ and $ne$ are the total number of member groups (i.e. design variables) and the total number of elements in the structure, respectively; $\sigma_{\text{mn}}$ and $\sigma_{\text{all}}$ are the stress in the $m$th member due to the loading condition $n$ and the allowable stress, respectively; $\Delta_{kn}$ and $\Delta_{all}$ are the nodal displacement of the $k$th translational degree of freedom due to the loading condition $n$ and the allowable displacement, respectively; $nl$ and $nd$ are the number of loading conditions and the total number of translational degrees of freedom in the structure, respectively; and $\mathbf{A}^L$ and $\mathbf{A}^U$ are the lower and upper bounds for cross-sectional areas, respectively.

Numerous constraint-handling techniques for metaheuristic algorithms have been proposed in the literature (e.g. dynamic penalties and adaptive penalties). The interested reader is referred to a comprehensive survey by Coello Coello [28]. In the present study, the objective function of the designs violating imposed constraints were penalized using the external penalty approach as [29–31]:

$$W^p(\mathbf{A}) = W(\mathbf{A})\,(1 + R_p\, P_f) \tag{2}$$

where $W^p(\mathbf{A})$ is the penalized objective function of a constraint-violating solution; $R_p$ is an adjusting coefficient (see [32]); and $P_f$ is the total penalty and represents the degree of constrain violation, which is the sum of all normalized constraint violations and defined as:

$$P_f = \sum_{m=1}^{ne}\sum_{n=1}^{nl} \max\left(\left|\frac{\sigma_{mn}}{\sigma_{all}}\right| - 1,\ 0\right)^2 + \sum_{k=1}^{nd}\sum_{n=1}^{nl} \max\left(\left|\frac{\Delta_{kn}}{\Delta_{all}}\right| - 1,\ 0\right)^2 \tag{3}$$

## 3. Multi-GSA method

### 3.1. Gravitational search algorithm

In the iteration $t$ of GSA, the agent $i$ as a potential solution for the optimization problem is associated with a position vector $\mathbf{X}_i$ and a velocity vector $\mathbf{V}_i$ denoted as:

$$\begin{aligned}
\mathbf{X}_i(t) &= \{\ x_i^1\ ,\ .\ .\ .\ .\ ,\ x_i^d\ ,\ .\ .\ .\ .\ ,\ x_i^D\} \\
\mathbf{V}_i(t) &= \{\ v_i^1,\ .\ .\ .\ .\ ,\ v_i^d\ ,\ .\ .\ .\ .\ ,\ v_i^D\}
\end{aligned} \tag{4}$$

where $D$ is the dimension of the search space and is equal to the number of design variables. For a minimization problem, the mass of the agent $i$ at the iteration $t$, $M_i(t)$, is updated after computing the fitness value of all agents in the population as:

$$M_i(t) = \frac{\widetilde{f}_i(t) - worst(t)}{\sum_{j=1}^{N}(\widetilde{f}_j(t) - worst(t))} \tag{5}$$

where $N$ and $\widetilde{f}_i(t)$ represent the population size and the fitness value of the $i$th agent at the $t$th iteration, respectively; and $worst(t)$ is defined as the worst fitness value among all agents at the iteration $t$.

Using the laws of motion, the acceleration of the agent $i$ in the direction $d$ is given as:

$$a_i^d(t) = \sum_{j \in kbest,\, j \neq 1} r_j G(t) \frac{M_j(t)}{R_{ij}(t) + \alpha}(x_j^d(t) - x_i^d(t)) \tag{6}$$

where $r$ is a uniform random number in the interval [0, 1]; $G(t)$ is a proportionality factor analogous to the universal gravitational constant; α is a small positive constant that is used to avoid division by zero ($2^{-52}$ in this study); $R_{i,j}(t)$ is the Euclidean distance between agents $i$ and $j$, $\|\mathbf{X}_i(t) - \mathbf{X}_j(t)\|_2$; and $k_{best}$ is the set of the first $k$ agents with the best fitness values (i.e. largest masses) in each iteration. These are the only agents that have gravitational interaction with one another and other ($N$ - $k$) agents. The contribution of the ($N$ - $k$) unfit agents to the acceleration of the agents of the population is set to zero in each iteration. $k_{best}$ is expressed in terms of a percentage of the population and changes with the iteration number. In this study, $k_{best}$ started from $N$ (i.e. 100% of the population) in the first iteration and decreased linearly to 2% of the population in the final iteration. This component of acceleration (Eq. (6)) is then added to a fraction of the current velocity of the agent in the corresponding direction in order to determine its velocity and, subsequently, its position in the same direction for the next iteration as:

$$v_i^d(t+1) = r_i v_i^d(t) + a_i^d(t) \tag{7}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{8}$$

where $v_i^d$ and $x_i^d$ are the velocity and the position of the agent $i$ in the direction $d$, respectively.

Proportionality factor (gravitational constant), $G(t)$, could have a significant influence on the performance of GSA. Some variations have been proposed for $G(t)$ in the literature (e.g. see [33]). In this study, however, the original definition of $G(t)$ given below was used [15]:

$$G(t) = G_0 \exp(-\beta \frac{t}{t_{\max}}) \tag{9}$$

where $G_0$ is the initial value of $G$; $\beta$ is a constant; and $t_{max}$ is the maximum number of iterations. The values used for these variables in this study are given in Section 7.

Drawbacks of GSA, namely weak local exploitation and slow convergence rate in final iterations, were touched on in the Introduction section. This recognition motivated, for instance, Khatibinia et al. [22] to incorporate GSA with the particle swarm optimizer with passive congregation (PSOPC), which exhibits superior exploitation capability. PSOPC [34] takes its impetus from passive congregation, one of the four biological mechanisms underlying congregations of creatures that routinely swarm and cluster or crowd together (see [35]). In the PSOPC-improved GSA, the velocity of the agent $i$ in the next iteration is influenced not only by its current velocity and the gravitational forces exerted by other agents (Eq. (7)) but also by the best previous position of the very agent (also known as the memory of agent $i$), $pbest_i^d$, the global best position found by the entire population up to the current iteration (same for all agents), $gbest^d$, and the position of a randomly-selected agent, $p_i^d$, in the $d$th direction as:

$$\begin{aligned} v_i^d(t+1) = {} & r_{i,1} v_i^d(t) + a_i^d(t) + r_{i,2}(pbest_i^d(t) - x_i^d(t)) \\ & + r_{i,3}(gbest^d(t) - x_i^d(t)) + r_{i,4}(p_i^d(t) - x_i^d(t)) \end{aligned} \tag{10}$$

where $r_{i,1}$ to $r_{i,4}$ are uniform random numbers in the interval [0, 1].

### 3.2. Multi-GSA

Khatibinia and Naseralavi [24] proposed the multi-GSA (MGSA) algorithm to enhance the exploration capability of GSA. MGSA is comprised of two main stages. In the first stage, the population is broken into several subpopulations, and in the second stage, each subpopulation is independently evaluated by GSA while communicating its information with its counterparts. Doing so, MGSA divides the whole search space into subspaces and reduces the possibility for the entire population to fall into local optima, thereby improving the exploration of the search space.

The MGSA algorithm (see [24]) begins with determining the feasibility of the randomly-generated initial population and assigning its agents into two categories of feasible and infeasible solutions. The feasible category is then sorted in an ascending order according to the fitness value of feasible solutions, while the infeasible category is arranged based on the degree of constrain violation, $P_f$, of its components. Subsequently, given each subpopulation is composed of $N_s$ agents, the first subpopulation is created by the top agent of the feasible category together with its ($N_s$ − 1) farthest neighbors, irrespective of the feasibility category they belong to. Creation of the first subpopulation is then followed by eliminating its constituting agents from the entire population and rearranging the feasibility categories. Following the same procedure, the second subpopulation is formed by the top agent of the updated feasible category and ($N_s$ − 1) of the remaining agents having the greatest distances from it, followed by removing them from the population. This procedure continues until the population is divided into $N/N_s$ subpopulations. Eventually, GSA is used to find the optimum within each subpopulation.

## 4. Simplex crossover (SPX)

The simplex crossover [25] is a stochastic operator in the real-coded GA that combines multiple parents to produce a new offspring without requiring their fitness values—real-coded GA is an extension to GA where, as opposed to the binary-coded GA, each chromosome is coded as a vector of floating-point numbers. Simplex is a geometric element in a Euclidean space that has the minimum number of boundary points (vertices), e.g. a line in one-dimensional space or a triangle in two-dimensional space. Therefore, in $\mathbf{R}^n$ ($n$-dimensional space), ($n$ + 1) independent individuals (vertices) form a simplex. In such space, the SPX stochastically mates ($n$ + 1) parents to produce one offspring.

For simplicity, let us describe the SPX in two-dimensional space where three individuals $\mathbf{X}_1$, $\mathbf{X}_2$, and $\mathbf{X}_3$ form a simplex (i.e. a triangle – Fig. 1). If the simplex is expanded by a rate of $\varepsilon$ along each direction, vertices of the enlarged simplex (i.e. $\mathbf{Y}_1$, $\mathbf{Y}_2$, and $\mathbf{Y}_3$) can be expressed as:

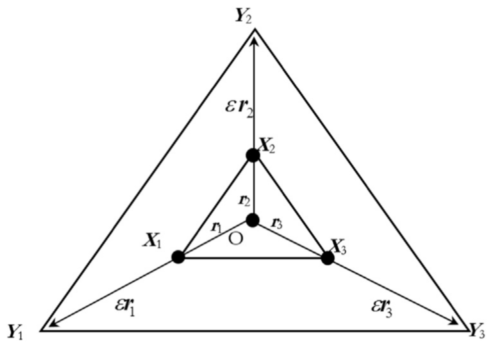

**Fig. 1.** 2D representation of simplex crossover.

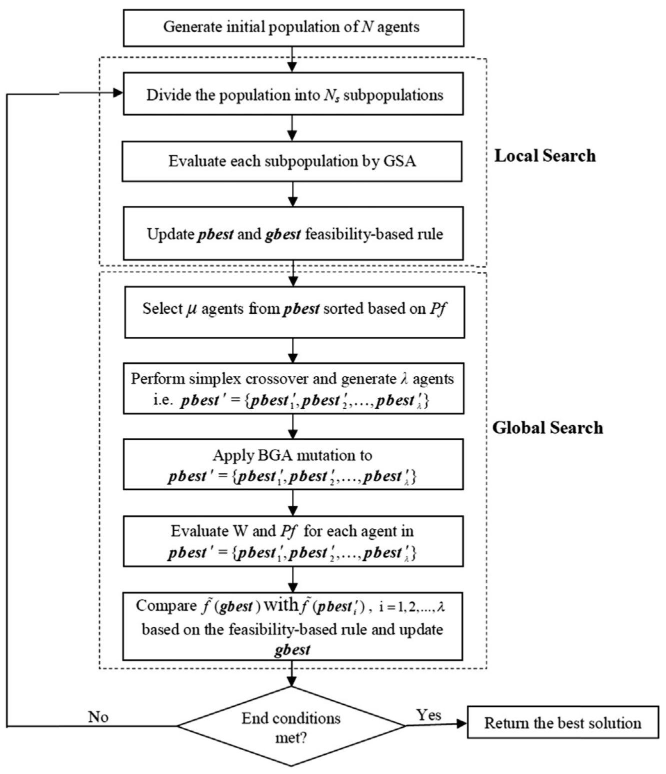
Generate initial population of N agents
Divide the population into $N_s$ subpopulations
Evaluate each subpopulation by GSA
Update pbest and gbest feasibility-based rule
Local Search
Select μ agents from pbest sorted based on Pf
Perform simplex crossover and generate λ agents i.e. $pbest' = \{pbest'_1, pbest'_2, \ldots, pbest'_\lambda\}$
Apply BGA mutation to $pbest' = \{pbest'_1, pbest'_2, \ldots, pbest'_\lambda\}$
Global Search
Evaluate W and Pf for each agent in $pbest' = \{pbest'_1, pbest'_2, \ldots, pbest'_\lambda\}$
Compare $f(gbest)$ with $f(pbest'_i)$, i = 1,2,...,λ based on the feasibility-based rule and update gbest
No
End conditions met?
Yes
Return the best solution

**Fig. 2.** Flowchart of AMGSA.

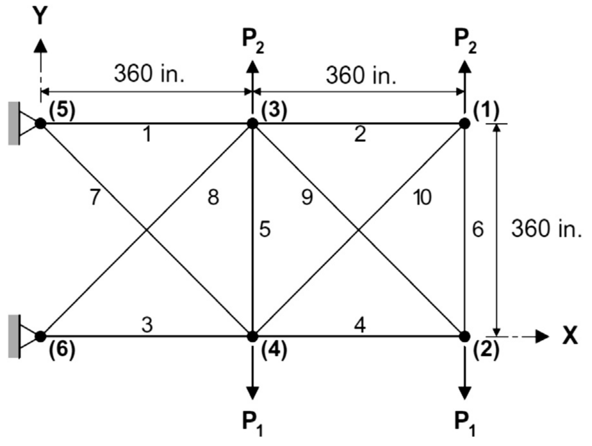

**Fig. 3.** The 10-bar planar truss.

$$\mathbf{Y}_i = (1+\varepsilon)(\mathbf{X}_i - \mathbf{O})\;; \quad where \quad \mathbf{O} = \frac{1}{3}\sum_{i=1}^{3}\mathbf{X}_i \tag{11}$$

where **O** represents the center of gravity. A new offspring **Z** is then produced by combining new vertices as:

$$\mathbf{Z} = k_1\,\mathbf{Y}_1 + k_2\,\mathbf{Y}_2 + k_3\,\mathbf{Y}_3 + \mathbf{O} \tag{12}$$

where $k_1$, $k_2$, and $k_3$ are uniform random numbers that sum to 1.

The procedure above can readily be generalized to higher dimensions. An SPX operation is generally specified as SPX-$\mu$-$\lambda$-$\varepsilon$, where $\mu$ is the number of parents mated to produce $\lambda$ offspring. The advantages of the SPX are twofold: 1) it maintains an exploration-exploitation balance, and 2) the computational complexity involved is only $O(n)$ [25].

**Table 1**
Sensitivity analysis of AMGSA for the 10-bar planar truss under the loading Case 1.

| $N$ | $N_S$ | $\lambda$ | Weight (lb) | | | | # of analyses |
|---|---|---|---|---|---|---|---|
| | | | Best | Worst | Mean | SD | |
| 20 | 4 | 10 | 5060.9 | 5061.4 | 5061.1 | 0.2 | 9000 |
| 20 | 4 | 15 | 5060.8 | 5061.0 | 5060.9 | 0.1 | 10,500 |
| 20 | 5 | 10 | 5060.9 | 5061.6 | 5060.9 | 0.2 | 9000 |
| 20 | 5 | 15 | 5060.8 | 5061.2 | 5060.1 | 0.1 | 10,500 |
| 20 | 10 | 10 | 5060.9 | 5061.3 | 5061.1 | 0.1 | 9000 |
| 20 | 10 | 15 | 5060.8 | 5060.9 | 5060.9 | 0.0 | 10,500 |
| 30 | 5 | 5 | 5060.9 | 5061.7 | 5061.1 | 0.2 | 10,500 |
| 30 | 5 | 10 | 5060.9 | 5061.5 | 5061.1 | 0.2 | 12,000 |
| 30 | 5 | 15 | 5060.9 | 5061.1 | 5061.3 | 0.2 | 13,500 |
| 30 | 6 | 5 | 5060.9 | 5061.7 | 5061.2 | 0.2 | 10,500 |
| 30 | 6 | 10 | 5060.9 | 5061.6 | 5061.2 | 0.2 | 12,000 |
| 30 | 6 | 15 | 5060.9 | 5062.3 | 5061.3 | 0.3 | 13,500 |
| 30 | 10 | 5 | 5060.9 | 5062.2 | 5061.2 | 0.2 | 10,500 |
| 30 | 10 | 10 | 5060.9 | 5061.6 | 5061.2 | 0.1 | 12,000 |
| 30 | 10 | 15 | 5060.8 | 5060.9 | 5060.9 | 0.0 | 13,500 |
| 40 | 5 | 5 | 5060.9 | 5061.5 | 5061.1 | 0.1 | 13,500 |
| 40 | 5 | 10 | 5060.9 | 5061.4 | 5061.1 | 0.1 | 15,000 |
| 40 | 5 | 15 | 5060.9 | 5061.3 | 5061.1 | 0.1 | 16,500 |
| 40 | 8 | 5 | 5060.9 | 5061.9 | 5061.2 | 0.2 | 13,500 |
| 40 | 8 | 10 | 5060.9 | 5061.9 | 5061.2 | 0.2 | 15,000 |
| 40 | 8 | 15 | 5060.9 | 5061.6 | 5061.1 | 0.2 | 16,500 |
| 40 | 10 | 5 | 5060.9 | 5061.6 | 5061.1 | 0.2 | 13,500 |
| 40 | 10 | 10 | 5060.8 | 5061.6 | 5061.1 | 0.2 | 15,000 |
| 40 | 10 | 15 | 5060.8 | 5060.9 | 5060.9 | 0.0 | 16,500 |
| 50 | 5 | 5 | 5060.9 | 5061.6 | 5061.1 | 0.1 | 16,500 |
| 50 | 5 | 10 | 5060.9 | 5061.4 | 5061.1 | 0.1 | 18,000 |
| 50 | 5 | 15 | 5060.9 | 5061.3 | 5061.0 | 0.1 | 19,500 |
| 50 | 10 | 5 | 5060.9 | 5061.0 | 5061.0 | 0.0 | 16,500 |
| 50 | 10 | 10 | 5060.8 | 5060.9 | 5060.8 | 0.0 | 18,000 |
| 50 | 10 | 15 | 5060.8 | 5060.9 | 5060.8 | 0.0 | 19,500 |

**Table 2**
Sensitivity analysis of AMGSA for the 10-bar planar truss under the loading Case 2.

| $N$ | $N_S$ | $\lambda$ | Weight (lb) | | | | # of analyses |
|---|---|---|---|---|---|---|---|
| | | | Best | Worst | Mean | SD | |
| 20 | 4 | 10 | 4677.5 | 4680.0 | 4678.3 | 0.7 | 9000 |
| 20 | 4 | 15 | 4677.1 | 4678.0 | 4677.5 | 0.3 | 10,500 |
| 20 | 5 | 10 | 4677.7 | 4680.0 | 4678.7 | 0.3 | 9000 |
| 20 | 5 | 15 | 4677.0 | 4678.2 | 4677.4 | 0.2 | 10,500 |
| 20 | 10 | 10 | 4677.5 | 4680.6 | 4678.9 | 0.3 | 9000 |
| 20 | 10 | 15 | 4677.0 | 4679.9 | 4678.6 | 0.3 | 10,500 |
| 30 | 5 | 5 | 4677.4 | 4682.4 | 4679.3 | 1.4 | 10,500 |
| 30 | 5 | 10 | 4677.3 | 4682.2 | 4679.2 | 1.4 | 12,000 |
| 30 | 5 | 15 | 4677.3 | 4681.1 | 4678.8 | 1.2 | 13,500 |
| 30 | 6 | 5 | 4677.4 | 4686.6 | 4679.9 | 2.2 | 10,500 |
| 30 | 6 | 10 | 4677.3 | 4684.7 | 4680.1 | 1.8 | 12,000 |
| 30 | 6 | 15 | 4677.3 | 4681.1 | 4679.0 | 1.0 | 13,500 |
| 30 | 10 | 5 | 4677.4 | 4683.9 | 4679.7 | 1.2 | 10,500 |
| 30 | 10 | 10 | 4677.4 | 4683.9 | 4679.7 | 1.2 | 12,000 |
| 30 | 10 | 15 | 4677.0 | 4679.1 | 4678.9 | 0.2 | 13,500 |
| 40 | 5 | 5 | 4677.5 | 4682.6 | 4678.8 | 1.3 | 13,500 |
| 40 | 5 | 10 | 4677.3 | 4681.3 | 4678.8 | 1.1 | 15,000 |
| 40 | 5 | 15 | 4677.2 | 4680.9 | 4678.8 | 1.0 | 16,500 |
| 40 | 8 | 5 | 4677.4 | 4686.1 | 4679.0 | 2.0 | 13,500 |
| 40 | 8 | 10 | 4677.3 | 4682.1 | 4678.8 | 1.2 | 15,000 |
| 40 | 8 | 15 | 4677.3 | 4680.6 | 4678.0 | 0.9 | 16,500 |
| 40 | 10 | 5 | 4677.7 | 4683.3 | 4679.2 | 1.3 | 13,500 |
| 40 | 10 | 10 | 4677.3 | 4682.1 | 4679.1 | 1.2 | 15,000 |
| 40 | 10 | 15 | 4677.0 | 4679.1 | 4678.7 | 0.2 | 16,500 |
| 50 | 5 | 5 | 4677.6 | 4683.1 | 4678.7 | 1.3 | 16,500 |
| 50 | 5 | 10 | 4677.3 | 4681.6 | 4678.5 | 1.0 | 18,000 |
| 50 | 5 | 15 | 4677.2 | 4681.2 | 4678.7 | 1.0 | 19,500 |
| 50 | 10 | 5 | 4677.5 | 4680.5 | 4678.6 | 0.9 | 16,500 |
| 50 | 10 | 10 | 4677.3 | 4679.6 | 4678.2 | 0.7 | 18,000 |
| 50 | 10 | 15 | 4677.0 | 4678.8 | 4677.3 | 0.1 | 19,500 |

**Table 3**
Comparison of MGSA and AMGSA for the 10-bar planar truss. Note: the best, worst, mean and SD data are in lbs.

| Loading condition | $N$ | MGSA [24] | | | | AMGSA (20,10,15) | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Best | Worst | Mean | SD | Best | Worst | Mean | SD |
| Case 1 | 20 | 5060.9 | 5062.4 | 5061.3 | 0.4 | 5060.9 | 5061.0 | 5060.9 | 0.0 |
| | 30 | 5060.9 | 5062.2 | 5061.2 | 0.3 | 5060.9 | 5061.0 | 5060.9 | 0.0 |
| | 40 | 5060.9 | 5061.7 | 5061.2 | 0.2 | 5060.9 | 5061.0 | 5060.9 | 0.0 |
| | 50 | 5060.9 | 5061.7 | 5061.1 | 0.2 | 5060.9 | 5060.9 | 5060.8 | 0.0 |
| Case 2 | 20 | 4677.3 | 4681.1 | 4678.8 | 1.2 | 4677.1 | 4679.9 | 4678.6 | 0.3 |
| | 30 | 4677.2 | 4681.1 | 4678.9 | 1.1 | 4677.1 | 4679.1 | 4678.9 | 0.2 |
| | 40 | 4677.2 | 4680.9 | 4678.8 | 1.0 | 4677.1 | 4679.1 | 4678.7 | 0.2 |
| | 50 | 4677.2 | 4679.6 | 4678.2 | 1.0 | 4677.0 | 4678.8 | 4677.3 | 0.1 |

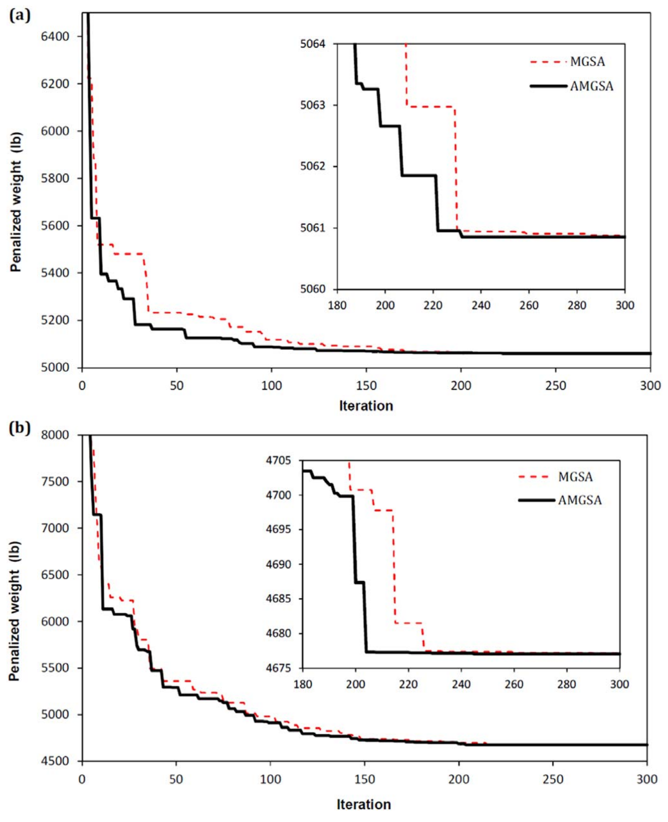

**Fig. 4.** Convergence histories of MGSA and AMGSA for the 10-bar truss under a) Case 1 and b) Case 2.

## 5. Breeder genetic algorithm (BGA) mutation

The mutation operator of BGA [26] was used in this study to enhance the local search capability of MGSA. BGA is a stochastic search method that can be applied to both discrete and continuous functions. BGA adopts truncation selection to select potential candidate solutions for recombination (crossover) while using the crossover and mutation operators of GA, essentially making it an amalgam of the evolution strategy and GA characteristics. BGA mutates a parent string $\mathbf{X}_i(t) = \{x_i^1, \ldots, x_i^d, \ldots, x_i^D\}$ by first selecting a variable $x_i$ with the probability $p_m$. Typically, $p_m = 1/D$ is used in order to ensure that on average at least one variable is mutated. The mutated variable $x_i'$ is then given by:

$$x_i' = x_i \pm range_i.\ \delta \tag{13}$$

where $range_i$ defines the mutation range and is normally set to $0.1(u_i - l_i)$, where $u_i$ and $l_i$ are the upper and lower bounds of $x_i$, respectively; the ± sign has the same probability of 0.5 for plus and minus; and $\delta$ is given by:

$$\delta = \sum_{k=0}^{15} \alpha_k 2^{-k} \qquad \alpha_k \in 0,\ 1 \tag{14}$$

where $\alpha_k$ are coefficients that are initially set to 0 and then mutated to 1 with the probability $p_\delta = 1/16$. This mutation scheme is able to locate the optimum $x_i$ up to a precision of $2^{-15} \times range_i$.

## 6. Accelerated MGSA (AMGSA)

As previously stated, AMGSA is the result of synthesizing the SPX, the BGA mutation scheme, and the feasibility-based rule with MGSA in order to improve ***gbest*** in each iteration of the optimization process.

The overall framework of AMGSA involves two main search stages, as summarized in the flowchart shown in Fig. 2. The first stage includes carrying out a global search using MGSA. In the second stage, first, $\mu$ agents are randomly selected from $\boldsymbol{pbest} = \{\boldsymbol{pbest}_1, \boldsymbol{pbest}_2, \ldots, \boldsymbol{pbest}_N\}$ and sorted in an ascending order according to their degree of constrain violation, $P_f$. These agents are then mated using the SPX so as to generate $\lambda$ agents (i.e. $\boldsymbol{pbest}' = \{\boldsymbol{pbest}'_1, \boldsymbol{pbest}'_2, \ldots, \boldsymbol{pbest}'_\lambda\}$). This operation is particularly effective when the set ***pbest*** contains very similar agents. Subsequently, the generated set is perturbed using the BGA mutation operator. Eventually, ***gbest*** is updated by comparing $W^p(\boldsymbol{gbest})$ with $W^p(\boldsymbol{pbest}'_i)(i = 1, 2, \ldots, \lambda)$ based on the feasibility-based rule.

## 7. Numerical examples

Seven benchmark truss optimization examples were selected to demonstrate the efficiency and efficacy of AMGSA and compare its performance with that of several other optimization metaheuristics. In all examples, $t_{max}$, $G_0$, and $\beta$ (Eq. (9)) and $\varepsilon$ (Eq. (11)) were set to 300, 100, 20, and 10 respectively. These values were selected based on the authors' experience and the general recommendations given in Refs. [15,36]. Due to the stochastic nature of the optimization process, the lowest (best), highest (worst), and mean structural weights of 20 independent runs that were carried out for each example together with their corresponding standard deviation (SD) are reported. Both optimization and finite element analysis of the examples were carried out using MATLAB®.

### 7.1. Ten-bar planar truss

The 10-bar planar truss shown in Fig. 3 was considered as the first design example. A Young's modulus of 10,000 ksi and a density of 0.1 lb/in$^3$ were assumed for the truss members. The allowable tensile and compressive stresses for all members were set to 25 ksi. The displacements of all free nodes in the $X$ and $Y$ directions were restricted within ± 2.0 in. The cross-sectional areas of truss members with a minimum value of 0.1 in$^2$ were considered as design variables. Two loading conditions were considered: Case 1, where $P_1 = 100$ kips and $P_2 = 0$; and Case 2, where $P_1 = 150$ kips and $P_2 = 50$ kips.

A sensitivity analysis was first carried out in order to find the best combination of AMGSA parameters that would maximize its performance. To this end, the population size $N$, the subpopulation size $N_s$, and the number of the offspring created λ denoted here by the triple ($N$, $N_s$, λ) were regulated to maximize the performance of AMGSA. An ensemble of 30 combinations was considered, and for each combination 20 independent optimization runs were performed. AMGSA results in terms of the best, worst, and mean of structural weights and the corresponding standard deviation (SD) as well as the number of structural analyses required in the optimization process are summarized in Tables 1 and 2.

Results shown in the tables indicate that given population and

**Table 4**
Performance comparison of AMGSA (20,10,15) with several other metaheuristics for the 10-bar planar truss under the loading condition Case 1.

| | HS [14] | PSO [38] | PSOPC [34] | HPSO [33] | HPSACO [39] | IHS [37] | ABC-AP [40] | EHS [41] | SAHS [41] | TLBO [42] | MGSA [24] | AMGSA (this study) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $A_1$ (in$^2$) | 30.2 | 33.5 | 30.6 | 30.7 | 30.3 | 30.5 | 30.5 | 30.2 | 30.4 | 30.4 | 30.5 | 30.5 |
| $A_2$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_3$ (in$^2$) | 22.7 | 22.8 | 23.0 | 23.2 | 23.4 | 23.2 | 23.2 | 22.7 | 23.1 | 23.2 | 23.2 | 23.2 |
| $A_4$ (in$^2$) | 15.3 | 14.4 | 15.1 | 15.2 | 15.5 | 15.2 | 15.2 | 15.3 | 15.5 | 15.4 | 15.2 | 15.2 |
| $A_5$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_6$ (in$^2$) | 0.5 | 0.1 | 0.5 | 0.6 | 0.5 | 0.6 | 0.6 | 0.5 | 0.5 | 0.6 | 0.5 | 0.6 |
| $A_7$ (in$^2$) | 7.5 | 7.5 | 7.5 | 7.5 | 7.4 | 7.5 | 7.5 | 7.6 | 7.5 | 7.4 | 7.5 | 7.5 |
| $A_8$ (in$^2$) | 21.6 | 20.5 | 21.2 | 21.0 | 21.1 | 21.0 | 21.1 | 21.6 | 21.2 | 21.0 | 21.1 | 21.0 |
| $A_9$ (in$^2$) | 21.5 | 20.4 | 21.2 | 21.5 | 21.2 | 21.5 | 21.5 | 21.5 | 21.3 | 21.5 | 21.1 | 21.6 |
| $A_{10}$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Weight (lb) | 5057.9 | 5024.2 | 5061.0 | 5060.9 | 5056.6 | 5060.8 | 5060.9 | 5062.4 | 5061.4 | 5061.0 | 5060.9 | 5060.9 |
| Mean weight (lb) | NR[a] | NR | NR | NR | NR | NR | NR | 5063.7 | 5061.9 | 5062.0 | 5,0601.1 | 5060.9 |
| SD (lb) | NR | NR | NR | NR | NR | NR | NR | 1.9 | 0.7 | 0.8 | 0.2 | 0.0 |
| Constraint violation (%) | 0.2 | 1.9 | None | None | 0.099 | None | None | None | None | None | None | None |
| # of analyses | 20,000 | NR | 150,000 | 125,000 | 10,650 | 1350 | 500,000 | 9791 | 7081 | 16,872 | 15,000 | 10,500 |

[a] Not reported.

**Table 5**
Performance comparison of AMGSA (20,10,15) with several other metaheuristics for the 10-bar planar truss under the loading condition Case 2.

| | HS [14] | PSO [38] | PSOPC [34] | HPSO [33] | HPSACO [39] | ABC-AP [40] | EHS [41] | SAHS [41] | TLBO [42] | MGSA [24] | AMGSA (this study) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $A_1$ (in$^2$) | 23.3 | 22.9 | 23.5 | 23.4 | 23.2 | 23.5 | 23.6 | 23.5 | 23.5 | 23.6 | 23.5 |
| $A_2$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_3$ (in$^2$) | 25.7 | 25.4 | 25.3 | 25.5 | 24.6 | 25.2 | 25.4 | 25.4 | 25.4 | 25.4 | 25.4 |
| $A_4$ (in$^2$) | 14.5 | 14.4 | 14.4 | 14.3 | 14.2 | 14.4 | 14.5 | 14.5 | 14.5 | 14.2 | 14.3 |
| $A_5$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_6$ (in$^2$) | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 |
| $A_7$ (in$^2$) | 12.2 | 12.3 | 12.4 | 12.4 | 12.5 | 12.4 | 12.4 | 12.4 | 12.3 | 12.4 | 12.4 |
| $A_8$ (in$^2$) | 12.6 | 12.9 | 12.7 | 13.0 | 12.9 | 12.9 | 12.7 | 12.7 | 12.7 | 12.8 | 12.8 |
| $A_9$ (in$^2$) | 20.4 | 20.7 | 20.3 | 20.4 | 21.0 | 20.3 | 20.3 | 20.3 | 20.4 | 20.5 | 20.3 |
| $A_{10}$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Weight (lb) | 4668.8 | 4679.5 | 4677.7 | 4677.3 | 4675.8 | 4677.1 | 4679.0 | 4678.8 | 4678.3 | 4677.2 | 4677.0 |
| Mean weight (lb) | NR | NR | NR | NR | NR | NR | 4681.6 | 4680.0 | 4680.1 | 4678.1 | 4677.2 |
| SD (lb) | NR | NR | NR | NR | NR | NR | 2.5 | 1.9 | 1.0 | 0.9 | 0.1 |
| Constraint violation (%) | 0.2 | None | None | None | 0.8 | None | None | None | None | None | None |
| # of analyses | 15,000 | 150,000 | 150,000 | 125,000 | 9925 | 500,000 | 11,402 | 7081 | 14,857 | 15,000 | 10,500 |

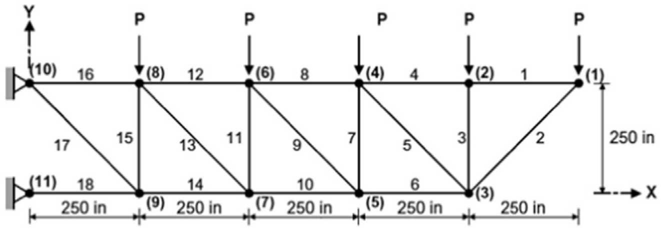

**Fig. 5.** The 18-bar planar truss.

subpopulation sizes, with a few exceptions due to the stochastic nature of the optimization algorithm, increasing the number of offspring $\lambda$ decreases the mean structural weight and results in smaller standard deviations, which indicate improved consistency across several experiments. For example, the mean and standard deviation of the combination (40, 10, $\lambda$) in Case 1 reduce from 5061.178 lb and 0.280 lb to 5060.901 lb and 0.041 lb, respectively, as $\lambda$ increases from 5 to 15. Another observation from Tables 1 and 2 is that, given the population size and $\lambda$, results improve with increasing the subpopulation size particularly for population sizes greater than 30. This suggests that the agents of a sufficiently large population should be assigned to an ideal number of subpopulations such that the population and number of subpopulations are large enough for the effective exploration of the entire search space, while each subpopulation contains adequate agents

**Table 6**
Performance comparison of AMGSA (20,10,15) with a few other optimization techniques for the 18-bar planar truss.

| | Multiplier method [43] | HS [14] | MGSA [24] | AMGSA (this study) |
|---|---|---|---|---|
| $A_{1,4,8,12,16}$ (in$^2$) | 10.0 | 10.0 | 10.0 | 10.0 |
| $A_{2,6,10,14,18}$ (in$^2$) | 21.6 | 21.6 | 21.6 | 21.6 |
| $A_{3,7,11,15}$ (in$^2$) | 12.5 | 12.5 | 12.5 | 12.5 |
| $A_{5,9,13,17}$ (in$^2$) | 7.1 | 7.1 | 7.1 | 7.1 |
| Weight (lb) | 6430.0 | 6421.9 | 6430.5 | 6430.5 |
| Mean weight (lb) | NR | NR | 6431.0 | 6430.5 |
| SD (lb) | NR | NR | 0.0 | 0.0 |
| Constraint violation (%) | None | 0.0014 | None | None |
| # of structure analyses | NR | 2000 | 15,000 | 10,500 |

for sharing information and exploiting potential zones in the search space. These allocations are essentially made based on the user's experience and the computational power available. In this study, the results corresponding to the triple (20,10,15) are frequently used for comparison with those of other techniques. This is because this triple provides an adequately good solution with a relatively low number of structural analyses.

The results corresponding to the triple (20,10,15) extracted from

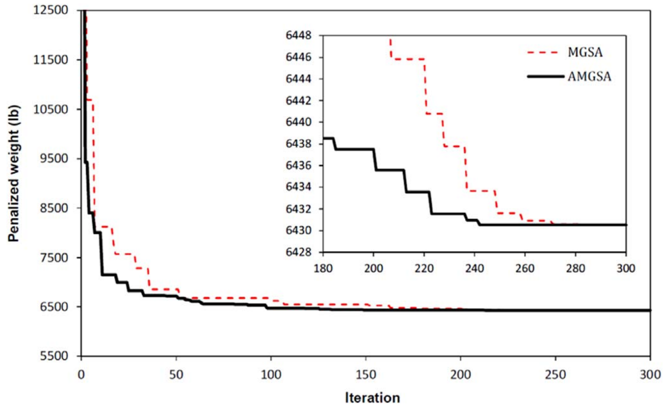

**Fig. 6.** Convergence histories of MGSA and AMGSA for the 18-bar truss.

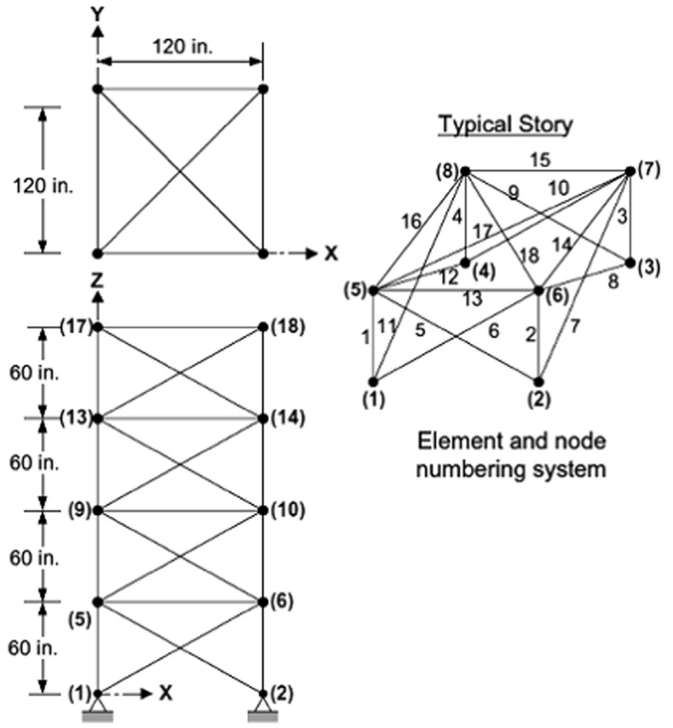

**Fig. 7.** The 72-bar space truss.

**Table 7**
Load cases for the 72-bar space truss.

| Node | Case 1 (kips) | | | Case 2 (kips) | | |
|---|---|---|---|---|---|---|
| | $P_X$ | $P_Y$ | $P_Z$ | $P_X$ | $P_Y$ | $P_Z$ |
| 17 | 5.0 | 5.0 | −5.0 | 0.0 | 0.0 | −5.0 |
| 18 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | −5.0 |
| 19 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | −5.0 |
| 20 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | −5.0 |

Tables 1 and 2 are compared in Table 3 with those obtained using MGSA [24] in order to show the influence of the SPX and the BGA mutation operator on the performance of MGSA. The comparison indicates that this incorporation results in consistently lighter designs and considerably smaller standard deviations. In addition, this incorporation accelerates the optimization procedure. As observed from the example convergence histories of MGSA and AMGSA shown in Fig. 4, AMGSA was able to find the minimum weights after 232 and 245 iterations (8120 and 8575 structural analyses) for Cases 1 and 2, respectively. In contrast, MGSA required over 259 and 261 iterations (over 9065 and 9135 structural analyses) to cluster candidates around those of the highest fitness.

A comparison between the performances of AMGSA and several other metaheuristics in optimizing the 10-bar truss is presented in Tables 4 and 5. The bases of comparison include cross-sectional areas of the members of the combination (20,10,15) and its total weight, the mean weight and its corresponding standard deviation across the runs carried out, the percentage of constraint violations, and the total number of analyses executed to converge to the optimum design. The comparison indicates that, except for the improved harmony search (IHS) algorithm [37], AMGSA achieves a lighter structure at a fewer number of analyses and with a lower standard deviation. The far fewer number of analyses required by IHS is due to a mechanism in it that directs the search toward the optimum using gradient descent.

## 7.2. Eighteen-bar planar truss

AMGSA was further benchmarked against other optimization techniques using the 18-bar planar truss shown in Fig. 5 which was originally studied by Imai and Schmit [43]. A Young's modulus of 10,000 ksi and a density of 0.1 lb/in$^3$ were considered for the truss members. In addition to an allowable stress of 20 ksi for both tensile and compressive members, stress in any compressive member $i$ was not allowed to exceed the Euler's critical buckling stress defined as:

$$\sigma_i^b = \frac{KEA_i}{L_i^2} \tag{15}$$

where $K$ is the member effective length factor assumed here as 4; $A_i$ and $L_i$ are the member's cross-sectional area and unsupported length; and $E$ is the Young's modulus. The truss was subjected to a set of 20-kip point loads acting downward at the upper nodes. The members were ensured to have a cross-sectional area greater than 0.1 in$^2$. The members were assigned to four groups as: Group 1: 1, 4, 8, 12, 16; Group 2: 2, 6, 10, 14, 18; Group 3: 3, 7, 11, 15; and Group 4: 5, 9, 13, 17.

The performances of AMGSA, MGSA, and HS in optimizing the 18-bar truss are compared in Table 6, where the results obtained using the multiplier method [43] are also listed for reference. Similar to the previous examples, AMGSA finds the minimum weight without violating any constraints. In addition, it can be seen that the structural analyses required to converge to the optimum drops from 15,000 for MGSA to 10,500 for AMGSA. This arises from incorporating the SPX and the BGA mutation operator with MGSA which in turn improves its convergence rate (i.e. 243 iterations or 7290 structural analyses vs. 276 iterations or 13,800 structural analyses – Fig. 6).

## 7.3. Seventy-two-bar space truss

The 72-bar space truss shown in Fig. 7 was used to further examine the performance of AMGSA. The Young's modulus and density of the members were assumed equal to 10,000 ksi and 0.10 lb/in$^3$, respectively. The members were divided into 16 groups. The allowable compressive and tensile stresses were set to 25 ksi for all members. Displacement limits of ±0.25 in were imposed on all nodes in all directions. The truss was subjected to the two loading conditions listed in Table 7. Minimum cross-sectional areas of 0.1 in$^2$ and 0.01 in$^2$ were considered for loading cases 1 and 2, respectively.

A performance comparison based on the results summarized in Tables 8 and 9 indicates the relatively higher capability of AMGSA versus MGSA and several other optimization metaheuristics. In line with previous examples, AMGSA achieves the optimum design after a relatively reasonable number of structural analyses and without violating any constraints. In addition, the convergence histories shown

**Table 8**
Performance comparison of AMGSA (20,10,15) with some other metaheuristics for the 72-bar space truss (loading case 1).

| | HS [14] | PSO [38] | PSO [33] | PSOPC [33] | HPSACO [39] | BB-BC [44,45] | HBB–BC [46] | EHS [41] | SAHS [41] | TLBO [42] | MGSA [24] | AMGSA (this study) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $A_1$–$A_4$ | 1.8 | 1.7 | 41.8 | 1.9 | 1.9 | 1.9 | 1.9 | 2.0 | 1.9 | 1.9 | 1.9 | 1.9 |
| $A_5$–$A_{12}$ | 0.5 | 0.5 | 0.2 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{13}$–$A_{16}$ | 0.1 | 0.1 | 10.8 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{17}$–$A_{18}$ | 0.1 | 0.1 | 6.9 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{19}$–$A_{22}$ | 1.2 | 1.3 | 0.4 | 1.3 | 1.3 | 1.2 | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 |
| $A_{23}$–$A_{30}$ | 0.5 | 0.5 | 0.3 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{31}$–$A_{34}$ | 0.1 | 0.1 | 18.3 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{35}$–$A_{36}$ | 0.1 | 0.1 | 1.2 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{37}$–$A_{40}$ | 0.5 | 0.5 | 5.9 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{41}$–$A_{48}$ | 0.5 | 0.5 | 19.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{49}$–$A_{52}$ | 0.1 | 0.1 | 0.2 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{53}$–$A_{54}$ | 0.1 | 0.1 | 0.2 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{55}$–$A_{58}$ | 0.2 | 0.2 | 10.1 | 0.1 | 0.1 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 |
| $A_{59}$–$A_{66}$ | 0.5 | 0.5 | 7.3 | 0.5 | 0.5 | 0.6 | 0.5 | 0.5 | 0.6 | 0.5 | 0.5 | 0.5 |
| $A_{67}$–$A_{70}$ | 0.4 | 0.5 | 3.8 | 0.4 | 0.4 | 0.4 | 0.4 | 0.5 | 0.4 | 0.4 | 0.4 | 0.4 |
| $A_{71}$–$A_{72}$ | 0.6 | 0.6 | 18.2 | 0.5 | 0.5 | 0.6 | 0.6 | 0.6 | 0.5 | 0.6 | 0.6 | 0.6 |
| Weight (lb) | 370.3 | 381.9 | 6818.7 | 369.7 | 369.7 | 379.9 | 379.7 | 381.0 | 380.6 | 379.6 | 379.7 | 379.6 |
| Mean weight (lb) | NR | NR | NR | NR | NR | 382.1 | 381.8 | 383.5 | 382.4 | 380.2 | 379.6 | 379.6 |
| SD (lb) | NR | NR | NR | NR | NR | 1.9 | 1.2 | 1.9 | 1.4 | 0.4 | 0.2 | 0.0 |
| Constraint violation (%) | 0.2 | None | None | 39.051 | 39.075 | None | None | None | None | None | None | None |
| # of structure analyses | 20,000 | NR | 150,000 | 125,000 | 125,000 | 19,621 | 13,200 | 15,044 | 13,742 | 19,709 | 15,000 | 10,500 |

**Table 9**
Performance comparison of AMGSA (20,10,15) with some other metaheuristics for the 72-bar space truss (loading case 2).

| | HS [14] | PSO [33] | PSOPC [33] | HPSO [33] | ABC-AP [32] | EHS [41] | SAHS [41] | TLBO [42] | MGSA [24] | AMGSA (this study) |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_{1�4}$ (in$^2$) | 2.0 | 41.8 | 1.9 | 1.9 | 1.9 | 2.0 | 1.9 | 1.9 | 1.9 | 1.9 |
| $A_{5-12}$ (in$^2$) | 0.5 | 0.2 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{13-16}$ (in$^2$) | 0.1 | 10.8 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{17,18}$ (in$^2$) | 0.1 | 0.7 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{19-22}$ (in$^2$) | 1.2 | 22.1 | 1.1 | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 | 1.3 |
| $A_{23-30}$ (in$^2$) | 0.5 | 0.3 | 0.6 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{31-34}$ (in$^2$) | 0.1 | 1.7 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{35,36}$ (in$^2$) | 0.1 | 10.3 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{37-40}$ (in$^2$) | 0.5 | 0.6 | 0.6 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{41-48}$ (in$^2$) | 0.5 | 12.9 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{49-52}$ (in$^2$) | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{53,54}$ (in$^2$) | 0.2 | 0.2 | 0.2 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| $A_{55-58}$ (in$^2$) | 0.2 | 29.0 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 |
| $A_{59-66}$ (in$^2$) | 0.5 | 0.6 | 0.6 | 0.5 | 0.5 | 0.6 | 0.5 | 0.5 | 0.5 | 0.5 |
| $A_{67-70}$ (in$^2$) | 0.5 | 3.0 | 0.5 | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 |
| $A_{71,72}$ (in$^2$) | 0.6 | 1.7 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 |
| Weight (lb) | 364.3 | 5417.0 | 368.5 | 364.9 | 363.8 | 364.4 | 364.1 | 363.8 | 363.9 | 363.8 |
| Mean weight (lb) | NR | NR | NR | NR | NR | 366.8 | 366.6 | 364.4 | 363.9 | 363.9 |
| SD (lb) | NR | NR | NR | NR | NR | 2.0 | 2.0 | 0.5 | 0.0 | 0.0 |
| Constraint violation (%) | 0.1 | None | None | 0.1 | None | None | None | None | None | None |
| # of structure analyses | 20,000 | 150,000 | 125,000 | 125,000 | 400,000 | 13,755 | 12,852 | 17,954 | 15,000 | 10,500 |

in Fig. 8 indicate the higher speed of AMGSA in finding the optimum solution.

### *7.4. Two-hundred-bar planar truss*

A 200-bar planar truss shown in Fig. 9 was the largest structure used in this study to investigate the performance of AMGSA. The Young's modulus and density of the members were assumed equal to 30,000 ksi and 0.283 lb/in$^3$, respectively. The members were assigned to 29 groups as listed in Table 10. A minimum cross-sectional area of 0.1 in$^2$ and an allowable stress of 10 ksi were assumed for both compressive and tensile members, while no nodal displacement limit was imposed. The truss was subjected to two simultaneously-applied sets of loads: 1) a set of 1-kip forces acting at nodes 1, 6, 15, 20, 29, 34, 43, 48, 57, 62, and 71 along the positive X-direction, and 2) a set of 10-kip forces acting downward at nodes 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 15, 16, 17, 18, and 19.

The results of AMGSA, MGSA, and some other optimization techniques for the 200-bar truss are compared in Table 11. Consistent with the previous examples, AMGSA finds the minimum weight without violating any constraints. Results also indicate that incorporating the SPX and the BGA mutation operator with MGSA improves its convergence rate (i.e. 281 iterations or 9835 function evaluations vs. 294 iterations or 14,700 function evaluations – Fig. 10).

Comparison of the optimization results of the 72-bar space truss and the 200-bar planar truss (16 vs. 29 design variables) shows that the efficiency of AMGSA becomes more pronounced as the problem size increases.

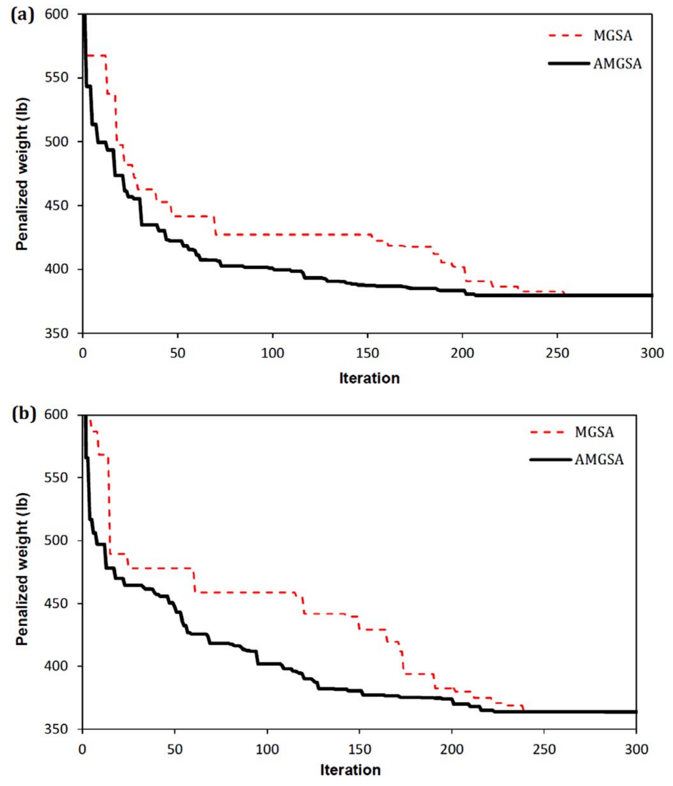

**Fig. 8.** Convergence histories of MGSA and AMGSA for the 72-bar truss under a) Case 1 and b) Case 2.

## 8. Conclusions

The simplex crossover (SPX) and the operator mutation of the breeder genetic algorithm (BGA) were incorporated with the multi-gravitational search algorithm (MGSA) so as to improve its exploration and exploitation capabilities. Several benchmark truss optimization examples were used to demonstrate the efficiency of the resulting algorithm, denoted as the accelerated MGSA (AMGSA). A sensitivity analysis was carried out in order to find the best combination of the population size $N$, the subpopulation size $N_s$, and the number of the offspring created λ denoted here by the triple ($N$, $N_s$, λ) that would maximize the performance of AMGSA at a reasonable computational cost. The combination (20,10,15) was found to return an adequately good yet computationally inexpensive solution. Results of the example problems consistently showed a good exploration-exploitation balance for AMGSA and its competitive promise in effectively and efficiently solving large-scale optimization problems. This improvement was found to become more pronounced as the size of the problem increases. The BGA mutation scheme was shown to be particularly effective when the best previous positions of agents are very similar. In this case, this operator slightly changes the agents, thus improving the algorithm's local search capability. Compared to MGSA, AMGSA offered lighter designs with considerably lower standard deviations while requiring fewer structural analyses and, therefore, less computational burden. AMGSA was further evaluated by comparing its performance with that of several other metaheuristics. Similarly, results indicated that AMGSA requires considerably fewer structural analyses to achieve the optimal solution, making the optimization process significantly faster, especially for large-scale structures.

The technique proposed in this study is not limited to the size optimization of truss structures with discrete/continuous sizing variables. Similar to the original GSA that has successfully been used in a variety of

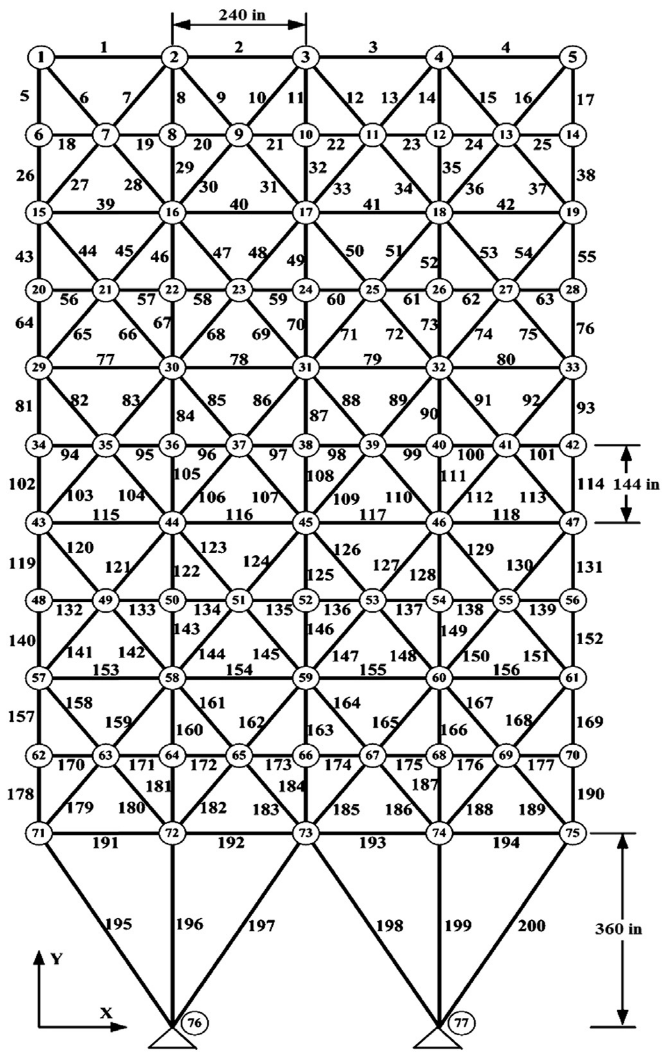

**Fig. 9.** The 200-bar planar truss.

**Table 10**
Groups and their constituting members for the 200-bar planar truss.

| Group | Member ID | Group | Member ID |
|---|---|---|---|
| 1 | 1, 2, 3, 4 | 16 | 82, 83, 85, 86, 88, 89, 91, 92, 103, 104, 106, 107, 109, 110, 112, 113 |
| 2 | 5, 8, 11, 14, 17 | 17 | 115, 116, 117, 118 |
| 3 | 19, 20, 21, 22, 23, 24 | 18 | 119, 122, 125, 128, 131 |
| 4 | 18, 25, 56, 63, 94, 101, 132, 139, 170, 177 | 19 | 133, 134, 135, 136, 137, 138 |
| 5 | 26, 29, 32, 35, 38 | 20 | 140, 143, 146, 149, 152 |
| 6 | 6, 7, 9, 10, 12, 13, 15, 16, 27, 28, 30, 31, 33, 34, 36, 37 | 21 | 120, 121, 123, 124, 126, 127, 129, 130, 141, 142, 144, 145, 147, 148, 150, 151 |
| 7 | 39, 40, 41, 42 | 22 | 153, 154, 155, 156 |
| 8 | 43, 46, 49, 52, 55 | 23 | 157, 160, 163, 166, 169 |
| 9 | 57, 58, 59, 60, 61, 62 | 24 | 171, 172, 173, 174, 175, 176 |
| 10 | 64, 67, 70, 73, 76 | 25 | 178, 181, 184, 187, 190 |
| 11 | 44, 45, 47, 48, 50, 51, 53, 54, 65, 66, 68, 69, 71, 72, 74, 75 | 26 | 158, 159, 161, 162, 164, 165, 167, 168, 179, 180, 182, 183, 185, 186, 188, 189 |
| 12 | 77, 78, 79, 80 | 27 | 191, 192, 193, 194 |
| 13 | 81, 84, 87, 90, 93 | 28 | 195, 197, 198, 200 |
| 14 | 95, 96, 97, 98, 99, 100 | 29 | 196, 199 |
| 15 | 102, 105, 108, 111, 114 | | |

**Table 11**
Performance comparison of AMGSA (20,10,15) with some other metaheuristics for the 200-bar truss.

| | HS [14] | PSO [33] | PSOPC [33] | HPSO [33] | CMLPSA [47] | EHS [41] | SAHS [41] | TLBO [42] | MGSA [24] | AMGSA (this study) |
|---|---|---|---|---|---|---|---|---|---|---|
| Group 1 (in$^2$) | 0.1 | 0.8 | 0.8 | 0.1 | 0.1 | 0.2 | 0.2 | 0.1 | 0.1 | 0.1 |
| Group 2 (in$^2$) | 1.0 | 2.4 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 | 0.9 |
| Group 3 (in$^2$) | 0.1 | 4.3 | 1.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 4 (in$^2$) | 0.1 | 5.7 | 1.0 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 5 (in$^2$) | 1.9 | 4.0 | 2.1 | 1.9 | 1.9 | 1.9 | 1.9 | 1.9 | 1.9 | 1.9 |
| Group 6 (in$^2$) | 0.3 | 0.6 | 0.4 | 0.3 | 0.3 | 0.3 | 0.3 | 0.3 | 0.3 | 0.3 |
| Group 7 (in$^2$) | 0.1 | 5.6 | 1.0 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 8 (in$^2$) | 3.0 | 9.2 | 2.8 | 3.0 | 3.1 | 3.2 | 3.1 | 3.1 | 3.1 | 3.1 |
| Group 9 (in$^2$) | 0.1 | 4.5 | 1.0 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 10 (in$^2$) | 4.2 | 4.6 | 3.8 | 3.9 | 4.1 | 4.2 | 4.1 | 4.2 | 4.1 | 4.1 |
| Group 11 (in$^2$) | 0.4 | 0.6 | 0.5 | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 | 0.4 |
| Group 12 (in$^2$) | 0.4 | 18.8 | 0.4 | 0.5 | 0.2 | 0.2 | 0.2 | 0.2 | 0.1 | 0.2 |
| Group 13 (in$^2$) | 5.2 | 6.0 | 5.3 | 5.0 | 5.4 | 5.4 | 5.4 | 5.4 | 5.4 | 5.4 |
| Group 14 (in$^2$) | 0.2 | 0.1 | 1.0 | 1.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 15 (in$^2$) | 6.2 | 8.2 | 6.0 | 6.0 | 6.4 | 6.4 | 6.4 | 6.4 | 6.4 | 6.4 |
| Group 16 (in$^2$) | 0.7 | 0.3 | 0.8 | 0.8 | 0.6 | 0.6 | 0.6 | 0.6 | 0.5 | 0.6 |
| Group 17 (in$^2$) | 0.1 | 11.2 | 0.6 | 0.7 | 0.1 | 0.2 | 0.2 | 0.2 | 0.4 | 0.1 |
| Group 18 (in$^2$) | 7.8 | 7.1 | 8.2 | 7.4 | 8.0 | 8.0 | 8.0 | 8.0 | 8.0 | 8.0 |
| Group 19 (in$^2$) | 0.1 | 4.5 | 1.0 | 0.7 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 20 (in$^2$) | 8.8 | 9.2 | 9.2 | 8.3 | 9.0 | 9.0 | 9.0 | 9.0 | 9.0 | 9.0 |
| Group 21 (in$^2$) | 0.7 | 2.8 | 1.5 | 0.7 | 0.7 | 1.2 | 0.7 | 0.7 | 0.8 | 0.7 |
| Group 22 (in$^2$) | 1.6 | 0.6 | 1.8 | 1.0 | 0.4 | 0.5 | 0.4 | 0.5 | 0.2 | 0.3 |
| Group 23 (in$^2$) | 11.0 | 16.2 | 10.6 | 10.8 | 10.9 | 10.9 | 10.9 | 10.9 | 11.0 | 10.8 |
| Group 24 (in$^2$) | 0.1 | 0.5 | 1.0 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 | 0.1 |
| Group 25 (in$^2$) | 12.1 | 16.2 | 12.5 | 11.7 | 11.9 | 12.0 | 11.9 | 11.9 | 12.0 | 11.8 |
| Group 26 (in$^2$) | 1.6 | 1.0 | 2.0 | 1.4 | 1.0 | 1.1 | 1.0 | 1.1 | 0.9 | 1.0 |
| Group 27 (in$^2$) | 5.0 | 3.6 | 4.5 | 5.0 | 6.7 | 6.5 | 6.6 | 6.5 | 6.8 | 7.0 |
| Group 28 (in$^2$) | 9.4 | 8.4 | 9.8 | 8.8 | 10.8 | 10.8 | 10.8 | 10.8 | 10.9 | 11.0 |
| Group 29 (in$^2$) | 15.1 | 15.6 | 14.5 | 14.7 | 13.8 | 13.9 | 13.9 | 13.9 | 13.8 | 13.7 |
| Weight (lb) | 25,447.1 | 24,081.4 | 28,537.8 | 25,156.5 | 25,445.6 | 25,542.5 | 25,491.9 | 25,488.2 | 25,463.4 | 25,461.0 |
| Mean weight (lb) | NR | NR | NR | NR | NR | 25659.7 | 25610.2 | 25533.1 | 25559.5 | 25547.4 |
| SD (lb) | NR | NR | NR | NR | NR | 164.2 | 141.8 | 27.4 | 159.5 | 110.1 |
| Constraint violation (%) | 3.7 | 39.1 | 7.4 | 9.9 | 0.1 | None | None | None | None | None |
| # of structure analyses | 48,000 | 150,000 | 150,000 | 9875 | 9650 | 22,851 | 19,670 | 28,059 | 15,000 | 10,500 |

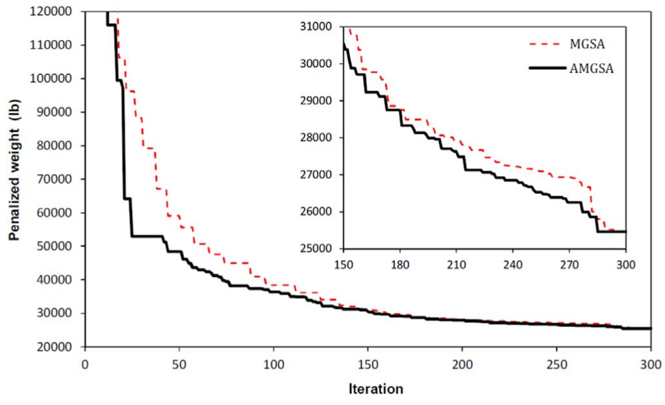

**Fig. 10.** Convergence histories of MGSA and AMGSA for the 200-bar truss.

optimization problems, this technique holds promise for application in any optimization problems, including plates, shells, and frame structures subjected to static and dynamic loads. Further research, however, is required to validate its robustness and efficacy in larger scales. Additional research is also required to accurately determine the time complexity of AMGSA. We presume that, if $n$ is the length of the string representing the population, the division of the population would require $O(n)$ time, and the interactions among the agents would accumulate $O(2^n)$ time, resulting in a time complexity of $O(n2^n)$.

## References

[1] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution – an updated survey, Swarm Evol. Comput. 27 (2016) 1–30. http://dx.doi.org/10.1016/j.swevo.2016.01.004.
[2] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search, Simulation 76 (2001) 60–68. http://dx.doi.org/10.1177/003754970107600201.
[3] S.K. Azad, O. Hasançebi, Discrete sizing optimization of steel trusses under multiple displacement constraints and load cases using guided stochastic search technique, Struct. Multidiscip. Optim. 52 (2015) 383–404. http://dx.doi.org/10.1007/s00158-015-1233-0.
[4] K. Sorensen, M. Sevaux, F. Glover, A history of metaheuristics, in: R. Marti, P. Pardalo, M. Resende (Eds.), Handbook of Heuristics, Springer, 2017 ⟨http://www.springer.com/us/book/9783319071237⟩ (Accessed 3 June 2016).
[5] I. Rechenberg, Evolution strategy: nature's way of optimization, in: Optim. Methods Appl. Possibilities Limit., Springer, Berlin, Heidelberg, 1989: pp. 106–126. http://dx.doi.org/10.1007/978-3-642-83814-9_6.
[6] L.J. Fogel, A.J. Owens, M.J. Walsh, Artificial Intelligence Through Simulated Evolution, Wiley, 1966.
[7] J.H. Holland, Adaptation in natural and artificial systems: an introductory analysis with applications to biologycontrol, and artificial intelligence, University of Michigan Press, 1975.
[8] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, 1st ed, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1989.
[9] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680. http://dx.doi.org/10.1126/science.220.4598.671.
[10] F. Glover, Applications of integer programming future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (1986) 533–549. http://dx.doi.org/10.1016/0305-0548(86)90048-1.
[11] A. Colorni, M. Dorigo, V. Maniezzo, Distributed optimization by ant colonies, IEEE Trans. Syst. Man Cybern. Part B Cybern. (1991) 134–142.
[12] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference Neural Netw., Perth, Australia, 1995: pp. 1942–1948.
[13] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359. http://dx.doi.org/10.1023/A:1008202821328.
[14] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, Comput. Methods Appl. Mech. Eng. 194 (2005) 3902–3933. http://dx.doi.org/10.1016/j.cma.2004.09.007.
[15] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, Inf. Sci. 179 (2009) 2232–2248. http://dx.doi.org/10.1016/j.ins.2009.03.004.
[16] P. Haghbayan, H. Nezamabadi-pour, S. Kamyab, A niche GSA method with nearest neighbor scheme for multimodal optimization, Swarm Evol. Comput., n.d. doi:http://dx.doi.org/10.1016/j.swevo.2017.03.002.
[17] A. Hatamlou, S. Abdullah, H. Nezamabadi-pour, A combined approach for

clustering based on K-means and gravitational search algorithms, Swarm Evol. Comput. 6 (2012) 47–52. http://dx.doi.org/10.1016/j.swevo.2012.02.003.
[18] S. Yazdani, H. Nezamabadi-pour, S. Kamyab, A gravitational search algorithm for multimodal optimization, Swarm Evol. Comput. 14 (2014) 1–14. http://dx.doi.org/10.1016/j.swevo.2013.08.001.
[19] J.K. Liu, Z.T. Wei, Z.R. Lu, Y.J. Ou, Structural damage identification using gravitational search algorithm, Struct. Eng. Mech. Int. J. 60 (2016) 729–747.
[20] S. Darzi, S.K. Tiong, M.T. Islam, H.R. Soleymanpour, S. Kibria, An experience oriented-convergence improved gravitational search algorithm for minimum variance distortionless response beamforming optimum, PLoS One 11 (2016) e0156749. http://dx.doi.org/10.1371/journal.pone.0156749.
[21] S. Mirjalili, S.Z. Mohd Hashim, H. Moradian Sardroudi, Training feed forward neural networks using hybrid particle swarm optimization and gravitational search algorithm, Appl. Math. Comput. 218 (2012) 11125–11137. http://dx.doi.org/10.1016/j.amc.2012.04.069.
[22] M. Khatibinia, E. Salajegheh, J. Salajegheh, M.J. Fadaee, Reliability-based design optimization of reinforced concrete structures including soil–structure interaction using a discrete gravitational search algorithm and a proposed metamodel, Eng. Optim. 45 (2013) 1147–1165. http://dx.doi.org/10.1080/0305215X.2012.725051.
[23] M. Khatibinia, S. Khosravi, A hybrid approach based on an improved gravitational search algorithm and orthogonal crossover for optimal shape design of concrete gravity dams, Appl. Soft Comput. 16 (2014) 223–233. http://dx.doi.org/10.1016/j.asoc.2013.12.008.
[24] M. Khatibinia, S. Sadegh Naseralavi, Truss optimization on shape and sizing with frequency constraints based on orthogonal multi-gravitational search algorithm, J. Sound Vib. 333 (2014) 6349–6369. http://dx.doi.org/10.1016/j.jsv.2014.07.027.
[25] S. Tsutsui, M. Yamamura, T. Higuchi, Multi-parent recombination with simplex crossover in real coded genetic algorithms, in: Proceedings of the 1st Annu. Conference Genet. Evol. Comput. - Vol. 1, Morgan Kaufmann Publishers Inc., Orlando, FL, 1999, pp. 657–664. ⟨http://dl.acm.org/citation.cfm?Id=2933923.2933986⟩ (Accessed 9 June 2016).
[26] Heinz Mühlenbein, Dirk Schlierkamp-Voosen, Predictive models for the breeder genetic algorithm I. Continuous parameter optimization, Evol. Comput. 1 (1993) 25–49. http://dx.doi.org/10.1162/evco.1993.1.1.25.
[27] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Methods Appl. Mech. Eng. 186 (2000) 311–338. http://dx.doi.org/10.1016/S0045-7825(99)00389-8.
[28] C.A. Coello Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, Comput. Methods Appl. Mech. Eng. 191 (2002) 1245–1287. http://dx.doi.org/10.1016/S0045-7825(01)00323-1.
[29] E. Salajegheh, S. Gholizadeh, M. Khatibinia, Optimal design of structures for earthquake loads by a hybrid RBF-BPSO method, Earthq. Eng. Eng. Vib. 7 (2008) 13–24. http://dx.doi.org/10.1007/s11803-008-0778-y.
[30] H. Yazdani, K. Hatami, E. Khosravi, Ant colony optimization method for design of piled-raft foundations, DFI J. - J. Deep Found. Inst. 7 (2013) 17–27. http://dx.doi.org/10.1179/dfi.2013.7.2.002.
[31] S. Gharehbaghi, M. Khatibinia, Optimal seismic design of reinforced concrete structures under time-history earthquake loads using an intelligent hybrid algorithm, Earthq. Eng. Eng. Vib. 14 (2015) 97–109. http://dx.doi.org/10.1007/s11803-015-0009-2.
[32] T.-Y. Chen, C.-J. Chen, Improvements of simple genetic algorithm in structural design, Int. J. Numer. Methods Eng. 40 (1997) 1323–1334. http://dx.doi.org/10.1002/(SICI)1097-0207(19970415)40:7 < 1323::AID-NME117 > 3.0.CO;2-T.
[33] L.J. Li, Z.B. Huang, F. Liu, Q.H. Wu, A heuristic particle swarm optimizer for optimization of pin connected structures, Comput. Struct. 85 (2007) 340–349. http://dx.doi.org/10.1016/j.compstruc.2006.11.020.
[34] S. He, Q.H. Wu, J.Y. Wen, J.R. Saunders, R.C. Paton, A particle swarm optimizer with passive congregation, Biosystems 78 (2004) 135–147. http://dx.doi.org/10.1016/j.biosystems.2004.08.003.
[35] J.K. Parrish, W.M. Hamner (Eds.), Animal Groups in Three Dimensions, Cambridge University Press, Cambridge, England, 1997 ⟨http://ebooks.cambridge.org/ref/id/CBO9780511601156⟩ (Accessed 7 June 2016).
[36] Y. Wang, Z. Cai, Y. Zhou, Z. Fan, Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique, Struct. Multidiscip. Optim. 37 (2008) 395–413. http://dx.doi.org/10.1007/s00158-008-0238-3.
[37] L. Lamberti, C. Pappalettere, An improved harmony-search algorithm for truss structure optimization, in: Proceedings of the Twelfth International Conference Civ. Struct. Environ. Eng. Comput., Stirlingshire, Scotland, 2009.
[38] R.E. Perez, K. Behdinan, Particle swarm approach for structural design optimization, Comput. Struct. 85 (2007) 1579–1588. http://dx.doi.org/10.1016/j.compstruc.2006.10.013.
[39] A. Kaveh, S. Talatahari, B. Farahmand Azar, An improved HPSACO for engineering optimum design problems, 12, 2011, pp. 133–141.
[40] M. Sonmez, Artificial Bee Colony algorithm for optimization of truss structures, Appl. Soft Comput. 11 (2011) 2406–2418. http://dx.doi.org/10.1016/j.asoc.2010.09.003.
[41] S.O. Degertekin, Improved harmony search algorithms for sizing optimization of truss structures, Comput. Struct. 92–93 (2012) 229–241. http://dx.doi.org/10.1016/j.compstruc.2011.10.022.
[42] S.O. Degertekin, M.S. Hayalioglu, Sizing truss structures using teaching-learning-based optimization, Comput. Struct. 119 (2013) 177–188. http://dx.doi.org/10.1016/j.compstruc.2012.12.011.
[43] Kanji Imai, Lucien A. Schmit, Configuration optimization of trusses, J. Struct. Div. 107 (1981) 745–756.
[44] C.V. Camp, Design of space trusses using Big Bang–Big Crunch optimization, J. Struct. Eng. 133 (2007) 999–1008. http://dx.doi.org/10.1061/(ASCE)0733-9445(2007)133:7(999).
[45] O.K. Erol, I. Eksin, A new optimization method: Big Bang–Big Crunch, Adv. Eng. Softw. 37 (2006) 106–111. http://dx.doi.org/10.1016/j.advengsoft.2005.04.005.
[46] A. Kaveh, S. Talatahari, Size optimization of space trusses using Big Bang–Big Crunch algorithm, Comput. Struct. 87 (2009) 1129–1140. http://dx.doi.org/10.1016/j.compstruc.2009.04.011.
[47] L. Lamberti, An efficient simulated annealing algorithm for design optimization of truss structures, Comput. Struct. 86 (2008) 1936–1953. http://dx.doi.org/10.1016/j.compstruc.2008.02.004.