Chapter 14
Global Sensitivity Analysis-Based Optimization Algorithm

14.1 Introduction

In this chapter a single-solution metaheuristic optimizer, namely, global sensitivity analysis-based (GSAB) algorithm [1], is presented that uses a basic set of mathematical techniques, namely, global sensitivity analysis. Sensitivity analysis (SA) studies the sensitivity of the model output with respect to its input parameters (Rahman [2]). This analysis is generally categorized as local SA and global SA techniques. While local SA studies the sensitivity of the model output about variations around a specific point, the global SA considers variations of the inputs within their entire feasibility space (Pianosi and Wagener [3], Zhai et al. [4]). One important feature of the GSA is factor prioritization (FP), which aims at ranking the inputs in terms of their relative contribution to output variability. The GSAB comprises of a single-solution optimization strategy and GSA-driven procedure, where the solution is guided by ranking the decision variables using the GSA approach, resulting in an efficient and rapid search. The proposed algorithm can be studied within the family of search algorithms such as the random search (RS) by Rastrigin [5], pattern search (PS) by Hooke and Jeeves [6], and vortex search (VS) by Dog and Ölmez [7] algorithms. In this method, similar to these algorithms, the search process is achieved in the specified boundaries. Contrary to these algorithms that use different functions for decreasing the search space, in the present method, the well-known GSA approach is employed to decrease the search boundaries. The minimization of an objective function is then performed by moving these search spaces into around the best global sample.

The present chapter is organized as follows: In Sect. 14.2, we describe the well-known variance-based sensitivity approach. In Sect. 14.3, the new method is presented. Two well-studied constrained optimization problems and three structural design examples are studied in Sect. 14.4. Conclusions are derived in Sect. 14.5.

A. Kaveh, *Advances in Metaheuristic Algorithms for Optimal Design of Structures*,
DOI 10.1007/978-3-319-46173-1_14

14.2 Background Study

The single-solution search algorithm proposed in this chapter uses the SA theory. In this section the main strategies used for taking SA into account which are based on the works of Zhai et al. [4], Saltelli et al. [8], and Archer et al. [9] are described.

14.2.1 Variance-Based Sensitivity Indices

The variance-based sensitivity indices can be estimated by a numerical model, $Y = g(\boldsymbol{X})$, with $\boldsymbol{X} = [x_1, x_2, \ldots, x_n]$ being the input vector, Y being the output scalar of the model, and $g()$ being a deterministic mapping function. Here, the input X is a random variable. Because of the uncertainty of $\boldsymbol{X}$ propagating through $g(.)$, Y is also a random variable. As the uncertainty of the output model is represented by its variance, V(Y), to find the effect of an input $\boldsymbol{X_i}$ on the output, it is assumed that the true value of $\boldsymbol{X_i}$ can be determined by the variance reduction in the output, i.e., $V(Y) - V\left(Y \middle| X_i = x_i^0\right)$, where x_i^0 is the true value of X_i and $V\left(Y \middle| X_i = x_i^0\right)$ is the conditional expected value of V(Y). Since the true value is unknown, one can employ $V(Y) - E_{X_i}\left(V\left(Y \middle| X_i\right)\right)$ to evaluate the expected variance reduction in the output (Zhai et al. [4]; Saltelli et al. [8]; Archer et al. [9]).

The variance of output model is calculated utilizing the following equation:

$$V(Y) = V_{X_i}\left(E\left(Y \middle| X_i\right)\right) + E_{X_i}\left(V\left(Y \middle| X_i\right)\right) \tag{14.1}$$

And the sensitivity indicator of X_i can be expressed as (Zhai et al. [4]):

$$SI_i = \frac{V(Y) - E_{X_i}\left(V\left(Y \middle| X_i\right)\right)}{V(Y)} = 1 - \frac{E_{X_i}\left(V\left(Y \middle| X_i\right)\right)}{V(Y)} = \frac{V_{X_i}\left(E\left(Y \middle| X_i\right)\right)}{V(Y)} \tag{14.2}$$

In sensitivity analysis, SI_i varies between 0 and 1. The lower value of SI_i corresponds to the less influential X_i, the higher value of SI_i corresponds to the much influential X_i, and for $SI_i = 0$, the X_i will have no influence on Y.

14.2.2 The Variance-Based Sensitivity Analysis Using Space-Partition Method

The most well-known methods for calculating the variance-based sensitivity indicators are the Monte Carlo simulations; however, they do not make full use of each output model evaluation. In order to calculate the variance-based sensitivity indicators from a given data, the scatterplot partitioning method can be utilized

(Zhai et al. [4]). For this method a single set of samples suffices to estimate all the sensitivity indicators. For estimating the variance-based sensitivity indices, a space-partition method is used in the following:

Suppose we have M points/samples $\{X^1, \ldots, X^M\}$ and M model output samples $\{y^1, \ldots, y^M\}$ obtained using the model $y = g(X)$. The variance of Y can be calculated by the sample variance $\hat{V}(y)$. For the sample bounds of X_i as [b1, b2], let it be decomposed into s successive, equal-probability, and nonoverlapping subintervals $A_k = [a_{k-1}, a_k)$, with k=1, ...s, $b_1 = a_0 < a_1 < \ldots < a_k < \ldots < a_s = b_2$ and $Pr(A_k)=1/s$. Decompose the output samples $\{y^1, \ldots, y^M\}$ into s subsets according to the decomposition of X_i, where $B_k = \left\{ y^j \middle| x_i^j \in A_k \right\}, \quad k = 1, \ldots, s$. The variance $V(Y|x_i \in A_k)$ can then be estimated by:

$$\hat{V}\left(Y\middle|x_i \in A_k\right) = V(B_k) \tag{14.3}$$

The expected conditional variance $E_{x_i}\left(V\left(Y\middle|x_i\right)\right)$ can now be approximately estimated using the following relationship:

$$\hat{E}_{x_i}\left(V\left(Y\middle|x_i\right)\right) \approx \frac{1}{s}\sum_{k=1}^{s} V(B_k) \tag{14.4}$$

And ultimately, SI_i is estimated by:

$$\widehat{SI}_i = 1 - \frac{\hat{E}_{x_i}\left(V\left(Y\middle|x_i\right)\right)}{\hat{V}(Y)} \tag{14.5}$$

14.3 A Global Sensitivity Analysis-Based Algorithm

This section introduces a global sensitivity analysis-based (GSAB) algorithm, which is a single-solution metaheuristic method. The proposed algorithm is named a global sensitivity analysis (GSA) because of determining the sensitivity indicator (*SI*) of decision variables.

Metaheuristic algorithms can be divided into two categories based on their search mechanism: population based and single solution (Kaveh and Mahdavi [10]). In the first group, a number of populations/agents are first generated, and then all agents updated iteratively until the termination condition is satisfied. On the other hand, single-solution metaheuristics are also known as trajectory methods, in which these algorithms produce single solution by exploring the search space efficiently while reducing the effective size of the search space. The GSAB algorithm consists of some samples for estimating the *SI* of decision variables. As these samples do not update iteratively and these are used only for calculating the

SIs, the proposed GSBA is studied within the single-solution metaheuristic category. The feasibility space of samples in the GSAB algorithm updates for searching the optimal solution over several iterations. In each iteration, the feasibility space is updated using two values: the sensitivity indicators and the global best sample. It is assumed that the problem is a minimization problem in R^D. The notations used are as follows:

S^t: The sample matrix in the tth iteration, $S^t = \left[X_i^t \middle| i = 1,\ 2,\ \ldots, N\right]$

X_i^t: The position of sample vector i in the th iteration, $X_i^t = \left\{x_{ij}^t \middle| j = 1, 2, .., D\right\}$

X_{min}: The minimum allowable value vector of variables, $X_{min} = \left\{x_{min_j} \middle| j = 1, 2, .., D\right\}$

X_{max}: The maximum allowable value vector of variables, $X_{max} = \left\{x_{max_j} \middle| j = 1, 2, .., D\right\}$

$f(X_i)$: The fitness of vector i

UB^t: The upper search boundary vector of variables in the tth iteration, $UB^t = \left\{ub_j^t \middle| j = 1, 2, .., D\right\}$

LB^t: The lower search boundary vector of variables in the tth iteration, $LB^t = \left\{lb_j^t \middle| j = 1, 2, .., D\right\}$

BW^t: The bandwidth of search space of variables in the tth iteration, $BW^t = \left\{bw_j^t \middle| j = 1, 2, .., D\right\}$

SF^t: The scale factor of bandwidth of search space in the tth iteration, $SF^t = \left\{sf_j^t \middle| j = 1, 2, .., D\right\}$

Sbest: The global best sample (i.e., with lower fitness), $Sbest = \left\{sbest_j \middle| j = 1, 2, .., D\right\}$

R: A random vector within [0,1].

14.3.1 Methodology

The following steps outline the main procedure in the implementation of the GSAB.

Step 1: Initialization The initial positions of samples are determined with random initialization in the search space:

$$X_i^0 = X_{\min} + R(X_{\max} - X_{\min}), \quad i = 1, 2, \ldots, N \tag{14.6}$$

where X_i^0 determines the initial value vector of the ith sample and N is the number of samples. In the first step, some parameter settings must also be predefined for the proposed algorithm. There are two parameters: the number of samples, N, and the number of subintervals for estimating the sensitivity indices, s. The number of samples is considered according to the problem's complexity. More complex problems require a higher number of samples. The last parameter is used for GSA

as mentioned in Sect. 14.2.2. These values affect the estimation of *SI* [Eq. (14.6)]. A more detailed discussion of these parameters is given in the subsequent subsections.

Step 2: Detection of the Most Sensitive Variable In this step the output model, i.e., the objective function of optimization problem, is first calculated. The sensitivity analysis is performed next for the generated samples, and the sensitivity indicators (*SI*s) of variables are calculated in Eqs. (14.3)–(14.6). Once the *SI*s are known, the most sensitive variable (which it has the high *SI*) and the amount of its *SI* are saved for the next step.

Step 3: Defining the Search Boundaries In the GSAB algorithm, the search boundaries are moved to the global best sample (which is updated and memorized in each iteration), *Sbest*, to push the samples into a feasible search space. The search boundaries are also decreased based on the values of the most sensitive variable, which is evaluated in the previous step. Hence, the upper boundary and lower boundary of the search space of variables in the $t+1$th iteration can be computed by:

$$\begin{aligned} UB^{t+1} &= Sbest + BW^t \times SF^t \leq X_{\max} \\ LB^{t+1} &= Sbest - BW^t \times SF^t \geq X_{\min} \end{aligned} \tag{14.7}$$

where BW^t and SF^t are the bandwidth and scale factor of boundaries in the tth iteration (Fig. 14.1), respectively. Equation (14.8) ensures that the current search space is moved around *Sbest* with the bandwidth BW^t in the D-dimensional space. The vector BW^t can be calculated as:

$$BW^t = \max(Sbest - LB^t, UB^t - Sbest) \tag{14.8}$$

For the algorithm to converge to a near-optimal solution, further exploitation (strong locality) is required to move the current solution toward to the optimal one. In the proposed GSAB algorithm, this is achieved by using a scale factor, *SF*. For this purpose, once *SI* values of variables are calculated, the most sensitive variable,

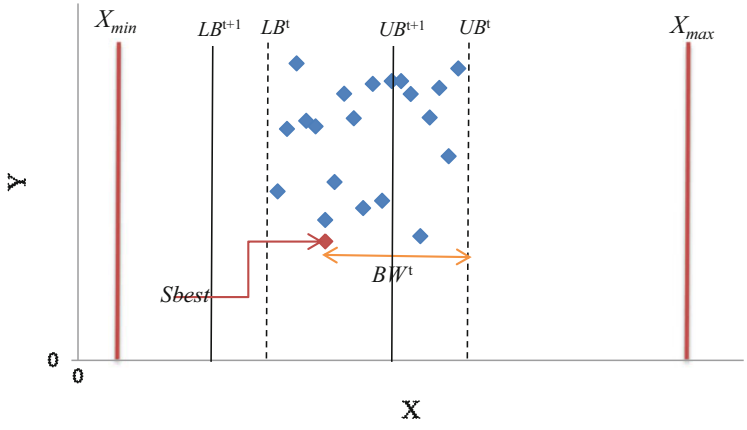

Fig. 14.1 An illustrative sketch of the search process

i.e., variable with high *SI* value, for reducing the bandwidth is identified, and then the *SF* is calculated as:

$$SF_j = \begin{cases} 1 - si_j & if \quad si_j = \max(SI) \\ 1 & Otherwisw \end{cases}, \quad \forall j = 1, \ldots, D \tag{14.9}$$

This equation shows that the bandwidth of the most sensitive variable is decreased, while other bandwidths are constant in the *t*h iteration.

Step 4: Replacement of the Current Samples In this step, the samples must be ensured to be inside the new search boundaries. For this purpose, the samples that exceed the boundaries are randomly regenerated into the new search boundaries, shown in Fig. 14.1, as:

$$X_i^{t+1} = \begin{cases} X_i^t, & LB^{t+1} \le X_i^t \le UB^{t+1} \\ LB^{t+1} + R\left(UB^{t+1} - LB^{t+1}\right), & Otherwise \end{cases} \tag{14.10}$$

where $i=1, 2, \ldots, n$ and t represents the iteration index.

Step 5: Termination The optimization process is repeated from Step 2 until a termination criterion, such as maximum iteration number or no improvement of the best sample, is satisfied. In the GSAB algorithm, if the maximum bandwidth of the search space, $max(W)$, becomes smaller than 0.000001, the optimization process will be stopped. This is because the GSAB cannot change the search space of the agents. For the sake of clarity, the flowchart of optimization procedure using the proposed GSAB is shown in Fig. 14.2.

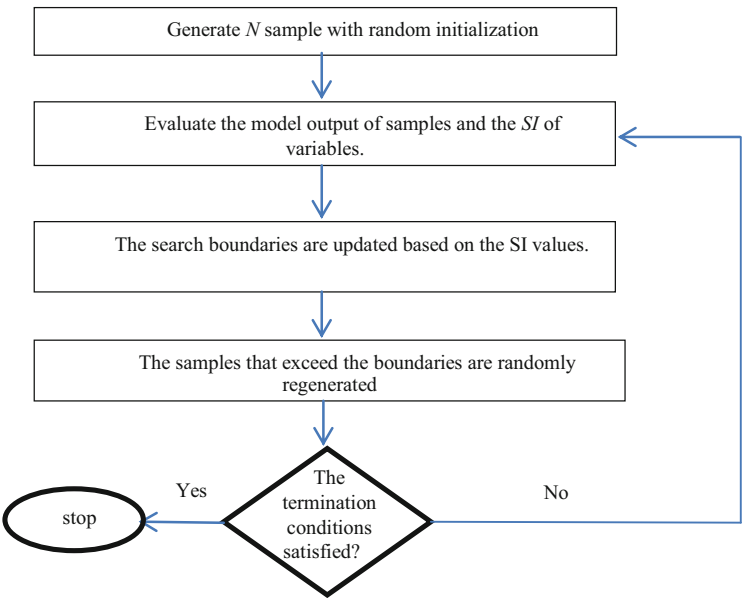

Fig. 14.2 Flowchart of the GSAB

14.4 Numerical Examples

In this section the efficiency of the proposed algorithm, GSAB, is shown through two mathematical constrained functions and three well-studied truss structures under static loads taken from the optimization literature. These examples have been previously solved using a variety of other techniques and are good examples to show the validity and effectiveness of the proposed algorithm. Examples 1 and 2 show the applicable of GSAB for optimization of the constrained problems. In Example 4, a planar truss structure is studied for finding the optimal cross sections. Examples 4 and 5 are selected to show the importance of selection of optimization algorithm in reduction of the number of function evaluations.

In structural optimization problems, the main objective is to minimize the weight of the structures under some constraints. The optimization problem for a truss structure can be stated as follows:

$$\begin{aligned} &\text{Find}\, X = [x_1, x_2, x_3, \ldots, x_n] \\ &\text{to minimizes}\, W(X) = \sum_{i=1}^{ne} \rho_i A_i l_i \\ &\text{subjected to}\, g_j(X) \leq 0, j = 1, 2, \ldots, m \\ &x_{lmin} \leq x_l \leq x_{lmax} \end{aligned} \tag{14.11}$$

where X is the vector of all design variables with n unknowns; W is the weight of truss structure; ρ_i, A_i, and l_i are the mass density, cross-sectional area, and length of the ith member, respectively; ne is the number of the structural elements; g_j is the jth constraint from m inequality constraints; and, also, x_{lmin} and x_{lmax} are the lower and upper bounds of design variable vector, respectively.

The employed constraint handling is the penalty function approach proposed by Deb [11]. It should be noted that the output model of SA method is the penalized objective function. For truss design and engineering design examples, the numbers of $N = 40$ and $N = 20$ samples are utilized, respectively. Also, all examples are independently optimized 20 times. The algorithm is coded in MATLAB. Structural analysis is performed with the direct stiffness method.

14.4.1 Design of a Tension/Compression Spring

This problem was first described by Belegundu [12] and Arora [13]. It consists of minimizing the weight of a tension/compression spring subject to constraints on shear stress, surge frequency, and minimum deflection as shown in Fig. 14.3.

The design variables are the mean coil diameter $D(= x_1)$, the wire diameter $d(= x_2)$, and the number of active coils $N(= x_3)$. The problem can be stated as follows:

Fig. 14.3 Schematic of the tension/compression spring with indication of design variables

$$\text{Find}\,\{x_1, x_2, x_3\} \tag{14.12}$$

To minimize:

$$\cos t(x) = (x_3 + 2)x_2x_1^2 \tag{14.13}$$

Subject to

$$\begin{aligned} g_1(x) &= 1 - \frac{x_2^3 x_3}{71785x_1^4} \le 0 \\ g_2(x) &= \frac{4x_2^2 - x_1x_2}{12566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5108x_1^2} - 1 \le 0 \\ g_3(x) &= 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\ g_4(x) &= \frac{x_1 + x_2}{1.5} - 1 \le 0 \end{aligned} \tag{14.14}$$

The bounds on the design variables are:

$$0.05 \le x_1 \le 2, \quad 0.25 \le x_2 \le 1.3, \quad 2 \le x_3 \le 15, \tag{14.15}$$

This problem has been solved by Belegundu [12] using eight different mathematical optimization techniques. Arora [13] solved this problem using a numerical optimization technique called a constraint correction at the constant cost. Coello [14] as well as Coello and Montes [15] solved this problem using GA-based method. Additionally, He and Wang [16] utilized a coevolutionary particle swarm optimization (CPSO). Recently, Montes and Coello [17], Kaveh and Talatahari [18], and Kaveh and Mahdavi [19] used the ES, CSS, and CBO to solve this problem, respectively.

Tables 14.1 and 14.2 compare the best results obtained in this chapter and those of the other researches. The GSAB found the best cost as 0.0126652 after 3729 fitness function evaluations. Although the best cost found is more than the standard CSS, it is the lowest fitness function evaluations among the existing literature results. It should be noted that the lighter design found by Kaveh and Talatahari [18] slightly violates the first two optimization constraints.

In order to show the performance of the GSA method in the GSAB algorithm, a study is focused on the influence of the *SI*s on the proposed algorithm result. As

Table 14.1 Comparison of GSAB optimized designs with literature for the tension/compression spring problem

	Optimal design variables			
Methods	x_1 (d)	x_2 (D)	x_3 (N)	f(x)
Belegundu [12]	0.050000	0.315900	14.250000	0.0128334
Arora [13]	0.053396	0.399180	9.185400	0.0127303
Coello [14]	0.051480	0.351661	11.632201	0.0127048
Coello and Montes [15]	0.051989	0.363965	10.890522	0.0126810
He and Wang [16]	0.051728	0.357644	11.244543	0.0126747
Montes and Coello [17]	0.051643	0.355360	11.397926	0.012698
Kaveh and Talatahari [18]	0.051744	0.358532	11.165704	0.0126384
Kaveh and Mahdavi [19]	0.051894	0.3616740	11.007846	0.0126697
Present work [1]	0.05171604	0.3573671	11.2509979	0.0126652

Table 14.2 Statistical results from different optimization methods for tension/compression string problem

Methods	Best result	Average optimized cost	Worst result	Std dev	Fitness function evaluations
Belegundu [12]	0.0128334	N/A	N/A	N/A	N/A
Arora [13]	0.0127303	N/A	N/A	N/A	N/A
Coello [14]	0.0127048	0.012769	0.012822	3.9390e-5	900,000
Coello and Montes [15]	0.0126810	0.0127420	0.012973	5.9000e-5	N/A
He and Wang [16]	0.0126747	0.012730	0.012924	5.1985e-5	200,000
Montes and Coello [17]	0.012698	0.013461	0.16485	9.6600e-4	25,000
Kaveh and Talatahari [18]	0.0126384	0.012852	0.013626	8.3564e-5	4000
Kaveh and Mahdavi [19]	0.0126697	0.01272964	0.0128808	5.00376e-5	4000
Present work [1]	0.0126652	0.012875334	0.01334400	2.31935e-4	3729

described in Sect. 14.3.1, the GSA method requires two predefined parameters: the number of samples, N, and the number of subintervals, s. A larger number of samples lead to an increase of the accuracy of the sensitivity indicators. On the other hand, because of generating the output model of the GSA method, the fitness function (or output) evaluations increase with the number of samples. The number of subintervals can also be affected to the SI values. As Zhai et al. [4] underline, the appropriate number of subintervals can be considered as $s = \frac{N}{5}$. The scatter plots of $X_{i=1,2,3}$ and $cost$ for $N = 100$ samples are shown in Fig. 14.4. It can be noticed that (i) x_1 seems to be the most influential input and (ii) x_2 and x_3 seem to be the low

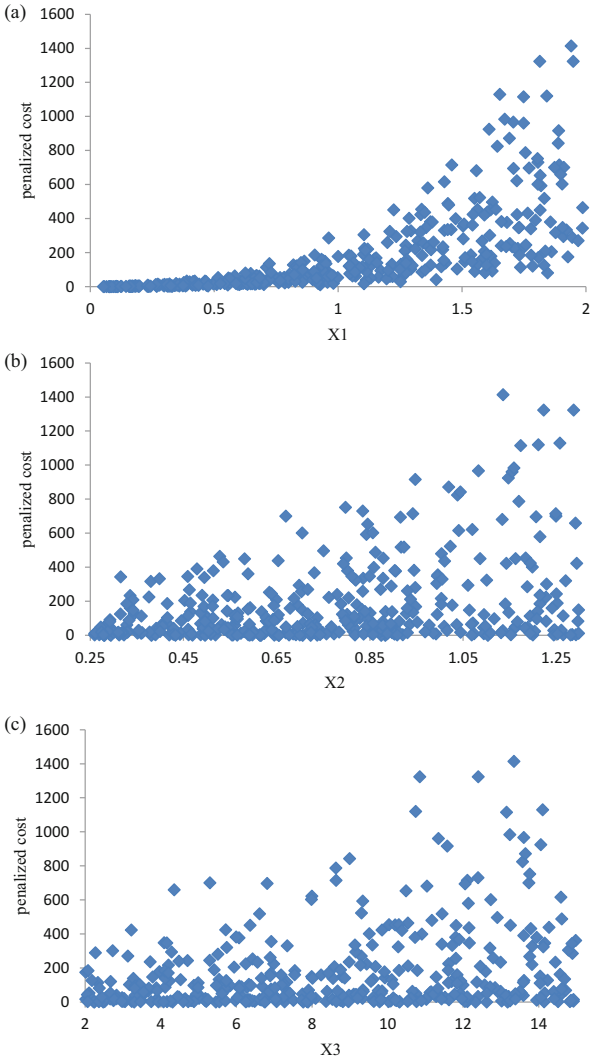

Fig. 14.4 Scatter plots for variables: (**a**) X_1, (**b**) X_2, (**c**) X_3 of the first example

influential inputs, because the distribution of samples against the first variable, x_1, is more dense compared to other variables. This is confirmed by the GSA method. If we apply the space-partition variance-based sensitivity analysis approach, we obtained the sensitivity indicators, *SI*s, in Fig. 14.5. As it can be seen from this figure, the *SI* of the first variable is much than other variables; then, the most influential/sensitive variable is the first variable.

Figure 14.6 shows the convergence rates of the upper and lower boundary of the search space and the best ones in the optimization process. It should be noted that, as mentioned before, the number of samples is considered as $N = 40$ in the optimization process. It can be seen, with respect to the second and third variables, that the

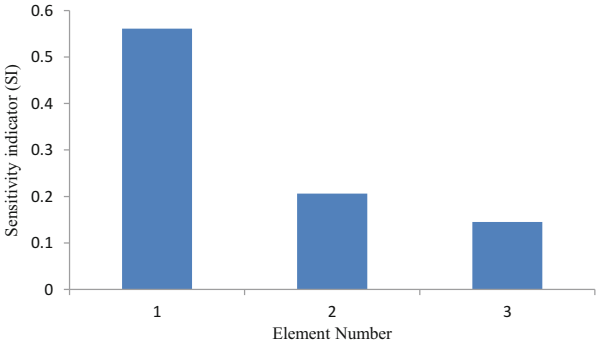

Fig. 14.5 The obtained sensitivity indicator of variables for penalized cost function of the first example

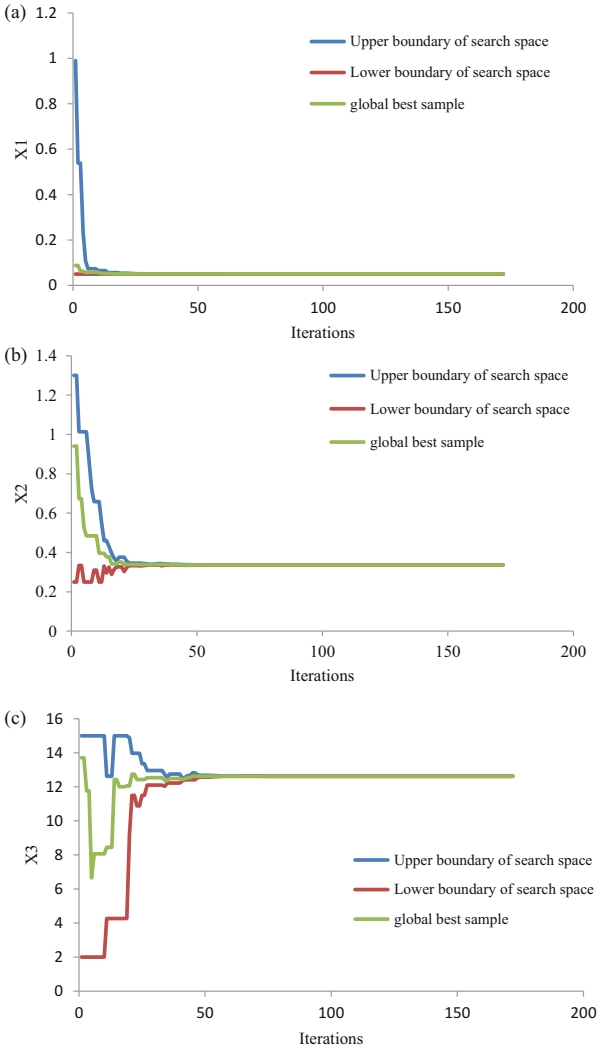

Fig. 14.6 The convergence history graphs of search space for variables: (**a**) X_1, (**b**) X_2, (**c**) X_3

search space of the first variable is rapidly decreased in the early iterations because it has more sensitivity to output (i.e., objective function). Hence, despite the fewer samples, the proposed GSA approach could appropriately rank the variables based on these sensitivities.

The optimum variables found with different algorithms can also be used for comparing the *SI* of variables. As shown in Table 14.1, although the optimal objective functions found by different optimization algorithms have no significant difference, the values of the optimum second and third variables have significant difference compared to the first variable.

14.4.2 A Constrained Function

This is a 10-variable problem which challenges the algorithm ability to deal with the problem of optimization. This problem also has eight nonlinear inequality constraints and it is defined as

Find

$$\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\} \tag{14.16}$$

To minimize:

$$f(x) = x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 \\ +2(x_6 - 1)^2 + 5x_7^2 + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45 \tag{14.17}$$

Subjected to:

$$\begin{aligned} g_1(x) &= 105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \geq 0, \\ g_2(x) &= -10x_1 + 8x_2 + 17x_7 - 2x_8 \geq 0, \\ g_3(x) &= 8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 \geq 0, \\ g_4(x) &= -3(x_1 - 2)^2 - 4(x_2 - 3)^2 - 2x_3^2 + 7x_4 + 120 \geq 0, \\ g_5(x) &= -5x_1^2 - 8x_2 - (x_3 - 6)^2 + 2x_4 + 40 \geq 0, \\ g_6(x) &= -x_1^2 - 2(x_2 - 2)^2 + 2x_1x_2 - 14x_5 + 6x_6 \geq 0, \\ g_7(x) &= -0.5(x_1 - 8)^2 - 2(x_2 - 4)^2 - 3x_5^2 + x_6 + 30 \geq 0, \\ g_8(x) &= 3x_1 - 6x_2 - 12(x_9 - 8)^2 + 7x_{10} \geq 0. \end{aligned} \tag{14.18}$$

The bounds on the design variables are:

$$-10 \leq x_i \leq 10 \quad (i = 1 - 10) \tag{14.19}$$

This problem has been solved by Deb [11] utilizing an efficient constraint handling method for the GA. Lee and Geem [20] and Kaveh and Mahdavi [21]

employed the harmony search and colliding bodies optimization algorithms, respectively.

Tables 14.3 and 14.4 compare the optimal variables, best cost, mean cost, and standard deviation of the results obtained using GSAB with the outcomes of other algorithms. As anticipated, GSAB led to a much better results in terms of best cost and also fitness function evaluations. Figure 14.7 provides the amount of *SI*s founded by the GSA method, and high sensitivity value of the ninth variable is shown. As it can be seen also in Table 14.3, the best objective function and the optimum value of the ninth variable, with respect to other variables, obtained by the HS algorithm, are similar to these outcomes of the CBO algorithm. However, the best objective function and the optimum value of the ninth variable obtained using the GSAB algorithm are much better than the other two algorithms.

14.4.3 A Planar 17-Bar Truss Problem

A 17-bar planar truss is schematized in Fig. 14.8. The single vertical downward load of 100 kips at node 9 is considered, and there are 17 independent design

Table 14.3 Optimal design variables obtained by different researchers for the constrained function

Optimal design variables *(x)*	Deb [11]	Lee and Geem [20]	Kaveh and Mahdavi [21]	Present study
x_1	Unavailable	2.155225	2.142755	2.193229
x_2		2.407687	2.441786	2.229117
x_3		8.778069	8.772559	8.747274
x_4		5.102078	5.089189	5.074095
x_5		0.967625	0.976804	1.011086
x_6		1.357685	1.36545	1.38219
x_7		1.287760	1.261765	1.347327
x_8		9.800438	9.778372	9.902594
x_9		8.187803	8.196755	8.308814
x_{10}		8.256297	8.362651	8.22824

Table 14.4 Statistical results from different optimization methods for the constrained function

Methods	Best objective function	Average objective function	Std dev	Fitness function evaluations
Deb [11]	24.37248	24.40940	N/A	350,070
Lee and Geem [20]	24.36679	N/A	N/A	230,000
Kaveh and Mahdavi [21]	24.38470	24.86188	0.580431	100,000
Present study	23.91122644	24.87359216	0.768176491	24,693

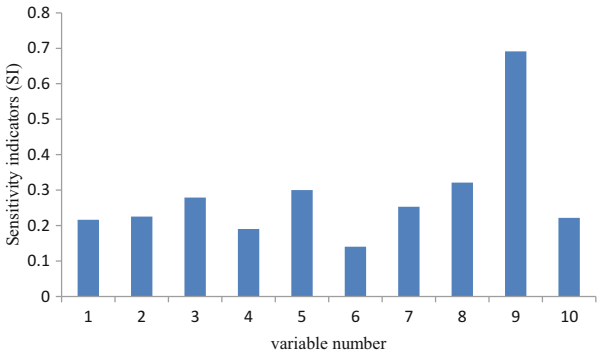

Fig. 14.7 The obtained sensitivity indicator of variables for penalized cost function of the constrained function example

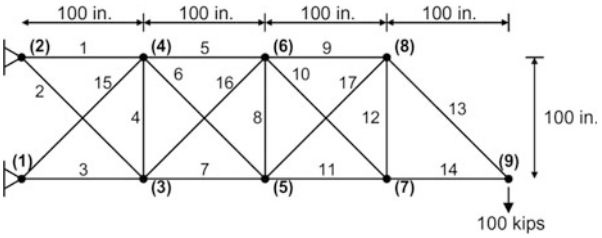

Fig. 14.8 Schematic of the planar 17-bar truss problem

variables. The elastic modulus is 30,000 ksi and the material density is 0.268 lb/in^3 for all elements. The members are subjected to the stress limits of 50 ksi both in tension and compression. Displacement limitations of ± 2.0 in are imposed on all nodes in both directions (x and y). The allowable minimum cross-sectional area of all the elements is set to 0.1 in^2.

Table 14.5 presents the optimum designs obtained by Khot and Berke [22], Adeli and Kumar [23], standard CBO, ECBO, Kaveh and Ilchi Ghazaan [24], and the proposed GSA algorithms. Although, the best design is obtained by the ECBO and the work of Khot and Berke [22], the average weight and standard deviation of independent runs obtained by the GSAB are the lowest. The optimization process of the best run of the GSAB is completed in 12,255 analyses. Standard CBO and ECBO required 15,560 and 14,180 analyses to converge to the optimum. The SI values of variables are shown in Fig. 14.9. It can be seen in Fig. 14.9 and Table 14.5 that the sensitivities of members 1, 3, 5, 7, 9, 11, and 13 are more than the remaining members, and the larger optimum designs obtained using optimization algorithms have the high value of *SI*s.

14.4.4 A 72-Bar Spatial Truss Structure

Schematic topology and element numbering of a 72-bar space truss are shown in Fig. 14.10. The elements are classified into 16 design groups according to

Table 14.5 Comparison of the optimized designs for the 17-bar planar truss

Element group	Khot and Berke [22]	Adeli and Kumar [23]	Optimal cross-sectional areas		Present work [1]
			Kaveh and Ilchi [24]		
			CBO	ECBO	
A_1	15.930	16.029	15.9674	15.9158	15.8916
A_2	0.100	0.107	0.1386	0.1001	0.10088
A_3	12.070	12.183	12.1735	12.0762	12.00129
A_4	0.100	0.110	0.1000	0.1000	0.100015
A_5	8.067	8.417	7.8524	8.0527	8.078015
A_6	5.562	5.715	5.5447	5.5611	5.571161
A_7	11.933	11.331	11.9648	11.9470	11.98603
A_8	0.100	0.105	0.1002	0.1000	0.100602
A_9	7.945	7.301	7.9385	7.9425	8.009118
A_{10}	0.100	0.115	0.1003	0.1000	0.100585
A_{11}	4.055	4.046	4.1146	4.0589	4.06476
A_{12}	0.100	0.101	0.1000	0.1000	0.100046
A_{13}	5.657	5.611	5.8134	5.6644	5.577003
A_{14}	4.000	4.046	4.0556	4.0057	4.004148
A_{15}	5.558	5.152	5.4973	5.5565	5.611166
A_{16}	0.100	0.107	0.1329	0.1000	0.104159
A_{17}	5.579	5.286	5.4043	5.5740	5.568715
Best weight (lb)	2581.89	2594.42	2582.79	2581.89	2582.032
Average weight (lb)	N/A	N/A	2631.07	2597.11	2585.62
Std dev (lb)	N/A	N/A	49.45	22.43	9.248879

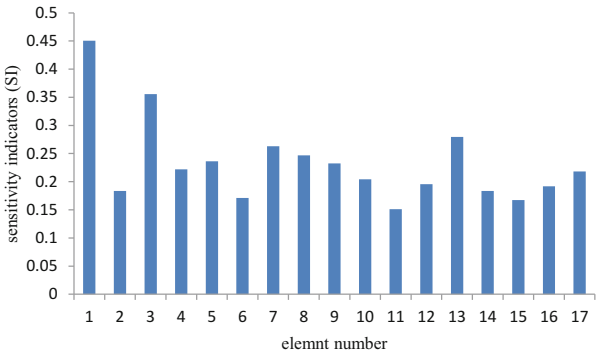

Fig. 14.9 The obtained sensitivity indicator of variables for the penalized weight of the planar 17-bar truss problem

Table 14.6. The material density is 0.1 lb/in^3 (2767.990 kg/m^3), and the modulus of elasticity is taken as 10,000 ksi (68,950 MPa). The members are subjected to the stress limits of ±25 ksi (±172.375 MPa). The uppermost nodes are subjected to the displacement limits of ±0.25 in (±0.635 cm) in both x and y directions. The

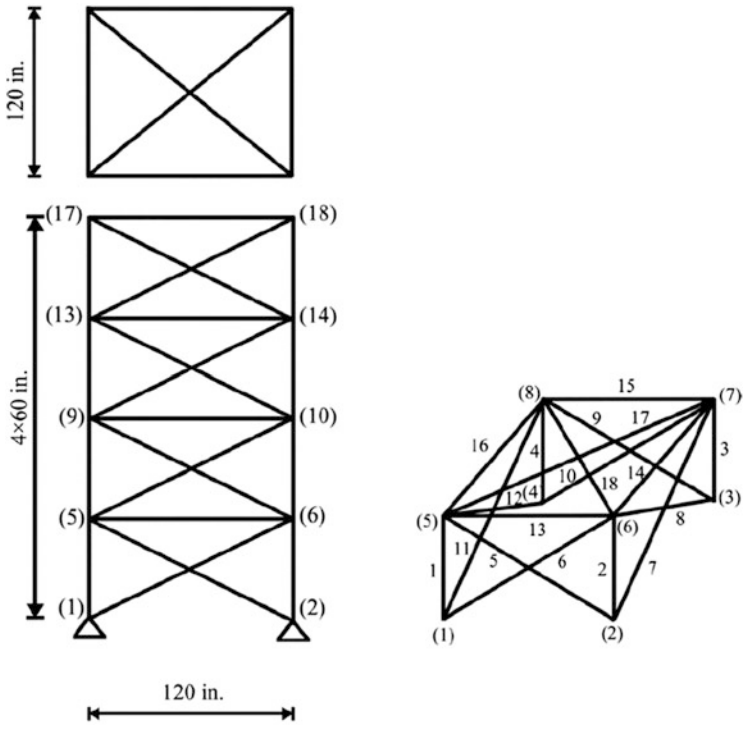

Fig. 14.10 Schematic of the 72-bar spatial truss

minimum permitted cross-sectional area of each member is taken as 0.10 in^2 (0.6452 cm^2), and the maximum cross-sectional area of each member is 4.00 in^2 (25.81 cm^2). The loading conditions are considered as:

1. Loads 5, 5, and −5 kips in the *x*, *y*, *and z* directions at node 17, respectively
2. A load equal to −5 kips in the *z* direction at nodes 17, 18, 19, and 20

Table 14.6 shows the optimum design variables using the GSAB algorithm, which is compared to the results of the other algorithms. The best result of the GSAB approach is 379.7689, while it is 385.76, 380.24, 381.91, 379.85, 380.458, 379.75, and 379.77 *lb* for the GA Erbatur et al. [25], ACO Camp and Bichon [26], PSO Perez and Behdinan [27], BB–BC Camp [28], RO Kaveh and Khayatazad [29], CBO and ECBO, and Kaveh and Ilchi Ghazaan [24] algorithms, respectively. Also, the number of analyses of the GSAB is 13,795, while it is 18,500, 19,621, 19,084, 16,000, and 18,000 for the ACO, BB–BC, RO, CBO, and ECBO algorithms, respectively. It is evident in Table 14.6 that although the statistical results of 20 independent runs for the CBO are less than the GSAB algorithm, the number of function evaluations for the GSAB algorithm is less than that of the CBO. Figure 14.11 shows the *SI* values of the variables for this example. Figure 14.12 shows the maximum stress ratios in truss group members obtained using the GSAB. As it

Table 14.6 Comparison of GSAB optimized designs with those of literature for the 72-bar spatial truss (in^2)

	Optimal cross-sectional areas (in^2)							
Element group	Erbatur et al. [25] GA	Camp et al. [26] ACO	Perez et al. [27] PSO	Camp [28] BB–BC	Kaveh and Khayat [29] RO	Kaveh and Mahdavi [24] CBO	Kaveh and Ilchi [24] ECBO	Present work [1]
1–4	1.755	1.948	1.7427	1.8577	1.8365	1.9170	1.8519	1.909083519
5–12	0.505	0.508	0.5185	0.5059	0.5021	0.5031	0.5141	0.515793736
13–16	0.105	0.101	0.1000	0.1000	0.1000	0.1000	0.1000	0.100097411
17–18	0.155	0.102	0.1000	0.1000	0.1004	0.1001	0.1000	0.100154463
19–22	1.155	1.303	1.3079	1.2476	1.2522	1.2721	1.2819	1.292073465
23–30	0.585	0.511	0.5193	0.5269	0.5033	0.5050	0.5091	0.524173699
31–34	0.100	0.101	0.1000	0.1000	0.1002	0.1000	0.1000	0.100059742
35–36	0.100	0.100	0.1000	0.1012	0.1001	0.1000	0.1000	0.100103578
37–40	0.460	0.561	0.5142	0.5209	0.5730	0.5184	0.5312	0.515818559
41–48	0.530	0.492	0.5464	0.5172	0.5499	0.5362	0.5173	0.513756471
49–52	0.120	0.1	0.1000	0.1004	0.1004	0.1000	0.1000	0.100010199
53–54	0.165	0.107	0.1095	0.1005	0.1001	0.1000	0.1000	0.100509039
55–58	0.155	0.156	0.1615	0.1565	0.1576	0.1569	0.1560	0.157384016
59–66	0.535	0.550	0.5092	0.5507	0.5222	0.5374	0.5572	0.526496976
67–70	0.480	0.390	0.4967	0.3922	0.4356	0.4062	0.4259	0.407510273
71–72	0.520	0.592	0.5619	0.5922	0.5971	0.5741	0.5271	0.56965198
Best weight (Ib)	385.76	380.24	381.91	379.85	380.458	379.75	379.77	379.7689
Average weight (Ib)	N/A	383.16	N/A	382.08	382.553	380.03	380.39	380.3613
Std dev	N/A	3.66	N/A	1.912	1.221	0.278	0.8099	0.519822
No. of analyses	N/A	18,500	N/A	19,621	19,084	16,000	18,000	13,795

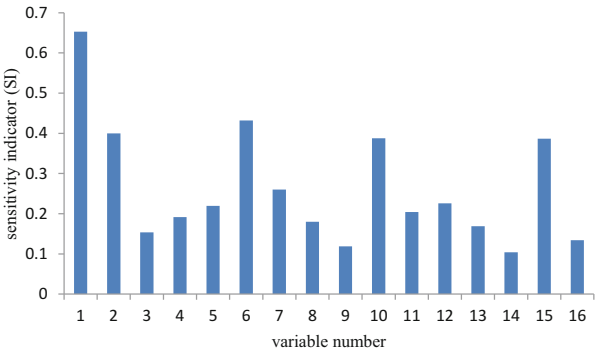

Fig. 14.11 The obtained sensitivity indicator of variables for the penalized weight of 72-bar spatial truss

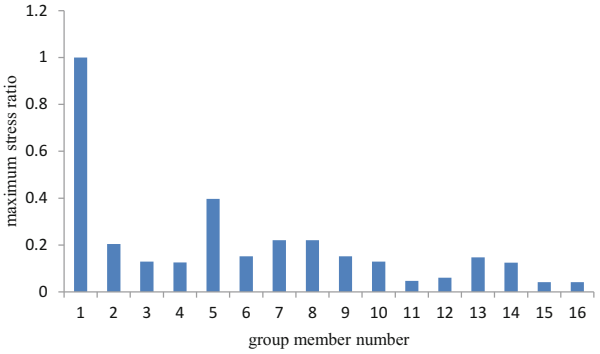

Fig. 14.12 The maximum stress ratio in the group elements of the 72-bar truss structure

can be seen in Figs. 14.11 and 14.12 and Table 14.6, the first design variable, i.e., the first-story column area, is the most sensitive variable because of the high amount of axial force in the first-story columns. The design variables corresponding to the vertical braces area are also the sensitive variables, with respect to other truss group members, because these can affect the displacement constraints and can have high length in the shape of truss.

14.4.5 A 120-Bar Truss Dome

The last test case solved in this study is the weight minimization problem of the 120-bar truss dome shown in Fig. 14.13. This test case was investigated by Soh and Yang [30] as a configuration optimization problem. It has been solved later as a sizing optimization problem by Kaveh and Talatahari [18], Kaveh and Khayatazad [29], and Kaveh and Mahdavi [19].

The allowable tensile and compressive stresses are set according to the ASD-AISC [31] code, as follows:

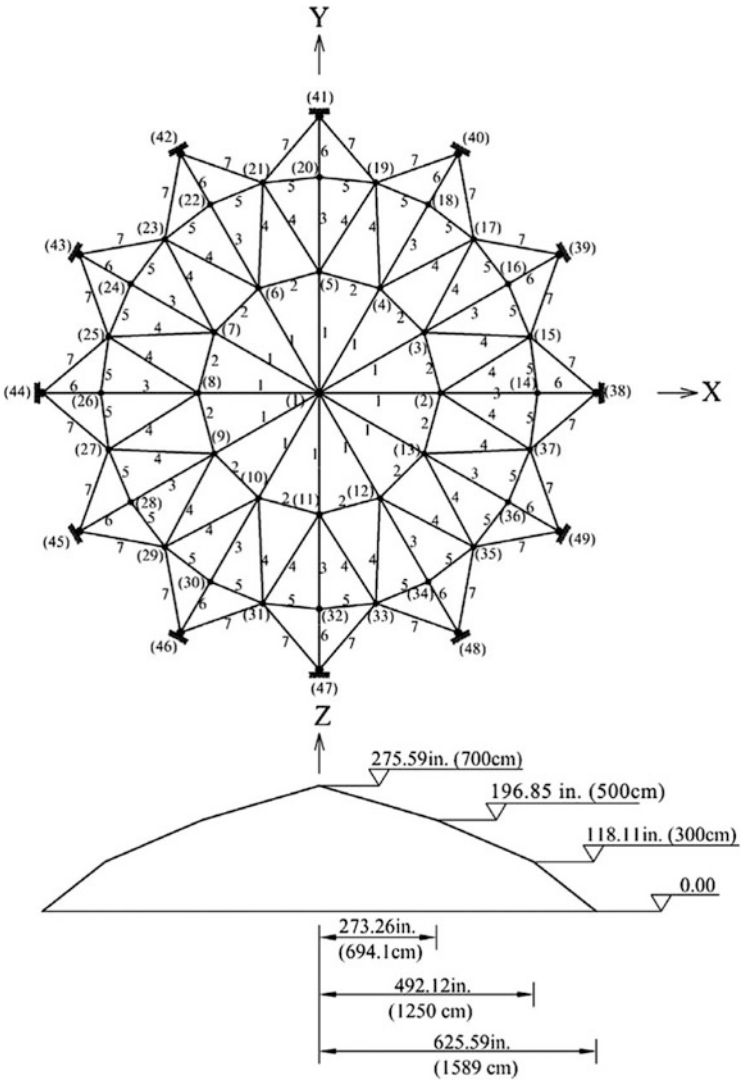

Fig. 14.13 Schematic of the spatial 120-bar dome truss with indication of design variables and main geometric dimensions

$$\begin{cases} \sigma_i^+ = 0.6F_y & for \sigma_i \geq 0 \\ \sigma_i^- & for \sigma_i \leq 0 \end{cases} \tag{14.20}$$

where σ_i^- is calculated according to the slenderness ratio:

$$\sigma_i^- = \begin{cases} \left[\left(1 - \frac{\lambda_i^2}{2C_c^2}\right)F_y\right] \Big/ \left(\frac{5}{3} + \frac{3\lambda_i}{8C_c} - \frac{\lambda_i^3}{8C_c^3}\right) & for \ \lambda_i < C_c \\ \frac{12\pi^2 E}{23\lambda_i^2} & for \ \lambda_i \geq C_c \end{cases} \tag{14.21}$$

where E is the modulus of elasticity, F_y is the yield stress of steel, C_c is the slenderness ratio (λ_i) dividing the elastic and inelastic buckling regions $\left(C_c = \sqrt{2\pi^2 E/F_y}\right)$, λ_i is the slenderness ratio $\left(\lambda_i = \frac{KL_i}{r_i}\right)$, K is the effective length factor, L_i is the member length, and r_i is the radius of gyration.

The modulus of elasticity is 30,450 ksi and the material density is 0.288 lb/in^3. The yield stress of steel is taken as 58.0 ksi. On the other hand, the radius of gyration (r_i) is expressed in terms of cross-sectional areas as $r_i = aA_i^{\,bi}$ (Saka [32]). Here, a and b are constants depending on the types of sections adopted for the members such as pipes, angles, and tees. In this example, pipe sections ($a = 0.4993$ and $b = 0.6777$) are adopted for bars. All members of the dome are divided into seven groups, as shown in Fig. 14.7. The dome is considered to be subjected to vertical loads at all unsupported joints. These are taken as −13.49 kips (60 kN) at node 1, −6.744 kips (30 kN) at nodes 2 through 14, and −2.248 kips (10 kN) at the remaining of the nodes. The minimum cross-sectional area of elements is 0.775 in^2 (cm^2). In this example, the constraints are considered: Stress constraints and displacement limitations of ±0.1969 in are imposed on all nodes in all directions. The maximum cross-sectional area is also considered as 20.0 in^2.

Table 14.7 summarizes the results obtained by the present work and those of the previously reported researches. As it can be seen, the best results obtained using the GSAB is better than those of the other methods (except for the HPSACO). The standard deviations of results are also better than the RO and CBO algorithms. In this example, the GSAB needs 5823 analyses to find the optimum result, while this number is 10,000, 125,000, 19,800, and 16,000 for the HPSACO, PSOPC, RO, and CBO algorithms as reported, respectively.

14.5 Concluding Remarks

In this chapter, a new single-solution global sensitivity analysis-based optimizer called GSAB is developed. Compared to other metaheuristic algorithms, the GSAB has several distinct features. Firstly, the population/agents in GSAB are directly represented by the samples, which are used to find the sensitivity values of the decision variables as well as the optimization search in sequence at each iteration. Hence, one can consider the proposed algorithm as a single-solution metaheuristic category. Secondly, the search boundaries are considered, and these are decreased

Table 14.7 Comparison of the GSAB optimized designs with those of literature for the 120-bar dome problem

Element group	Optimal cross-sectional areas (in^2)					
	Kaveh and Talat [18] PSO	Kaveh and Talat [18] PSOPC	Kaveh and Talat [18] HPSACO	Kaveh and Khayat [29] RO	Kaveh and Mahdavi [19] CBO	Present work [1]
1	12.802	3.040	3.095	3.030	3.0284	3.024214
2	11.765	13.149	14.405	14.806	14.9543	14.8525
3	5.654	5.646	5.020	5.440	5.4607	5.064194
4	6.333	3.143	3.352	3.124	3.1214	3.134918
5	6.963	8.759	8.631	8.021	8.0552	8.457656
6	6.492	3.758	3.432	3.614	3.3735	3.283562
7	4.988	2.502	2.499	2.487	2.4899	2.49657
Best weight (Ib)	51986.2	33481.2	33248.9	33317.8	33286.3	33249.68
Average weight (Ib)	–	–	–	–	33398.5	33253.32
Std dev (Ib)	–	–	–	354.333	67.09	4.112399

based on the sensitivity values of the variables at each iteration. The sample, which is the best one, is also selected to push the search boundaries around this sample, and it is selected as solution of the GSAB algorithm. Then, the samples that exceed the search boundaries are randomly regenerated into the boundaries. Unlike the common metaheuristic algorithms where the agents of a population move to the new positions without considering any information about the sensitivity of variables, in this algorithm the search boundaries are decreased based on the sensitivity indices of the variables, and this accelerates the converge of the solution.

The GSAB algorithm is tested over five benchmark optimization problems consisting of mathematical and truss structure optimization problems with different dimensions. The results are compared to those of some population-based metaheuristics. This comparison reveals that besides its simplicity, the proposed GSAB algorithm is also competitive, especially from the number of function evaluations' point of view, when compared to the performance of the some other algorithms.

References

1. Kaveh A, Mahdavi VR (2016) Optimal design of truss structures using a new metaheuristic algorithm based on global sensitivity analysis. Struct Eng Mech Int J 60(26)
2. Rahman S (2011) Global sensitivity analysis by polynomial dimensional decomposition. Reliab Eng Syst Saf 96:825–837

3. Pianosi F, Wagener T (2015) A simple and efficient method for global sensitivity analysis based on cumulative distribution functions. Environ Modell Softw 67:1–11
4. Zhai Q, Yang J, Zhao Y (2014) Space-partition method for the variance-based sensitivity analysis: optimal partition scheme and comparative study. Reliab Eng Syst Saf 131:66–82
5. Rastrigin LA (1963) The convergence of the random search method in the extremal control of a many parameter system. Autom Remote Control 24(10):1337–1342
6. Hooke R, Jeeves TA (1961) Direct search solution of numerical and statistical problems. J Assoc Comput Mach 8(2):212–229
7. Dog B, Ölmez T (2015) A new meta-heuristic for numerical function optimization: Vortex Search algorithm. Inform Sci 293:125–145
8. Saltelli A, Annoni P, Azzini I, Campolongo F, Ratto M, Tarantola S (2010) Variance based sensitivity analysis of model output. Design and estimator for the total sensitivity index. Comput Phys Commun 181:259–270
9. Archer G, Saltelli A, Sobol I (1997) Sensitivity measures, ANOVA-like techniques and the use of bootstrap. J Statist Comput Simul 58:99–120
10. Kaveh A, Mahdavi VR (2015) Colliding bodies optimization; extensions and applications. Springer, Switzerland
11. Deb K (2000) An efficient constraint handling method for genetic algorithms. Comput Meth Appl Mech Eng 186(2–4):311–338
12. Belegundu AD (1982) A study of mathematical programming methods for structural optimization. Ph.D. thesis, Department of Civil and Environmental Engineering, University of Iowa, Iowa, USA
13. Arora JS (1989) Introduction to optimum design. McGraw-Hill, New York, NY
14. Coello CAC (2000) Use of a self-adaptive penalty approach for engineering optimization problems. Comput Indust Eng 41:113–127
15. Coello CAC, Montes EM (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament. IEEE Trans Reliab 41:576–582
16. He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problem. Eng Appl Artific Intell 20:89–99
17. Montes EM, Coello CAC (2008) An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. Int J General Syst 37:443–473
18. Kaveh A, Talatahari S (2010) A novel heuristic optimization method: charged system search. Acta Mech 213:267–289
19. Kaveh A, Mahdavi VR (2014) Colliding bodies optimization: a novel meta-heuristic method. Comput Struct 139:18–27
20. Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput Meth Appl Mech Eng 194 (36–38):3902–3933
21. Kaveh A, Mahdavi VR (2015) A hybrid CBO–PSO algorithm for optimal design of truss structures with dynamic constraints. Appl Soft Comput 34:260–273
22. Khot NS, Berke L (1984) Structural optimization using optimality criteria methods. In Atrek E, Gallagher H, Ragsdell KM, Zienkiewicz OC (eds). Wiley, New York
23. Adeli H, Kumar S (1995) Distributed genetic algorithm for structural optimization. J Aerosp Eng 8(3):156–163
24. Ilchi KA, Ghazaan M (2014) Enhanced colliding bodies optimization for design problems with continuous and discrete variables. Adv Eng Softw 77:66–75
25. Erbatur F, Hasançebi O, Tütüncü I, Kiliç H (2014) Optimal design of planar and space structures with genetic algorithms. Comput Struct 75:209–224
26. Camp CV, Bichon BJ (2004) Design of space trusses using ant colony optimization. J Struct Eng 130:741–751
27. Perez RE, Behdinan K (2007) Particle swarm approach for structural design optimization. Comput Struct 85:1579–1588

28. Camp CV (2007) Design of space trusses using Big Bang–Big Crunch optimization. J Struct Eng 133:999–1008
29. Kaveh A, Khayatazad M (2012) A novel meta-heuristic method: ray optimization. Comput Struct 112–113:283–294
30. Soh CK, Yang J (1996) Fuzzy controlled genetic algorithm search for shape optimization. J Comput Civil Eng 10:143–150
31. American Institute of Steel Construction (AISC) (1989) Manual of steel construction allowable stress design, 9th edn. Chicago, IL, USA
32. Saka MP (1990) Optimum design of pin-jointed steel structures with practical applications. J Struct Eng 116:2599–2620