Prem Singh* and Himanshu Chaudhary

# A Modified Jaya Algorithm for Mixed-Variable Optimization Problems

https://doi.org/10.1515/jisys-2018-0273
Received June 25, 2018; previously published online October 23, 2018.

**Abstract:** Mixed-variable optimization problems consist of the continuous, integer, and discrete variables generally used in various engineering optimization problems. These variables increase the computational cost and complexity of optimization problems due to the handling of variables. Moreover, there are few optimization algorithms that give a globally optimal solution for non-differential and non-convex objective functions. Initially, the Jaya algorithm has been developed for continuous variable optimization problems. In this paper, the Jaya algorithm is further extended for solving mixed-variable optimization problems. In the proposed algorithm, continuous variables remain in the continuous domain while continuous domains of discrete and integer variables are converted into discrete and integer domains applying bound constraint of the middle point of corresponding two consecutive values of discrete and integer variables. The effectiveness of the proposed algorithm is evaluated through examples of mixed-variable optimization problems taken from previous research works, and optimum solutions are validated with other mixed-variable optimization algorithms. The proposed algorithm is also applied to two-plane balancing of the unbalanced rigid threshing rotor, using the number of balance masses on plane 1 and plane 2. It is found that the proposed algorithm is computationally more efficient and easier to use than other mixed optimization techniques.

**Keywords:** Modified Jaya algorithm, mixed variables, constraint handling, penalty function, balancing.

## 1 Introduction

Various optimization problems deal with integer, discrete, and continuous variables, known as mixed-variable optimization problems. However, many practical design problems consider only discrete and integer variables. Like structural design, the number of bolts for a connection, balancing of the rotor using a set of the balance masses on each plane, and standard diametric pitch of the gear are some examples of discrete variable optimization problems due to a predefined set of standard values. Integer variables are often used for identical elements in engineering design problems such as the number of teeth of a gear [4]. Most researchers have been focused on continuous variable optimization algorithms, where optimum values of design variables lie within their bounds. However, these algorithms are not sufficient for practical design problems due to a predefined set of design variables. Therefore, in recent years, the mixed-variable optimization methods are developed for real-world optimization problems. However, mixed-variable optimization problems can be solved by two classes of optimization techniques: classical and evolutionary techniques. Classical techniques, such as sequential linear programming [24], branch and bound methods [5, 22, 40], a penalty function approach [14, 41], Lagrangian relaxation [15, 18], rounding-off techniques based on continuous variables, cutting plane techniques, and zero-one variable techniques (integer programming) [1] have been applied to mixed-variable optimization problems in order to find out the optimum design variables. However, these methods include more computational cost, low efficiency, and low complexity due to the determination of derivatives and the Hessian matrix of the objective function [3]. Moreover, most of

***Corresponding author: Prem Singh,** Mechanical Engineering Department, Malaviya National Institute of Technology, Jaipur, Rajasthan, India, e-mail: premsingh001@gmail.com
**Himanshu Chaudhary:** Mechanical Engineering Department, Malaviya National Institute of Technology, Jaipur, Rajasthan, India

these algorithms work on continuous and differentiable objective function. Further, these give local optimal solution due to convergence on optimal solution near the start point [2].

Recently, evolutionary optimization techniques are considered effective tools for mixed-variable optimization problems. The differential evolution (DE) [21], evolutionary programming (EP) [7], evolutionary algorithms (EAs) [11], genetic algorithms (GAs) based on mixed variables [10, 34, 46] and discrete variables [30], particle swarm optimization (PSO) algorithms applied to mixed variables [16, 17, 19, 29, 44] and discrete variables [23], ant colony optimization (ACO) [6], artificial bee colony (ABC) [43], teaching-learning based optimization (TLBO) [12], and simulated annealing (SA) [20, 47] are some evolutionary optimization algorithms. Moreover, these algorithms do not require any calculation of derivatives and the Hessian matrix as in the case of classical optimization algorithms. Non-convex and non-differentiable objective function can also be handled using these algorithms.

However, the performance of these algorithms can be affected due to the requirement of algorithmic parameters for its convergence. The selection of these optimization parameters increases the complexity of algorithms. However, these algorithms cannot guarantee giving the global solutions within a definite time. Further, the convergence of these algorithms is slow, thus increasing the computational cost. Moreover, the TLBO algorithm also does not require any algorithmic parameter for its convergence. However, the optimization problems have been solved using two phases: teacher phase and learner phase [9, 31, 35].

The Jaya algorithm is also an evolutionary algorithm. Initially, it has been developed for continuous design variable optimization problems by Rao [32]. Recently, this algorithm has been implemented for solving integer and discrete variable optimization problems [13]. However, no relevant algorithm has been published in which the Jaya algorithm has been applied for solving the mixed-variable optimization problems or optimization problems associated with all variables. Therefore, a modification is made to the original Jaya algorithm for solving the mixed-variable optimization problems. In the modified Jaya algorithm, integer, discrete, and continuous variables are treated as continuous variables in the initial solution.

Further, continuous variables remain in the continuous domain while continuous domains of integer and discrete variables are converted into discrete and integer domains applying bound constraint of the middle point of corresponding two consecutive values of integer and discrete variables. It works on one phase only and uses an initial population to find an optimum global solution. Moreover, this algorithm finds the optimal solution rapidly and updates the worse solution in every iteration [38, 39]. It is easier to use than the other evolutionary optimization algorithms such as ABC, GA, SA, and PSO, etc., and any algorithmic parameters are not required in this algorithm [36]. Furthermore, this algorithm gives a right balance between exploitation and exploration in design space. The robustness and effectiveness of the proposed algorithm are validated through examples of mixed-variable optimization problems taken from the literature, and optimum results are compared with other evolutionary optimization algorithms. It is found that the proposed algorithm is more efficient and easier to use for mixed-variable optimization problems. Continuous and discrete optimization problems can also be solved separately using this algorithm.

The rest of the paper is structured as follows: the optimization problem based on mixed variables is formulated in Section 2. A modified Jaya optimization algorithm is proposed in Section 3, while Section 4 analyzes and validates the results of the proposed algorithm through standard design examples. It is also applied for two-plane balancing of the unbalanced rigid threshing rotor. Finally, conclusions are presented in Section 5.

## 2 Formulation of Mixed-Variable Optimization Problems

This section describes the formulation of optimization problems based on mixed variables. The optimization problem is generally formulated similarly as general optimization problems; the only difference is that variables may be in any form of an integer, discrete, and continuous variable. The optimization problems are formulated by considering continuous and discrete variables as a design variable; the optimization problems are known as continuous and discrete optimization problems. Moreover, problems associated with discrete, integer, and continuous variables are known as mixed-variable optimization problems. The mixed-variable

optimization problem is defined in mathematical form as

$$
\begin{aligned}
&Minimize/Maximize\ Z(\boldsymbol{x}),\\
&\text{Subjected to } g_s(\boldsymbol{x}) \le 0,\ s = 1, 2, \ldots\ldots\ldots\ldots cin.\\
&\boldsymbol{x} = \left[\mathbf{x^c}, \mathbf{x^{int}}, \mathbf{x^d}\right]^{\mathbf{T}} = \left[x_1^c, x_2^c., ., ., x_{nc}^c,\ x_1^{int}, x_2^{int}., ., ., ., x_{nint}^{int},\ x_1^d, x_2^d., ., ., ., x_{nd}^d\right]\\
&x_i^d \in D_i,\ D_i\left(d_{i1}, d_{i2}, d_{i3} \ldots\ldots\ldots\ldots d_{im_i}\right),\ i = 1, ., ., ., .n_d\\
&x_i^{int} \in G_i,\ G_i\left(g_{i1}, g_{i2}, g_{i3} \ldots\ldots\ldots\ldots g_{iq_i}\right),\ i = 1, ., ., ., .n_{nint}\\
&x_i^{cL} \le x_i^c \le x_i^{cU}\\
&x_i^{dL} \le x_i^d \le x_i^{dU}\\
&x_i^{intL} \le x_i^{int} \le x_i^{intU},
\end{aligned}
\tag{1}
$$

where $Z(\mathbf{x})$ and $g(\mathbf{x})$ denote the objective functions and non-equality constraints, respectively. *cin* denotes the numbers of total inequality constraints. $\mathbf{x} = \left[\mathbf{x^c}, \mathbf{x^{int}}, \mathbf{x^d}\right]^{\mathbf{T}}$ is the vector of design variables. $\mathbf{x^c}$, $\mathbf{x^d}$, and $\mathbf{x^{int}}$ present the vector of continuous, discrete, and integer variables. *nc*, *nd*, and *nint* represent the number of continuous, discrete, and integer variables, respectively. The total number of variables is given as $n = nc + nint + nd$. $d_{ij}$ and $g_{ij}$ are the $j$th discrete and integer values for the $i$th variable, respectively. $m_i$ and $q_i$ are the number of discrete and integer values for the $i$th variable, respectively. $D_i$ and $G_i$ are sets of discrete values and integer values for the $i$th variable, respectively. However, the number of discrete values may be different for each variable. $x_i^{cL}$, $x_i^{dL}$, and $x_i^{intL}$ are the lower bounds of $i$th continuous, discrete, and integer variables, respectively. $x_i^{cU}$, $x_i^{dU}$, and $x_i^{intU}$ are the upper bounds of $i$th continuous, discrete, and integer variables, respectively. If there are any equality constraints in the optimization problem, these can be converted into inequality constrains.

The constraint optimization problem described in Eq. (1) is changed into an unconstrained optimization problem using penalty function [42]. The objective function is penalized for an infeasible solution for each constraint violation. Hence, the global optimum solutions, those that satisfy all the constraints, are obtained. Finally, the unconstrained optimization problem is posed as a combination of the objective function and penalty function:

$$\varphi(x) = f(\boldsymbol{x}) + \sum_{j=1}^{cin} C_j * pn^r, \tag{2}$$

$$\mathbf{x_i^L} \le \mathbf{x_i} \le \mathbf{x_i^U}, \tag{3}$$

where $pn^r$ ($r = 1$ to *cin*) presents the penalty value of $10^5$ assigned to objective function for constraint violation. The Boolean function [28] is represented by $C_j$, defined as

$$C_j = \begin{cases} 0 & \text{if } g_s(x) \le 0 \\ 1 & \text{otherwise} \end{cases}. \tag{4}$$

# 3 A Modified Jaya Algorithm for Mixed-Variable Optimization Problems

This section describes a modified Jaya algorithm proposed for mixed-variable optimization problems.

The original Jaya algorithm has been developed by Rao in 2016. It is a population-based evolutionary algorithm that does not require any parameter for its convergence. It works only on one phase compared to TLBO that works on two phases (teacher and learner phases). This algorithm converges rapidly toward the

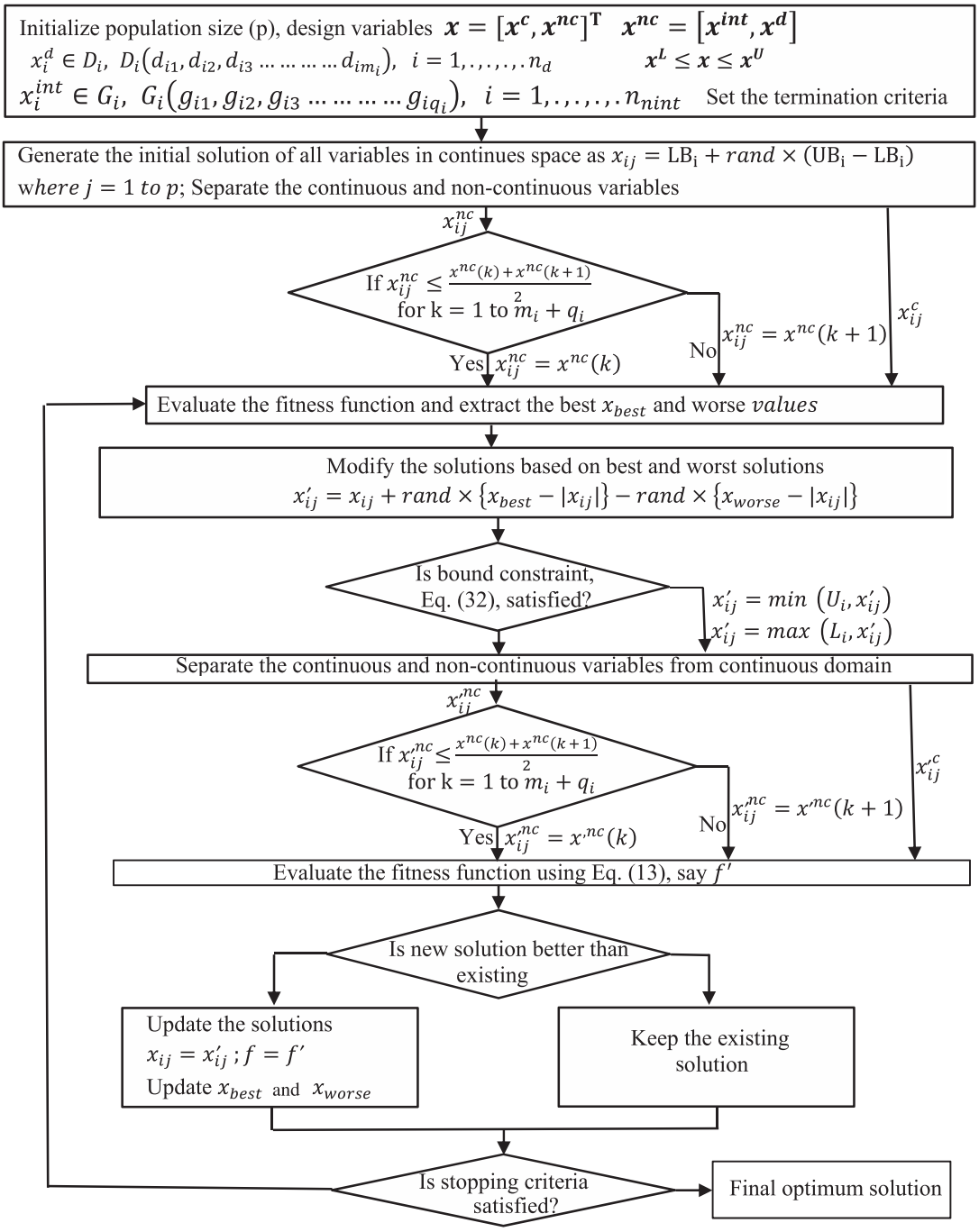


**Figure 1:** A Modified Jaya Algorithm for Mixed-Variable Optimization Problems.

optimal solution in each iteration [33, 37]. The readers may refer to Ref. [32] for the flowchart of original Jaya algorithm. Generally, continuous variables are converted into non-continuous variables using rounding-off operators. Rounding-off operation of a continuous variable may violate the constraints due to the existence of optimal continuous solutions on the boundaries of the functional constraints. Further, researchers check the optimum values for integer and discrete variables before rounding off corresponding continuous variables so that constraints are not violated after rounding off, although this operation increases calculation time.

Therefore, the original Jaya algorithm is modified to handle the various design variables in optimization problems without violation of constraints. This algorithm has no algorithmic parameter and converges fast to the optimal global solution. This algorithm begins with the initialization of parameters. Initial solutions

of all variables are generated in continuous space randomly. Further, continuous variables remain in continuous space while the continuous domain of discrete/integer variables is converted into discrete domain by using bound constraint of a middle point of corresponding two consecutive values of discrete and integer variables. For example, the continuous solutions of discrete and integer variables ($x_{ij}^{nc}$) lie between corresponding discrete values $x^{nc}(k)$ and $x^{nc}(k+1)$. Then, the continuous solutions of discrete and integer variables are converted into discrete solutions if $x_{ij}^{nc} \leq \frac{x^{nc}(k)+x^{nc}(k+1)}{2}$, $x_{ij}^{nc} = x^{nc}(k)$ otherwise $x_{ij}^{nc} = x^{nc}(k+1)$, as shown in Figure 1. Further, the best and worse solutions of the objective function are compared with previous solutions at each iteration. Thus, the best solution is stored and the worse solution is updated in each iteration. The procedure of this algorithm continues until the termination criteria are satisfied. The termination criteria are described by function evaluations and the number of generations. The number of function evaluations is the product of the number of iterations and initial populations or population size, i.e. (number of function evaluations = population size $\times$ number of iterations). Thus, function evaluations are not affected by design variables, but the computational time of the algorithm can be increased. Generally, an algorithm is efficient if it takes the fewer number of function evaluations. The detailed procedure of this algorithm is explained by the flowchart shown in Figure 1. Moreover, this algorithm reduces the computational effort than the other mixed optimization algorithms. However, this is the first time it is applied to mixed optimization problems.

# 4 Design Problems

In this section, the effectiveness of the proposed algorithm as described in the previous section is validated through five design problems taken from the literature. These design problems have been tested using other evolutionary mixed optimization algorithms such as EP [7], EA [11], GAs [10, 30, 34, 46], PSOs [16, 17, 19, 23, 29, 44], ACO [6], ABC [43], TLBO [12], and SAs [20, 47]. However, these problems involve continuous, discrete, and integer variables. Moreover, the optimum results of the proposed algorithm are validated to optimum solutions achieved by other algorithms. This algorithm is also applied to two-plane balancing of unbalance rigid threshing rotor. This algorithm is implemented in MatLab. Bold faces in Tables represent the best values of the design variables and the objectives.

## 4.1 Validation of the Proposed Algorithm Through Five Design Problems

The five design problems are given as follows:

**Example 1:** Design of a welded beam

This design problem includes mixed variables taken from Refs. [21, 29], and the objective of this example is to determine the minimum cost of the welded beam design shown in Figure 2. There are seven non-linear and linear constraints. The length of the welded joint ($l$), thickness of the weld ($h$), bar breadth ($b$), and bar thickness ($t$) are taken as design variables. $l$ and $h$ are two integer variables, while $b$ and $t$ are two discrete variables whose values are multiples of 0.5. These design variables are defined in vector form as $x = [x_1,\ x_2, x_3,\ x_4]^T = [l,\ h,\ b,\ t]^T$.

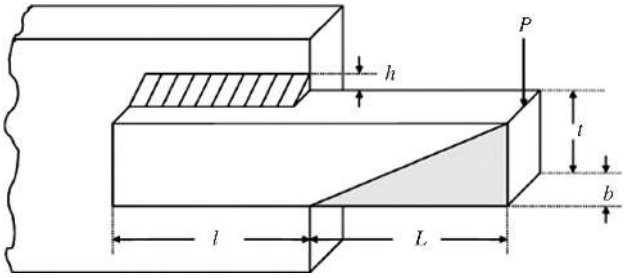


**Figure 2:** Welded Beam Design [29].

The optimization formulation of this design is given as

$$\min f(x) = 1.10471x_2^2x_1 + 0.04811x_3x_4(14 + x_1).$$

Subjected to

$$\left.\begin{aligned}
&g_1(x) = x_2 - x_3 \le 0\\
&g_2(x) = \sigma(x) - \sigma_{\max} \le 0\\
&g_3(x) = 0.10471x_2^2 + 0.04811x_3x_4(14 + x_1) - 5.0 \le 0\\
&g_4(x) = \tau(x) - \tau_{\max} \le 0\\
&g_5(x) = 0.125 - x_2 \le 0\\
&g_6(x) = P - P_c(x) \le 0\\
&g_7(x) = \delta(x) - \delta_{\max} \le 0\\
&0.1 \le x_1 \le 10.0\\
&0.1 \le x_2 \le 2.0\\
&x_3, x_4 \in [0.5, 1.0, 1.5, 2.0, 2.5, ., ., ., ., ., ., ., ., ., .9.5, 10.0]
\end{aligned}\right\}, \tag{5}$$

where

$$\delta(x) = \frac{4PL^3}{Ex_4^3x_3},\ \sigma(x) = \frac{6PL}{x_3x_4^2},\ P_c(x) = \frac{4.013\sqrt{EG\left(x_4^2x_3^6/36\right)}}{l^2}\left(1 - \frac{x_4}{2L}\sqrt{\frac{E}{4G}}\right)$$

$$\tau(x) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \tau' = \frac{P}{\sqrt{2}x_1x_2},\ \tau'' = \frac{MR}{J},\ M = P\left(L + \frac{x_1}{2}\right),\ R = \sqrt{\frac{x_1^2}{4} + \left(\frac{x_2 + x_4}{2}\right)^2}$$

$$J = 2\sqrt{2}x_1x_2\left(\frac{x_1^2}{12} + \left(\frac{x_2 + x_4}{2}\right)^2\right).$$

The design parameters are taken as

$$P = 6000 \text{ lb},\ E = 30 \times 10^6 \text{ lbf/in}^2,\ G = 12 \times 10^6 \text{ lbf/in}^2,\ L = 14 \text{ in},$$

$$\tau_{\max} = 13{,}600 \text{ lbf/in}^2,\ \delta_{\max} = 0.25 \text{ in},\ \sigma_{\max} = 30{,}000 \text{ lbf/in}^2.$$

A total of 20 initial populations and 30 iterations are chosen for this design example. The best objective function values and the mean of all function values corresponding to the best run are obtained in 20 independent runs. The convergence performance of the best and mean values of the objective function is presented in Figure 3. The best and mean values of the objective function are obtained in 600 function evaluations as 4.3521 and 4.3521, respectively. The optimum solutions of the welded beam design are validated with the optimum solutions of other algorithms, as shown in Table 1. Table 1 presents that the proposed algorithm gives a better optimum design than that of Ref. [34] and equal to that of Refs. [29, 44]. However, the proposed algorithm takes less function evaluation for finding the best objective function value compared to other evolutionary optimization algorithms.

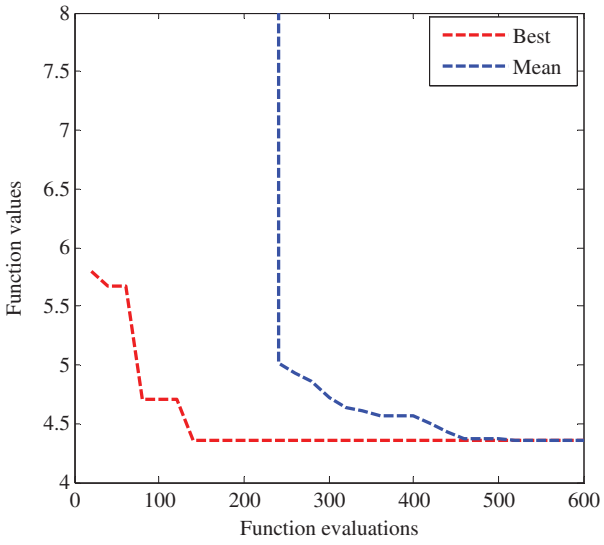


**Figure 3:** Convergence Graph of Best and Mean Values of Objective Function for Welded Beam Design.

**Table 1:** Optimum Designs of the Welded Beam.

| Design variables ($x$) | MDHGA [34] | HPB [29] | MPSO [44] | This study |
|---|---|---|---|---|
| $x_1$ | 2.0 | 1 | 1 | 1 |
| $x_2$ | 1.0 | 1 | 1 | 1 |
| $x_3$ | 1.0 | 1 | 1 | 1 |
| $x_4$ | 4.5 | 4.5 | 4.5 | 4.5 |
| $f(x)$ | 5.67334 | 4.3521 | 4.3521 | **4.3521** |
| Function evaluations | - | 800 | 6750 | **600** |
| Constraints violation | None | None | None | **None** |

MDHGA, Mixed discrete hybrid genetic algorithm; HPB, Hybrid particle swarm branch and bound; MPSO, Modified particle swarm optimization. Bold face represent the best values of the design variables and the objective.

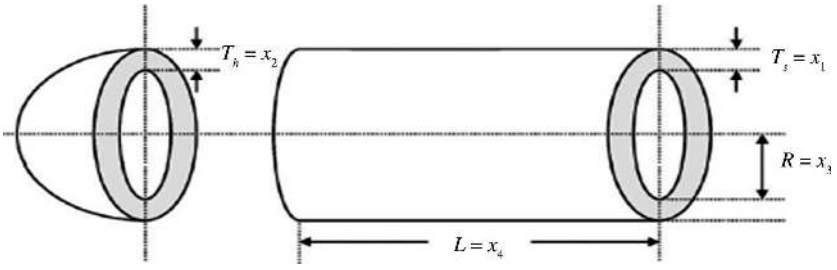


**Figure 4:** Pressure Vessel Design [40].

**Example 2:** Pressure vessel design

This design deals with the pressure vessel design taken from the literature [40], as shown in Figure 4. The objective of this design is to minimize the manufacturing cost of the pressure vessel with specific design constraints. The design variables are taken as shell thickness ($T_s$), spherical head thickness ($T_h$), radius ($R$), and shell length ($L$). $T_s$ and $T_h$ are the discrete variables whose values are multiples of 0.0625 in, while $R$ and $L$ are defined as continuous variables. The design variables are described in vector form as

$$\boldsymbol{x} = [x_1,\ x_2, x_3,\ x_4]^T = [T_h, T_s, l, R]^T.$$

The optimization problem is posed [40] as

$$\min f(\boldsymbol{x}) = 0.6224x_2x_3x_4 + 1.7781x_1{x_4}^2 + 3.1661{x_2}^2x_3 + 19.84{x_2}^2x_4.$$

Subjected to

$$\left.\begin{aligned} &g_1(\boldsymbol{x}) = 0.0193x_4 - x_2 \leq 0\\ &g_2(\boldsymbol{x}) = 0.00954x_4 - x_1 \leq 0\\ &g_3(\boldsymbol{x}) = 1296{,}000 - \pi {x_4}^2 x_3 - \frac{4}{3}\pi {x_4}^2 \leq 0\\ &g_4(\boldsymbol{x}) = x_3 - 240 \leq 0\\ &10 \leq x_3 \leq 200\\ &10 \leq x_4 \leq 200\\ &x_1, x_2 \in [0.0625, 0.1250, 0.1875,\ 0.2500, ., ., ., ., ., 6.00, 6.0625, 6.1250, 6.1875] \end{aligned}\right\}. \quad (6)$$

The number of iterations and initial populations are considered as 100 and 20, respectively, for this example. Ten independent runs are chosen to find the best values of the objective function and mean of all objective function value corresponding to the best run. The convergence rates of the best and mean objective function values are shown in Figure 5. The best and mean values of the objective function are obtained in 2000 function evaluations as 6059.70 and 6059.74, respectively. The optimum results for the design of pressure vessel are shown in Table 2. Table 2 shows that the optimum value of the objective function is better than that of other optimization algorithms. Moreover, the proposed algorithm takes less function evaluation for finding the best objective function value compared to different evolutionary optimization algorithms.

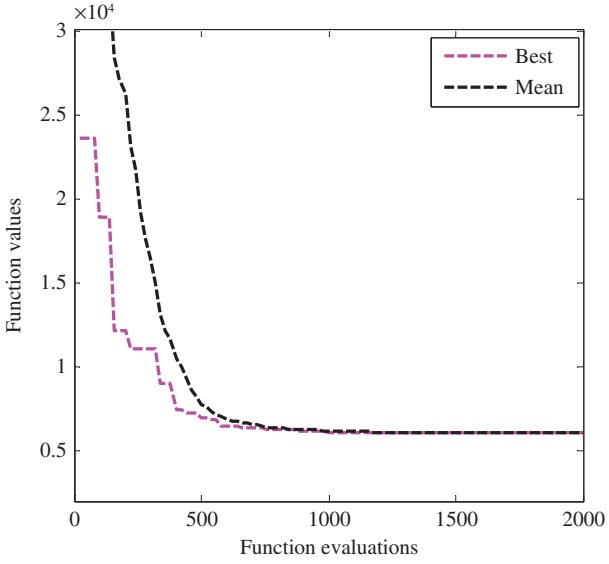


**Figure 5:** Convergence of Best and Mean Values of Objective Function for Pressure Vessel Design.

**Table 2:** Optimum Solutions for the Design of Pressure Vessel.

| Design variables ($x$) | EP [7] | EA [11] | GA [10] | PSO [17] | HPB [29] | MPSO [44] | This study |
|---|---|---|---|---|---|---|---|
| $x_1$ | 0.625 | 0.5 | 0.4375 | 0.4375 | 0.4375 | 0.4375 | **0.4375** |
| $x_2$ | 1 | 0.9345 | 0.8125 | 0.8125 | 0.8125 | 0.8125 | **0.8125** |
| $x_3$ | 90.7821 | 112.679 | 176.654 | 176.6366 | 176.6366 | 176.636792 | **176.636792** |
| $x_4$ | 51.1958 | 48.329 | 40.0974 | 42.09845 | 42.09845 | 42.098446 | **42.098446** |
| $f(x)$ | 7108.616 | 6410.381 | 6059.946 | 6059.714 | 6059.714 | 6059.718932 | **6059.700** |
| Function evaluations | 100,000 | 42,000 | 30,000 | 30,000 | 4013 | 400,000 | **2000** |
| Constraints violation | None | None | None | None | None | None | None |

HPB, Hybrid particle swarm branch and bound; MPSO, Modified particle swarm optimization. Bold face represents the best values of the design variables and the objective.

**Example 3:** Ten-bar planar truss design

A 10-bar planar truss design is taken as an optimization problem shown in Figure 6 [30]. In this truss structure design, a minimization optimization problem is formulated by considering the weight of the truss as the objective function with the constraints of displacements at each nodal point and the stress induced in each member. This problem is based on discrete optimization problems in which cross-sectional areas of each member are discrete variables. The nodes 2 and 4 are subjected to a vertical nodal load of 100 kips. The modulus of elasticity of the material of each bar and the density are considered as $E = 10{,}000$ ksi and $\rho = 0.1$ lb/in$^3$, respectively. The allowable displacements for the free nodes and the allowable stress for all members are taken as $\pm 2$ in for both directions and $\pm 25$ ksi, respectively. Ten discrete design variables and their values are selected from the standard values $D = \{1.62, 2.38, 1.99, 1.80, 2.13, 2.62, 2.88, 3.09, 2.93, 2.63, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.5, 13.5, 14.2, 13.9, 15.5, 16.0, 16.9, 18.8, 1.99, 22.0, 22.9, 26.5, 30.0, 33.5\}$ (in$^2$) [1]. The vector form of design variables is expressed as

$$\boldsymbol{x} = [x_i] = [A_i]\ i = 1 \text{ to } 10.$$

The formulation of the problem is presented as

$$\textit{Minimize weight } (f) = \sum_{i=1}^{n} \rho x_i L_i.$$

Subjected to

$$\left.\begin{array}{l} \dfrac{\sigma_i}{\sigma_a} - 1 \le 0 \\ \dfrac{u_i}{u_a} - 1 \le 0 \end{array}\right\}, \tag{7}$$

where $L_i$, $\sigma_i$, $u_i$, and $A_i$ are length, stresses, deflection, and cross-sectional area of the $i$-th member, respectively.

For this problem, the initial populations and the number of iterations are set to 10 and 95, respectively. Ten independent runs are performed to find out the best values of the objective function, and the mean of all objective function values correspond to best run. The best and mean values of the objective function in 10 runs are 5490.74 and 5493.54, respectively. This algorithm takes 9500 function evaluations. The convergence plot of the best and mean values of the objective function is shown in Figure 7. The comparison of optimum solutions for planar 10-bar truss is presented in Table 3. Table 3 shows that the optimum objective function value is better than that of Refs. [23, 30] and close to that of Refs. [6, 12, 20, 32, 43]. However, the proposed algorithm takes less function evaluation for finding the best objective function value compared to other evolutionary optimization algorithms. Thus, the proposed approach is more efficient and reduces the computational cost.

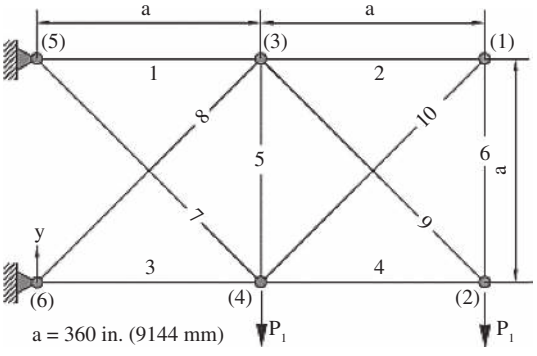


**Figure 6:** A Planar 10-Bar Truss Structure [30].

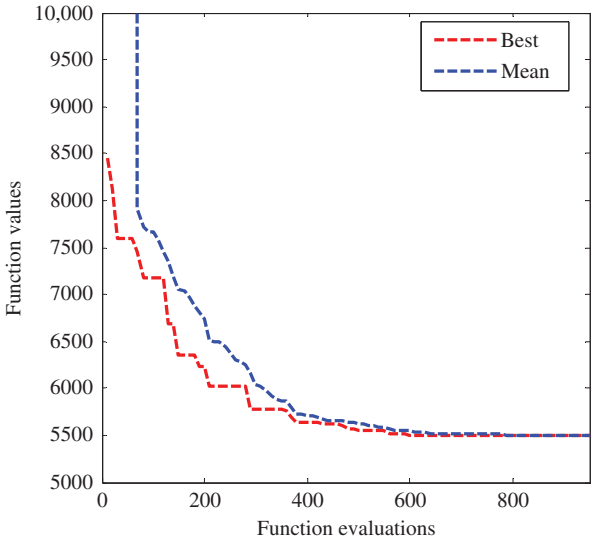


**Figure 7:** Convergence of Best and Mean Values of Objective Function for Planar 10-Bar Planar Truss.

**Table 3:** Comparison of Optimum Solutions for Planar 10-Bar Truss Structure.

| Design variables (in$^2$) | GA [30] | SA [20] | HPSO [23] | ACO [6] | ABC [43] | TLBO [12] | Jaya [32] | This study |
|---|---|---|---|---|---|---|---|---|
| $A_1$ | 33.5 | 33.5 | 30 | 33.5 | 33.5 | 33.5 | 33.5 | **33.5** |
| $A_2$ | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | **1.62** |
| $A_3$ | 22 | 22.9 | 22.9 | 22.9 | 22.9 | 22.9 | 22.9 | **22.9** |
| $A_4$ | 15.5 | 14.2 | 13.5 | 14.2 | 14.2 | 14.2 | 14.2 | **14.2** |
| $A_5$ | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | **1.62** |
| $A_6$ | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | 1.62 | **1.62** |
| $A_7$ | 14.2 | 7.97 | 7.97 | 7.97 | 7.97 | 7.97 | 7.97 | **7.97** |
| $A_8$ | 19.9 | 22.9 | 26.5 | 22.9 | 22.9 | 22.9 | 22.9 | **22.9** |
| $A_9$ | 19.9 | 22 | 22 | 22 | 22 | 22 | 22 | **22** |
| $A_{10}$ | 2.62 | 1.62 | 1.8 | 1.62 | 1.62 | 1.62 | 1.62 | **1.62** |
| $W$ (lb) | 5613.84 | 5490.74 | 5531.9 | 5490.74 | 5490.74 | 5490.74 | 5490.74 | **5490.74** |
| Function evaluation | N/A | N/A | 50,000 | 10,000 | 25,800 | 1000 | 950 | **950** |
| Constraints violation | None | None | None | None | None | None | None | **None** |

Bold face represents the best values of the design variables and the objective.

**Example 4:** A helical spring design

This example consists of the design of helical spring under the constant and axial load, as shown in Figure 8 [29]. The minimization of spring weight is considered as an objective function with certain inequality constraints. The number of spring coils ($N$), outside diameter of the spring ($D$), and spring wire diameter ($d$) are the design variables. This problem involves integer, discrete, and continuous variables, where the number of coils ($N$) is an integer variable, the outside diameter of the spring ($D$) is a continuous variable, and the spring wire diameter is a discrete variable, whose standard values are chosen. Design variables are in vector form as

$$\boldsymbol{x} = [x_1,\ x_2, x_3\ ]^T = [D,\ N,\ d]^T.$$

The formulation of the optimization problem is posed as

$$\min f(x) = \frac{\pi^2 x_2 x_1^2 (x_3 + 2)}{4}.$$

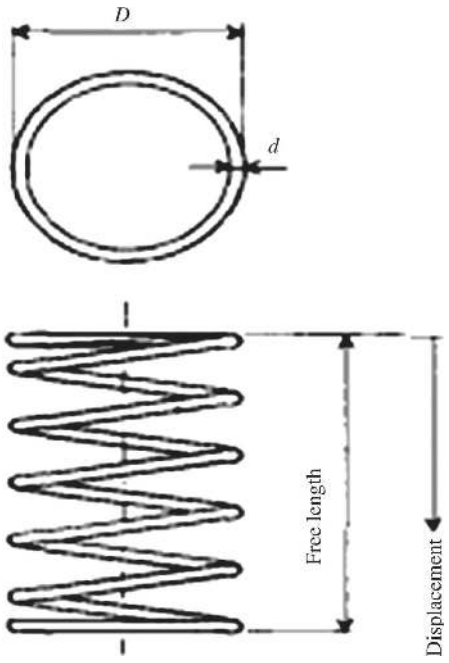


**Figure 8:** Design of Helical Spring [29].

Subjected to

$$
\left.
\begin{aligned}
&g_1(x) = \frac{8C_f F_{\max} x_2}{\pi x_3^3} - S \le 0\\
&g_2(x) = l_f - l_{\max} \le 0\\
&g_3(x) = d_{\min} - x_3 \le 0\\
&g_4(x) = x_2 - D_{\max} \le 0\\
&g_5(x) = 3.0 - \frac{x_1}{x_3} \le 0\\
&g_6(x) = \delta - \delta_m \le 0\\
&g_7(x) = \delta + \frac{F_{\max} - F}{K} + 1.05(x_2 + 2)x_3 - l_f \le 0\\
&g_8(x) = \delta_w - \frac{F_{\max} - F}{K} \le 0\\
&0.6 \le x_1 \le 3\\
&1 \le x_2 \le 70\\
&x_3 \in [0.009, 0.0095, 0.0104, 0.0118, 0.0128, 0.0132, 0.014, 0.015,\\
&0.0162, 0.0173, 0.018, 0.020, 0.023, 0.025, 0.028, 0.032, 0.035,\\
&0.041, 0.047, 0.054, 0.063, 0.072, 0.080, 0.092, 0.105, 0.120,\\
&0.135, 0.148, 0.162, 0.177, 0.192, 0.207, 0.225, 0.244, 0.263,\\
&0.283, 0.307, 0.331, 0.362, 0.394, 0.4375, 0.500]
\end{aligned}
\right\} \quad (8)
$$

where

$$
C_f = \frac{4\left(\frac{x_1}{x_3}\right) - 1}{4\left(\frac{x_1}{x_3}\right) - 4} + \frac{0.615x_3}{x_1},\ K = \frac{Gx_3^4}{8x_2x_1^3},\ \delta = \frac{F}{K},\ l_f = \frac{F_{\max}}{K} + 1.05(x_2 + 2)x_3.
$$

The values of predefined parameters of spring are given as

$F_{\max} = 1000.0$ lb, $l_{\max} = 14.0$ in, $d_{\min} = 0.2$ in, $S = 189{,}000.0$ lbf/in$^2$, $d_{\max} = 3.0$ in, $F = 300.0$ lb.

$\delta_m = 6.0$ in, $\delta_w = 1.25$ in, $G = 11.5 \times 10^6$ lbf/in$^2$.

For this design problem, the number of iterations and initial populations are set to 40 and 20, respectively. Twenty independent runs are chosen to find the best values of the objective function and the mean of all objective function values corresponding to the best run. The convergence performance of the best and mean objective function values is shown in Figure 9. The best and mean values of the objective function are obtained in 800 function evaluations as 2.6585 and 2.7746, respectively. The optimum design of the spring is validated with the results obtained by other algorithms as shown in Table 4. Table 4 shows that the proposed algorithm gives a better optimum value of the objective function than that of Ref. [11] and equal to that of Refs. [17, 21, 29, 44]. However, the proposed algorithm takes less function evaluation for finding the best objective function value compared to other evolutionary optimization algorithms.

**Example 5:** Compound gear train design

The purpose of this design is to obtain the optimum gear ratio of the gear train arrangement as presented in Figure 10 [16]. The ratio of the output shaft angular velocity to the input shaft angular velocity is known as the gear ratio of the gear train. The effective overall gear ratio $G_r$ is expressed as

$$G_r = \frac{\omega_{\text{out}}}{\omega_{\text{in}}} = \frac{T_b T_d}{T_a T_f},$$

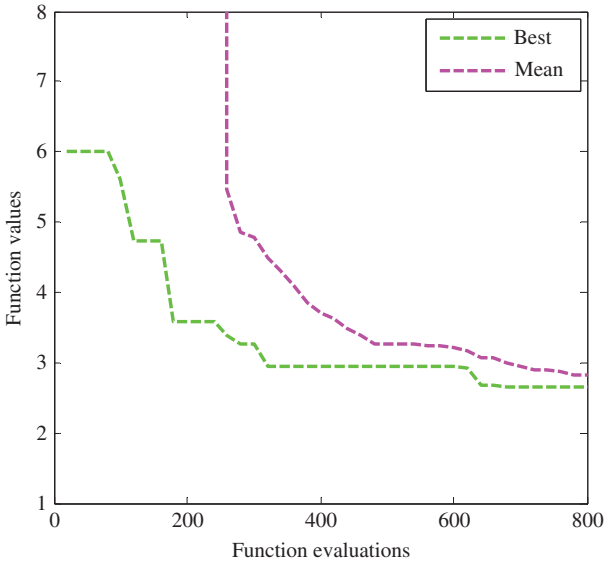


**Figure 9:** Convergence Characteristic of Objective Function for Spring Design.

**Table 4:** Optimal Solutions Comparison for the Spring Design.

| Design variables ($x$) | EA [11] | DE [21] | PSO [17] | HPB [29] | MPSO [14, 44] | This study |
|---|---|---|---|---|---|---|
| $x_1$ | 1.226 | 1.223041 | 1.223041 | 1.223041 | 1.223041 | **1.2231** |
| $x_2$ | 9 | 9 | 9 | 9 | 9 | **9** |
| $x_3$ | 0.283 | 0.283 | 0.283 | 0.283 | 0.283 | **0.283** |
| $f(x)$ | 2.665 | 2.65856 | 2.65856 | 2.6585 | 2.6585 | **2.6586** |
| Function evaluations | 30,000 | 30,000 | 30,000 | 835 | 10,000 | **800** |
| Constraints violation | None | None | None | None | None | **None** |

HPB, hybrid particle swarm branch and bound; MPSO, modified particle swarm optimization. Bold face represent the best values of the design variables and the objective.

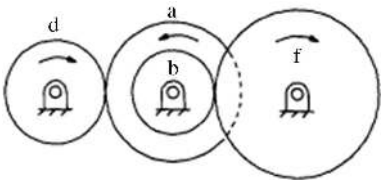


**Figure 10:** Gear Train Design [16].

where $\omega_{\text{in}}$ and $\omega_{\text{out}}$ represent input and output shaft angular velocities, respectively, and the number of teeth of each gear is represented by $T$. The teeth number of each gear is taken as a design variable. However, all design variables are integers whose values lie between 12 and 60. A vector form of design variables is expressed as

$$\boldsymbol{x} = [T_b \; T_d \; T_a \; T_F]^T = [x_1 \; x_2 \; x_3 \; x_4]^T.$$

The optimization problem is posed as

$$f(x) = \left(\frac{1}{6.931} - \frac{x_1 x_2}{x_3 x_4}\right)^2,$$

$$12 \leq x_i \leq 60, \; i = 1 \text{ to } 4. \tag{9}$$

The initial populations of 150 and the number of iterations of 100 are decided for this example. The best value of the objective function and the mean of all function values corresponding to best run are obtained for 30 independent runs. The convergence performance of objective function values is shown in Figure 11. The best and mean values of the objective function are obtained in 15,000 function evaluations as $2.7 \times 10^{-12}$ and $1.4311 \times 10^{-7}$, respectively. The optimum design of the gear train is validated with the design achieved by other algorithms as shown in Table 5. Table 5 presents that the proposed algorithm gives a nearly same optimum solution of the objective function to that of the different evolutionary algorithm. However, it minimizes the percentage of error known as the difference between the mean and best values of objective functions, compared to other optimization algorithms. Thus, the proposed algorithm can be effectively applied to integer optimization problems.

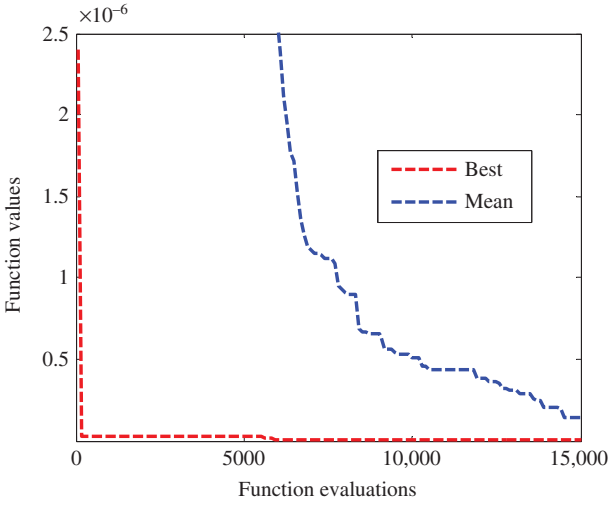


**Figure 11:** Convergence Graph of Best and Mean Values of Objective Function for Gear Train Design.

**Table 5:** Optimal Design for the Compound Gear Train.

| Design variables ($x$) | SA [47] | EP [7] | HSIA [16] | MPSO [44] | Jaya [32] | This study |
|---|---|---|---|---|---|---|
| $x_1$ | 30 | 30 | 16 | 16 | 16 | **16** |
| $x_2$ | 15 | 15 | 19 | 19 | 19 | **19** |
| $x_3$ | 52 | 52 | 43 | 43 | 43 | **43** |
| $x_4$ | 60 | 60 | 49 | 49 | 49 | **49** |
| $f(x)$ | $2.36 \times 10^{-6}$ | $2.36 \times 10^{-6}$ | $2.7 \times 10^{-12}$ | $2.7 \times 10^{-12}$ | $2.7 \times 10^{-12}$ | $\mathbf{2.7 \times 10^{-12}}$ |
| Function evaluations | – | – | – | 400,000 | 18,000 | **15,000** |
| % Error | 0.033 | 0.033 | 0.0011 | 0 | $9.8 \times 10^{-13}$ | $\mathbf{9.8 \times 10^{-13}}$ |
| Gear ratio | 0.14423 | 0.14423 | 0.14428 | 0.14428 | 0.14428 | **0.14428** |

HSIA, Hybrid swarm intelligence approach; MPSO, Modified particle swarm optimization. Bold face represent the best values of the design variables and the objective.

## 4.2 Application – Two-Plane Balancing of Unbalanced Rigid Threshing Rotor

The proposed algorithm developed in Section 3 is applied for balancing of the rigid threshing rotor. The threshing rotor is an important part of agricultural thresher machine and detaches grains from the panicles by impact action of the beaters attached to the rotor [45]. In this problem, the rotor is balanced by placing the number of discrete masses at corresponding discrete angles on two balanced planes. Moreover, this balancing problem involves only discrete variables. A rotor is mounted on bearings $p$ and $q$, as shown in Figure 12. A fixed coordinate system is denoted by $(x, y, z)$, while a rotating coordinate $(x_r, y_r, z_r)$ is attached to the shaft rotating at a constant angular velocity $\omega$ about the $z$-axis. The point $O$ denotes the origin of the coordinate system. Two balance planes are centered at points $c_1$ and $c_2$, respectively. The rotor's center of mass is eccentric at a distance of $e$ from the axis of rotation due to imbalance of the rotor. $F_p$ and $F_q$ are lengths measured from $O$ to the bearings $p$ and $q$, respectively. The numbers of discrete masses $m_{ij}$ ($j = 1$ to $M_i$) placed at radius $R_i$ correspond to discrete angular positions $\alpha_{ij}$ made to $x$-axis on balance plane $i$ ($i = 1, 2$), as shown in Figure 12.

The rotor is described by its mass $m$ and inertia tensor $I_G$. Inertia tensor $I_G$ is given as

$$[I_G] = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{yz} & I_{yz} & I_{zz} \end{bmatrix}. \tag{10}$$

The position of its center of mass $G$ is given as $OG(e_x, e_y)$,

$$\text{where } e_x = e\cos(\theta + \omega t) \text{ and } e_y = e\sin(\theta + \omega t).$$

The equilibrium forces and moments acting on supports are determined.

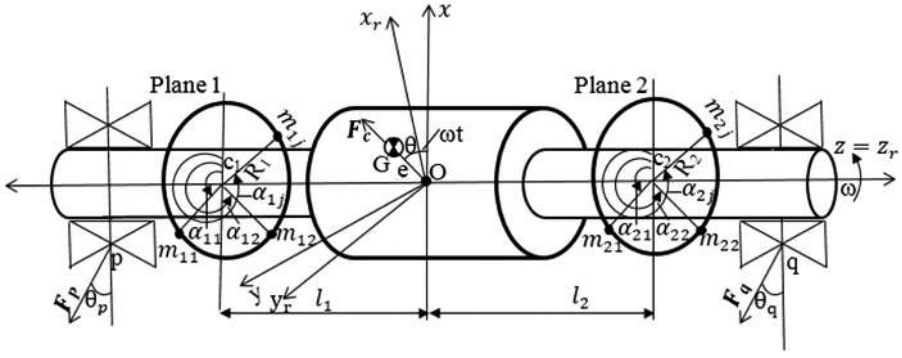


**Figure 12:** Balancing Model of Rigid Threshing Rotor.

The resultant forces $F_p$ and $F_q$ acting on supports $p$ and $q$ are determined using the Newton–Euler equation [8] as

$$F_p = \frac{\omega^2}{l_p - l_q}\sqrt{\left(I_{xz} + ml_q e_x + \sum_{i=1}^{2}\sum_{j=1}^{M_i} m_{ij}R_i(l_q - l_i)\cos\alpha_{ij}\right)^2 + \left(I_{yz} + ml_q e_y + \sum_{i=1}^{2}\sum_{j=1}^{M_i} m_{ij}R_i(l_q - l_i)\sin\alpha_{ij}\right)^2}. \tag{11}$$

$$F_q = \frac{\omega^2}{l_p - l_q}\sqrt{\left(I_{xz} + ml_p e_x + \sum_{i=1}^{2}\sum_{j=1}^{M_i} m_{ij}R_i(l_p - l_i)\cos\alpha_{ij}\right)^2 + \left(I_{yz} + ml_p e_y + \sum_{i=1}^{2}\sum_{j=1}^{M_i} m_{ij}R_i(l_p - l_i)\sin\alpha_{ij}\right)^2}. \tag{12}$$

Moreover, the resultant reaction forces for the unbalanced rotor (when balance masses per plane are placed as zero value) are calculated using Eqs. (15) and (16) as

$$F_{pu} = \frac{\omega^2}{l_p - l_q}\sqrt{(I_{xz} + ml_q e_x)^2 + (I_{yz} + ml_q e_y)^2}. \tag{13}$$

$$F_{qu} = \frac{\omega^2}{l_p - l_q}\sqrt{(I_{xz} + ml_p e_x)^2 + (I_{yz} + ml_p e_y)^2}. \tag{14}$$

Normalized forces $F_{P,norm}$ and $F_{Q,norm}$ with respect to $F_{Pu}$ and $F_{qu}$ is written as

$$F_{p,norm} = \frac{F_P}{F_{Pu}}. \tag{15}$$

$$F_{q,norm} = \frac{\mathrm{F}_q}{F_{qu}}. \tag{16}$$

The minimization of reaction forces $F_p$ and $F_q$ acting on supports is expressed as multi-objective functions with discrete constraints on design variables. These multi-objective functions are converted into a single objective function using appropriate weighting factors having any values of 0, 1, and between 0 and 1 based on the importance of the objective functions. The rotor is balanced by placing the masses at different angular positions at a fixed radius from the finite sets of masses and available angular position for each balancing plane. These masses and angular position per plane are taken as design variables, where $x_i$ of design variable for the $i$-th plane is expressed in vector form as

$$\boldsymbol{x}_i = \left[m_{i1}\ \alpha_{i1}\ m_{i2}\ \alpha_{i2}\ m_{i3}\ \alpha_{i3} \ldots\ m_{iM_i}\ \alpha_{iM_i}\right]^T, \tag{17}$$

where $m_{ij}$ and $\alpha_{ij}$ are the $j$-th mass and corresponding angular position of the $i$-th plane, respectively. Hence, the design vector, $\mathbf{x}$, for the rotor having two balancing planes as shown in Figure 12 is given by

$$\boldsymbol{x} = \left[\boldsymbol{x}_1^T\ \boldsymbol{x}_2^T\right]^T. \tag{18}$$

The formulation of the optimization problem is expressed as a weighted sum of the reaction forces acting on supports as given in Eqs. (17) and (18) as

$$\text{Minimize } Z = w_1 F_{P,norm} + w_2 F_{Q,norm}. \tag{19}$$

The discrete constraint to design variables is defined as

$$\left.\begin{aligned} m_{ij} &\in \left\{m_1; m_2; m_3 \ldots\ldots\ldots\ldots\ldots; m^{D_1}\right\} \\ \alpha_{ij} &\in \left\{\alpha_1; \alpha_2; \alpha_3 \ldots\ldots\ldots\ldots\ldots; \alpha^{D_2}\right\} \end{aligned}\right\} \text{for } i = 1, 2 \text{ and } j = 1, 2 \ldots, \ldots, \ldots M_i.$$

$$LB_i \leq x_i \leq UB_i \; i = 1, \ldots\ldots\ldots, \; N, \tag{20}$$

Where $D_1$ and $D_2$ are total numbers of discrete values of masses and discrete values of corresponding angular position, $LB_i$ and $UB_i$ denote the lower and upper side constraints of $i$-th design variable, and $M$ and $N$ represent the number of balancing masses and number of design variables, respectively. The weighting factors $w_1$ and $w_2$ are used to assign weightage to forces acting on supports. These factors transform the multiple objective functions into a single objective function. The various approaches for selection of the weighting factors are presented in Refs. [25, 26]. The weightage defines the importance of the various objective functions. However, both the objective functions have equal importance in the balancing of the unbalance rotor. Therefore, $w_1 = 0.5$ and $w_2 = 0.5$ are chosen for this study.

**Table 6:** Comparison of Performance of the Modified Jaya Algorithm with a GA Algorithm [27].

| Algorithm | No. of masses | Population size | No. of iterations | No. of function evaluations | Possible solutions |
|---|---|---|---|---|---|
| GA | $M_1 = M_2 = 1$ | 50 | 150 | 7500 | 7056 |
| | $M_1 = M_2 = 2$ | 150 | 300 | 45,000 | $49.8 \times 10^6$ |
| | $M_1 = M_2 = 3$ | 400 | 3000 | $12 \times 10^5$ | $3.51 \times 10^{11}$ |
| Modified Jaya | $M_1 = M_2 = 1$ | 10 | 100 | 1000 (−87%) | 7056 |
| | $M_1 = M_2 = 2$ | 100 | 200 | 20,000 (−56%) | $49.8 \times 10^6$ |
| | $M_1 = M_2 = 3$ | 300 | 2500 | $7.5 \times 10^5$ (−38%) | $3.51 \times 10^{11}$ |

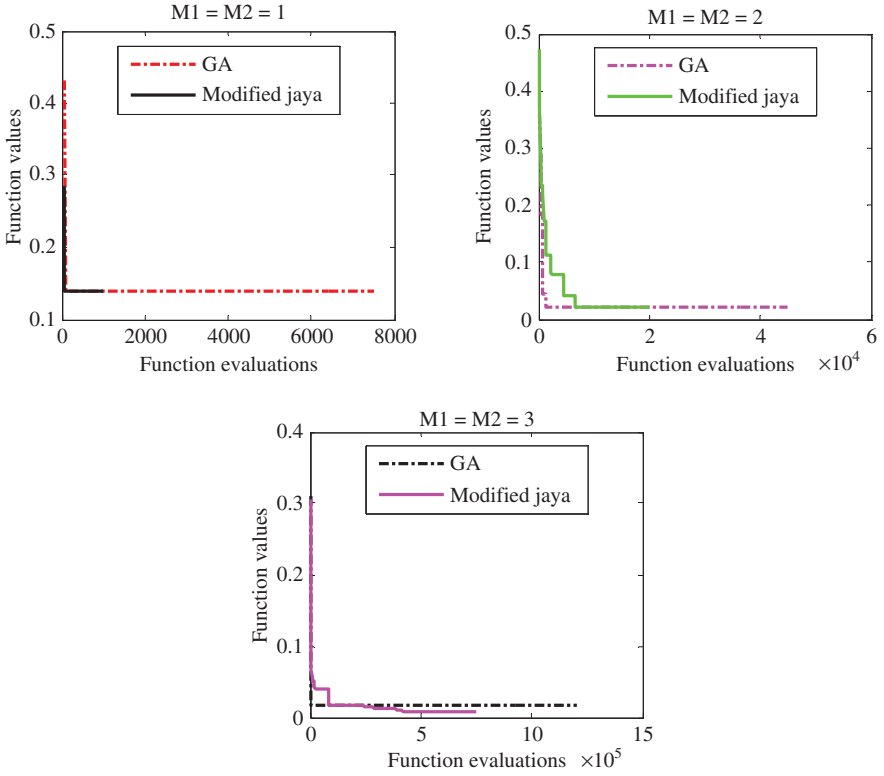


**Figure 13:** Convergence Rate of Best Objective Function Values in GA and Modified Jaya for the Number of Balance Masses for Case 1.

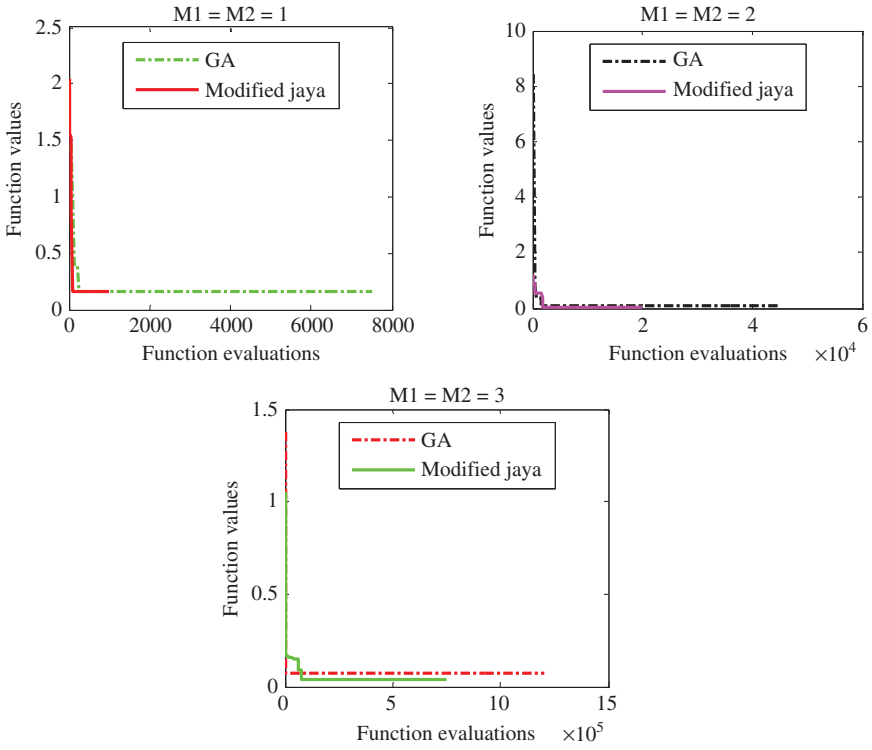


**Figure 14:** Convergence Rate of Best Objective Function Values in GA and Modified Jaya for the Number of Balance Masses for Case 2.

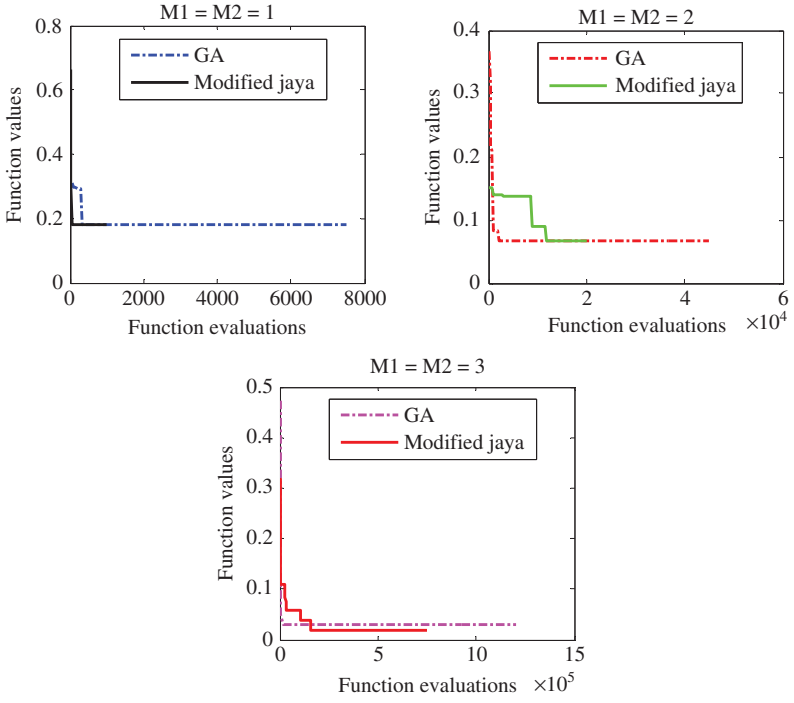


**Figure 15:** Convergence Rate of Best Objective Function Values in GA and Modified Jaya for the Number of Balance Masses for Case 3.

Further, the dimensions of the rotor are taken from Ref. [45] as $R_1 = R_2 = 0.326$, $l_p = -0.356$, $l_q = 0.356$, $l_1 = -1.430$, and $l_2 = 0.470$ (in m). The total mass of rotor $m = 60$ kg and constant rotating speed

**Table 7:** Comparison of Optimum Results of Modified Jaya Algorithm to GA [27] for Case 1.

| Algorithm | No. of masses | | Plane 1 | | Plane 2 | | $F_p$ (N) | $F_q$ (N) |
|---|---|---|---|---|---|---|---|---|
| | $M_1$ | $M_2$ | $m_{1j}$(g) | $\alpha_{1j}$ (°) | $m_{2j}$(g) | $\alpha_{2j}$ (°) | | |
| GA | 1 | 1 | 100 | 180 | 100 | 210 | 5.12 | 7.08 |
| | 2 | 2 | 50 | 180 | 20 | 300 | 0.3898 | 2.5052 |
| | | | 50 | 210 | 100 | 180 | | |
| | 3 | 3 | 200 | 270 | 50 | 150 | 0.82 | 0.105 |
| | | | 100 | 180 | 50 | 180 | | |
| | | | 10 | 360 | 50 | 270 | | |
| Modified Jaya | 1 | 1 | 100 | 180 | 100 | 210 | **5.12** | **7.08** |
| | 2 | 2 | 50 | 180 | 20 | 300 | **0.3898** | **2.5052** |
| | | | 50 | 210 | 100 | 180 | | |
| | 3 | 3 | 100 | 210 | 10 | 120 | **0.23** | **0.67** |
| | | | 10 | 60 | 20 | 90 | | |
| | | | 20 | 120 | 100 | 210 | | |

**Table 8:** Comparison of Optimum Results of Modified Jaya Algorithm to GA [27] for Case 2.

| Algorithm | No. of masses | | Plane 1 | | Plane 2 | | $F_p$ (N) | $F_q$ (N) |
|---|---|---|---|---|---|---|---|---|
| | $M_1$ | $M_2$ | $m_{1j}$(g) | $\alpha_{1j}$ (°) | $m_{2j}$(g) | $\alpha_{2j}$ (°) | | |
| GA | 1 | 1 | 50 | 60 | 50 | 240 | 1.67 | 1.67 |
| | 2 | 2 | 10 | 210 | 50 | 270 | 1.11 | 0.35 |
| | | | 50 | 60 | 20 | 150 | | |
| | 3 | 3 | 50 | 330 | 50 | 270 | 1.11 | 0.32 |
| | | | 20 | 90 | 20 | 210 | | |
| | | | 50 | 120 | 20 | 90 | | |
| Modified Jaya | 1 | 1 | 50 | 60 | 50 | 240 | **1.67** | **1.67** |
| | 2 | 2 | 10 | 180 | 50 | 240 | **0.48** | **0.48** |
| | | | 50 | 60 | 10 | 360 | | |
| | 3 | 3 | 20 | 90 | 20 | 270 | **0.35** | **0.35** |
| | | | 10 | 30 | 20 | 240 | | |
| | | | 20 | 60 | 10 | 210 | | |

**Table 9:** Comparison of Optimum Results of Modified Jaya Algorithm to GA [27] for Case 3.

| Algorithm | No. of masses | | Plane 1 | | Plane 2 | | $F_p$ (N) | $F_q$ (N) |
|---|---|---|---|---|---|---|---|---|
| | $M_1$ | $M_2$ | $m_{1j}$(g) | $\alpha_{1j}$ (°) | $m_{2j}$(g) | $\alpha_{2j}$ (°) | | |
| GA | 1 | 1 | 100 | 180 | 100 | 210 | 7.06 | 4.57 |
| | 2 | 2 | 100 | 180 | 20 | 210 | 2.90 | 0.32 |
| | | | 20 | 30 | 100 | 210 | | |
| | 3 | 3 | 10 | 60 | 10 | 240 | 0.832 | 1.989 |
| | | | 20 | 30 | 20 | 210 | | |
| | | | 100 | 180 | 100 | 210 | | |
| Modified Jaya | 1 | 1 | 100 | 180 | 100 | 210 | **7.06** | **4.57** |
| | 2 | 2 | 100 | 180 | 20 | 210 | **2.90** | **0.32** |
| | | | 20 | 30 | 100 | 210 | | |
| | 3 | 3 | 20 | 30 | 300 | 270 | **0.707** | **0.443** |
| | | | 100 | 180 | 50 | 330 | | |
| | | | 10 | 60 | 300 | 120 | | |

$N = 400$ rpm are chosen. The balance masses and corresponding angular positions on two planes are chosen from set $D_1 = 7$, $m_{ij} \in [0; 10; 20; 50; 100; 200; 300]$ values in grams, and set $D_2 = 12$ values in degree, $\alpha_{ij} \in [30; 60; 90; 120; 150; 180; 210; 240; 270; 300; 330; 360]$, respectively.

The optimization problem formulated in Eqs. (19) and (20) is applied in three different unbalancing cases of rigid threshing rotor: static, couple, and dynamic unbalanced, respectively. The different inertia and eccentricities components are considered in each case. The numbers of masses $M_1$ and $M_2$ are placed on balance plane 1 and 2, respectively, for each case. However, the total numbers of design variables depend on the number of masses. The lower and higher values of discrete variables represent the lower and upper bound of design variables. This algorithm is coded in MatLab. The effectiveness of the algorithm is compared with that of GA for the same problems by placing the number of masses as $M_1 = M_2 = 1$, $M_1 = M_2 = 2$, and $M_1 = M_2 = 3$ on two balance planes for each case. In the case of GA, the population size in the number of masses is taken as 50, 150, and 400, respectively, while the number of iterations is taken as 150, 300, and 3000 in all cases. Twenty independent runs of GA have been carried out to find out the optimum value of the objective function. The function evaluations for three cases are 7500, 45,000, and $12 \times 10^5$, respectively. However, the modified Jaya algorithm takes population size for the number of masses as 10, 100, and 300, while 100, 200, and 2500 number of generations are considered in all cases, respectively. Twenty independent runs of the Jaya algorithm have been carried out to find out the optimum value of the objective function. The function evaluations for the number of masses are 1000, $2 \times 10^4$, and $7.5 \times 10^5$, respectively. The function evaluations of the modified Jaya algorithm for the number of masses are compared with those of GA. Moreover, the modified Jaya algorithm requires 87%, 56%, and 38% less the function evaluations for the number of masses than those needed by GA, as shown in Table 6.

The computational efficiencies of GA and the modified Jaya algorithm for the number of masses in three cases are shown in Figures 13–15, respectively. The optimal solutions for three cases obtained using a modified Jaya algorithm validated with those of the GA algorithm are shown in Tables 7–9, respectively. It is observed that the optimum solutions obtained by the proposed algorithm are better or in good agreement to those of GA. Moreover, the proposed algorithm takes fewer function evaluations to find out the best values of objective functions. Hence, the computation efficiency of the modified Jaya algorithm is better than that of GA.

# 5 Conclusion

This paper proposed a modified Jaya algorithm for the mixed-variable optimization problems. The original Jaya algorithm has been developed for continuous optimization problems. Therefore, the Jaya algorithm is further extended for solving mixed-variable optimization problems. In the proposed algorithm, continuous variables remain in the continuous domain while continuous domains of discrete and integer variables are converted into discrete and integer domains applying bound constraint of the middle point of corresponding two consecutive values of discrete and integer variables. Furthermore, the efficiency of the modified Jaya algorithm is demonstrated using five design problems taken from the literature. This algorithm is also applied to two-plane balancing of an unbalanced rigid threshing rotor. Moreover, the optimum results obtained from the proposed algorithm are compared with the results of well-known optimization algorithms. The results show that it takes fewer function evaluations without violation of the design constraints and gives better and nearly close results compared to other optimization algorithms. It also provides better balancing solutions for the unbalanced rigid threshing rotor for all cases with less computational effort. Other mixed, continuous, and discrete variable optimization problems can also be effectively solved using this algorithm. Hence, a modified Jaya algorithm may be an essential tool for a wide range of mixed-variable problems.

# Bibliography

[1] J. S. Arora, Methods for discrete variable structural optimization, *Adv. Technol. Struct. Eng.* (2000), 1–8. DOI: 10.1061/40492(2000)23.

[2] J. Arora, *Introduction to Optimum Design*, 3rd ed., Elsevier, Amsterdam, 2004.

[3] J. S. Arora, M. W. Huang and C. C. Hsieh, Methods for optimization of nonlinear problems with discrete variables: a review, *Struct. Optim.* **8** (1994), 69–85.

[4] R. Beck, J. N. Katz, A. D. Martin and K. M. Quinn, A review of discrete optimization algorithms, *Polit. Methodol.* **7** (1995), 6–10.
[5] B. Borchers and J. E. Mitchell, An improved branch and bound algorithm for mixed integer nonlinear programs, *Comput. Oper. Res.* **21** (1994), 359–367.
[6] C. V. Camp and B. J. Bichon, Design of space trusses using ant colony optimization, *J. Struct. Eng.* **130** (2004), 741–751.
[7] Y. J. Cao, An evolutionary programming approach to mixed-variable optimization problems, *Appl. Math. Model.* **24** (2000), 931–942.
[8] S. K. Chaudhary and H. Saha, *Dynamics and Balancing of Multibody Systems*, Springer Verlag, Germany, 2009.
[9] K. Chaudhary and H. Chaudhary, Optimal dynamic design of planar mechanisms using teaching – learning-based optimization algorithm, *Mech. Eng. Sci.* **230** (2016), 3442–3456.
[10] C. A. Coello Coello and E. M. Montes, Use of dominance-based tournament selection to handle constraints in genetic algorithms, *Intell. Eng. Syst. Through Artif. Neural Netw.* **11** (2001), 177–182.
[11] K. Deb, GeneAS: a robust optimal design technique for mechanical component design, in: *Evolutionary Algorithms in Engineering Applications*, pp. 497–514, Springer, Berlin, 1997.
[12] T. Dede, Application of teaching-learning-based-optimization algorithm for the discrete optimization of truss structures, *KSCE J. Civ. Eng.* **18** (2014), 1759–1767.
[13] S. O. Degertekin, L. Lamberti and I. B. Ugur, Sizing, layout and topology design optimization of truss structures using the Jaya algorithm, *Appl. Soft Comput. J.* **70** (2017), 903–928.
[14] J. F. Fu, R. G. Fenton and W. L. Cleghorn, A mixed integer-discrete-continuous programming method and its application to engineering design optimization, *Eng. Optim.* **17** (1991), 263–280.
[15] A. M. Geoffrion, Lagrangean relaxation, *Math. Program. Stud.* **2** (1974), 82–114.
[16] C.-X. Guo, J.-S. Hu, B. Ye and Y.-J. Cao, Swarm intelligence for mixed-variable design optimization, *J. Zhejiang Univ. Sci.* **5** (2004), 851–60.
[17] S. He, E. Prempain and Q. Wu, An improved particle swarm optimizer for mechanical design optimization problems, *Eng. Optim.* **36** (2004), 585–605.
[18] V. Jeet and E. Kutanoglu, Lagrangian relaxation guided problem space search heuristics for generalized assignment problems, *Eur. J. Oper. Res.* **182** (2007), 1039–1056.
[19] S. Kitayama, M. Arakawa and K. Yamazaki, Penalty function approach for the mixed discrete nonlinear problems by particle swarm optimization, *Struct. Multidiscip. Optim.* **32** (2006), 191–202.
[20] M. Kripka, Discrete optimization of trusses by simulated annealing, *J. Braz. Soc. Mech. Sci. Eng.* **26** (2004), 170–173.
[21] J. Lampinen and I. Zelinka, Mixed integer discrete continuous optimization by differential evolution: part 2. A practical example, in: *Proceedings of MENDEL'99, 5th International Mendel Conference on Soft Computing*, pp. 77–81, Brno, Czech Republic, 1999.
[22] S. Leyffer, Integrating SQP and branch-and-bound for mixed integer nonlinear programming, *Comput. Optim. Appl.* **18** (2001), 295–309.
[23] L. J. Li, Z. B. Huang and F. Liu, A heuristic particle swarm optimization method for truss structures with discrete variables, *Comput. Struct.* **87** (2009), 435–443.
[24] H. T. Loh and P. Y. Papalambros, A sequential linearization approach for solving mixed-discrete nonlinear design optimization problems, *J. Mech. Des.* **113** (1991), 325.
[25] R. T. Marler and J. S. Arora, Survey of multi-objective optimization methods for engineering, *Struct. Multidiscip. Optim.* **26** (2004), 369–395.
[26] R. T. Marler and J. S. Arora, The weighted sum method for multi-objective optimization: new insights, *Struct. Multidiscip. Optim.* **41** (2010), 853–862.
[27] T. Messager and M. Pyrz, Discrete optimization of rigid rotor balancing, *J. Mech. Sci. Technol.* **27** (2013), 2231–2236.
[28] D. Mundo, G. Gatti and D. B. Dooner, Optimized five-bar linkages with non-circular gears for exact path generation, *Mech. Mach. Theory* **44** (2009), 751–760.
[29] S. Nema, J. Goulermas, G. Sparrow and P. Cook, A hybrid particle swarm branch-and-bound (HPB) optimizer for mixed discrete nonlinear programming, *IEEE Trans. Syst. Man, Cybern. Part A Syst. Hum.* **38** (2008), 1411–1424.
[30] S. Rajeev and S. C. Krishnamoorthy, Discrete optimization of structures using genetic algorithms, *J. Struct. Eng.* **118** (1992), 1233–1250.
[31] R. V. Rao, *Teaching Learning Based Optimization Algorithm and its Engineering Applications*, Springer, Cham, Switzerland, 2015.
[32] R. V. Rao, Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems, *Int. J. Ind. Eng. Comput.* **7** (2016), 19–34.
[33] R. V. R. Rao, *Jaya: an Advanced Optimization Algorithm and its Engineering Applications*, Springer International Publishing AG, part of Springer Nature, Switzerland, 2019.
[34] S. S. Rao and Y. Xiong, A hybrid genetic algorithm for mixed-discrete design optimization, *J. Mech. Des.* **127** (2005), 1100.
[35] R. V. Rao, V. J. Savsani and D. P. Vakharia, Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems, *CAD Comput. Aided Des.* **43** (2011), 303–315.

[36] R. V. Rao, K. C. More, J. Taler and P. Ocłoń, Optimal design of Stirling heat engine using an advanced optimization algorithm, *Sadhana – Acad. Proc. Eng. Sci.* **41** (2016), 1321–1331.
[37] R. V. Rao, K. C. More, L. S. Coelho and V. C. Mariani, Multi-objective optimization of the Stirling heat engine through self-adaptive Jaya algorithm, *J. Renew. Sustain. Energy* **9** (2017), 033703.
[38] R. V. Rao, D. P. Rai and J. Balic, A multi-objective algorithm for optimization of modern machining processes, *Eng. Appl. Artif. Intell.* **61** (2017), 103–125.
[39] R. V. Rao, D. P. Rai and J. Balic, Multi-objective optimization of abrasive waterjet machining process using Jaya algorithm and PROMETHEE method, *J. Intell. Manuf.* (2017), 1–27. https://doi.org/10.1007/s10845-017-1373-8.
[40] E. Sandgren, Nonlinear integer and discrete programming in mechanical design optimization, *J. Mech. Des.* **112** (1990), 223.
[41] D. K. Shin, Z. Gurdal and O. H. Griffin, A penalty approach for nonlinear optimization with discrete design variables, *Eng. Optim.* **16** (1990), 29–42.
[42] R. Singh, H. Chaudhary and A. K. Singh, Defect-free optimal synthesis of crank-rocker linkage using nature-inspired optimization algorithms, *Mech. Mach. Theory J.* **116** (2017), 105–122.
[43] M. Sonmez, Discrete optimum design of truss structures using artificial bee colony algorithm, *Struct. Multidiscip. Optim.* **43** (2011), 85–97.
[44] C. Sun, J. Zeng and J. S. Pan, A modified particle swarm optimization with feasibility-based rules for mixed-variable optimization problems, *Int. J. Innov. Comput. Inf. Control* **7** (2011), 3081–3096.
[45] A. C. Varshney, *Data Book for Agricultural Machinery Design*, Central Institute of Agricultural Engineering, Bhopal, India, 2004.
[46] S.-J. Wu and P.-T. Chow, Genetic algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization, *Eng. Optim.* **24** (1995), 137–159.
[47] C. Zhang and H. P. (Ben) Wang, Mixed-discrete nonlinear optimization with simulated annealing, *Eng. Optim.* **21** (1993), 277–291.