OXFORD CDE

*Journal of Computational Design and Engineering*, 2022, 9(2), 706–730

https://doi.org/10.1093/jcde/qwac013
Journal homepage: www.jcde.org



RESEARCH ARTICLE

# A hybrid genetic–firefly algorithm for engineering design problems

M. A. El-Shorbagy [1,2,*] and Adel M. El-Refaey [3]

[1]Department of Mathematics, College of Science and Humanities in Al-Kharj, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia; [2]Department of Basic Engineering Science, Faculty of Engineering, Menoufia University, Shebin El-Kom 32511, Egypt and [3]Basic and Applied Science Department, College of Engineering and Technology, Arab Academy for Science, Technology and Maritime Transport, Smart Village Campus 12577, Egypt

*Corresponding author. E-mail: mohammed_shorbagy@yahoo.com https://orcid.org/0000-0002-8115-0638

## Abstract

Firefly algorithm (FA) is a new random swarm search optimization algorithm that is modeled after movement and the mutual attraction of flashing fireflies. The number of fitness comparisons and attractions in the FA varies depending on the attraction model. A large number of attractions can induce search oscillations, while a small number of attractions can cause early convergence and a large number of fitness comparisons that can add to the computational time complexity. This study aims to offer H-GA–FA, a hybrid algorithm that combines two metaheuristic algorithms, the genetic algorithm (GA) and the FA, to overcome the flaws of the FA and combine the benefits of both algorithms to solve engineering design problems (EDPs). In this hybrid system, which blends the concepts of GA and FA, individuals are formed in the new generation not only by GA processes but also by FA mechanisms to prevent falling into local optima, introduce sufficient diversity of the solutions, and make equilibrium between exploration/exploitation trends. On the other hand, to deal with the violation of constraints, a chaotic process was utilized to keep the solutions feasible. The proposed hybrid algorithm H-GA–FA is tested by well-known test problems that contain a set of 17 unconstrained multimodal test functions and 7 constrained benchmark problems, where the results have confirmed the superiority of H-GA–FA overoptimization search methods. Finally, the performance of the H-GA–FA is also investigated on many EDPs. Computational results show that the H-GA–FA algorithm is competitive and better than other optimization algorithms that solve EDPs.

*Keywords:* hybrid algorithms; genetic algorithm; firefly algorithm; engineering design problems

## 1. Introduction

The engineering design problems (EDPs) are extremely significant from both the manufacturing and scientific perspectives, where it is a very important and challenging area especially in the field of engineering for getting designs that have efficient form and are more accurate. Generally, EDPs are treated as nonlinear constrained optimization problems (NCOPs). Oftentimes, the complex nature of constraints leads to that solving NCOPs is very difficult and the problem feasible region may be a thin subset of the search domain. There is no exact method of locating the global optimal to the general NCOP (Michael, 2008).

Traditionally, NCOPs are solved by some efficient methods such as recursive quadratic programming, projection method, generalized reduced gradient method, penalty method, and a multiplier method (Rao, 2009). To use these methods, the objective function must be differentiable and the feasible set is

**Received:** 13 August 2021; **Revised:** 14 January 2022; **Accepted:** 16 January 2022

convex. In addition, traditional methods are not efficient since they may only compute local optima. So, it is very hard to apply these methods to problems that its feasible region is not convex or the objective function is not differentiable. Moreover, we find that most of EDPs are discontinuities problems and their search space is noise, which makes traditional methods unable to find the global solutions for this kind of problem. Generally, we can say that conventional optimization methods depend on the existence of the derivative, local in scope, and they are not sufficiently robust in discontinuous, noisy, and multimodal search spaces (Michalewicz, 1994).

There are optimization methods that have appeared, which are completely dissimilar from the conventional methods and labeled as nontraditional or metaheuristic algorithms. They are considered popular methods to solve NCOPs. These methods simulated the behavior and characteristics of molecular, biological, swarms, and neurobiological systems. Furthermore, they can conquer the restrictions and difficulties of conventional techniques. In addition, these metaheuristic algorithms are simple in concepts, have a low probability to fall into local optima, and require simple information about the optimization problem without requiring that the objective function or the constraints are derivable or continuous (Onwubolu & Babu, 2004).

Researchers have relied on metaheuristic algorithms based on simulations for solving EDPs. The main feature of these algorithms is that they merge randomness and rules to simulate natural phenomena. Among the existing metaheuristic algorithms, famous algorithms are simulated annealing (SA; Alrefaei & Diabat, 2009), differential evolution (DE; Parouha & Das, 2016), genetic algorithms (GAs; El-Desoky *et al.*, 2016; El-Shorbagy *et al.*, 2017, 2019a, b), artificial immune system (Saurabh & Verma, 2016), neural network-based methods (Zhou, 2005), particle swarm optimization (PSO; El-Shorbagy, 2010; El-Shorbagy & Mousa, 2017; El-Shorbagy & Hassanien, 2018), ant colony optimization (ACO; Dorigo & Stützle, 2004), artificial bee colony (ABC; Karaboga, 2005), bacterial foraging algorithm (Passino, 2002; Zhao & Wang, 2016), evolution strategies (ESs; Beyer & Schwefel, 2002), glowworm swarm optimization algorithm (Marinaki & Marinakis, 2016), firefly algorithm (FA; Verma & Mukherjee, 2016), monkey algorithm (MA; Zhou *et al.*, 2016), krill herd algorithm (KHA; Bolaji *et al.*, 2016), whale optimization algorithm (WOA; Mirjalili & Lewis, 2016), sine cosine algorithm (SCA; Abo-elnaga & El-Shorbagy, 2020; El-Shorbagy *et al.*, 2020), grasshopper optimization algorithm (GOA; Saremi *et al.*, 2017), monarch butterfly optimization (Ghetas *et al.*, 2015), slime mould algorithm (Li *et al.*, 2020), moth search algorithm (Wang, 2018), hunger games search (Yang *et al.*, 2021), Runge–Kutta method (Ahmadianfar *et al.*, 2021), Harris hawks optimization (Alabool *et al.*, 2021), salp swarm algorithm (Mirjalili *et al.*, 2017), grey wolf optimizer (Gupta & Deep, 2020), tunicate swarm algorithm (Kaur *et al.*, 2020), seagull optimization algorithm (SOA; Dhiman & Kumar, 2019), gravitational search algorithm (GSA; Rashedi *et al.*, 2009), spherical search optimizer (Zhao *et al.*, 2020), equilibrium optimizer (El-Shorbagy & Mousa, 2021), backtracking search algorithm (Nama *et al.*, 2021), teaching-learning-based optimization (TLBO; Rao et al., 2011), brain storm optimization (Shi, 2011), bus transportation algorithm (Bodaghi & Samieefar, 2019), socio evolution and learning optimization algorithm (Kumar *et al.*, 2018), hummingbirds optimization algorithm (Zhuoran *et al.*, 2018), etc. The categorization of metaheuristic algorithms based on algorithmic behaviors is depicted in Fig. 1.

Although metaheuristic algorithms perform well compared to traditional methods, they may encounter difficulties such as being stuck in a local optimum, insufficient diversity of solutions, and an imbalance between exploitation/exploration trends in some complex cases. To overcome these weaknesses, most of the researchers have proposed hybridization strategies between metaheuristic algorithms to improve the solution quality, benefit from their advantages, and overcome any deficiencies such as hybrid ACO (Goel & Maini, 2018), hybrid GA (Nasr *et al.*, 2015; Al Malki *et al.*, 2016), hybrid PSO (El-Shorbagy *et al.*, 2011), hybrid GOA (El-Shorbagy & Ayoub, 2021), hybrid GSO (Chen *et al.*, 2017), hybrid ABC (Jadon *et al.*, 2017), hybrid BF (Turanoğlu & Akkaya, 2018), hybrid FA (Ekinci *et al.*, 2019), hybrid KHA (Abualigah *et al.*, 2017), hybrid MA (Marichelvam *et al.*, 2017), hybrid SCA (Wang *et al.*, 2018), etc. Furthermore, these hybrid strategies have various properties, such as providing robust algorithms with faster performance and handling large optimization problems. Also, they are one of the most interesting recent trends in optimization.

The FA is one of the new metaheuristic algorithms that is modeled according to the mutual attraction and movement of flashing fireflies. It is commonly utilized in a wide range of optimization issues. Depending on the attraction model in FAs, the number of fitness comparisons and attractions varies. A large number of attractions can induce search oscillations, a small number can cause early convergence, and a large number of fitness comparisons can add to the computational time complexity. So we can conclude that FA has several limits, such as (i) an imbalance between the processes of exploitation and exploration, (ii) an unstable convergence speed, and (iii) the possibility of falling into the local optimum. As a result, various hybrid algorithms combining FA and other metaheuristic algorithms have been presented in the literature. In Goel and Maini (2018), a hybrid method called hybrid of ant colony and firefly algorithms (HAFA) was created to solve one of the classic NP-hard optimization problems: vehicle routing. It combines characteristics of FA and ant colony system (ACS) techniques. FA was employed to look for the unknown solution space, while ACS served as the core framework for HAFA. Furthermore, the pheromone shaking technique was applied in ACS to minimize pheromone stagnation on exploited locations, allowing it to escape from local optima. The electromagnetism-like FA (EFA), a novel hybrid between the electromagnetism-like algorithm (EM) and the FA, is proposed in Le *et al.* (2019). Some of the benefits of EFA include (i) the use of modified formulas of interactive forces to increase population diversification; (ii) to avoid becoming trapped in infeasible domains, constraint violations are embedded in the charges of all electromagnetic fireflies; and (iii) to balance EFA's exploration and exploitation abilities, a mechanism known as "current-to-best" electromagnetic movement is combined with traditional interactive movements. Aydilek (2018) proposed a hybrid algorithm combining firefly and PSO (HFPSO). The advantages of both the particle swarm and the FA methods can be used in the suggested approach to avoid becoming trapped in local optima, where HFPSO examines the previous global best fitness values to appropriately decide the commencement of the local search phase. Al-Thanoon *et al.* (2019) introduce a hybrid firefly method and PSO that can efficiently leverage the strengths of both determining the best solution with outstanding classification performance and escape from the local optimum. In Lieu *et al.* (2018), the authors presented an adaptive hybrid evolutionary firefly algorithm that is a hybridization of the DE algorithm and the FA. To select an acceptable mutation strategy for an effective tradeoff between the global and local search abilities, an automatically adapted parameter is utilized. Furthermore, an elitist strategy was used in the selection process to identify the best

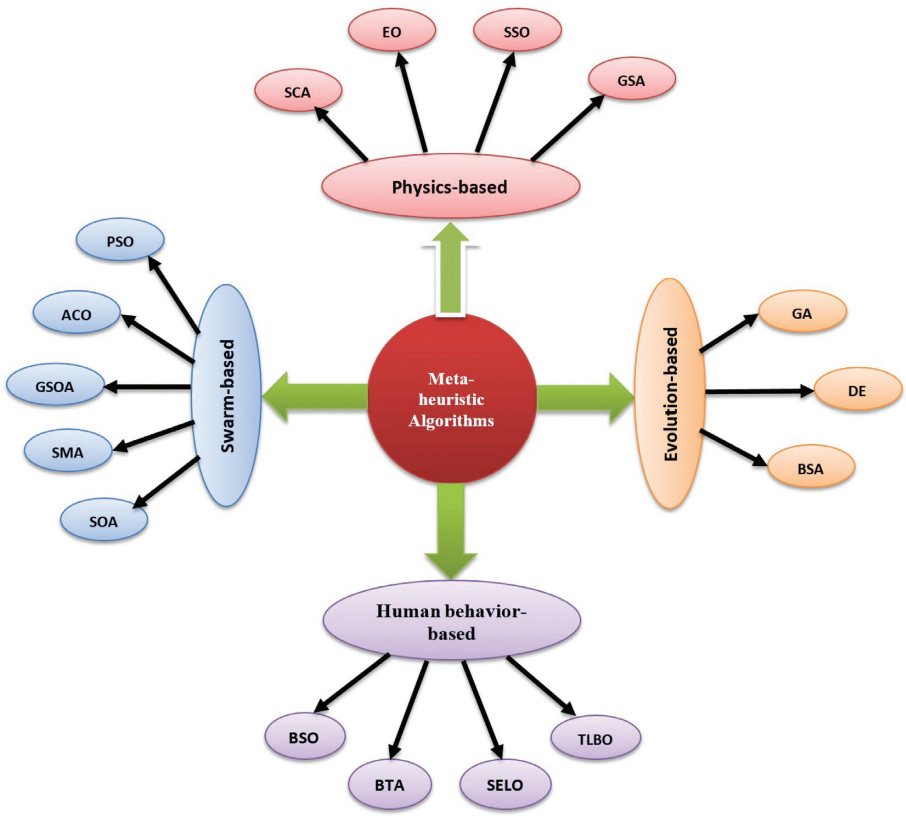


**Figure 1:** The categorization of metaheuristic algorithms based on algorithmic behaviors (Nama *et al.*, 2021).

individuals. In Cheng *et al.* (2021), a novel attraction grouping model is developed to eliminate FA difficulties, which may successfully minimize the number of attractions and fitness comparisons, where the fireflies in the group with better fitness are added with the supervision of the firefly with the best fitness. In addition, a combination mutation operator is added to FA to lessen the likelihood of stagnation and early convergence, as well as to better balance the FA's exploration and exploitation capabilities. Finally, nature-inspired computing technique, GA, is combined with the FA. In Sharma *et al.* (2021), an improved clinical decision support system query is proposed that is a combination between the FA and the controlled GA in a restricted divergence environment. In Kaushik and Arora (2015), a firefly-based GA is proposed for clustering problems, where the initial population is selected from a pool of population on the basis of the FA.

The GA is a metaheuristic algorithm that was established in 1975 (Holland, 1975) and defined in 1989 (Goldberg, 1989) as a competent global approach for addressing large optimization problems based on natural selection, evolution, and genetics. GA is highly suited to tackling optimization problems, and academics continue to pay close attention to it. In Farag *et al.* (2015), binary–real coded GA-based k-means clustering is proposed for the unit commitment problem, in El-Shorbagy and El-Refaey (2020) hybridization of the GOA with the GA is introduced for solving a system of nonlinear equations, in Mousa *et al.* (2020) steady-state sine cosine GA-based chaotic search was proposed for nonlinear programming and engineering applications, and in Ayoub *et al.* (2020) cell blood image segmentation based on the GA and k-means algorithm is presented. However, when solving complicated and massive systems, GA has some drawbacks, including being highly sluggish, and hence it is impossible to identify the global optimal solution where it necessitates an increase in the number of iterations. Hence, it is suggested that the implementation of GA and FA in a hybrid form results in superior characteristics, where GA with its operators (ranking, selection, crossover, and mutation) has good exploitation ability. However, FA constantly updates each solutions' position based on movement and the mutual attraction of flashing fireflies. This indicates that FA is an excellent explorer. As a result of this motivation, this paper makes the following contributions:

1. Introducing a newfangled hybrid algorithm H-GA–FA for solving engineering design issues (EDPs).
2. Presenting sufficient diversity of the solutions, and preventing H-GA–FA to fall into local optima by forming the new generation in H-GA–FA not only by GA processes but also by FA mechanisms.
3. Making a balance between exploration/exploitation trends in H-GA–FA by integrating GA's exploitation and FA's exploration capabilities.
4. Adopting a chaotic repair procedure to tackle the constraints and unfeasible solution.
5. Testing H-GA–FA by well-known test problems as well as several EDPs.
6. Using Wilcoxon and Friedman tests to assess the significance of the H-GA–FA findings.

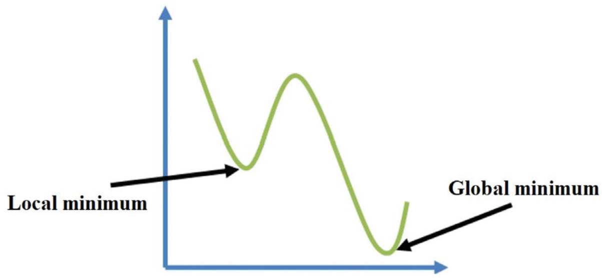


**Figure 2:** Global minimum and local minimum.

7. Showing that H-GA–FA, by computational results and statistical analysis, is competitive and better than other optimization algorithms.

The remainder of this paper is arranged as follows: In Section 2, the preliminaries about the problem are presented. In Section 3, the proposed methodology is described in detail. In Section 4, computational experiments with discussions are done. In Section 5, a brief conclusion is offered along with its future scope.

## 2. Preliminaries

The formulation of the EDPs is discussed in this section. As previously stated, EDPs are commonly expressed as NCOPs. NCOP's mathematical model is as follows:

Minimize or (maximize) $f(X)$

Subject to:

$$C_i(X) \le 0, \ i = 1, \ldots, p, \tag{1}$$

$$H_e(X) = 0, \ e = 1, \ldots, m, \tag{2}$$

$$L_d \le x_d \le U_d, \forall d = 1, \ldots, n, \tag{3}$$

where $f, C_1, \ldots, C_P, H_1, \ldots, H_m$ are functions defined on $\mathbb{R}_n$, $X = (x_1, \ldots, x_n)$ is a vector of $n$ component subset of $\mathbb{R}_n$. The above problem must be solved for the values of the variables $x_1, \ldots, x_n$ that satisfy the restrictions and minimize or (maximize) the function $f$. The function $f$ is the objective function or the criterion function. If there are no constraints, the problem is called an unconstrained problem. Each of the constraints $C_i(X) \le 0$ for $i = 1, \ldots, p$ is called an inequality constraint, and each of the constraints $H_e(X) = 0$ for $e = 1, \ldots, m$ is called an equality constraint. $L_d$ represents the lower bounds and $U_d$ the upper bounds for the decision variables $x_d \forall d = 1, \ldots, n$.

When solving NCOP, we seek a global solution rather than relying on a local one. The local solution of an optimization problem is the optimal solution (either maximal or minimal) within a nearby set of candidate solutions. However, the global optimum is the best solution between every possible solution, not just those in a particular zone of search region (Michalewicz, 1994). Definition 1 introduces the difference between a local solution and a global solution. Figure 2 illustrates this definition.

**Definition 1:**

Let $X = (x_1, x_2, \ldots, x_n)$ be a feasible solution to a minimization problem with objective function $f(X)$ (Michalewicz, 1994). Then, $X$ is

1. A global minimum if $f(X) \le f(Y)$ for every feasible point $Y = (y_1, y_2, \ldots, y_n)$.

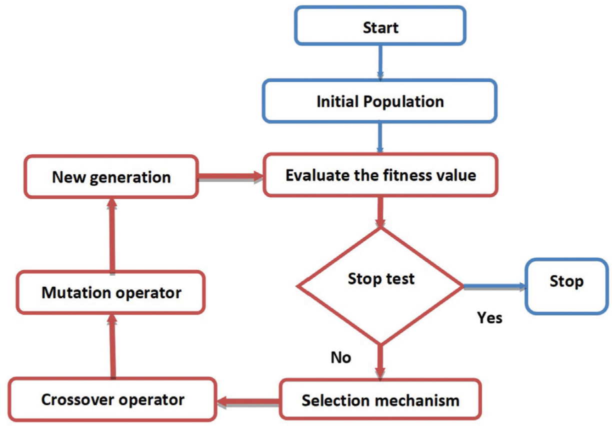


**Figure 3:** The GA's flowchart.

2. A local minimum if $f(X) \le f(Y)$ for all feasible points $Y = (y_1, y_2, \ldots, y_n)$ sufficiently close to $X$.

## 3. Methodology

In this section, we provide a brief overview of both the GA and the FA. The proposed algorithm is then thoroughly described.

### 3.1 Genetic algorithm

GA was first proposed in Holland (1975) as an optimization approach for locating the global or near-global optimal solution. It starts with a group of chromosomes (solutions). Then, the operators of genetic selection, crossover, and mutation are applied one by one to get a new set of solutions. The quality of the newly generated chromosomes is predictable to be better than the initial generation. These steps are repeated until the criterion of termination is met. Algorithmically, GA main steps are described as follows:

Step I: Randomly create an initial group of solutions so that they are appropriate for the problem.

Step II: The fitness value of every solution, in the group, is evaluated.

Step III: Generate a new group of solutions by repeating and applying the following steps:

a) Two parents are selected from the group of solutions according to the selection mechanism.
b) Crossover the parents to create new offspring.
c) New offsprings are mutated.

Step IV: If the satisfying criteria are met, stop; otherwise, go to step II.

The flowchart that represents GA is shown in Fig. 3.

### 3.2 Firefly algorithm

The FA is a metaheuristic optimization technique that was proposed in Yang (2008). FA simulates the social performance of fireflies in nature for solving optimization problems. The major advantage of FA is that all local modes, as well as global modes, will be visited. This is due to the decreasing of light intensity with distance, leading to that the attraction among fireflies can be global or local, depending on the absorbing coefficient. So, it has taken much interest and has been applied effectively to

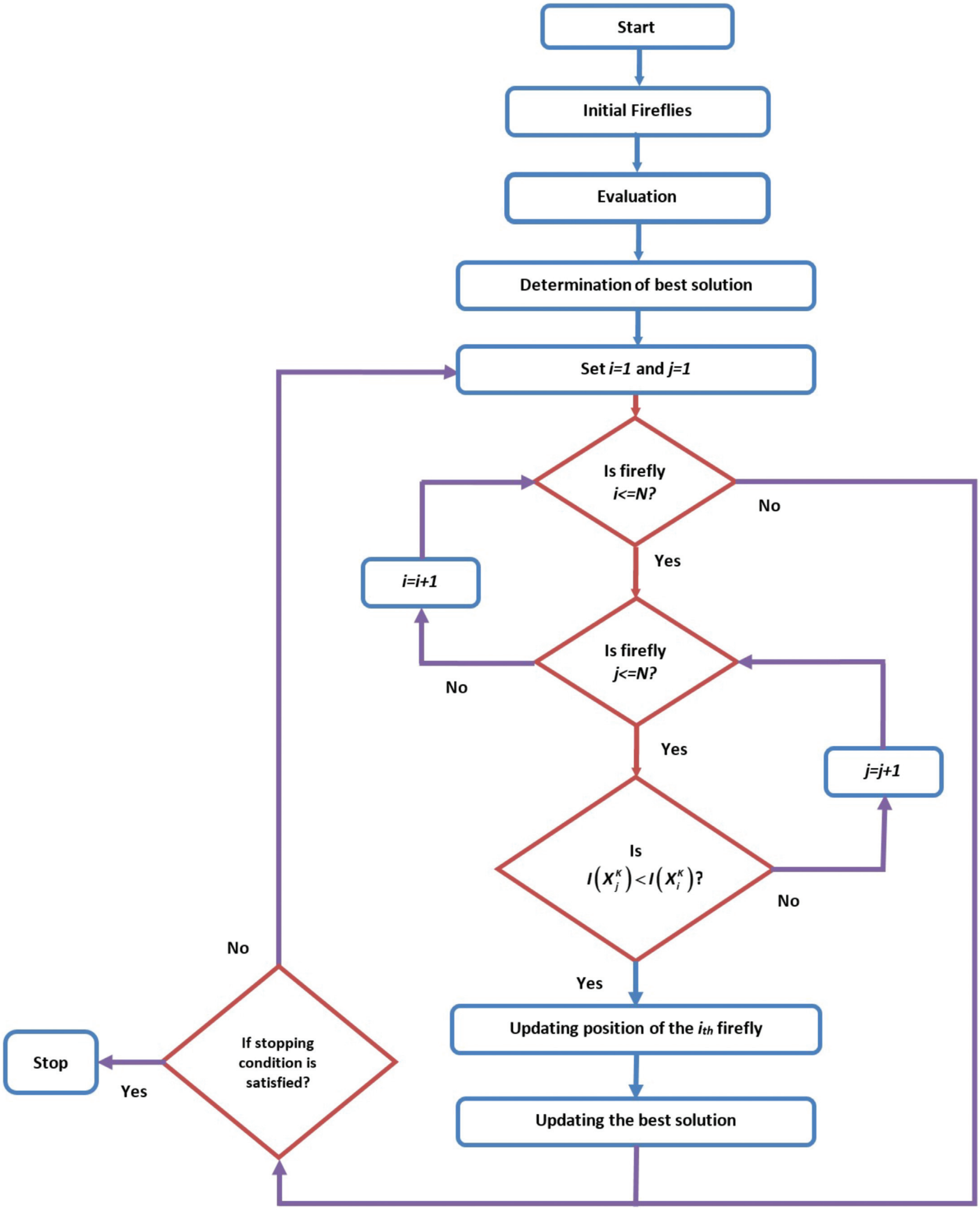


**Figure 4:** Flowchart of the FA.

solve a lot of optimization problems. The main steps of FA are described as follows:

**Step 1 Initialization:**

A population of random $N$ fireflies (solutions) is initialized ($K = 0$), where the position of the $i$-th firefly in $d$-dimensional space is denoted as $X_i$ and represented as $X_i^K = (x_{i1}, x_{i2}, \ldots, x_{id})$.

**Step 2 Evaluation:**

Evaluate the fitness value [the light intensity $I(X_i^K)\forall i = 1, \ldots, N$] of each firefly in the population or simply $I(X_i^K) = f(X_i^K)\forall i = 1, \ldots, N$.

**Step 3 Determination of best solution:**

For minimization problems, the firefly that has minimum light intensity is the best solution $X_b$.

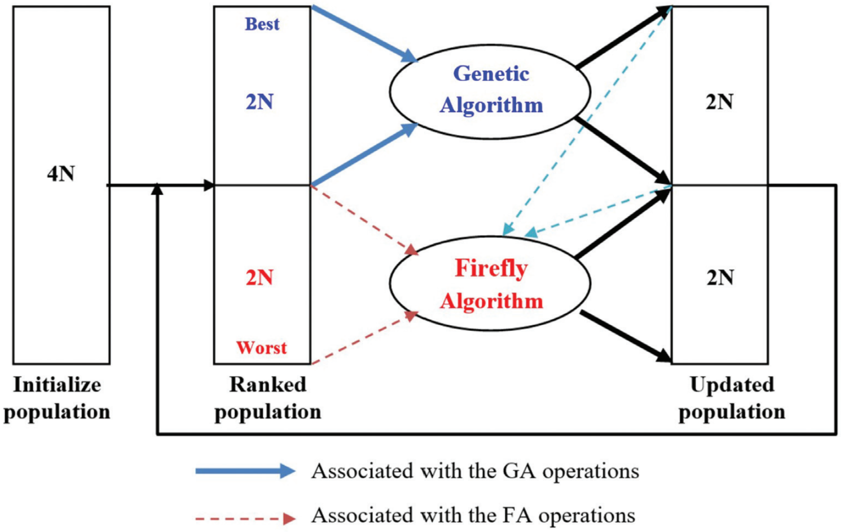


**Figure 5:** Diagram depiction of H-GA–FA.

**Step 4_ Updating positions of fireflies:**

For every firefly $i = 1, \ldots, N$ and every firefly $j = 1, \ldots, N$ Do:

If $I(X_j^K) < I(X_i^K)$, the $i$-th firefly is attracted to the firefly $j$ and updating its position $X_i^K$ according to the following equation:

$$X_i^{K+1} = X_i^K + \beta_0 e^{(-\gamma r_{ij}^2)} \left(X_j^K - X_i^K\right) + \alpha_k \varepsilon_k,$$

$$r_{ij} = \left|X_i^K - X_j^K\right| = \sqrt{\sum_{d=1}^{d} \left(x_{id}^K - x_{jd}^K\right)^2}, \tag{4}$$

where $\beta_0$ is attractiveness at $r_{ij} = 0$, $\gamma$ is the light absorption coefficient, $r_{ij}$ is the Cartesian distance between the two fireflies $i$ and $j$, $\alpha_k$ is a parameter controlling the step size, and $\varepsilon_k$ is a vector drawn from a Gaussian or other distribution.

**Step 5_ Updating the best solution $X_b$:**

If any of the new positions for the fireflies $X_i^{K+1} \forall i = 1, \ldots, N$ is better than the best solution $X_b$, i.e. $I(X_i^{K+1}) < I(X_b) \forall i = 1, \ldots, N$, then $X_b = X_i^{K+1}$.

**Step 6_ Stopping condition:**

If the stopping condition is satisfied, stop; otherwise, go to step 4.

The flowchart that represents FA is shown in Fig. 4.

## 3.3 Hybrid genetic–firefly algorithm

This section shows the idea of the proposed algorithm H-GA–FA. H-GA–FA integrates the benefits of the two metaheuristic algorithms, GA and FA. FA has strong exploration capabilities since it visits all local and global modes, and finds good solutions, while GA has high exploitation capabilities where its operators (ranking, selection, crossover, and mutation) make a comprehensive change in the shape of the solutions, which makes the proposed algorithm get out of the local optima when needed. Figure 5 describes the diagram depiction of the H-GA–FA. According to Fig. 5, H-GA–FA is initiated by an initial population. Assuming the initial population contains $4N$ individuals that are randomly generated. The $4N$ individuals are arranged by fitness. The top $2N$ individuals (the best individuals) are used by GA as chromosomes to generate $2N$ new individuals by its operators (Fig. 3). The reason for this is that the best solutions may be located in the local optima and with the help of GA operators (crossover and mutation), the best solutions can escape the local optima. However, the other $2N$ individuals (the worst individuals) with the $2N$ individuals generated by GA are used by FA as fireflies to generate $2N$ new fireflies (Fig. 4), which leads to an improved variety of solutions. In addition, FA's exploration capabilities can move the worst solutions to better areas in the search space, which leads to rapidly converging to the optimal solution. The process of enhancing the $2N$ fireflies in FA includes the determination of the best solution. The generated $4N$ individuals are arranged to prepare to repeat the previous steps. H-GA–FA is ended either when the maximum number of generations *Kmax* is completed or when the individuals' convergence occurs. Convergence occurs when all individuals' positions are identical. In this case, updating the position of any individual will have no further effect.

By using a simple optimization problem, we show how H-GA–FA works as follows:

Minimizing $F(x, y) = x^2 + y^2$ subject to $-1 \le x, \ y \le 1$

Then, the procedures described above are repeated until the maximum number of iterations is reached.

### 3.3.1 *Repairing infeasible solution by a chaotic process*

A chaotic process is used to violate the constraint, where it repairs any infeasible solution in the population at each generation. To begin, an initial reference point $R$ is specified as any feasible solution in the search area. Then, if the new position $X_i^{K+1}$ obtained in any generation is infeasible, it is chaotically repaired according to the following equation:

$$X_i^{K+1} = \phi_t \times X_i^{K+1} + (1 - \phi_t) \times R. \tag{5}$$

Or according to the following equation:

$$X_i^{K+1} = \phi_t \times R + (1 - \phi_t) \times X_i^{K+1}, \tag{6}$$

where $\phi$ is a chaotic number that is created by the following chaotic logistic map (Mousa *et al.*, 2021):

$$\phi_{t+1} = c\phi_t (1 - \phi_t)\,;\; c = 4,\; x_0 \in (0, 1) \text{ and } \phi_0 \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}, \tag{7}$$

where $t$ is the age of the infeasible position.

In other words, the feasibility for each newly generated solution $X_i^{K+1}$ is checked. If it is infeasible, the chaotic $\phi$ parameter

**Step 1:** Initialize population: H-GA–FA is initiated by an initial population. We assume that the initial population contains four individuals that are randomly generated.

| Initialization | 4N, N = 1 | | | |
|---|---|---|---|---|
| $(x, y)$ | [−0.4096, 0.3408] | [−0.8488, 0.5966] | [−0.8138, 0.1217] | [−0.3467, −0.4951] |
| $F(x, y)$ | 0.2839 | 1.0763 | 0.6770 | 0.3653 |

**Step 2:** Ranke the population: The four individuals are arranged by fitness.

| Ranke the population | Best 2N (GA) | | Worst 2N (FA) | |
|---|---|---|---|---|
| $(x, y)$ | [−0.4096, 0.3408] | [−0.3467, −0.4951] | [−0.8138, 0.1217] | [−0.8488, 0.5966] |
| $F(x, y)$ | 0.2839 | 0.3653 | 0.6770 | 1.0763 |

**Step 3:** GA process: The best two individuals are used by GA as chromosomes to generate two new individuals by its operators.

| GA process | Parent | | New offspring | |
|---|---|---|---|---|
| $(x, y)$ | [−0.4096, 0.3408] | [−0.3467, −0.4951] | [−0.4290, 0.1720] | [−0.0644, 0.4932] |
| $F(x, y)$ | 0.2839 | 0.3653 | 0.2136 | 0.2474 |

**Step 4:** FA process: The other two worst individuals with the two individuals generated by GA are used by FA as fireflies to generate two new fireflies.

| FA process | Old fireflies | | New offspring of GA | |
|---|---|---|---|---|
| $(x, y)$ | [−0.8138, 0.1217] | [−0.8488, 0.5966] | [−0.4290, 0.1720] | [−0.0644, 0.4932] |
| $F(x, y)$ | 0.6770 | 1.0763 | 0.2136 | 0.2474 |
| | New fireflies | | | |
| $(x, y)$ | [−0.4096, 0.3408] | [−0.3283, 0.0480] | | |
| $F(x, y)$ | 0.2839 | 0.1101 | | |
| New generated population | 4N, N = 1 | | | |
| $(x, y)$ | [−0.4290, 0.1720] | [−0.0644, 0.4932] | [−0.4096, 0.3408] | [−0.3283, 0.0480] |
| $F(x, y)$ | 0.2136 | 0.2474 | 0.2839 | 0.1101 |

is implemented as in equations (5)–(7), where $t$ is increased with the number of failed trials to keep the feasibility of the solution.

#### 3.3.2 *Computational complexity of the H-GA–FA algorithm*

When evaluating the process time of any metaheuristic optimization method, it is critical to consider the computational complexity, which is related to the algorithm's structure and implementation. It should be noted that the computational cost of the suggested H-GA–FA is primarily determined by three factors: the process of the initialization, fitness function evaluation, and updating the solutions. The complexity of the initialization process is $O(4N)$, where $4N$ shows the population size. The complexity of updating solutions by GA is $O(Kmax \times 2N \times n)$, where $Kmax$ indicates iterations and $n$ is the number of parameters in the problem (dimension). The complexity of updating solutions by FA is $O(Kmax \times 4N \times n)$. Therefore, the computational complexity of updating solutions is $O(Kmax \times 2N \times n) + O(Kmax \times 4N \times n)$. Hence, the computational complexity of the proposed H-GA–FA is $O(4N) + O(Kmax \times 2N \times n) + O(Kmax \times 4N \times n)$. In the next section, various benchmark test functions and real optimization problems are used to validate and confirm the performance of the proposed H-GA–FA in addressing optimization problems.

## 4. Computational Experiments

In this section, a suite of test problems, which have different characteristics and are widely used in the literature, are used to benchmark the performance of the proposed algorithm H-GA–FA and demonstrate its robustness and efficiency. These test problems are divided into three groups: 17 unconstrained multimodal test functions (available in Appendix A), 7 constrained benchmark problems (available in Appendix B), and many EDPs. The results were compared with popular algorithms that solve this type of problem.

The maximization problem is solved as a minimization problem by transforming $-f(X)$ to $-f(X)$. The equality constraints were handled by a reduction strategy (Rao, 2009). To show how this strategy works, we provide an illustrative example. Consider

The generated four individuals are arranged to prepare to repeat the previous steps as follows:

| Ranke the population | Best 2N (GA) | | Worst 2N (FA) | |
|---|---|---|---|---|
| (x, y) | [−0.3283, 0.0480] | [−0.4290, 0.1720] | [−0.0644, 0.4932] | [−0.4096, 0.3408] |
| F (x, y) | 0.1101 | 0.2136 | 0.2474 | 0.2839 |
| GA process | Parent | | New offspring | |
| (x, y) | [−0.3283, 0.0480] | [−0.4290, 0.1720] | [−0.4281, 0.1229] | [−0.3874, 0.0568] |
| F (x, y) | 0.1101 | 0.2136 | 0.1983 | 0.1533 |
| FA process | Old fireflies | | New offspring of GA | |
| (x, y) | [−0.0644, 0.4932] | [−0.4096, 0.3408] | [−0.4281, 0.1229] | [−0.3874, 0.0568] |
| F (x, y) | 0.2474 | 0.2839 | 0.1983 | 0.1533 |
| | New fireflies | | | |
| (x, y) | [−0.1866, −0.1369] | [−0.2567, −0.0457] | | |
| F (x, y) | 0.0536 | 0.0680 | | |
| New generated population | 4N, N = 1 | | | |
| (x, y) | [−0.4281, 0.1229] | [−0.3874, 0.0568] | [−0.1866, −0.1369] | [−0.2567, −0.0457] |
| F (x, y) | 0.1983 | 0.1533 | 0.0536 | 0.0680 |
| Ranke the population | Best 2N (GA) | | Worst 2N (FA) | |
| (x, y) | [−0.1866, −0.1369] | [−0.2567, −0.0457] | [−0.4281, 0.1229] | [−0.3874, 0.0568] |
| F (x, y) | 0.0536 | 0.0680 | 0.1983 | 0.1533 |
| GA process | Parent | | New offspring | |
| (x, y) | [−0.1866, −0.1369] | [−0.2567, −0.0457] | [−0.1146, 0.0050] | [−0.1556, −0.0380] |
| F (x, y) | 0.0536 | 0.0680 | 0.0132 | 0.0256 |
| FA process | Old fireflies | | New offspring of GA | |
| (x, y) | [−0.4281, 0.1229] | [−0.3874, 0.0568] | [−0.1146, 0.0050] | [−0.1556, −0.0380] |
| F (x, y) | 0.1983 | 0.1533 | 0.0132 | 0.0256 |
| | New fireflies | | | |
| (x, y) | [−0.0315, 0.0315] | [−0.0505, 0.0146] | | |
| F (x, y) | 0.0020 | 0.0028 | | |
| New generated population | 4N, N = 1 | | | |
| (x, y) | [−0.1146, 0.0050] | [−0.1556, −0.0380] | [−0.0315, 0.0315] | [−0.0505, 0.0146] |
| F (x, y) | 0.0132 | 0.0256 | 0.0020 | 0.0028 |

the following constrained problem:

$$\begin{aligned} &\text{Min } f(X) = x_1^2 + x_2^2 \\ &\text{Subject to: } \quad x_1 - 3 = 0 \\ &\qquad\qquad - x_2 + 2 \le 0 \\ &\qquad\qquad - 10 \le x_i \le 10, \quad i = 1, 2. \end{aligned} \tag{8}$$

From equation (8), we can obtain a relationship $x_1 = 3$ from the equality constraint. The constrained problem presented in (8) can be transformed into

$$\begin{aligned} &\text{Min } f = 9 + x_2^2 \\ &\text{Subject to: } \quad - x_2 + 2 \le 0 \\ &\qquad\qquad - 10 \le x_2 \le 10. \end{aligned} \tag{9}$$

In comparison, the solution space of constrained problem (8) is 2D, and an equality constraint has to be considered during the solution search process. However, the solution space of constrained problem (9), processed by reduction strategy, becomes one dimensional with only one bound inequality constraint. Noticeably, the complexity of the original constrained problem is reduced.

Simulations of the proposed algorithm H-GA–FA and numerical solutions are coded in the MATLAB programming software and run on an Intel(R) Core (TM) i3 CPU M430 @ 2.4 GHz processor, installed memory (RAM): 3.00 GB. The parameters used in the execution of H-GA–FA are listed in Table 1.

## 4.1 Results for unconstrained multimodal test functions

The unconstrained multimodal test functions have many optimum solutions, where one of them is called global optimum and the others are called local optimum. Any algorithm should avoid all local optima to determine and access the global optimum. These test functions are more difficult than unimodal test functions, Also, they are more difficult than unimodal test functions, where it is not easy to get the globally optimum solution.

The unconstrained multimodal test functions were solved by five algorithms: continuous GA (CGA; Chelouah & Siarry, 2000), continuous hybrid algorithm (CHA; Chelouah & Siarry, 2003), hybrid GA and PSO (H-GA–PSO; Kao & Zahara, 2008), integrating PSO–GA (Abd-El-Wahed *et al.*, 2011), and the proposed algorithm H-GA–FA.

Table 2 shows the results for the unconstrained multimodal test functions. In Table 2, we can see a comparison between the solutions obtained by the proposed algorithm H-GA–FA and the best-known solutions, where H-GA–FA found the best-known solutions in 13 multimodal test functions (RC, B2, ES, GP, SH, DJ, $S_{4,5}$, $R_2$, $R_5$, $R_{10}$, $Z_2$, $Z_5$, and $Z_{10}$). In the other four multimodal test functions, H-GA–FA found solutions very close to the best-known solutions in $H_{6,4}$, $S_{4,7}$, and $S_{4,10}$, while in $H_{3,4}$ H-GA–FA found a solution better than the best-known solution.

Also, in Table 2, we can see the calculated average error for the proposed algorithm H-GA–FA and the other approaches mentioned above. As a result of Table 2, the average error of H-GA–FA is smaller than those of other comparison algorithms in

**Table 1:** The parameters used in the execution of H-GA–FA.

| | Parameter | Notation | Value |
|---|---|---|---|
| **Initialization** | Size of population (individuals) | $4N$ | 40 |
| | Generations maximum number (iterations) | $K_{max}$ | 60–400 |
| **FA** | Initial attractiveness | $\beta_0$ | 1 |
| | The light absorption coefficient | $\gamma$ | 1 |
| | The step size factor | $\alpha$ | 0.95 |
| **GA** | | | |
| | Selection operator | Stochastic universal sampling | |
| | Crossover operator | Single-point crossover | |
| | Crossover rate | Pc | 0.8 |
| | Mutation operator | Real value mutation | |
| | Mutation rate | Pm | 0.06 |

**Table 2:** Results for unconstrained multimodal test functions.

| Test function | Best-known solution | Solution calculated by H-GA–FA | Average error | | | | |
|---|---|---|---|---|---|---|---|
| | | | CGA | CHA | H-GA–PSO | Integrating PSO–GA | H-GA–FA |
| RC | 0.397887 | 0.397887 | 0.0001 | 0.0001 | 0.00009 | 4.59E-7 | 0 |
| B2 | 0 | 0 | 0.0003 | 0.0000002 | 0.00001 | 1E-25 | 0 |
| ES | −1 | −1 | 0.0010 | 0.0010 | 0.00003 | 1E-30 | 0 |
| GP | 3 | 3 | 0.0010 | 0.0010 | 0.00012 | −6.3060E-14 | 0 |
| SH | −186.7309 | −186.7309 | 0.0050 | 0.0050 | 0.00007 | 8.83064E-6 | 0 |
| DJ | 0 | 0 | 0.0002 | 0.0002 | 0.00004 | 8.443663E-15 | 0 |
| $H_{3,4}$ | −3.86278 | −3.863433477876346 | 0.0050 | 0.0050 | 0.00020 | 3E-05 | 1.6917E-04 |
| $H_{6,4}$ | −3.32237 | −3.322368 | 0.0400 | 0.0080 | 0.00024 | 2E-6 | 6.0198E-07 |
| $S_{4,5}$ | −10.1532 | −10.1532 | 0.1400 | 0.0090 | 0.00014 | 0 | 0 |
| $S_{4,7}$ | −10.40294 | −10.40291634 | 0.1200 | 0.0100 | 0.00015 | 2E-05 | 2.2744E-06 |
| $S_{4,10}$ | −10.53641 | −10.53638558 | 0.1500 | 0.0150 | 0.00012 | 2E-05 | 2.3177E-06 |
| $R_2$ | 0 | 0 | 0.0040 | 0.0040 | 0.00064 | 1E-30 | 0 |
| $R_5$ | 0 | 0 | 0.1500 | 0.0180 | 0.00013 | 1E-20 | 0 |
| $R_{10}$ | 0 | 0 | 0.0200 | 0.0080 | 0.00005 | 1E-18 | 0 |
| $Z_2$ | 0 | 0 | 0.000003 | 0.000003 | 0.00005 | 1E-15 | 0 |
| $Z_5$ | 0 | 0 | 0.0004 | 0.00006 | 0.00000 | 1E-17 | 0 |
| $Z_{10}$ | 0 | 0 | 0.000001 | 0.000001 | 0.00000 | 1E-25 | 0 |

all test functions except in $H_{3,4}$ where H-GA–FA found a solution better than the best-known solution of this test function.

## 4.2 Results for constrained benchmark problems

Constrained benchmark problems are available in Appendix B, where the details, the variable bounds, objective function, and constraints, of all problems are displayed. The constrained benchmark problems were solved by three algorithms: Augmented Lagrange PSO (ALPSO; Sedlaczek & Eberhard, 2005), chaotic genetic algorithm (CGA; El-Shorbagy *et al.*, 2016), and the proposed algorithm H-GA–FA. As in ALPSO and CGA, the constrained benchmark problems were solved, by the proposed algorithm H-GA–FA, 30 independent times.

Table 3 shows the results for the constrained benchmark problems, where a comparison is made between the known optimal solutions for these problems and best solutions obtained by the three algorithms: ALPSO, CGA, and H-GA–FA. According to Table 3, the proposed algorithm H-GA–FA found the optimum solution in five cases (C1, C2, C3, C5, and C7), just like CGA, and outperformed ALPSO in C1 and C2. In C4, the H-GA–FA method obtained a solution that was close to the optimal solution. However, H-GA–FA achieved a solution that was near to the optimal solution in the test issue C6, which was not solved by CGA or ALPSO. In general, it can be said that the proposed algorithm H-GA–FA is capable of solving constrained benchmark problems, reaching the optimal solution, and showing superior performance compared to other algorithms.

## 4.3 Results for engineering design problems

In this subsection, the proposed algorithm performance is investigated in many EDPs that are the gear train design, cantilever design, pressure vessel design, welded beam design, tension/compression spring design, and speed reducer design. Each of the EDPs was solved 100 times independently, as in other compared algorithms. Statistically, to evaluate the proposed algorithm H-GA–FA, statistical measures have been recorded, which are the worst value, mean deviation value, best value, and standard deviation (SD), and we compare them to other algorithms. In addition, the number of function evaluations (NFEs) (computational cost) is determined by multiplying the number of $4N$ individuals that are randomly generated and the number of iterations. In other words, in this paper, the NFEs are considered as the value corresponding to the best-obtained solution.

**Table 3:** Results for constrained benchmark problems.

| Constrained benchmark problems | Known optimal solution | Best solution of | | |
|---|---|---|---|---|
| | | ALPSO | CGA | H-GA–FA |
| C1 | 13.0000 | 12.9995 | 13.0000 | 13.0000 |
| C2 | 0.01721 | 0.01719 | 0.01721 | 0.01721 |
| C3 | −0.09583 | −0.09583 | −0.09583 | −0.09583 |
| C4 | −6961.81 | −6963.57 | −6961.804 | −6961.80211 |
| C5 | 0.75 | 0.75 | 0.75 | 0.75 |
| C6 | 1.000 | NA | NA | 1.000001 |
| C7 | −30665.5 | −30 665.5 | 30665.8 | −30 665.5 |

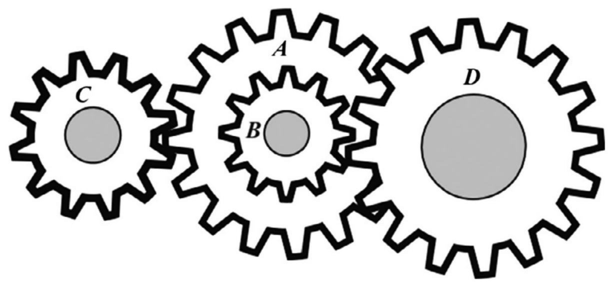


**Figure 6:** Gear train design problem (Gandomi *et al.*, 2013).

**A. The gear train design problem**:

The aim, of this problem, is to locate the optimal number of the teeth for four gears ($n_A \leftarrow x_1$, $n_B \leftarrow x_2$, $n_C \leftarrow x_3$, $n_D \leftarrow x_4$) to minimize the gear ratio cost ($n_B n_C / n_D n_A$) of a gear train as shown in Fig. 6 (Gandomi *et al.*, 2013). The mathematical description of this problem is as follows:

$$\text{Min } f(x) = \left(\frac{1}{6.931} - \frac{x_2 x_3}{x_1 x_4}\right)^2 \qquad (10)$$
$$\text{Subject to: } \quad 12 \le x_i \le 60, \quad i = 1, 2, 3, 4.$$

In this problem, the constraints just are lower and upper limits on the previous design variables. However, these design variables are in a discrete form where each gear must have an integer tooth number, which increases the complexity of the problem. In the proposed algorithm, every search solution was rounded to the nearby integer number previous to the step of evaluation.

This problem is solved by many algorithms such as cuckoo search (CS; Gandomi *et al.*, 2013), mine blast algorithm (MBA; Sadollah *et al.*, 2013), interior search algorithm (ISA; Gandomi, 2014), GA (Wu & Chow, 1995), combined genetic adaptive search (GeneAS; Deb & Goyal, 1996), augmented Lagrange multiplier (ALM; Kannan & Kramer, 1994) and ant lion optimizer (ALO; Mirjalili, 2015), unified PSO (UPSO; Parsopoulos & Vrahatis, 2005), and ABC (Akay & Karaboga, 2012). Table 4 explains a comparison of the best-optimized design variables and the best function value found by these algorithms, the original FA, the original GA, and the proposed algorithm H-GA–FA. Table 5 gives the statistical results using the different optimizers. In addition, Fig. 7 demonstrates the convergence curves of gear ratio cost (function values) versus the iteration number for the original FA, the original GA, and H-GA–FA.

Results show that H-GA–FA can solve discrete real problems efficiently where it found a design with the optimal value identical to that of ALO and outperforms other algorithms. Also, from the values of best, worst, mean, and SD, we can see that the solution accuracy and stability of H-GA–FA are better than other algorithms. Furthermore, H-GA–FA gives a new (design) solution ($n_A = 36$, $n_B = 12$, $n_C = 16$, $n_D = 37$) with an acceptable value for the function value. Finally, Fig. 7 shows that H-GA–FA has fast convergence to reach a solution better than original FA and original GA.

**B. Cantilever beam design problem**

As shown in Fig. 8, this design problem consists of five hollow elements with a square-shaped cross-section (Mirjalili, 2015). Every element is expressed as a variable, whereas the thickness is considered constant. So the problem includes five structural parameters and only one perpendicular displacement constraint. From Fig. 8, we can see that there is a perpendicular load applied to node 6 and node 1 is strictly supported. This problem aims to minimize the weight of this beam that is subject to one constraint; where the final optimal design must be fulfilled this

**Table 4:** Comparisons for gear train design problem.

| Method | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $f(x)$ | NFEs |
|---|---|---|---|---|---|---|
| CS | 43 | 16 | 19 | 49 | 2.7009E-012 | 5000 |
| MBA | 43 | 16 | 19 | 49 | 2.700857E-012 | 10 000 |
| ISA | NA | NA | NA | NA | 2.7009E-012 | 200 |
| GA | NA | NA | NA | NA | 2.33E-07 | 10 000 |
| GeneAS | 33 | 14 | 17 | 50 | 1.362E-09 | NA |
| ALM | 33 | 15 | 13 | 41 | 2.1469E-08 | NA |
| ALO | 49 | 19 | 16 | 43 | 2.7009E-012 | 120 |
| Original FA | 33 | 15 | 13 | 41 | 2.1469E-08 | 6000 |
| Original GA | 33 | 14 | 17 | 50 | 1.362E- 09 | 6000 |
| The proposed algorithm H-GA–FA | 49 | 19 | 16 | 43 | 2.7009E-012 | 6000 |
| | 36 | 12 | 16 | 37 | 1.8274E-08 | 6000 |

*Note.* "NA" means that the result is not available.

**Table 5:** Statistical result comparison for gear train problem.

| Method | Worst | Mean | Best | SD | NFEs |
|---|---|---|---|---|---|
| Unified PSO (UPSO) | NA | 3.805620E-08 | 2.700857E-12 | 1.09E-07 | 100 000 |
| ABC | NA | 3.641339E-10 | 2.700857E-12 | 5.52E-10 | 60 |
| MBA | 2.062904E-08 | 2.471635E-09 | 2.700857E-12 | 3.94E-09 | 1120 |
| H-GA–FA | 2.0453E-09 | 2.156400E-010 | 2.7009E-012 | 1.49E-010 | 6000 |

*Note.* "NA" means that the result is not available.

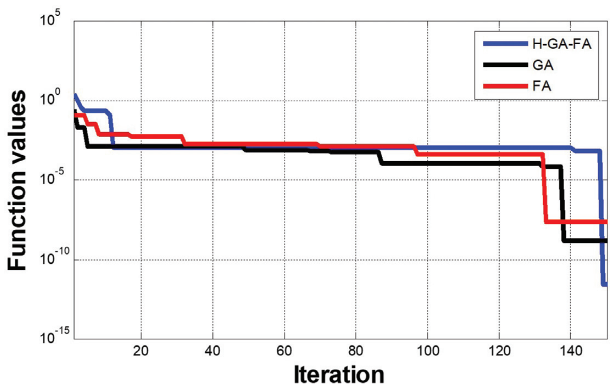


**Figure 7:** Convergence curve of gear ratio cost versus the iteration number for the original FA, the original GA, and H-GA–FA.

constraint. The mathematical formulation of this problem is as follows:

$$\begin{aligned} &\text{Min } f(x) = 0.6224x_1 + x_2 + x_3 + x_4 + x_5 \\ &\text{Subject to : } \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} \le 1 \\ &0.01 \le x_i \le 100, \quad i = 1, 2, \ldots, 5. \end{aligned} \tag{11}$$

This problem is solved by many algorithms such as symbiotic organisms search (SOS; Cheng & Prayogo, 2014), CS (Gandomi *et al.*, 2013), method of moving asymptotes (MMA; Chickermane & GEA, 1996), generalized convex approximation (GCA_I; Chickermane & GEA, 1996), GCA_II (Chickermane & GEA, 1996), enhanced leader PSO (ELPSO; Jordehi, 2015), WOA (Zhou *et al.*, 2018), improved WOA based on a Lévy flight trajectory (LWOA; Zhou *et al.*, 2018), and ALO (Mirjalili, 2015). Table 6 illustrates the comparison of the best optimal design variables and the best function value obtained by H-GA–FA and these algorithms. Statistical results of this problem using the different algorithms are shown in Table 7, while Fig. 9 demonstrates the convergence curve of the weight of this beam (function values) versus the iteration number for the original FA, the original GA, and H-GA–FA.

Results explain that H-GA–FA outperforms other algorithms, where the obtained weight by H-GA–FA is less than that obtained by other algorithms. This shows that H-GA–FA has a high performance in bringing the global best solution to this problem. Also, Fig. 9 shows that H-GA–FA has fast convergence to reach a solution better than original FA and original GA. In addition, from the values of best, worst, mean and SD, we can see that the solution accuracy and stability of the proposed algorithm are better than other algorithms. Finally, the maximum number of function evaluations in Tables 6 and 7 demonstrates the superiority of H-GA–FA over the other algorithms, where it found the global

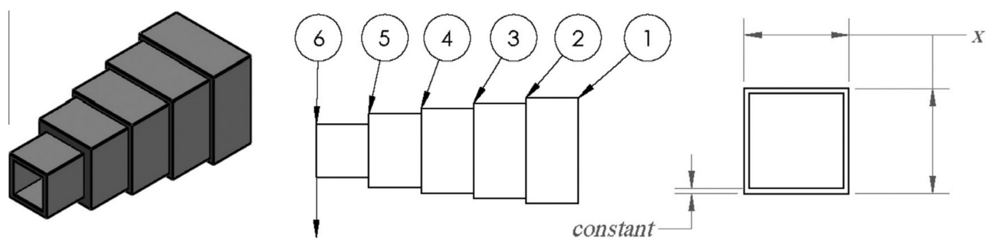


**Figure 8:** Cantilever beam design problem (Mirjalili, 2015).

**Table 6:** Comparisons for the cantilever design problem.

| Algorithm | Optimal values for variables | | | | | Optimum weight | NFEs |
|---|---|---|---|---|---|---|---|
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $f(x)$ | |
| SOS | 6.01878 | 5.30344 | 4.49587 | 3.49896 | 2.15564 | 1.33996 | 15 000 |
| CS | 6.0089 | 5.3049 | 4.5023 | 3.5077 | 2.1504 | 1.33999 | 2500 |
| MMA | 6.0100 | 5.3000 | 4.4900 | 3.4900 | 2.1500 | 1.3400 | NA |
| GCA_I | 6.0100 | 5.30400 | 4.4900 | 3.4980 | 2.1500 | 1.3400 | NA |
| GCA_II | 6.0100 | 5.3000 | 4.4900 | 3.4900 | 2.1500 | 1.3400 | NA |
| ELPSO | 6.0160 | 5.3092 | 4.4943 | 3.5015 | 2.1527 | 1.34000 | NA |
| WOA | 5.9518 | 5.0649 | 4.6744 | 3.5476 | 2.2916 | 1.34350 | NA |
| LWOA | 6.0105 | 5.3470 | 4.4468 | 3.5165 | 2.1546 | 1.34000 | NA |
| ALO | 6.01812 | 5.31142 | 4.48836 | 3.49751 | 2.158329 | 1.33995 | 14 000 |
| Original FA | 6.134208 | 5.35598 | 4.527011 | 3.378503 | 2.102520 | 1.33804996 | 2400 |
| Original GA | 6.197047 | 5.41243 | 4.413968 | 3.419296 | 2.064024 | 1.33858163 | 2400 |
| H-GA–FA | 6.017563478 | 5.311392298 | 4.497733036 | 3.494192221 | 2.152821173 | **1.336523225** | 2400 |

*Note.* "NA" means that the result is not available.

**Table 7:** Statistical result comparison for the cantilever design problem.

| Method | Worst | Mean | Best | SD | NFEs |
|---|---|---|---|---|---|
| ELPSO | 1.3400 | 1.3400 | 1.3400 | 0.00001755 | NA |
| WOA | 1.4865 | 1.3668 | 1.3435 | 0.0270 | NA |
| LWOA | 1.349137 | 1.34101 | 1.3400 | 0.002208 | NA |
| H-GA–FA | **1.343678930** | **1.340121354** | **1.336523225** | 0.0008745 | 2400 |

*Note.* "NA" means that the result is not available.

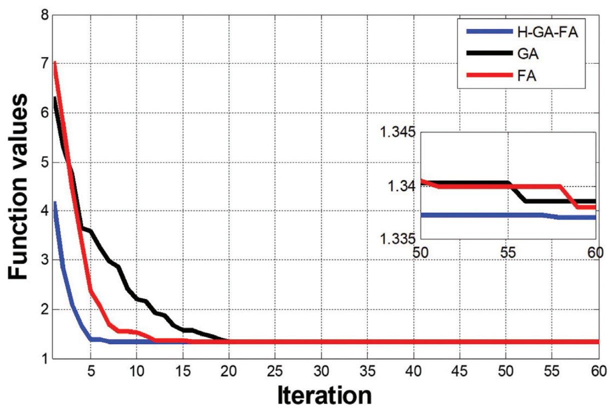


**Figure 9:** Convergence curve of cantilever beam weight versus the iteration numbers for the original FA, the original GA, and H-GA–FA.

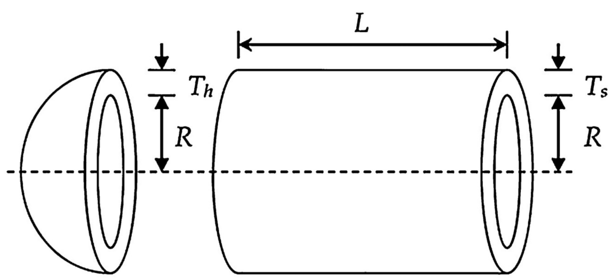


**Figure 10:** Pressure vessel design problem (Sadollah *et al.*, 2013).

optimum solution for this problem in the fewest number of function evaluations.

**C. Pressure vessel design problem**

As shown in Fig. 10, this design problem is specific to a cylindrical vessel that is capped at both ends by hemispherical heads (Sadollah *et al.*, 2013). It was proposed in Kannan and Kramer (1994) and aims to minimize the total cost, containing the cost of welding, forming, and material. It contains four design variables: the shell thickness $T_s$ ($x_1$), head thickness $T_h$ ($x_2$), inner radius R ($x_3$), and the cylindrical section length of the vessel with no head L ($x_4$). The two variables $R$ and $L$ are continuous, while the other two variables $T_s$ and $T_h$ are expected to be integer multiples of 0.0625 inch that is the available thickness of the rolled steel plates. This problem is written mathematically as

$$\begin{aligned} &\text{Min } f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\ &\text{Subject to: } \quad -x_1 + 0.0193x_3 \le 0 \\ &\qquad -x_2 + 0.00954x_3 \le 0 \\ &\qquad -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296\,000 \le 0 \\ &\qquad x_4 - 240 \le 0 \\ &\qquad 1 \times 0.0625 \le x_1, x_2 \le 99 \times 0.0625 \\ &\qquad 10 \le x_3, x_4 \le 200. \end{aligned} \tag{12}$$

Previously, this problem is solved using other algorithms such as GA-based co-evolution model (GA3; Coello, 2000), GA through the use of dominance-based tour tournament selection (GA4; Coello & Montes, 2002), co-evolutionary PSO (CPSO; He & Wang, 2006), hybrid PSO (HPSO; He & Wang, 2007), hybrid Nelder-Mead simplex search and PSO (NM-PSO; Zahara & Kao, 2009), Gaussian quantum-behaved PSO (G-QPSO; Coelho, 2010), PSO with DE (PSO-DE; Liu *et al.*, 2010), UPSO (Parsopoulos & Vrahatis,

**Table 8:** The comparisons for pressure vessel design problem.

| Method | Design variables | | | | $f(x)$ |
|---|---|---|---|---|---|
| | $T_s$ | $T_h$ | $R$ | $L$ | |
| GA3 | 0.8125 | 0.4375 | 40.3239 | 200.0000 | 6288.7445 |
| GA4 | 0.8125 | 0.4375 | 42.0974 | 176.6540 | 6059.9463 |
| CPSO | 0.8125 | 0.4375 | 42.0913 | 176.7465 | 6061.0777 |
| HPSO | 0.8125 | 0.4375 | 42.0984 | 176.6366 | 6059.7143 |
| NM-PSO | 0.8036 | 0.3972 | 41.6392 | 182.4120 | 5930.3137 |
| G-QPSO | 0.8125 | 0.4375 | 42.0984 | 176.6372 | 6059.7208 |
| MBA | 0.7802 | 0.3856 | 40.4292 | 198.4964 | 5889.3216 |
| Original FA | 0.862732839386900 | 0.426319500657094 | 44.6836939800188 | 147.117669089838 | 6049.94158296718 |
| Original GA | 0.924760935072174 | 0.457064486928402 | 47.8982898850724 | 116.128691460803 | 6193.19736445249 |
| H-GA–FA | 0.778169505064424 | 0.384649589721359 | 40.3196634724738 | 199.999377073430 | **5885.33425081451** |

Table 9: Statistical result comparison for pressure vessel design problem.

| Method | Worst | Mean | Best | SD | NFEs |
|---|---|---|---|---|---|
| GA3 | 6308.4970 | 6293.8432 | 6288.7445 | 7.4133 | 900 000 |
| GA4 | 6469.3220 | 6177.2533 | 6059.9463 | 130.9297 | 80 000 |
| CPSO | 6363.8041 | 6147.1332 | 6061.0777 | 86.45 | 240 000 |
| HPSO | 6288.6770 | 6099.9323 | 6059.7143 | 86.20 | 81 000 |
| NM-PSO | 5960.0557 | 5946.7901 | 5930.3137 | 9.161 | 80 000 |
| G-QPSO | 7544.4925 | 6440.3786 | 6059.7208 | 448.4711 | 8000 |
| PSO-DE | NA | 6059.714 | 6059.714 | NA | 42 100 |
| UPSO | NA | 9032.55 | 6544.27 | 995.573 | 100 000 |
| ABC | NA | 6245.308 | 6059.714 | 205 | 30 000 |
| ($\mu + \lambda$)-ES | NA | 6379.938 | 6059.7016 | 210 | 30 000 |
| TLBO | NA | 6059.71434 | 6059.714335 | NA | 10 000 |
| MBA | 6392.5062 | 6200.64765 | 5889.3216 | 160.34 | 70 650 |
| H-GA–FA | **5889.09968292685** | **5887.5687171994** | **5885.33425081451** | 1.05462 | 6000 |

*Note.* "NA" means that the result is not available.

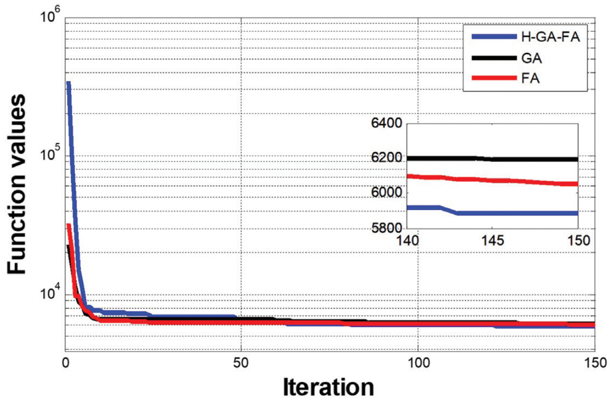


Figure 11: Convergence curve of pressure vessel design cost versus the iteration numbers for the original FA, the original GA, and H-GA–FA.

2005), ABC (Akay & Karaboga, 2012), ($\mu + \lambda$)-ES (Mezura-Montes & Coello, 2005), TLBO (Rao *et al.*, 2011), and MBA (Sadollah *et al.*, 2013). Table 8 demonstrates the comparisons of the best solution, for the optimization of pressure vessel design problem, for H-GA–FA and other reported algorithms, while statistical results are given in Table 9. In addition, Fig. 11 demonstrates the convergence curve of pressure vessel design cost (function values) versus the iteration number for the original FA, the original GA, and H-GA–FA.

Results show that H-GA–FA outperforms the other algorithms, where the obtained total cost by H-GA–FA is less than that obtained by other reported algorithms and illustrates that it has high performance in bringing the global best solution to this problem. Also, Fig. 11 shows that H-GA–FA has fast convergence to reach a solution better than original FA and original GA. In addition, from the values of best, worst, mean and SD, we can see that the solution accuracy and stability of the proposed algorithm are better than other reported algorithms. Finally, Tables 8 and 9 show that H-GA–FA has a low computational cost, where it found the global optimum solution for this problem in the least number of function evaluations.

**D. The welded beam design problem**

This design problem aims to optimize the designing cost of the welded beam subject to several constraints such as shear stress ($\tau$), bending stress ($\sigma$), buckling load (P), end deflection ($\delta$), and side constraints (Cheng & Prayogo, 2014). As shown

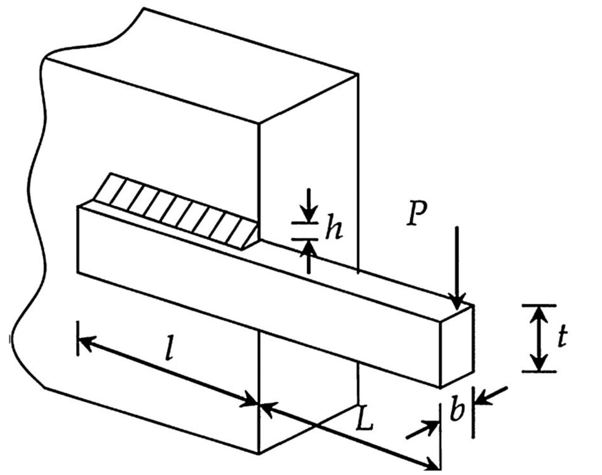


Figure 12: The welded beam design problem (Sadollah *et al.*, 2013).

in Fig. 12, this design problem includes four variables, which are the weld thickness $h(x_1)$, attached part length of bar $l(x_2)$, the bar height $t(x_3)$, and the bar thickness $b(x_4)$ (Sadollah *et al.*, 2013). The mathematical formulation of this problem is written as

$$
\begin{aligned}
&\text{Min } f = 1.10471h^2 l + 0.04811\, t\, b(14 + l)\\
&\text{Subject to: } \tau - \tau_{max} \le 0 \rightarrow \begin{cases} \tau = \sqrt{\tau_1^2 + 2\tau_1\tau_2\left(\dfrac{l}{2R}\right) + \tau_2^2} \\ \tau_1 = \dfrac{P}{\sqrt{2}hl} \\ \tau_2 = \dfrac{MR}{J} \\ \tau_{max} = 13\,600\text{psi} \end{cases} \rightarrow \begin{cases} R = \dfrac{1}{2}\sqrt{l^2 + h + t^2} \\ P = 6000\text{ lb} \\ M = P(L + 0.5) \\ J = 2\left(\dfrac{hl}{\sqrt{2}}\left[\dfrac{l^2}{12} + \left(\dfrac{h+t}{2}\right)^2\right]\right) \end{cases}\\
&\sigma - \sigma_{max} \le 0 \rightarrow \begin{cases} \sigma = \dfrac{6PL}{bt^2} \\ \sigma_{max} = 30\,000\text{psi} \end{cases} \rightarrow \{L = 14\text{ in}\\
&h - b \le 0,\ 0.125 - h \le 0\\
&\delta - 0.25 \le 0 \rightarrow \left\{\delta = \dfrac{4PL^3}{E\,b\,t^3}\right. \rightarrow \{E = 30 \times 10^6\text{psi}\\
&P - P_c \le 0 \rightarrow \left\{P_c = \dfrac{4.013\sqrt{EGt^2b^6}}{6L^2}\left(1 - \dfrac{t}{2L}\sqrt{\dfrac{E}{4G}}\right)\right. \rightarrow \{G = 12 \times 10^6\text{psi}\\
&0.1 \le h \le 2,\ 0.1 \le l, t \le 10,\ 0.1 \le b \le 2,
\end{aligned} \tag{13}
$$

where $\tau_1$ is the primary stress, $\tau_2$ is the secondary stress, $\tau_{max}$ is the maximum permissible shear stress of the weld, $L$ is the length of the overhang portion of the beam, $\sigma_{max}$ is the

**Table 10:** Comparisons for the welded beam design problem.

| Method | Design variables | | | | $f(x)$ |
|---|---|---|---|---|---|
| | $h$ | $l$ | $t$ | $b$ | |
| **Developed HS** | 0.2442 | 6.2231 | 8.2915 | 0.2443 | 2.38 |
| **A penalty-guided ABC** | 0.24436198 | 6.21767407 | 8.29163558 | 0.24436883 | 2.38099617 |
| **ARSAGA** | 0.223100 | 1.5815 | 12.84680 | 0.2245 | 2.25 |
| **GP** | 0.245500 | 6.196000 | 8.273000 | 0.245500 | 2.385937 |
| **SSM** | 0.24436895 | 6.21860635 | 8.29147256 | 0.24436895 | 2.3811341 |
| **SC algorithm** | 0.244438276 | 6.237967234 | 8.2885761430 | 0.2445661820 | 2.3854347 |
| **GSA** | 0.22349476917 | 7.8278697763206 | 8.28377214722 | 0.25644859066 | 2.6628178327 |
| **LXGSA** | 0.24434783486 | 6.2153684818530 | 8.29452799679 | 0.24434813295 | 2.3810923303 |
| **PMGSA** | 0.24265157269 | 6.2768500252288 | 8.29587136937 | 0.24436429788 | 2.3858653166 |
| **LXPMGSA** | 0.24436873660 | 6.2174949128463 | 8.29150645150 | 0.24436873664 | 2.3809581283 |
| **Original FA** | 0.323828994417369 | 4.95633313479088 | 6.46449574776373 | 0.402014677676637 | 2.9442673784997 |
| **Original GA** | 0.326565801979975 | 4.44908967197571 | 7.10293125986479 | 0.332992262286482 | 2.6234928286527 |
| **H-GA–FA** | 0.24436897580 | 5.3104279688297 | 8.29147139048 | 0.24436897580 | **2.2326937762** |

**Table 11:** Statistical result comparison for the welded beam design problem.

| Method | Worst | Mean | Best | SD |
|---|---|---|---|---|
| **Developed HS** | NA | NA | 2.38 | NA |
| **A penalty-guided ABC** | 2.38146999 | 2.38108932 | 2.38099617 | 0.00010123 |
| **ARSAGA** | 2.28 | 2.26 | 2.25 | NA |
| **GP** | NA | NA | 2.385937 | NA |
| **SSM** | 2.3812614 | 2.3811786 | 2.3811341 | NA |
| **SC algorithm** | 6.3996785 | 3.2551371 | 2.3854347 | 0.9590780 |
| **GSA** | 4.658170101252 | 3.509128043982 | 2.662817832721 | 0.522504812259 |
| **LXGSA** | 2.798319913618 | 2.486396607177 | 2.381092330370 | 0.109157595213 |
| **PMGSA** | 2.634437273036 | 2.467970463125 | 2.385865316644 | 0.075590736939 |
| **LXPMGSA** | 2.537728435508 | 2.416140783674 | 2.38095812831348 | 0.045576480782 |
| **H-GA–FA** | **2.2330487596** | **2.2328875241** | **2.2326937762** | **0.009913753581** |

*Note.* "NA" means that the result is not available.

maximum permissible normal stress for the beam material, $P_c$ is the bar buckling load, $M$ is called moment, and $J$ is called the polar moment of inertia.

This benchmark problem was solved by many other optimizers such as developed harmony search (HS) algorithm (Lee & Geem, 2005), a penalty-guided ABC algorithm (Garg, 2014), a novel adaptive real-parameter simulated annealing genetic algorithm (ARSAGA; Hwang & He, 2006), geometric programming (GP; Ragsdell & Phillips, 1976), simplex search method (SSM; Mehta & Dasgupta, 2012), society and civilization (SC) algorithm (Ray & Liew, 2003), GSA, and three new variants of GSA, namely LXGSA (Singh & Deep, 2017), PMGSA (Singh & Deep, 2017), and LXPMGSA (Singh & Deep, 2017) by embedded Laplace crossover and power mutation into GSA.

The comparison for the best solution given by H-GA–FA and these algorithms is offered in Table 10, where the proposed algorithm obtained the best solution $f(x) = 2.2326937762$, while Table 11 explains the statistical results for optimization, where the values of best, worst, mean, and SD indicated that the solutions of the proposed algorithm are more accurate and stable than other algorithms. In addition, Fig. 13 shows the convergence curve of the designing cost of the welded beam (function values) versus the iteration numbers for the original FA, the original GA, and H-GA–FA. We can see that in the proposed H-GA–FA algorithm, at the early iterations, the function values are reduced close to the optimum point.

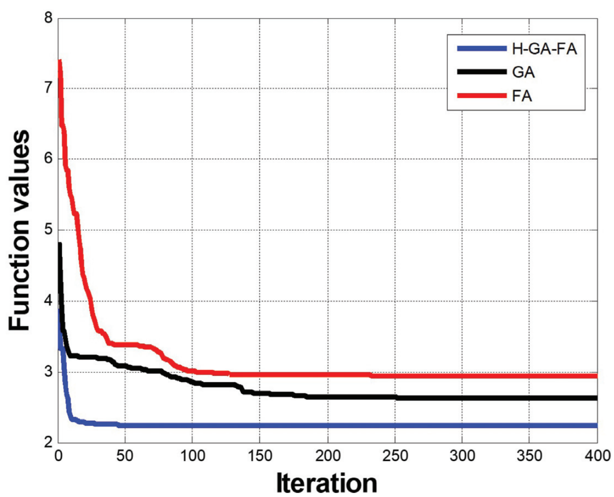


**Figure 13:** Convergence curve of designing cost of the welded beam versus the iteration numbers for the original FA, the original GA, and H-GA–FA.

**E. Tension/compression spring design problem**

The goal of this optimization problem is to minimize the weight of tension/compression spring according to many constraints such as shear stress, minimum deflection, surge frequency, outside diameter limits, and design variables. The tension/compression spring contains three variables, which are the wire diameter $d(x_1)$, the mean coil diameter $D(x_2)$, and active coil number $P(x_3)$ as shown in Fig. 14 (Sadollah *et al.*, 2013).

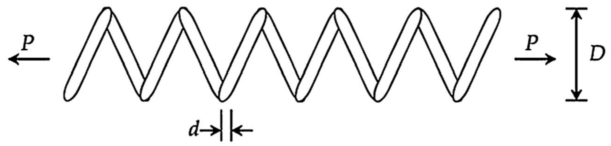


Figure 14: Tension/compression spring design problem (Sadollah *et al.*, 2013).

This problem can be described mathematically as

$$
\begin{aligned}
&\text{Min } f(x) = (x_3+2)x_2x_1^2\\
&\text{Subject to}: \quad 1-\frac{x_2^3x_3}{71\,785x_1^4}\le 0\\
&\qquad\frac{4x_2^2-x_1x_2}{12\,566(x_2x_1^3-x_1^4)}+\frac{1}{5108x_1^2}-1\le 0\\
&\qquad 1-\frac{140.45x_1}{x_2^2x_3}\le 0\\
&\qquad \frac{x_1+x_2}{1.5}-1\le 0\\
&\qquad 0.05\le x_1\le 2\\
&\qquad 0.25\le x_2\le 1.3\\
&\qquad 2\le x_3\le 15.
\end{aligned}
\tag{14}
$$

The methods that before used to solve this problem are GA3 (Coello, 2000), GA4 (Coello & Montes, 2002), cultural algorithms with evolutionary programming (CAEP; Coello & Becerra, 2004), UPSO (Parsopoulos & Vrahatis, 2005), CPSO (He & Wang, 2006), HPSO (He & Wang, 2007), NM-PSO (Zahara & Kao, 2009), G-QPSO (Coelho, 2010), quantum-behaved PSO (QPSO; Coelho, 2010), PSO-DE (Liu *et al.*, 2010), differential evolution with level comparison (DELC; Wang & Li, 2010), differential evolution with dynamic stochastic selection (DEDS; Zhang *et al.*, 2008), hybrid evolutionary algorithm and adaptive constraint handling technique (HEAA; Wang *et al.*, 2009), SC algorithm (Ray & Liew, 2003), ($\mu+\lambda$)-ES (Mezura-Montes & Coello, 2005), ABC (Akay & Karaboga, 2012), TLBO (Rao *et al.*, 2011), and MBA (Sadollah *et al.*, 2013).

The best solutions given by such algorithms are offered in Table 12, where the proposed method obtained the best solution $f(x) =$ 0.012665 as most comparison algorithms. However, the statistical results for the optimization of the tension/compression spring design problem, given in Table 13, show that the solutions resulting from the proposed algorithm are accurate and consistent as most comparison methods. Also, the maximum number of function evaluations in Table 13 shows that H-GA–FA is competitive with other algorithms, as it is the third-best value among the nineteen comparison methods. On the other hand, Fig. 15 demonstrates the convergence curve of tension/compression spring weight (function values) versus the iteration numbers for the original FA, the original GA, and H-GA–FA. We can see that H-GA–FA has fast convergence at the early iterations to the optimal solution.

**F. Speed reducer design problem**

The goal of this design problem is to minimize the weight of the speed reducer subject to several constraints such as the stress of gear teeth bending, the stress of surface, the shafts' transverse deflections, and stresses in the shafts. It includes

Table 12: Comparisons for tension/compression spring problem.

| D.V. | DELC | DEDS | CPSO | HPSO | NM-PSO | G-QPSO | HEAA | MBA | Original FA | Original GA | H-GA–FA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 0.051689 | 0.051689 | 0.051728 | 0.051706 | 0.051620 | 0.051515 | 0.051689 | 0.051656 | 0.06741053 | 0.05768662 | 0.05172433 |
| $x_2$ | 0.356717 | 0.356717 | 0.357644 | 0.357126 | 0.355498 | 0.352529 | 0.356729 | 0.355940 | 0.86479565 | 0.47742290 | 0.35756704 |
| $x_3$ | 11.288965 | 11.288965 | 11.244543 | 11.265083 | 11.333272 | 11.538862 | 11.288293 | 11.344665 | 9.56892128 | 8.83673351 | 11.2393464 |
| $g_1$ | – 3.40E-09 | 1.45E-09 | – 8.25E-04 | – 3.06E-06 | 1.01E-03 | -4.83E-05 | 3.96E-10 | 0 | – 3.1750223 | – 0.2096714 | – 2.6645E-15 |
| $g_2$ | 2.44E-09 | – 1.19E-09 | – 2.52E-05 | 1.39E-06 | 9.94E-04 | – 3.57E-05 | – 3.59E-10 | 0 | – 0.0012838 | – 0.0679039 | – 1.3323E-15 |
| $g_3$ | – 4.053785 | – 4.053785 | – 4.051306 | – 4.054583 | – 4.061859 | – 4.0455 | – 4.053808 | – 4.052248 | – 0.3229989 | – 3.0225234 | – 4.0555 |
| $g_4$ | – 0.727728 | – 0.727728 | – 0.727085 | – 0.727445 | – 0.728588 | – 0.73064 | – 0.727720 | – 0.728268 | – 0.3785292 | – 0.6432603 | – 0.7271 |
| $f(x)$ | 0.012665 | 0.012665 | 0.0126747 | 0.0126652 | 0.0126302 | 0.012665 | 0.012665 | 0.012665 | 0.04546339 | 0.01721678 | 0.012665 |

Table 13: Statistical result comparison for tension/compression spring problem.

| Method | Worst | Mean | Best | SD | NFEs |
|---|---|---|---|---|---|
| GA3 | 0.0128220 | 0.0127690 | 0.0127048 | 3.94E-05 | 900 000 |
| GA4 | 0.0129730 | 0.0127420 | 0.0126810 | 5.90E-05 | 80 000 |
| CAEP | 0.0151160 | 0.0135681 | 0.0127210 | 8.42E-04 | 50 020 |
| CPSO | 0.0129240 | 0.0127300 | 0.0126747 | 5.20E-04 | 240 000 |
| HPSO | 0.0127190 | 0.0127072 | 0.0126652 | 1.58E-05 | 81 000 |
| NM-PSO | 0.0126330 | 0.0126314 | 0.0126302 | 8.47E-07 | 80 000 |
| G-QPSO | 0.017759 | 0.013524 | 0.012665 | 0.001268 | 2000 |
| QPSO | 0.018127 | 0.013854 | 0.012669 | 0.001341 | 2000 |
| DELC | 0.012665575 | 0.012665267 | 0.012665233 | 1.3E-07 | 20 000 |
| DEDS | 0.012738262 | 0.012669366 | 0.012665233 | 1.3E-05 | 24 000 |
| HEAA | 0.012665240 | 0.012665234 | 0.012665233 | 1.4E-09 | 24 000 |
| PSO-DE | 0.012665304 | 0.012665244 | 0.012665233 | 1.2E-08 | 24 950 |
| SC algorithm | 0.016717272 | 0.012922669 | 0.012669249 | 5.9E-04 | 25 167 |
| UPSO | NA | 0.02294 | 0.01312 | 7.20E-03 | 100 000 |
| ($\mu+\lambda$)-ES | NA | 0.013165 | 0.012689 | 3.9E-04 | 30 000 |
| ABC | NA | 0.012709 | 0.012665 | 0.012813 | 30 000 |
| TLBO | NA | 0.01266576 | 0.012665 | NA | 10 000 |
| MBA | 0.012900 | 0.012713 | 0.012665 | 6.30E-05 | 7650 |
| H-GA–FA | 0.012665 | 0.012665 | 0.012665 | 0 | 7000 |

*Note*. "NA" means that the result is not available.

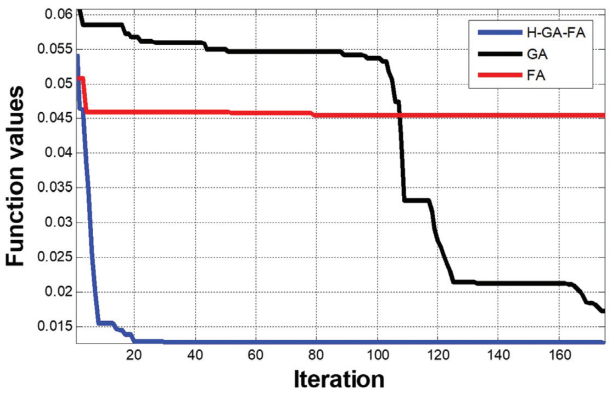


**Figure 15:** Convergence curve of tension/compression spring weight versus the iteration numbers for the original FA, the original GA, and H-GA–FA.

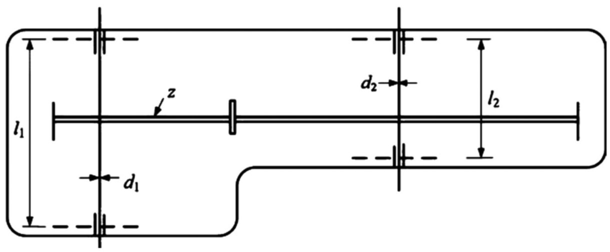


**Figure 16:** Speed reducer design problem (Sadollah *et al.*, 2013).

seven variables that represent the width of face $b(x_1)$, the teeth module $m(x_2)$, the number of the teeth in the pinion $z(x_3)$, the first shaft length between bearings $l_1(x_4)$, the second shaft length between bearings $l_2(x_5)$, the first shaft diameter $d_1(x_6)$, and the second shaft $d_2(x_7)$ as shown in Fig. 16 (Sadollah *et al.*, 2013). Mathematically, this problem can be described as

$$
\begin{aligned}
\text{Min } f(x) &= 0.7854x_1x_2^2\left(3.3333x_3^2 + 14.9334x_3 - 43.0934\right) \\
&\quad - 1.508x_1\left(x_6^2 + x_7^2\right) + 7.4777\left(x_6^3 + x_7^3\right) + 0.7854\left(x_4x_6^2 + x_5x_7^2\right) \\
\text{Subject to: } & \frac{27}{x_1x_2^2x_3} - 1 \le 0, \quad \frac{397.5}{x_1x_2^2x_3^2} - 1 \le 0, \quad \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \le 0 \\
& \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \le 0, \quad \frac{\sqrt{\left(\frac{745x_4}{x_2x_3}\right)^2 + 16.9\times10^6}}{110x_6^3} - 1 \le 0 \\
& \frac{\sqrt{\left(\frac{745x_5}{x_2x_3}\right)^2 + 157.5\times10^6}}{85x_7^3} - 1 \le 0, \quad \frac{x_2x_3}{40} - 1 \le 0 \\
& \frac{5x_2}{x_1} - 1 \le 0, \quad \frac{x_1}{12x_2} - 1 \le 0 \\
& \frac{1.5x_6 + 1.9}{x_4} - 1 \le 0, \quad \frac{1.1x_7 + 1.9}{x_5} - 1 \le 0 \\
& 2.6 \le x_1 \le 3.6, \quad 0.7 \le x_2 \le 0.8, \quad 17 \le x_3 \le 28 \\
& 7.3 \le x_4 \le 8.3, \quad 7.3 \le x_5 \le 8.3, \quad 2.9 \le x_6 \le 3.9 \\
& 5 \le x_7 \le 5.5.
\end{aligned}
\tag{15}
$$

This problem is solved previously by many algorithms as SC algorithm (Ray & Liew, 2003), PSO-DE (Liu *et al.*, 2010), DELC (Wang & Li, 2010), DEDS (Zhang *et al.*, 2008), HEAA (Wang *et al.*, 2009), modified differential evolution (MDE; Mezura-Montes *et al.*, 2006), $(\mu + \lambda)$-ES (Mezura-Montes & Coello, 2005), ABC (Akay & Karaboga, 2012), TLBO (Rao *et al.*, 2011), MBA (Sadollah *et al.*, 2013). Table 14 shows a comparison between the best solution obtained by such algorithms and the proposed algorithm H-GA–FA, while the statistical results for this problem of all algorithms are given in Table 15.

Table 14 shows that the proposed method obtained the best solution $f(x) =$ 2994.471066 as in DEDS and DELC methods and better than the other algorithms. The statistical results for the optimization of speed reducer design problem that is given in Table 15 show that the solutions resulting from the proposed algorithm are accurate and consistent as in $(\mu + \lambda)$-ES, ABC, and TLBO methods and better than the other algorithms. On the other hand, the maximum number of function evaluations in

**Table 14:** Comparisons for speed reducer design problem.

| D.V. | DEDS | DELC | HEAA | MDE | PSO-DE | MBA | Original FA | Original GA | H-GA–FA |
|---|---|---|---|---|---|---|---|---|---|
| $x_1$ | 3.5 + 09 | 3.5 + 09 | 3.500022 | 3.500010 | 3.50000 | 3.50000 | 3.502215 | 3.500703 | 3.50000 |
| $x_2$ | 0.7 + 09 | 0.7 + 09 | 0.70000039 | 0.70000 | 0.70000 | 0.70000 | 0.700067 | 0.700059 | 0.70000 |
| $x_3$ | 17 | 17 | 17.000012 | 17 | 17.0000 | 17.0000 | 17.007069 | 17.000985 | 17.0000 |
| $x_4$ | 7.3 + 09 | 7.3 + 09 | 7.300427 | 7.300156 | 7.30000 | 7.300033 | 7.393423 | 7.300368 | 7.30000 |
| $x_5$ | 7.715319 | 7.715319 | 7.715377 | 7.800027 | 7.800000 | 7.715772 | 7.716988 | 7.741199 | 7.715319 |
| $x_6$ | 3.350214 | 3.350214 | 3.350230 | 3.350221 | 3.350214 | 3.350218 | 3.352355 | 3.351213 | 3.350214 |
| $x_7$ | 5.286654 | 5.286654 | 5.286663 | 5.286685 | 5.2866832 | 5.286654 | 5.287034 | 5.287375 | 5.286654 |
| $f(x)$ | 2994.471066 | 2994.471066 | 2994.499107 | 2996.356689 | 2996.348167 | 2994.482453 | 2998.515145 | 2996.472564 | 2994.471066 |

**Table 15:** Statistical result comparison for speed reducer design problem.

| Method | Worst | Mean | Best | SD | NFEs |
|---|---|---|---|---|---|
| SC algorithm | 3009.964736 | 3001.758264 | 2994.744241 | 4.0 | 54 456 |
| PSO-DE | 2996.348204 | 2996.348174 | 2996.348167 | 6.4E-06 | 54 350 |
| DELC | 2994.471066 | 2994.471066 | 2994.471066 | 1.9E-12 | 30 000 |
| DEDS | 2994.471066 | 2994.471066 | 2994.471066 | 3.6E-12 | 30 000 |
| HEAA | 2994.752311 | 2994.613368 | 2994.499107 | 7.0E-02 | 40 000 |
| MDE | NA | 2996.367220 | 2996.356689 | 8.2E-03 | 24 000 |
| $(\mu + \lambda)$-ES | NA | 2996.348 | 2996.348 | 0 | 30 000 |
| ABC | NA | 2997.058 | 2997.058 | 0 | 30 000 |
| TLBO | NA | 2996.34817 | 2996.34817 | 0 | 10 000 |
| MBA | 2999.652444 | 2996.769019 | 2994.482453 | 1.56 | 6300 |
| H-GA–FA | 2994.471066 | 2994.471066 | 2994.471066 | 0 | 16 000 |

*Note.* "NA" means that the result is not available.

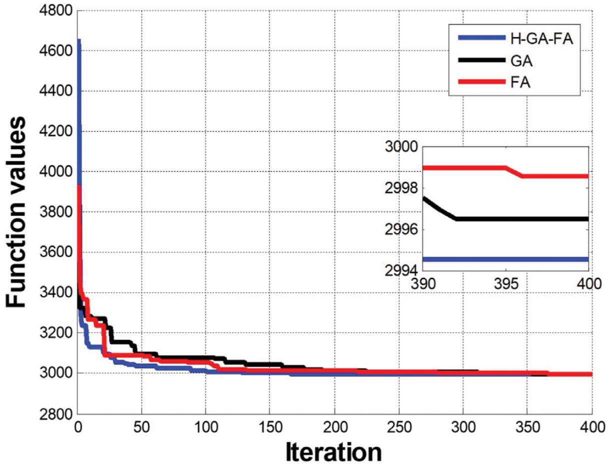


**Figure 17:** Convergence curve of the speed reducer weight versus the iteration numbers for the original FA, the original GA, and H-GA–FA.

Table 15 shows that H-GA–FA is competitive with other algorithms, where it found the global optimum solution for this problem in the third place according to the number of function evaluations. Figure 17 demonstrates the convergence curve of the speed reducer weight (function values) versus the iteration number for the original FA, the original GA, and H-GA–FA.

Finally, Table 16 shows an overall comparison between all methods in solving EDPs. The overall comparison shows that H-GA–FA is considered an efficient optimization system to solve EDPs and an attractive alternate optimizer offering quick convergence and high solution quality. This is due to the integration of FA with the GA process that leads to efficiently performing global exploration, rapid scanning of the feasible area, and effectively reaching the optimum solution. Also, in most EDPs, merging GA's exploitation and FA's exploration capabilities leads to a balance between exploration and exploitation in H-GA–FA and adequate variety of solutions, and prevents H-GA–FA from falling into local optima, as shown in convergent figures.

#### 4.3.1. *Engineering design problems' Friedman test and Wilcoxon signed-rank test*

The Friedman test (Derrac *et al.*, 2011) is used to examine the outcomes for the various algorithms of EDPs in this section. The Friedman test compares the algorithms' average ranks and produces Friedman statistics, where the smaller the ranking, the better the performance of the algorithm. The *p*-value is another essential term in the Friedman test; it indicates whether there is a significant difference between algorithms or not. A smaller *p*-value ($p < 0.05$) indicates greater evidence of a significant difference. Table 17 shows the results of the EDPs' Friedman test. From Table 17, we can see that the *p*-value is less than 0.05 in the majority of EDPs, which indicates that there are differences in the results obtained by all algorithms. Furthermore, the proposed H-GA–FA algorithm outperforms the compared algorithms in most EDPs, with a lower mean rank.

In addition, the Wilcoxon signed-rank test (García *et al.*, 2010) is used to show the significant differences in each design problem between the H-GA–FA and the other algorithms. It is a pairwise test that searches for the significant differences in the behavior of any two algorithms and is connected to the *p*-value. Table 18 gives the results of the EDPs' Wilcoxon signed-rank test. The total of positive ranks is $R^+$, whereas the sum of negative ranks is $R^-$. Table 18 shows that the *p*-value in some comparisons between algorithms is more than 0.05. This is because certain data were recorded insufficiently or not at all by some of the comparator algorithms. On the other hand, Table 18 shows that H-GA–FA obtains higher $R^-$ values than $R^+$ in all cases of EDPs, indicating that it is superior to other algorithms. Based on the Wilcoxon signed-rank test findings, it can be inferred that the H-GA–FA outperforms the majority of algorithms that processed EDPs.

Comparative studies were carried out in this section to assess the efficacy of the suggested hybrid algorithm solutions. Unlike traditional algorithms, the proposed H-GA–FA algorithm searches through a set of solutions rather than a single solution. in addition, H-GA–FA uses only objective function data, not derivatives or any ancillary information, enabling it to solve nonsmooth, noncontinuous, nondifferentiable problems in real-world applications.

On the other hand, it is well known that metaheuristic algorithms suffer from a lack of consistency in their solutions. So, H-GA–FA was used to improve the quality of the

**Table 16:** Overall comparison between all algorithms in solving EDPs.

| EDPs | Max. worst | Min. mean | Min. best | Min. SD | Min. NFEs |
|---|---|---|---|---|---|
| **Gear train** | MBA | **H-GA–FA** | MBA<br>UPSO<br>ABC | **H-GA–FA** | ABC |
| **Cantilever design** | WOA | ELPSO | **H-GA–FA** | ELPSO | **H-GA–FA** |
| **Pressure vessel design** | G-QPSO | **H-GA–FA** | **H-GA–FA** | **H-GA–FA** | **H-GA–FA** |
| **Welded beam design** | SC algorithm | **H-GA–FA** | **H-GA–FA** | A penalty-guided ABC | NA |
| **Tension/compression spring** | QPSO | NM-PSO | G-QPSO<br>ABC<br>TLBO<br>MBA<br>**H-GA–FA** | **H-GA–FA** | QPSO<br>G-QPSO |
| **Speed reducer design** | SC algorithm | DELC<br>DEDS<br>**H-GA–FA** | DELC<br>DEDS<br>**H-GA–FA** | $(\mu + \lambda)$-ES<br>ABC<br>TLBO<br>**H-GA–FA** | **MBA** |

*Note.* "NA" means that the result is not available.

**Table 17:** Engineering design problems' Friedman test.

**Tension/compression spring design problem**

| **Algorithm** | **MR** | | |
|---|---|---|---|
| GA3 | 16.33 | | |
| GA4 | 13.83 | | |
| CAEP | 5.50 | | |
| CPSO | 16.00 | | |
| HPSO | 14.33 | | |
| NM-PSO | 6.83 | | |
| G-QPSO | 8.00 | **Test statistics** | |
| QPSO | 6.67 | **N** | 3 |
| DELC | 7.50 | **Chi-square** | 33.991 |
| DEDS | 10.33 | **df** | 18 |
| HEAA | 7.17 | ***p*-value** | 0.013 |
| PSO-DE | 10.50 | | |
| SC algorithm | 12.33 | | |
| UPSO | 18.33 | | |
| $(\mu + \lambda)$-ES | 14.17 | | |
| ABC | 8.17 | | |
| TLBO | 5.00 | | |
| MBA | 6.00 | | |
| H-GA–FA | 3.00 | | |

**Pressure vessel design problem**

| **Algorithm** | **MR** | | |
|---|---|---|---|
| GA3 | 11.67 | | |
| GA4 | 8.50 | | |
| CPSO | 9.67 | | |
| HPSO | 7.33 | **Test statistics** | |
| NM-PSO | 4.50 | **N** | 3 |
| G-QPSO | 7.67 | **Chi-square** | 22.523 |
| PSO-DE | 4.83 | **df** | 12 |
| UPSO | 12.33 | ***p*-value** | 0.032 |
| ABC | 6.33 | | |
| $(\mu + \lambda)$-ES | 6.50 | | |
| TLBO | 5.00 | | |
| MBA | 5.67 | | |
| H-GA–FA | 1.00 | | |

**Speed reducer design problem**

| **Algorithm** | **MR** | | |
|---|---|---|---|
| SC algorithm | 9.75 | | |
| PSO-DE | 8.00 | | |
| DELC | 3.88 | **Test statistics** | |
| DEDS | 4.13 | **N** | 4 |
| HEAA | 6.75 | **Chi-square** | 17.324 |
| MDE | 7.50 | **df** | 10 |
| $(\mu + \lambda)$-ES | 5.25 | ***p*-value** | 0.067 |
| ABC | 7.50 | | |
| TLBO | 4.88 | | |
| MBA | 6.00 | | |
| H-GA–FA | 2.38 | | |

**Welded beam design problem**

| **Algorithm** | **MR** | | |
|---|---|---|---|
| Developed HS | 3.00 | | |
| A penalty-guided ABC | 4.67 | | |
| ARSAGA | 2.00 | **Test statistics** | |
| GP | 7.33 | **N** | 3 |
| SSM | 5.33 | **Chi-square** | 26.909 |
| SC algorithm | 9.67 | **df** | 10 |
| GSA | 10.67 | ***p*-value** | 0.003 |
| LXGSA | 8.00 | | |
| PMGSA | 8.33 | | |
| LXPMGSA | 6.00 | | |
| H-GA–FA | 1.00 | | |

**Gear train design problem**

| **Algorithm** | **MR** | **Test statistics** | |
|---|---|---|---|
| UPSO | 3.63 | **N** | 4 |
| ABC | 1.88 | **Chi-square** | 6.600 |
| MBA | 2.63 | **df** | 3 |
| H-GA–FA | 1.88 | ***p*-value** | 0.086 |

**Cantilever design problem**

| **Algorithm** | **MR** | **Test statistics** | |
|---|---|---|---|
| ELPSO | 1.38 | **N** | 4 |
| WOA | 4.00 | **Chi-square** | 10.385 |
| LWOA | 2.88 | **df** | 3 |
| H-GA–FA | 1.75 | ***p*-value** | 0.016 |

*Note.* "MR" means the Mean Rank.

solutions by integrating the benefits of the two metaheuristic algorithms, namely GA and FA, and making a balance between exploration/exploitation trends. FA has high exploration capabilities where all local modes will be visited, as well as global modes, and by all fireflies, good solutions are reached. GA has high exploitation capabilities where its operators (ranking, selection, crossover, and mutation) make a comprehensive change in the shape of the solutions, which makes the proposed algorithm get out of the local optima when needed.

By using unconstrained multimodal text functions, constrained benchmark problems, and engineering design challenges, the efficiency of the proposed H-GA–FA was compared to that of previous optimization techniques. The proposed H-GA–FA has displayed an exceptional execution where it performed better in terms of the best solution and the number of function evaluations and provided better solutions than other algorithms for most of the testing problems. Moreover, by comparing its results to the original GA and the original FA, H-GA–FA has been proven to improve the solutions' quality and balance between exploration/exploitation capabilities as shown in Figs 7, 9, 11, 13, 15, and 17.

Statistically according to the Wilcoxon rank-sum and Friedman rank tests (Tables 17 and 18) for EDPs, H-GA–FA is very competitive in obtaining the solution compared to other algorithms where the results of the Wilcoxon rank-sum test analysis show that the *p*-values are less than 0.05 in most of the design problems. In addition, the sum of the negative ranks ($R^-$) is greater than the sum of the positive ranks ($R^+$), which suggests that the results obtained by H-GA–FA are significantly different from the other compared algorithms. In contrast, according to the Friedman rank test results in Table 17, the rank of H-GA–FA is the least, which showed that it is superior to the compared algorithms on all EDPs.

Finally, the results demonstrated that the proposed method is capable of resolving EDPs, which are often formulated as NCOPs, and that, due to the simplicity of H-GA–FA, it can handle a wide range of applications that can be expressed as NCOPs such as design and manufacturing optimization, economic optimization of heat exchangers, conceptual design of automobile components, design optimization of highway guardrails, optimization of a vehicle engine connecting rod, design optimization of a cam-follower mechanism, optimum structural design of automobile brake components, etc.

**Table 18:** Engineering design problems' Wilcoxon signed-rank test.

| Design problem | Compared algorithms | | | | | Evaluations of solution | |
|---|---|---|---|---|---|---|---|
| **Gear train** | **Algorithm 1** | **Algorithm 2** | **R⁻** | **≈** | **R⁺** | **P-value** | **Best algorithm** |
| | H-GA–FA | UPSO | 3 | 1 | 0 | 0.108809430040546 | ≈ |
| | H-GA–FA | ABC | 2 | 1 | 1 | 1 | ≈ |
| | H-GA–FA | MBA | 3 | 1 | 1 | 0.715000654688089 | ≈ |
| **Cantilever** | **Compared algorithms** | | | | | **Evaluations of solution** | |
| | **Algorithm 1** | **Algorithm 2** | **R⁻** | **≈** | **R⁺** | **P-value** | **Best algorithm** |
| | H-GA–FA | ELPSO | 1 | 0 | 3 | 0.465208818452142 | ≈ |
| | H-GA–FA | WOA | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | LWOA | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| **Pressure vessel** | **Compared algorithms** | | | | | **Evaluations of solution** | |
| | **Algorithm 1** | **Algorithm 2** | **R⁻** | **≈** | **R⁺** | **P-value** | **Best algorithm** |
| | H-GA–FA | GA3 | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | GA4 | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | CPSO | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | HPSO | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | NM-PSO | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | G-QPSO | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | PSO-DE | 3 | 0 | 0 | 0.108809430040546 | ≈ |
| | H-GA–FA | UPSO | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | ABC | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | $(\mu + \lambda)$-ES | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | TLBO | 3 | 0 | 0 | 0.108809430040546 | ≈ |
| | H-GA–FA | MBA | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| **Welded beam** | **Compared algorithms** | | | | | **Evaluations of solution** | |
| | **Algorithm 1** | **Algorithm 2** | **R⁻** | **≈** | **R⁺** | **P-value** | **Best algorithm** |
| | H-GA–FA | Developed HS | 1 | 0 | 0 | NA | No decision |
| | H-GA–FA | A penalty-guided ABC | 3 | 0 | 1 | 0.144127034816015 | ≈ |
| | H-GA–FA | ARSAGA | 3 | 0 | 0 | 0.108809430040546 | ≈ |
| | H-GA–FA | GP | 1 | 0 | 0 | NA | No decision |
| | H-GA–FA | SSM | 3 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | SC algorithm | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | GSA | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | LXGSA | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | PMGSA | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | LXPMGSA | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| **Tension/compression spring** | **Compared algorithms** | | | | | **Evaluations of solution** | |
| | **Algorithm 1** | **Algorithm 2** | **R⁻** | **≈** | **R⁺** | **P-value** | **Best algorithm** |
| | H-GA–FA | GA3 | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | GA4 | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | CAEP | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | CPSO | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | HPSO | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | NM-PSO | 2 | 0 | 3 | 0.685830434451606 | ≈ |
| | H-GA–FA | G-QPSO | 3 | 1 | 1 | 0.715000654688089 | ≈ |
| | H-GA–FA | QPSO | 4 | 0 | 1 | 0.500184257070795 | ≈ |
| | H-GA–FA | DELC | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | DEDS | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | HEAA | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | PSO-DE | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | SC algorithm | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | UPSO | 4 | 0 | 0 | 0.067889154861829 | H-GA–FA |
| | H-GA–FA | $(\mu + \lambda)$-ES | 4 | 0 | 0 | 0.067889154861829 | ≈ |
| | H-GA–FA | ABC | 3 | 1 | 0 | 0.108809430040546 | ≈ |
| | H-GA–FA | TLBO | 2 | 1 | 0 | 0.179712494879 | ≈ |
| | H-GA–FA | MBA | 4 | 1 | 0 | 0.067889154861829 | ≈ |
| **Speed reducer** | **Compared algorithms** | | | | | **Evaluations of solution** | |
| | **Algorithm 1** | **Algorithm 2** | **R⁻** | **≈** | **R⁺** | **P-value** | **Best algorithm** |
| | H-GA–FA | SC algorithm | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | PSO-DE | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | DELC | 2 | 3 | 0 | 0.179712494879 | ≈ |
| | H-GA–FA | DEDS | 2 | 3 | 0 | 0.179712494879 | ≈ |
| | H-GA–FA | HEAA | 5 | 0 | 0 | **0.0431144467830754** | H-GA–FA |
| | H-GA–FA | MDE | 4 | 0 | 0 | 0.067889154861829 | ≈ |

**Table 18:** Continued

| Design problem | Compared algorithms | | Evaluations of solution | | | | |
|---|---|---|---|---|---|---|---|
| | H-GA–FA | $(\mu + \lambda)$-ES | 3 | 1 | 0 | 0.102470434859749 | ≈ |
| | H-GA–FA | ABC | 3 | 1 | 0 | 0.102470434859749 | ≈ |
| | H-GA–FA | TLBO | 2 | 1 | 1 | 1 | H-GA–FA |
| | H-GA–FA | MBA | 4 | 0 | 1 | 0.500184257070795 | H-GA–FA |

## 5. Conclusion

This paper presents a hybrid algorithm, called H-GA–FA, for solving EDPs that combines two optimization techniques, which are GA and FA. This hybrid technique creates individuals in a new generation by GA operations and mechanisms of FA. In addition, this combination aims to prevent falling into local optima, introduce sufficient diversity of the solutions, and make a balance between exploration and exploitation trends. The proposed hybrid algorithm H-GA–FA was tested by using a suite of 17 unconstrained multimodal test functions, 7 constrained benchmark problems taken from the literature, and many EDPs. The proposed algorithm showed several advantages, which we mention as follows:

1. H-GA–FA is a flexible and adaptive method to solve a wide range of optimization problems.
2. H-GA–FA has a high solution quality due to the combination of the advantages of the two optimization algorithms GA and FA.
3. Unlike traditional methods, H-GA–FA provides a globally optimal solution where it searches through a population of points.
4. H-GA–FA uses only the objective function information, so it can solve any practical optimization problem that may include noncontinuous, nonsmooth, and nondifferentiable functions.
5. Computational experiments have proven the superiority of H-GA–FA over those reported in the literature, as it is significantly better than other comparison methods.
6. H-GA–FA saves time where it converges more quickly to the optimal or near-optimal solution in the early iteration.
7. H-GA–FA has a low competitive computational cost, where it found the global optimum solution for most solved problems in the least number of function evaluations.
8. The use of the chaotic repair procedure enables H-GA–FA to retain the feasibility of the solutions.
9. Statistical results indicated that solutions obtained by H-GA–FA are accurate and stable than most solutions obtained by other algorithms.
10. Wilcoxon and Friedman's tests showed the significance of the H-GA–FA findings.
11. The proposed method can be used to handle large-scale engineering challenges such as resource allocation issues, cost-effective load transfer problems, unit commitment concerns, wind farm optimization of turbines, and real-time applications.

Without any prejudice, the suggested approach, like other metaheuristic algorithms, has the potential flaw of not guaranteeing an increase in computing speed or accuracy while tackling any optimization issue. Because metaheuristic algorithms are random techniques, the computational effectiveness and solution quality presented by the H-GA–FA depends on the nature and complexity of the problem. In our future works, we will concentrate on three directions: (i) developing H-GA–FA to use it in solving many-objective problems; (ii) applying H-GA–FA to solve optimization problems in different fields; and (iii) introducing new algorithms for these types of problems.

## Author contributions

All authors equally contributed to this article.

## Funding

The authors received no specific funding for this work.

## Data availability

All data used to support the findings of this study are included in the article.

## Conflict of interest statement

None declared.

## References

Abd-El-Wahed, W. F., Mousa, A. A., & El-Shorbagy, M. A. (2011). Integrating particle swarm optimization with genetic algorithms for solving nonlinear optimization problems. *Journal of Computational and Applied Mathematics*, 235, 1446–1453.

Abo-elnaga, Y., & El-Shorbagy, M. A. (2020). Multi-sine cosine algorithm for solving nonlinear bilevel programming problems. *International Journal of Computational Intelligence Systems*, *13*, 421–432.

Abualigah, L. M., Khader, A. T., Hanandeh, E. S., & Gandomi, A. H. (2017). A novel hybridization strategy for krill herd algorithm applied to clustering techniques. *Applied Soft Computing*, 60, 423–435.

Ahmadianfar, I., Heidari, A. A., Gandomi, A. H., Chu, X., & Chen, H. (2021). RUN beyond the metaphor: An efficient optimization algorithm based on Runge–Kutta method. *Expert Systems with Applications*, 181, 115079.

Akay, B., & Karaboga, D. (2012). Artificial bee colony algorithm for large-scale problems and engineering design optimization. *Journal of Intelligent Manufacturing*, 23, 1001–1014.

Alabool, H. M., Alarabiat, D., Abualigah, L., & Heidari, A. A. (2021). Harris hawks optimization: A comprehensive review of recent variants and applications. In *Neural computing and applications*(pp. 1–42).

Al Malki, A., Rizk, M. M., El-Shorbagy, M. A., & Mousa, A. A. (2016). Hybrid genetic algorithm with K-means for clustering problems. *Open Journal of Optimization*, 5, 71–83.

Alrefaei, M. H., & Diabat, A. H. (2009). A simulated annealing technique for multi-objective simulation optimization. *Applied Mathematics and Computation*, 215, 3029–3035.

Al-Thanoon, N. A., Qasim, O. S., & Algamal, Z. Y. (2019). A new hybrid firefly algorithm and particle swarm optimization for tuning parameter estimation in penalized support

vector machine with application in chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 184, 142–152.
Aydilek, I. B. (2018). A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. *Applied Soft Computing*, 66, 232–249.
Ayoub, A. Y., El-Shorbagy, M A., El-Desoky, I. M., & Mousa, A. A. (2020). Cell blood image segmentation based on genetic algorithm. In *Joint European–US Workshop on Applications of Invariance in Computer Vision*(pp. 564–573). Springer.
Beyer, H. G., & Schwefel, H P. (2002). Evolution strategies – A comprehensive introduction. *Natural Computing*, 1, 3–52.
Bodaghi, M., & Samieefar, K. (2019). Meta-heuristic bus transportation algorithm. *Iran Journal of Computer Science*, 2, 23–32. https://doi.org/10.1007/s42044-018-0025-2.
Bolaji, A. L., Al-Betar, M. A., Awadallah, M. A., Khader, A. T., & Abualigah, L. M. (2016). A comprehensive review: Krill herd algorithm (KH) and its applications. *Applied Soft Computing*, 49, 437–446
Chelouah, R., & Siarry, P. (2000). A continuous genetic algorithm designed for theglobal optimization of multimodal functions. *Journal of Heuristics*, 6, 191–213.
Chelouah, R., & Siarry, P. (2003). Genetic and Nelder–Mead algorithms hybridized for a more accurate global optimization of continuous multiminima functions. *European Journal of Operational Research*, 148, 335–348.
Chen, X., Zhou, Y., Tang, Z., & Luo, Q. (2017). A hybrid algorithm combining glowworm swarm optimization and complete 2-opt algorithm for spherical travelling salesman problems. *Applied Soft Computing*, 58, 104–114.
Cheng, M.-Y., & Prayogo, D. (2014). Symbiotic organisms search: A new metaheuristic optimization algorithm. *Computers and Structures*, 139, 98–112.
Cheng, Z., Song, H., Wang, J., Zhang, H., Chang, T., & Zhang, M. (2021). Hybrid firefly algorithm with grouping attraction for constrained optimization problem. *Knowledge-Based Systems*, 220, 106937.
Chickermane, H., & GEA, H. C. (1996). Structural optimization using a new local approximation method. *International Journal for Numerical Methods in Engineering*, 39, 829–846.
Coelho, L. D. S. (2010). Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems. *Expert Systems with Applications*, 37, 1676–1683.
Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry*, 41, 113–127.
Coello, C. A. C., & Becerra, R. L. (2004). Efficient evolutionary optimization through the use of a cultural algorithm. *Engineering Optimization*, 36, 219–236.
Coello, C. A. C., & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, 16, 193–203.
Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and Informatics*, 26, 30–45.
Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1, 3–18.
Dhiman, G., & Kumar, V. (2019). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems*, 165, 169–196. https://doi.org/10.1016/j.knosys. 2018.11.024.
Dorigo, M., & Stützle, T. (2004). *Ant colony optimization, ISBN: 978-0-262-04219-2*. MIT Press.
Ekinci, S., Hekimoğlu, B., Eker, E., & Sevim, D. (2019). Hybrid firefly and particle swarm optimization algorithm for PID controller design of buck converter. In *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*(pp. 1–6). https://doi.org/10.1109/ISMSIT.2019.8932733.
El-Desoky, I. M., El-Shorbagy, M. A., Nasr, S. M., Hendawy, Z. M., & Mousa, A. A. (2016). A hybrid genetic algorithm for job shop scheduling problems. In *International Journal of Advancement in Engineering, Technology and Computer Sciences (IJAETCS)*(Vol. 3, pp. 6–17).
El-Shorbagy, M. A. (2010). *Hybrid particle swarm algorithm for multiobjective optimization, Master of Engineering Thesis*. Menoufia University.
El-Shorbagy, M. A., & Ayoub, A. Y. (2021). Integrating grasshopper optimization algorithm with local search for solving data clustering problems. *International Journal of Computational Intelligence Systems*, 14, 783–793.
El-Shorbagy, M. A., & El-Refaey, A. M. (2020). Hybridization of grasshopper optimization algorithm with genetic algorithm for solving system of non-linear equations. *IEEE Access*, 8, 220944–220961.
El-Shorbagy, M. A., & Hassanien, A. E. (2018). Particle swarm optimization from theory to applications. *International Journal of Rough Sets and Data Analysis*, 5, 1–24.
El-Shorbagy, M. A., & Mousa, A. A. (2017). Chaotic particle swarm optimization for imprecise combined economic and emission dispatch problem. *Review of Information Engineering and Applications*, 4, 20–35.
El-Shorbagy, M. A., & Mousa, A. A. (2021). Constrained multiobjective equilibrium optimizer algorithm for solving combined economic emission dispatch problem. *Complexity*, 2021, 6672131.
El-Shorbagy, M. A., Mousa, A. A., & Fathi, W. (2011). *Hybrid particle swarm algorithm for multiobjective optimization: Integrating particle swarm optimization with genetic algorithms for multiobjective optimization*. Lambert Academic Publishing.
El-Shorbagy, M. A., Mousa, A. A., & Nasr, S. M. (2016). A chaos-based evolutionary algorithm for general nonlinear programming problems. *Chaos, Solitons and Fractals*, 85, 8–21.
El-Shorbagy, M. A., Mousa, A. A., & Farag, M. (2017). Solving nonlinear single-unit commitment problem by genetic algorithm based clustering technique. *Review of Computer Engineering Research*, 4, 11–29.
El-Shorbagy, M. A., Ayoub, A. Y., Mousa, A. A., & El-Desoky, I. M. (2019a). An enhanced genetic algorithm with new mutation for cluster analysis. *Computational Statistics*, 34, 1355–1392.
El-Shorbagy, M. A., Mousa, A. A., & Farag, M. A. (2019b). An intelligent computing technique based on a dynamic-size subpopulations for unit commitment problem. *OPSEARCH*, 56, 911–944.
El-Shorbagy, M. A., Farag, M. A., Mousa, A. A., & El-Desoky, I. M. (2020). A hybridization of sine cosine algorithm with steady state genetic algorithm for engineering design problems. In A. E. Hassanien et al. (Eds.), *Proceedings of The International Conference on Advanced Machine Learning Technologies and Applications*, AMLTA 2019, AISC 921(pp. 1–13). https://doi.org/10.1007/978-3-030-14118-9_15.
Farag, M. A., El-Shorbagy, M. A., El-Desoky, I. M., El-Sawy, A. A., & Mousa, A. A. (2015). Binary–real coded genetic

algorithm-based k-means clustering for unit commitment problem. *Applied Mathematics*, *6*, 1873–1890.
Gandomi, A. H. (2014). Interior search algorithm (ISA): A novel approach for global optimization. *ISA Transactions*, *53*, 1168–1183.
Gandomi, A. H., Yang, X.-S., & Alavi, A. H. (2013). Cuckoo search algorithm: A metaheuristic approach to solve structural optimization problems. *Engineering with Computers*, *29*, 17–35.
García, S., Fernández, A., Luengo, J., & Herrera, F. (2010). Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences*, *180*, 2044–2064.
Garg, H. (2014). Solving structural engineering design optimization problems using an artificial bee colony algorithm. *Journal of Industrial & Management Optimization*, 10, 777–794.
Ghetas, M., Yong, C. H., & Sumari, P. (2015). Harmony-based monarch butterfly optimization algorithm. In *2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*(pp. 156–161). https://doi.org/10.1109/ICCSCE.2015.7482176.
Goel, R., & Maini, R. (2018). A hybrid of ant colony and firefly algorithms (HAFA) for solving vehicle routing problems. *Journal of Computational Science*, 25, 28–37.
Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley.
Gupta, S., & Deep, K. (2020). A memory-based Grey wolf optimizer for global optimization tasks. *Applied Soft Computing*, *93*, 106367. https://doi.org/10.1016/j.asoc.2020.106367.
He, Q., & Wang, L. (2006). An effective co-evolutionary particle swarm optimization for engineering optimization problems. *Engineering Applications of Artificial Intelligence*, *20*, 89–99.
He, Q., & Wang, L. (2007). A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization. *Applied Mathematics and Computation*, *186*, 1407–1422.
Holland, J. H. (1975). *Adaptation in natural and artificial systems*(1st ed.). MIT Press.
Hwang, S. F., & He, R. S. (2006). A hybrid real-parameter genetic algorithm for function optimization. *Advanced Engineering Informatics*, *20*, 7–21.
Jadon, S. S., Tiwari, R., Sharma, H., & Bansal, J. C. (2017). Hybrid artificial bee colony algorithm with differential evolution. *Applied Soft Computing*, 58, 11–24.
Jordehi, A. R. (2015). Enhanced leader PSO (ELPSO), a new PSO variant for solving global optimization problems. *Applied Soft Computing Journal*, *26*, 401–417.
Kannan, B., & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design*, *116*, 405–411.
Kao, Y., & Zahara, E. (2008). A hybrid genetic algorithm and particle swarm optimization for multimodal functions. *Applied Soft Computing*, *8*, 849–857.
Karaboga, D. (2005). *An idea based on honey bee swarm for numerical optimization, Technical report-TR06*. Erciyes University, Engineering Faculty, Computer Engineering Department.
Kaur, S., Awasthi, L. K., Sangal, A. L., & Dhiman, G. (2020). Tunicate swarm algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Engineering Applications of Artificial Intelligence*, *90*, 103541. https://doi.org/10. 1016/j.engappai.2020.103541.
Kaushik, K., & Arora, V. (2015). A hybrid data clustering using firefly algorithm based improved genetic algorithm. *Procedia Computer Science*, 58, 249–256.
Kumar, M., Kulkarni, A. J., & Satapathy, S. C. (2018). Socio evolution & learning optimization algorithm: A socioinspired optimization methodology. *Future Generation Computer Systems*, *81*, 252–272. https://doi.org/10.1016/j. future.2017.10.052.
Le, D. T., Bui, D.-K., Ngo, T. D., Nguyen, Q.-H., & Nguyen-Xuan, H. (2019). A novel hybrid method combining electromagnetism-like mechanism and firefly algorithms for constrained design optimization of discrete truss structures. *Computers & Structures*, *212*, 20–42.
Lee, K. S., & Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, *194*, 3902–3933.
Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems*, *111*, 300–323.
Lieu, Q. X., Do, D. T.T., & Lee, J. (2018). An adaptive hybrid evolutionary firefly algorithm for shape and size optimization of truss structures with frequency constraints. *Computers & Structures*, *195*, 99–112.
Liu, H., Cai, Z., & Wang, Y. (2010). Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization. *Applied Soft Computing*, *10*, 629–640.
Marichelvam, M. K., Tosun, Ö., & Geetha, M. (2017). Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time. *Applied Soft Computing*, 55, 82–92.
Marinaki, M., & Marinakis, Y. (2016). A glowworm swarm optimization algorithm for the vehicle routing problem with stochastic demands. *Expert Systems with Applications*, *46*, 145–163.
Mehta, V. K., & Dasgupta, B. (2012). A constrained optimization algorithm based on the simplex search method. *Engineering Optimization*, *44*, 537–550.
Mezura-Montes, E., & Coello, C. A. C. (2005). Useful infeasible solutions in engineering optimization with evolutionary algorithms. In *MICAI 2005: Advances in Artificial Intelligence, 4th Mexican International Conference on Artificial Intelligence*(Vol. *3789*, pp. 652–662). http://dx.doi.org/10.1007/11579427 66.
Mezura-Montes, E., Velazquez-Reyes, J., & Coello, C. A. C. (2006). Modified differential evolution for constrained optimization. In *2006 IEEE International Conference on Evolutionary Computation*(pp. 25–32). IEEE.
Michael, B.-B. (2008). *Nonlinear optimization with engineering applications (Springer Optimization and Its Applications Book Series)*. Springer-Verlag.
Michalewicz, Z. (1994). Evolutionary computation techniques for nonlinear programming problems. *International Transactions in Operational Research*, *1*, 223–240.
Mirjalili, S. (2015). The ant lion optimizer. *Advances in Engineering Software*, *83*, 80–98.
Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67.
Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremia, S., Farisd, H., & Mirjalilie, S. M. (2017). Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, *114*, 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002.
Mousa, A. A., El-Shorbagy, M. A., & Farag, M. A. (2020). Steady-state sine cosine genetic algorithm-based chaotic search for nonlinear programming and engineering applications. *IEEE Access*, *8*, 212036–212054. doi:10.1109/ACCESS.2020.3039882.

Mousa, A. A., El-Shorbagy, M. A., Mustafa, I., & Alotaibi, H. (2021). Chaotic search-based equilibrium optimizer for dealing with nonlinear programming and petrochemical application. *Processes*, *9*, 200.
Nama, S., Sharma, S., Saha, A. K., & Gandomi, A H. (2021). A quantum mutation-based backtracking search algorithm. *Artificial Intelligence Review*, 1–55.
Nasr, S. M., El-Shorbagy, M. A., El-Desoky, I. M., Hendawy, Z. M., & Mousa, A. A. (2015). Hybrid genetic algorithm for constrained nonlinear optimization problems. *British Journal of Mathematics & Computer Science*, *7*, 466–480.
Onwubolu, G. C., & Babu, B. V. (2004). *New optimization techniques in engineering*(Vol. *141*). Springer Science & Business Media.
Parouha, R. P., & Das, K. N. (2016). A memory-based differential evolution algorithm for unconstrained optimization. *Applied Soft Computing*, *38*, 501–517.
Parsopoulos, K., & Vrahatis, M. (2005). Unified particle swarm optimization for solving constrained engineering optimization problems. In *Advances in Natural Computation, First International Conference, ICNC 2005*(Vol. *3612*, pp. 582–591). Springer-Verlag.
Passino, K. M. (2002). Biomimicry of bacteria foraging for distributed optimization and control. *IEEE Control Systems Magazine*, *22*, 52–67.
Ragsdell, K. M., & Phillips, D. T. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of Engineering for Industry*, *98*, 1021–1025.
Rao, S. S. (2009). *Engineering optimization: Theory and practice*(3rd ed.). Wiley.
Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*, 303–315.
Rashedi, E., Nezamabadi-pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences*, *179*, 2232–2248. https://doi.org/10.1016/j.ins.2009.03.004.
Ray, T., & Liew, K. M. (2003). Society and civilization: An optimization algorithm based on the simulation of social behavior. *IEEE Transactions on Evolutionary Computation*, *7*, 386–396.
Sadollah, A., Bahreininejad, A., Eskandar, H., & Hamdi, M. (2013). Mine blast algorithm: A new population-based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing*, *13*, 2592–2612.
Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper optimisation algorithm: Theory and application. *Advances in Engineering Software*, *105*, 30–47.
Saurabh, P., & Verma, B. (2016). An efficient proactive artificial immune system-based anomaly detection and prevention system. *Expert Systems with Applications*, *60*, 311–320.
Sedlaczek, K., & Eberhard, P. (2005). Constrained particle swarm optimization of mechanical systems. In *6th World Congresses of Structural and Multidisciplinary Optimization*.
Sharma, M., Singh, G., & Singh, R. (2021). Clinical decision support system query optimizer using hybrid firefly and controlled genetic algorithm. *Journal of King Saud University-Computer and Information Sciences*, *33*, 798–809.
Shi, Y. (2011). Brain storm optimization algorithm. In *Advances in swarm intelligence, ICSI 2011, Lecture Notes in Computer Science*(pp. 303–309).
Singh, A., & Deep, K. (2017). Hybridizing gravitational search algorithm with real coded genetic algorithms for structural engineering design problem. *OPSEARCH*, *54*, 505–536.
Turanoğlu, B., & Akkaya, G. (2018). A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem. *Expert Systems with Applications*, *98*, 93–104.
Verma, S., & Mukherjee, V. (2016). Firefly algorithm for congestion management in deregulated environment. *Engineering Science and Technology*, *19*, 1254–1265.
Wang, G.-G. (2018). Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Computing*, *10*, 151–164.
Wang, L., & Li, L. P. (2010). An effective differential evolution with level comparison for constrained engineering design. *Structural and Multidisciplinary Optimization*, *41*, 947–963.
Wang, Y., Cai, Z., Zhou, Y., & Fan, Z. (2009). Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint handling technique. *Structural and Multidisciplinary Optimization*, *37*, 395–413.
Wang, J., Yang, W., Du, P., & Niu, T. (2018). A novel hybrid forecasting system of wind speed based on a newly developed multi-objective sine cosine algorithm. *Energy Conversion and Management*, *163*, 134–150.
Wu, S.-J., & Chow, P.-T. (1995). Genetic algorithms for nonlinear mixed discrete-integer optimization problems via metagenetic parameter optimization. *Engineering Optimization*, *24*, 137–159.
Yang, X. S. (2008). *Nature-inspired metaheuristic algorithms, ISBN 1-905986-10-6*. Luniver Press.
Yang, Y., Chen, H., Heidari, A. A., & Gandomi, A. H. (2021). Hunger games search: Visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. *Expert Systems with Applications*, *177*, 114864.
Zahara, E., & Kao, Y. T. (2009). Hybrid Nelder–Mead simplex search and particle swarm optimization for constrained engineering design problems. *Expert Systems with Applications*, *36*, 3880–3886.
Zhang, M., Luo, W., & Wang, X. (2008). Differential evolution with dynamic stochastic selection for constrained optimization. *Information Sciences*, *178*, 3043–3074.
Zhao, W., & Wang, L. (2016). An effective bacterial foraging optimizer for global optimization. *Information Sciences*, *329*, 719–735.
Zhao, J., Tang, D., Liu, Z., Cai, Y., & Dong, S. (2020). Spherical search optimizer: A simple yet efficient meta-heuristic approach. *Neural Computing and Applications*, *32*, 9777–9808. https://doi.org/10.1007/s00521-019-04510-4.
Zhou, A. (2005). A genetic-algorithm-based neural network approach for short-term traffic flow forecasting. *Advances in Neural Networks*, *3498*, 965–969.
Zhou, Y., Chen, X., & Zhou, G. (2016). An improved monkey algorithm for a 0–1 knapsack problem. *Applied Soft Computing*, *38*, 817–830.
Zhou, Y., Ling, Y., & Luo, Q. (2018). Lévy flight trajectory-based whale optimization algorithm for engineering optimization. *Engineering Computations*, *35*, 2406–2428.
Zhuoran, Z., Changqiang, H., Hanqiao, H., Shangqin, T., & Kangsheng, D. (2018). An optimization method: Hummingbirds optimization algorithm. *Journal of Systems Engineering and Electronics*, *29*, 386–404.

## Appendix 1.

### A.1. Branin (RC) (two variables)

$$RC(x_1, x_2) = \left(x_2 - \left(\frac{5}{4\pi^2}\right)x_1^2 + \left(\frac{5}{\pi}\right)x_1 - 6\right)^2 + 10\left(1 - \left(\frac{1}{8\pi}\right)\right)\cos(x_1) + 10$$

Search domain: $-5 < x_1 < 10$, $0 < x_2 < 15$.

No local minimum; three global minima: $(x_1, x_2)^* = (-\pi, 12.275)$, $(\pi, 2.275)$, $(9.42478, 2.475)$; $RC[(x_1, x_2)^*] = 0.397887$.

### A.2. B2 (two variables)

$$B_2(x_1, x_2) = x_1^2 + 2x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7;$$

Search domain: $-100 < x_j < 100$, $j = 1, 2$.

Several local minima (exact number unspecified in usual literature); one global minimum: $(x_1, x_2)^* = (0, 0)$; $B2[(x_1, x_2)^*] = 0$.

### A.3. Easom (ES) (two variables)

$$ES(x_1, x_2) = -\cos(x_1)\cos(x_2)\exp(-((x_1 - \pi)^2 + (x_2 - \pi)^2));$$

Search domain: $-100 < x_j < 100$, $j = 1, 2$.

Several local minima (exact number unspecified in usual literature); one global minimum: $(x_1, x_2)^* = (\pi, \pi)$; $ES[(x_1, x_2)^*] = -1$.

### A.4. Goldstein and Price (GP) (two variables)

$$GP(x_1, x_2) = \left[1 + (x_1 + x_2 + 1)^2 \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right];$$

Search domain: $-2 < x_j < 2$, $j = 1, 2$.

Four local minima; one global minimum: $(x_1, x_2)^* = (0, -1)$; $B2[(x_1, x_2)^*] = 0$.

### A.5 Shubert (SH) (two variables)

$$SH(x_1, x_2) = \left\{\sum_{j=1}^{5} j\cos[(j+1)x_1 + j]\right\} \times \left\{\sum_{j=1}^{5} j\cos[(j+1)x_2 + j]\right\};$$

Search domain: $-10 < x_j < 10$, $j = 1, 2$.

760 local minima; 18 global minima: $SH[(x_1, x_2)^*] = -186.7309$.

### A.6. De Joung (DJ) (three variables)

$$DJ(x_1, x_2, x_3) = x_1^2 + x_2^2 + x_3^2;$$

Search domain: $-5.12 < x_j < 5.12$, $j = 1, 2, 3$.

One single minimum (local and global): $(x_1, x_2, x_3)^* = (0, 0, 0)$; $DJ[(x_1, x_2, x_3)^*] = 0$.

### A.7 Hartmann ($H_{3,4}$) (three variables)

$$H_{3,4}(X) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right];$$

Search domain: $0 < x_j < 1$, $j = 1, 2, 3$.

Four local minima; one global minimum: $\mathbf{x}^* = (0.11, 0.555, 0.855)$; $H_{3,4}(\mathbf{x}) = -3.86278$.

| $i$ | $a_{ij}$ | | | $c_i$ | $p_{ij}$ | | |
|---|---|---|---|---|---|---|---|
| 1 | 3.0 | 10.0 | 30.0 | 1.0 | 0.3689 | 0.1170 | 0.2673 |
| 2 | 0.1 | 10.0 | 35.0 | 1.2 | 0.4699 | 0.4387 | 0.7470 |
| 3 | 3.0 | 10.0 | 30.0 | 3.0 | 0.1091 | 0.8732 | 0.5547 |
| 4 | 0.1 | 10.0 | 35.0 | 3.2 | 0.0381 | 0.5743 | 0.8827 |

### A.8. Hartmann ($H_{6,4}$) (six variables)

$$H_{6,4}(X) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right];$$

Search domain: $0 < x_j < 1$, $j = 1, \ldots, 6$.

Four local minima; one global minimum: $\mathbf{x}^* = (0.20169, 0.150011, 0.47687, 0.275332, 0.311652, 0.6573)$; $H_{6,4}(\mathbf{x}) = -3.32237$.

### A.9. Shekel ($S_{4,n}$) (four variables)

$$S_{4,n}(X) = -\sum_{i=1}^{n}\left[(x - a_i)^T(x - a_i) + c_i\right]^{-1};$$

$$X = (x_1, x_2, x_3, x_4)^T;$$

$$a = (a_i^1, a_i^2, a_i^3, a_i^4)^T;$$

Three functions $S_{4,n}$ were considered: $S_{4,5}$, $S_{4,7}$, and $S_{4,10}$.

Search domain: $0 < x_j < 10$; $j = 1, \ldots, 4$.

$n$ local minima ($n = 5$, 7, or 10): $a_i^T = i$-th local minimum approximation; $S_{4,n}(a_i^T) \cong -1/c_i$;

$S_{4,5}$: Five minima with one global minimum; $S_{4,5}(\mathbf{x}) = -10.1532$.

$S_{4,7}$: Seven minima with one global minimum; $S_{4,7}(\mathbf{x}) = -10.40294$.

$S_{4,10}$: Ten minima with one global minimum; $S_{4,10}(\mathbf{x}) = -10.53641$.

### A.10 Rosenbrock ($R_n$) ($n$ variables)

$$R_n(x) = \sum_{j=1}^{n-1}\left[100(x_j^2 - x_{j+1})^2 + (x_j - 1)^2\right];$$

Three functions were considered: $R_2$, $R_5$, and $R_{10}$.

Search domain: $-5 < x_j < 10$, $j = 1, \ldots, n$.

Several local minima; one global minimum: $\mathbf{x}^* = (1, \ldots, 1)$; $R_n(\mathbf{x}^*) = 0$.

### A.11. Zakharov ($Z_n$) ($n$ variables)

$$Z_n(x) = \left(\sum_{j=1}^{n} x_j^2\right) + \left(\sum_{j=1}^{n} 0.5\,j\,x_j\right)^2 + \left(\sum_{j=1}^{n} 0.5\,j\,x_j\right)^4$$

Two functions were considered: $Z_2$ and $Z_5$.

Search domain: $-5 < x_j < 10$, $j = 1, \ldots, n$.

Several local minima; one global minimum: $\mathbf{x}^* = (0, \ldots, 0)$; $Z_n(\mathbf{x}^*) = 0$.

## Appendix 2.

### B.1. Constrained problem 1 (C1)

$$\begin{aligned} \text{Min} \quad & x_1^2 + x_2^2 \\ \text{Subject to:} \quad & x_1 - 3 = 0 \\ & -x_2 + 2 \le 0 \\ & -10 \le x_i \le 10, \quad i = 1, 2 \end{aligned}$$

| $i$ | $a_{ij}$ | | | | | | $c_i$ | $p_{ij}$ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 10.0 | 3.00 | 17.0 | 3.50 | 1.70 | 8.00 | 1.0 | 0.1312 | 0.1696 | 0.5569 | 0.0124 | 0.8283 | 0.5886 |
| 2 | 0.05 | 10.0 | 17.0 | 0.10 | 8.00 | 14.0 | 1.2 | 0.2329 | 0.4135 | 0.8307 | 0.3736 | 0.1004 | 0.9991 |
| 3 | 3.00 | 3.50 | 1.70 | 10.0 | 17.0 | 8.00 | 3.0 | 0.2348 | 0.1451 | 0.3522 | 0.2883 | 0.3047 | 0.6650 |
| 4 | 17.0 | 8.00 | 0.05 | 10.0 | 0.10 | 14.0 | 3.2 | 0.4047 | 0.8828 | 0.8732 | 0.5743 | 0.1091 | 0.0381 |

| $i$ | $a_{ij}$ | | | | $c_i$ |
|---|---|---|---|---|---|
| 1 | 4.0 | 4.0 | 4.0 | 4.0 | 0.1 |
| 2 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 |
| 3 | 8.0 | 8.0 | 8.0 | 8.0 | 0.2 |
| 4 | 6.0 | 6.0 | 6.0 | 6.0 | 0.4 |
| 5 | 3.0 | 7.0 | 3.0 | 7.0 | 0.4 |
| 6 | 2.0 | 9.0 | 2.0 | 9.0 | 0.6 |
| 7 | 5.0 | 5.0 | 3.0 | 3.0 | 0.3 |
| 8 | 8.0 | 1.0 | 8.0 | 1.0 | 0.7 |
| 9 | 6.0 | 2.0 | 6.0 | 2.0 | 0.5 |
| 10 | 7.0 | 3.6 | 7.0 | 3.6 | 0.5 |

## B.2. Constrained problem 2 (C2)

$$\begin{aligned}
&\text{Min} && \frac{1}{4000}(x_1^2 + x_2^2) - \cos\left(\frac{x_1}{\sqrt{1}}\right)\cos\left(\frac{x_2}{\sqrt{2}}\right) + 1 \\
&\text{Subject to :} && x_1 - 3 = 0 \\
& && -x_2 + 2 \le 0 \\
& && -10 \le x_i \le 10, \quad i = 1, 2
\end{aligned}$$

## B.3. Constrained problem 3 (C3)

$$\begin{aligned}
&\text{Min} && \frac{-\sin(2\pi x_1)^3 \sin(2\pi x_1)}{x_1^3(x_2 + x_1)} \\
&\text{Subject to :} && -x_1 + (x_2 - 4)^2 + 1 \le 0 \\
& && x_1^2 - x_2 + 1 \le 0 \\
& && 0.1 \le x_1 \le 10, \quad 0 \le x_2 \le 10
\end{aligned}$$

## B.4. Constrained problem 4 (C4)

$$\begin{aligned}
&\text{Min} && (x_1 - 10)^3 + (x_2 - 20)^3 \\
&\text{Subject to :} && -(x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0 \\
& && -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0 \\
& && 13 \le x_1 \le 10, \quad 0 \le x_2 \le 100
\end{aligned}$$

## B.5. Constrained problem 5 (C5)

$$\begin{aligned}
&\text{Min} && x_1^2 + (x_2 - 1)^2 \\
&\text{Subject to :} && -x_1^2 + x_2 = 0 \\
& && -1 \le x_i \le 1, \quad i = 1, 2
\end{aligned}$$

## B.6. Constrained problem 6 (C6)

$$\begin{aligned}
&\text{Min} && (\sqrt{n})^n \prod_{i=1}^{n} x_i \\
&\text{Subject to :} && \sum_{i=1}^{n} x_i^2 - 1 = 0 \\
& && 0 \le x_i \le 1, \quad i = 1, 2, \ldots, 4
\end{aligned}$$

## B.7. Constrained problem 7 (C7)

$$\begin{aligned}
&\text{Min} && 5.357857x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \\
&\text{Subject to :} && -85.334407 - 0.0006262x_1x_4 - 0.0056858x_2x_5 + 0.0022053x_3x_5 \le 0 \\
& && 85.334407 + 0.0006262x_1x_4 + 0.0056858x_2x_5 - 0.0022053x_3x_5 - 92 \le 0 \\
& && -80.51249 - 0.0029955x_1x_2 - 0.0071317x_2x_5 - 0.0021813x_3^2 + 90 \le 0 \\
& && 80.51249 + 0.0029955x_1x_2 + 0.0071317x_2x_5 + 0.0021813x_3^2 - 110 \le 0 \\
& && -9.300961 - 0.0012547x_1x_3 - 0.0047026x_3x_5 - 0.0019085x_3x_4 + 20 \le 0 \\
& && 9.300961 + 0.0012547x_1x_3 + 0.0047026x_3x_5 + 0.0019085x_3x_4 - 25 \le 0 \\
& && 8 \le x_1 \le 100, \\
& && 3 \le x_2 \le 45, \\
& && 27 \le x_3 \le 45, \\
& && 27 \le x_4 \le 45, \\
& && 27 \le x_5 \le 45.
\end{aligned}$$