Received November 15, 2020, accepted December 3, 2020, date of publication December 7, 2020, date of current version December 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043029

Hybridization of Grasshopper Optimization Algorithm With Genetic Algorithm for Solving System of Non-Linear Equations

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, Shibin El Kom 32511, Egypt
[3]Department of Basic and Applied Science, College of Engineering and Technology, Arab Academy for Science, Technology & Maritime Transport, Smart Village Campus, Cairo 125077, Egypt

Corresponding author: M. A. El-Shorbagy (mohammed_shorbagy@yahoo.com)

ABSTRACT A novel algorithm for optimization in this article, called hybrid grasshopper optimization algorithm (GOA) with genetic algorithm (GA): hybrid-GOA-GA, is proposed for solving the system of non-linear equations (SNLEs). First, the SNLEs are converted into an optimization problem. Then, the optimization problem is solved by hybrid-GOA-GA. In the hybrid-GOA-GA, a population of randomized solutions is initialized. These solutions, by GOA, are looking for an optimal solution for SNLE in the domain of optimization problem. During this process, the evolution of these solutions is carried out by GA. Hybrid-GOA-GA integrates the merits of both GOA and GA; where GOA's exploitability and GOA's exploration potential are combined. Furthermore, hybrid-GOA-GA has good capability for escaping from local optima with faster convergence. The hybrid-GOA-GA has been tested by eight benchmarks problems which include different applications. Additionally, the effect of changing the initial intervals of the variables on the efficiency of the proposed algorithm is discussed. Also, the computational cost of the proposed algorithm is studied and compared with other methods. The results show that the hybrid-GOA-GA algorithm is superior to other algorithms, and return the best solution of SNLEs. Finally, in terms of accuracy, the effect of changing initial intervals and computational cost, the proposed approach is competitive and better in most benchmark problems compared to other methods. So, we can say that hybrid-GOA-GA is effective to solve a SNLEs.

INDEX TERMS Grasshopper optimization algorithm, genetic algorithm, system of non-linear equations, hybrid algorithm, optimization.

I. INTRODUCTION

The system of non-linear equations (SNLEs) is known as the basis for many engineering and scientific models, and their efficient solution is very important to achieve progress in these fields. SNLEs appears directly in some applications, but they may also appear indirectly from the transformation of practical models into SNLEs [1], [2]. Theoretically, finding an effective and robust solution for the SNLEs can be a very challenging problem.

The single nonlinear equation solution is an easy process but, solving a set of nonlinear equations is very difficult. SNLEs traditionally solved by bisection method, false-position method, Muller's method, Levenberg–Marquardt algorithm, the steepest descent methods, Broyden method, Halley's method, branch and prune approach, Newton method, damped Newton methods, and Secant method [3]–[6]. In general, secant and Newton are the recommended methods to solve the system of nonlinear equations. On the other hand, some methods convert the SNLEs into a single optimization problem [7], [8], then used the augmented Lagrangian method to solve it [9]. These methods are very time-consuming, might be diverged, not efficient to solve a set of nonlinear equations, need a tedious task to calculate the partial derivatives to form the Jacobian matrix and sensitivity to initial guess [10].

Due to these limitations, the researchers turned to the population based approaches (PBAs) to solve SNLEs. PBAs

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Shen.

can be described as the collective behavior that emerged from social insects working under very few rules. The famous algorithms of PBAs come from the world of animals, such as fish school, birds flock and bugs swarm. In addition, PBAs are considered as computational models simulating natural swarm systems. Until now, many PBAs have been proposed in the literature and have been successfully applied to solve optimization problems. Examples of PBAs models are: genetic algorithm (GA) [11], [12], ant colony optimization (ACO) [13], particle swarm optimization (PSO) [14]–[16], artificial bee colony (ABC) [17], bacterial foraging (BF) [18], cat swarm optimization (CSO) [19], glowworm swarm optimization (GSO) [20], firefly algorithm (FA) [21], monkey algorithm (MA) [22], krill herd algorithm (KHA) [23], sine cosine algorithm (SCA) [24] and grasshopper optimization algorithm (GOA) [25], cuckoo search algorithm (CSA) [26], salp swarm algorithm (SSA) [27], gradient-based optimizer (GBO) [28], slime mould algorithm (SMA) [29], and harris hawks optimization (HHO) [30], etc.

There are many PBAs [31]–[38] that were used to solve SNLEs such as GA, PSO, ABC, CSA and FA. In [31] Chang proposed an improved real-coded GA for parameters estimation of the nonlinear system. In [32] the authors proposed a new perspective for the solution of complex SNLEs by introducing them as a multiobjective optimization problem and resolving it via GA. While, in [33] SNLEs are solved by an efficient GA with harmonious and symmetric individuals. In [34], Mo and Liu proposed conjugate direction to PSO for the solving of SNLEs, which would incorporate conjugate direction method (CDM) into PSO in order to improve PSO and enable high-dimensional optimization problems to be optimized effectively. CDM also helps PSO overcome local minima by shifting the problem of high-dimensional function optimization to low-dimensional function optimization. In [35] Jaberipour *et al.* proposed a new version of PSO for solving SNLEs. They modified the way of updating each particle to overcome the problems of the basic PSO method such as trapping in local minima and slowing convergence. In addition, In [36], Jia and He presented a hybrid algorithm for solving SNLEs through combining ABC algorithm and PSO together. The principle of the hybrid algorithm is using the advantages of both algorithms to ameliorate the defect of slumping into premature or into local optimum. Furthermore, In [37], Zhou and Li proposed an improvement CSA, to solve the SNLEs. They used a new encoding method which guarantees that the obtained solution is a feasible solution without needing to modify the evolution of cuckoo. Finally, in [38], Ariyaratne *et al.* presented an amended FA which treats the SNLEs as an optimization problem, where initial assumptions, differentiation, and even function continuity were not required and it is capable of providing several root approximations at the same time.

In order to improve the productivity of the solutions, benefit from their advantages, and correct any weaknesses, the majority of researchers suggest integration between PBAs such as: hybrid PSO [39], [40], hybrid ACO [41], hybrid GA [42], [43], hybrid ABC [44], hybrid BF [45], hybrid CSO [46], hybrid GSO [47], hybrid FA [48], hybrid MA [49], hybrid KHA [50], hybrid SCA [51], and etc.

Grasshopper optimization algorithm (GOA) is one of the novel PBAs which is based on the swarm nature of grasshoppers. It mainly depends on the forces of social interaction to find the global optimum values of the optimization problem. Due to its easy deployment and high accuracy, robustness and effectiveness, it is widely used in a variety of optimization problems. But, GOA has some limitations such as: 1) unbalancing between the processes of exploitation and exploration; 2) convergence speed is unstable; and 3) may be fall into the local optimum. So, there are several hybrid algorithms between GOA and other PBAs have been proposed in the literature [52]–[59]. In [52], the authors proposed a dynamic population quantum binary GOA based on shared knowledge and a rough set theory for the selection of features; where GOA was improved by the quantum method. The search scope was improved by the quantification of grasshopper individuals and a good balance between exploitation and exploration was achieved. In addition, premature and catastrophe methods have also been used to avoid premature convergence and failure to get an optimal position. While, In [53], Singh and Prakashthe optimized multiple optical network units (ONUs) placement in Fiber-Wireless (FiWi) access network using two recent optimization algorithms, HHO and GOA. Dwivedi *et al.* proposed in [54] a new technique for the selection of features by combining the feature Selection Ensemble and the chaotic adaptive GOA. The chaos principle in GOA has been used to create a uniformly distributed population to boost the efficiency of the initial populations, control the capacity to look for new space referred to as exploration, and use existing space referred to as exploitation in the optimization process. In addition, Raeesi *et al.* (55) proposed an inverse magneto-rheological (MR) damping model Takagi-Sugeno-Kang for the control of non-linear vibrations using enhanced GOA, which improved by incorporating opposition-based learning and merit function methods to enhance its exploration and exploitation capabilities. In [56], the author proposed a new model to reliably forecast the monthly volatilization of the price of iron ore. This model integrates chaotic behavior into GOA in order to form a new GOA called a chaotic GOA that is used to learn the neural network (NN) of multi-layer perceptions as a trainer. Purushothaman *et al.* [57] also suggested a hybrid algorithm, between GWO and GOA, for text selection and clustering; where it has a reasonable convergence rate and a minimum computational time. In [58] Ewees *et al.* suggest an improved GOA approach based on opposition-based learning (OBL) for the resolution of benchmarking optimization functions and engineering problems, where OBL is used to reduce time complexity by applying it to half of the solutions. Finally, in [59], Alphonsa and Sundaram have used a hybrid algorithm for the data sanitation and restoration process, known as the grasshopper optimization with GA. These hybrid algorithms

[52]–[59] aim to enhance GOA by generating a uniformly distributed population, improving the search scope, achieving good balance between exploration (the ability to search for new space) and exploitation (use existing space), producing a mature convergence rate or reducing the computational time.

On the other hand, GA is one of the PBAs that introduced in 1975 by Holland [60], and described in 1989 by Goldberg [61] as a proficient global technique for solving complex optimization problems based on the technicalities of natural selection, evolution, and genetics. GA is well suitable for solving optimization problems and has still garners great attention by researchers [62]–[65]. From the literature we can see that GA was widely applied to solve SNLEs [66]–[72]. However, GA when dealing with complex and large systems has many disadvantages such as: extreme slow and the difficulty in ensuring access to the global optimum solution as it requires an increase in the number of iterations (i.e. long search time). Hence, it is suggested that the implementation of GOA and GA in hybrid form results in superior characteristics; where GA with its operators (ranking, selection, crossover and mutation) has good exploitation ability. While, GOA updates the location of each agent on the basis of their actual position, the global best position, and the locations of all other search agents. This mean that GOA has good exploration ability. From this motivation, this study makes the following contributions:

1. Proposing a novel hybrid-GOA-GA algorithm for solving system of non-linear equations (SNLEs).
2. The proposed algorithm includes the key merits of the autonomous GOA and GA and integrates the GA exploitation and GOA exploration capabilities.
3. The novel hybrid algorithm is realized on eight benchmarks problems that includes different applications.
4. The effect of changing the initial intervals of the variables on the efficiency of the proposed algorithm is discussed.
5. The Computational cost of the proposed hybrid algorithm is investigated and compared with other techniques.
6. The outcomes show that the proposed hybrid-GOA-GA is superior to other algorithms and that it provides the best solution for SNLEs.
7. With the effective performance characteristics on the benchmarks problems, in terms of accuracy, effect of changing initial intervals and computational cost, the use of the hybrid algorithm is recommended for solving system of non-linear equations (SNLEs).

This article is organized as follows: Non-linear system of equations are described in Section II. In Section III The proposed methodology is presented. Section IV shows the numerical results. Discussions about the results are introduced in section V. Remarks and conclusions are given in the concluding Section VI.

II. NON-LINEAR SYSTEM OF EQUATIONS

The system of non-linear equations (SNLEs) are defined mathematically as:

$$\text{SNLE} = \begin{cases} f_1(x) = 0 \\ f_2(x) = 0 \\ \vdots \\ f_N(x) = 0; \end{cases} \tag{1}$$

where each function f_i $\forall i = 1, \ldots, N$ is a nonlinear function, which maps the vector $x = (x_1, x_2, \ldots, x_n)^T$ of the n-dimensional space R^n to real line. May be some of the functions are nonlinear and others linear. The solution for SNLEs involves finding solution so that each of the function above f_i $\forall i = 1, \ldots, N$ is equal to zero.

There are many methods that convert the SNLEs ($f_i = 0,\ i = 1, \ldots, N$) to an optimization problem. The first technique is used by in Nie [7], [73]; where the SNLEs ($f_i = 0,\ i = 1, \ldots, N$) is transformed into a constrained optimization problem. The original equations is divided into two groups S_1 and S_2; S_1 contains the equations that form the objective function and S_2 contains the equations that used as equality constraints. In addition, the two groups are updated due to the optimization process in every step. Then, the constrained optimization problem can be written as:

$$\begin{aligned} F(x) &= \min \sum_{i \in S_1} f_i^2(x) \\ &\text{Subject to: } f_j(x) = 0, \quad j \in S_2 \end{aligned} \tag{2}$$

Secondly, the SNLEs are converted to a multi-objective optimization problem [32], [74] as follow:

$$\begin{cases} f_1(x) = 0 \\ f_2(x) = 0 \\ \vdots \\ f_N(x) = 0 \end{cases} \longrightarrow \begin{cases} F_1 = \min \operatorname{abs}(f_1(x)) = 0 \\ F_2 = \min \operatorname{abs}(f_2(x))) = 0 \\ \vdots \\ F_N = \min \operatorname{abs}(f_N(x)) = 0 \end{cases} \tag{3}$$

This function (Eq. (3)) is aimed to minimize the difference in the absolute value of the equations between the left and the right sides. This technique has no additional constraints are required and has the capability to find the solutions even for large-scale SNLEs. But, the functions values ($f_i = 0, i = 1, \ldots, N$) is not sufficiently close to zero.

Finally, with the inclusion of the left side of all equations and the use of the absolute value function, the SNLEs is transformed into a constrained optimization problem as follows [71]:

$$\begin{aligned} F(x) &= \operatorname{abs}(f_1(x) + f_2(x) + \cdots + f_N(x)) \\ &\text{Subject to: } \begin{cases} f_1(x) = 0 \\ f_2(x) = 0 \\ \vdots \\ f_N(x) = 0 \end{cases} \end{aligned} \tag{4}$$

The definition of this objective function is quite different from that proposed by Grosan and Abraham [32].

The objective function in Eq. (4) has a global minimum if all the nonlinear equations equal to zero ($f_i = 0 \;\; \forall i = 1, \ldots, N$).

III. PROPOSED METHODOLOGY

This section introduces a brief description of GA and GOA. Then, the hybrid version between them is presented in detail.

A. GENETIC ALGORITHM

Holland introduced the genetic algorithm (GA) and Goldberg defined it in 1975 and 1989, respectively [60], [61] as an optimization technique. It starts with a group of chromosomes (solutions). Genetic operators (selection, mutation, and crossovers) were then used to generate a new set of chromosomes (solutions). The efficiency of the newly produced chromosomes is predicted to be better than the initial generation. The procedures will proceed until the completion test is completed and the finishing solution is registered on the best chromosome (solution) of the last generation. The basic steps of GA are defined as follows:

Step I: An initial population of chromosomes are generated randomly and it is must be suitable for the problem.

Step II: The fitness value is evaluated of each chromosome in the population.

Step III: A new population is generated by applying the following steps repeatedly:

1) Two parents are selected from the population depending on selection mechanism.
2) Crossover the parents to create new offspring.
3) Mutate new offspring at each locus.

Step IV: If the termination criterion is met, the run is stopped; otherwise, the algorithm returns to **step II**. The flowchart that represents GA is shown in Figure 1.

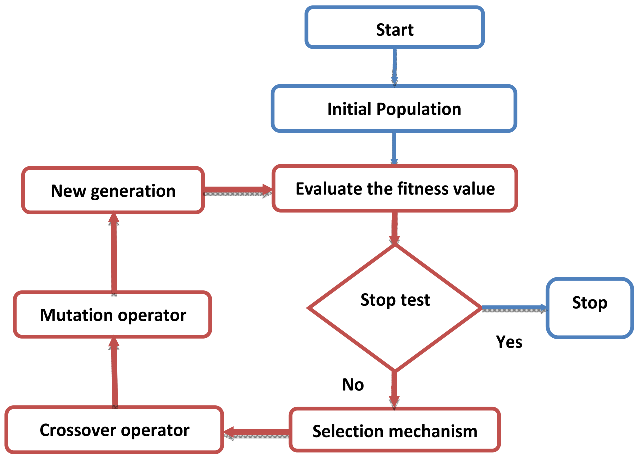


FIGURE 1. The flowchart of GA.

B. GRASSHOPPER OPTIMIZATION ALGORITHM

Grasshopper optimization algorithm (GOA) is one of the newly algorithms for optimization that proposed by Saremi *et al.* [25]. GOA is an evolutionary computation technique simulates the social behavior of grasshoppers in nature to solve optimization problems. As in Figure 2,

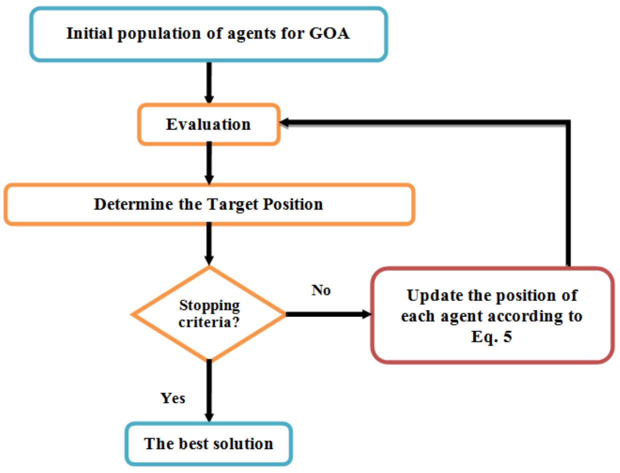

Initial population of agents for GOA
Evaluation
Determine the Target Position
Stopping criteria?
No
Update the position of each agent according to Eq. 5
Yes
The best solution

FIGURE 2. The flowchart of GOA.

The algorithm initially has a population of random (grasshoppers) solutions; where the position of the i-th grasshopper in a d-dimensional space is denoted as X_i and represented as $X_i = (x_{i1}, x_{i2}, \ldots, x_{id})$.

The grasshoppers positions are updated according to the following equations:

$$X_i = c\left(\sum_{\substack{j=1\\ j\neq i}}^{N} c\frac{ub_d - lb_d}{2} s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right) + \widehat{T_d} \quad \forall i = 1, \ldots, N_{grasshoppers},$$

$$s\left(\left|x_j^d - x_i^d\right|\right) = fe^{\frac{-\left|x_j^d - x_i^d\right|}{l}} - e^{-\left|x_j^d - x_i^d\right|},$$

$$d_{ij} = \left|x_j - x_i\right|; \tag{5}$$

where X_i is the position of the *i-th* grasshopper, ub_d and lb_d are the upper bound and the lower bound in the *d-th* dimension respectively, x_i^d and x_j^d are the *i-th* and the *j-th* grasshopper in the *d-th* dimension respectively, s is a function to define the strength of social forces, f is the intensity of attraction, l is the attractive length scale, d_{ij} is the distance between the *i-th* and the *j-th* grasshopper, $\widehat{T_d}$ is the value of the *d-th* dimension in the target (the best grasshopper among all the grasshopper in the population found so far) and c is a decreasing coefficient proportional to the number of iterations and is calculated as follows:

$$c = c_{\max} - t\frac{c_{\max} - c_{\min}}{T}, \tag{6}$$

where $c_{\max}$ is the maximum value, $c_{\min}$ is the minimum value, t indicates the current iteration, and T is the maximum number of iterations.

C. COMBINED ALGORITHM: HYBRID-GOA-GA

Hybrid-GOA-GA blends the merits of both the GOA and GA algorithms. It blends GA exploitation capacity and GOA exploration capacity; where GA operators (crossover and mutation) prevent the proposed algorithm from falling into the local optima while GOA's procedures accelerate the search process and convergence to reach a global solution. In the proposed algorithm, a population of solutions are randomly initialized. These solutions, by GOA, are searching in the domain of the optimization problem to obtain optimal solution. The evolution of these solutions is carried out by GA during this process. The flow chart of the proposed approach is shown in Figure 3. The proposed algorithm details are explained as follows:

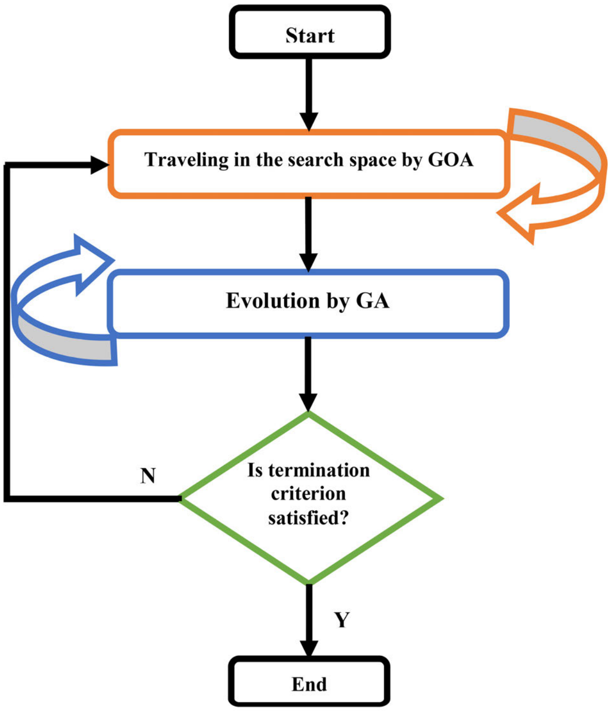

Start
Traveling in the search space by GOA
Evolution by GA
Is termination criterion satisfied?
N
Y
End

FIGURE 3. The flow chart of the proposed algorithm.

Step 1 (Initialization):

- A population of agents (in d-dimensions) are Initialized with random positions in the search space and set GOA and GA parameters.
- For each agent, the desired fitness function is evaluated in d variables.
- Determine the target position T as the position of best initial agent.

Step 2 (Traveling in the Search Space by GOA):

- Update the position of each agent as in (5).
- For each agent, the desired fitness function is evaluated in d variables.
- The target position T is updated as the best agent position found so far.

Step 3 (Evolution by GA):

- Selection [75]: In the proposed approach, tournament selection (TS) will be used; where a number *Tour* of agents are chosen randomly from the population and the best agent from this group is selected as parent. This process is repeated as often as agents must be chosen.
- Crossover [75]: Two agents, $A = [a_1, b_1]$ and $B = [a_2, b_2]$, are selected as parents by tournament selection. Suppose a is the crossover point for a particular A and B, and the a-values in the offsprings are determined by:

$$a_{new1} = (1-\gamma)\, a_1 + \gamma a_2,$$
$$a_{new2} = (1-\gamma)\, a_2 + \gamma a_2; \tag{7}$$

where γ is a random number between 0 and 1. The remaining parameter (b in this case) is inserted directly from each parent, so the new offsprings are:

$$\text{offspring}_1 = [a_{new1}, b_1],$$
$$\text{offspring}_2 = [a_{new2}, b_2]. \tag{8}$$

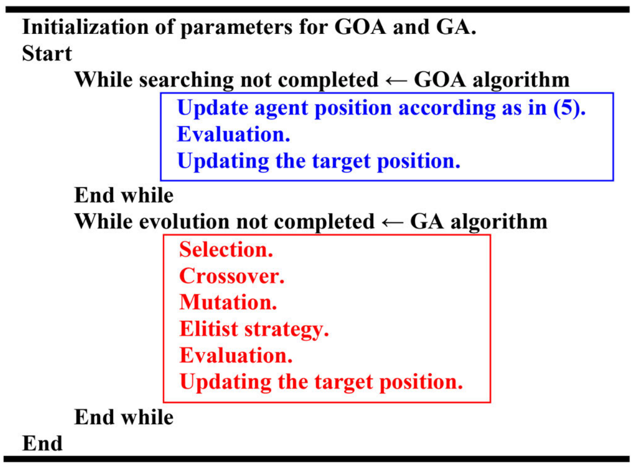
- Mutation [77]: We mutate each agent in a string $x_i \in [a_i, b_i]$ with mutation probability Pm by addition of small random values according to the equations below:

$$X_i = \begin{cases} X_i + \Delta\,(t, b_i - X_i) & \text{if } \delta = 0 \\ X_i - \Delta\,(t, X_i - a_i) & \text{if } \delta = 1, \end{cases}$$
$$\Delta\,(t, y) = y\left[1 - r^{(1-t/T)^{\beta}}\right]; \tag{9}$$

where r is a random number $r \in [0, 1]$ and β is a positive constant chosen arbitrarily.
- Elitist strategy: The list of best agents directly entered into new set of agents are elitism.
- Evaluation: For each agent, the desired fitness function is evaluated in d variables.
- Updating: Updating the target position T as the best agent position found so far.

Step 4 (Termination Criteria):

If the maximum number of generations is reached or the agents converge, the proposed algorithm will be terminated. Finally, the target position T is set as the optimal solution. Pseudo code of the proposed algorithm is shown in Figure 4.



```
Initialization of parameters for GOA and GA.
Start
    While searching not completed ← GOA algorithm
            Update agent position according as in (5).
            Evaluation.
            Updating the target position.
    End while
    While evolution not completed ← GA algorithm
            Selection.
            Crossover.
            Mutation.
            Elitist strategy.
            Evaluation.
            Updating the target position.
    End while
End
```

FIGURE 4. The pseudo code of the proposed algorithm.

IV. NUMERICAL RESULTS

In order to evaluate the proposed approach, eight systems of nonlinear equations are solved. These eight test cases are general problems and known as benchmarks which were studied by other researchers. The proposed algorithm was programmed in Matlab (R2016b) on a PC with 3.00 GHz P4 CPU, 1 GB of RAM for i5 processor, and Windows 7 for OS.

A. BENCHMARK 1

This benchmark problem consists of the following two of nonlinear algebraic equations [33]:

$$F(x) = \begin{cases} f_1(x_1, x_2) = x_1^2 + x_2^2 + x_1 + x_2 - 8 = 0, \\ f_2(x_1, x_2) = x_1 x_2 + x_1 + x_2 - 5 = 0, \end{cases}$$
$$x_1 \in [-3.5, 2.5], \quad x_2 \in [-3.5, 2.5]. \quad (10)$$

B. BENCHMARK 2

This benchmark problem consists of the following two of nonlinear algebraic equations [33]:

$$F(x) = \begin{cases} f_1(x_1, x_2, x_3), \\ f_2(x_1, x_2, x_3), \\ f_3(x_1, x_2, x_3), \end{cases} \quad \text{where}$$

$$\begin{aligned} f_1(x_1, x_2, x_3) &= 3x_1^2 + \sin(x_1 x_2) - x_3^2 + 2 = 0, \\ f_2(x_1, x_2, x_3) &= 2x_1^3 - x_2^2 - x_3 + 3 = 0, \\ f_3(x_1, x_2, x_3) &= \sin(2x_1) + \cos(x_2 x_3) + x_2 - 1 = 0, \end{aligned}$$
$$x_1 \in [-5, 5], x_2 \in [-1, 3], x_3 \in [-5, 5]. \quad (11)$$

C. BENCHMARK 3

This benchmark problem consists of the following two of nonlinear equations [33]; where f is non-differentiable:

$$F(x) = \begin{cases} f_1(x_1, x_2) = x_1^2 - x_2 + 1 + \frac{1}{9}|x_1 - 1| = 0, \\ f_2(x_1, x_2) = x_2^2 - x_1 - 7 + \frac{1}{9}|x_2| = 0, \end{cases}$$
$$x_1 \in [-2, 2], \quad x_2 \in [1, 6]. \quad (12)$$

These benchmark problems are solved by EGA: efficient GA, SHEGA: GA with harmonious and symmetric individuals [33] and the proposed algorithm. In [33], the authors introduced $[\lambda \times N]$ pairs of harmonious and symmetric individuals; where N is the size of population ($N = 40$ for benchmark 1 and 3, $N = 50$ for benchmark 2) and $\lambda \in [0, 0.5)$ is a parameter, while G is the fixed maximal generation. Benchmark problem 1 is solved 30 times by the proposed algorithm, while benchmark problems 2 and 3 are solved 20 times. The comparison results between the proposed algorithm, EGA and SHEGA with different values of λ for the three benchmarks are given in Tables 1-3. Figures 5-7 show the convergence curves of the best F obtained so far by the proposed algorithm, in different 10 runs that randomly selected from among the total number of runs. Tables and Figures show that the proposed algorithm converges well and performs better than EGA and SHEGA; where our results are completely less than that that obtained by both algorithms.

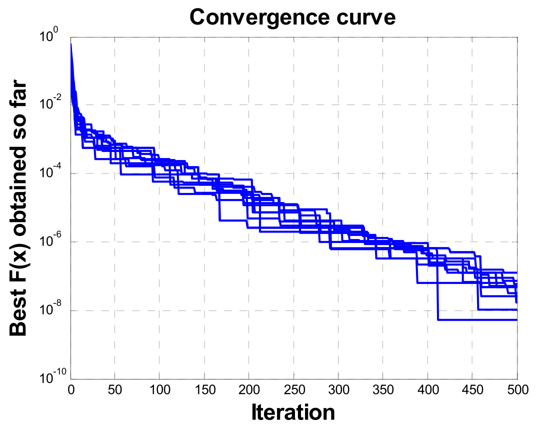


FIGURE 5. The convergence curves of the best *F* obtained so far by the proposed algorithm in 10 random runs out of 30 different runs for benchmark 1.

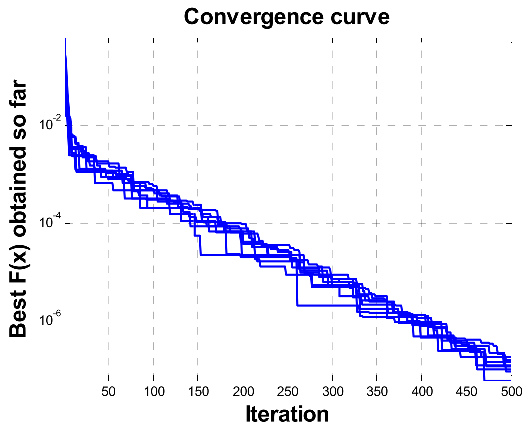


FIGURE 6. The convergence curves of the best *F* obtained so far by the proposed algorithm in 10 random runs out of 20 different runs for benchmark 2.

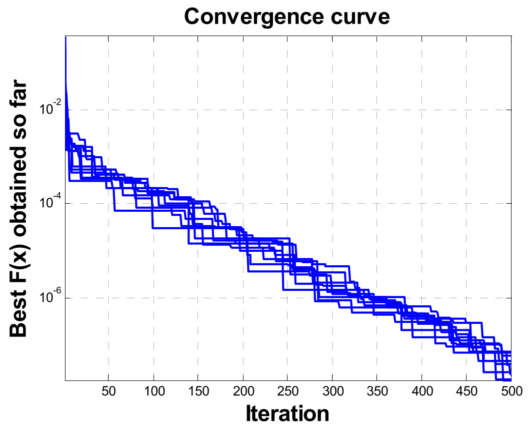


FIGURE 7. The convergence curves of the best *F* obtained so far by the proposed algorithm in 10 random runs out of 20 different runs for benchmark 3.

TABLE 1. The comparison results of the best function value F for benchmark 1.

The proposed algorithm: The best solutions obtained in 30 runs			EGA and SHEGA [33]: The average of the best function (in 30 runs) value F with $G = 300$	
f_1,f_2	x_1,x_2	F		F
1.9929e-08,4.1568e-08	1.00000002752194,1.99999997950108	3.0748e-08	EGA	0.129795093953972
1.1812e-08, 9.3856e-08	2.00000003522273,0.999999945232846	5.2834e-08	SHEGA ($\lambda = 0.05$)	0.053101422547384
2.1311e-08,7.6810e-08	1.00000003793659,1.99999998150026	4.9061e-08	SHEGA ($\lambda = 0.10$)	0.041427669194903
1.1961e-07, 2.6574e-08	2.00000003101142,0.999999988183792	7.3091e-08	SHEGA ($\lambda = 0.15$)	0.034877317449825
5.9117e-09, 4.8096e-09	0.999999996014262,2.00000000357378	5.3607e-09	SHEGA ($\lambda = 0.20$)	0.035701604675096
1.0585e-08, 2.3503e-08	1.99999998863761, 1.00000001540913	1.7044e-08	SHEGA ($\lambda = 0.25$)	0.038051665705034
2.1473e-07,2.8525e-08	0.999999968129972,2.00000006206758	1.2163e-07	SHEGA ($\lambda = 0.30$)	0.039332883168632
3.8687e-09,1.6598e-08	0.999999991638524,2.00000000424314	1.0233e-08	SHEGA ($\lambda = 0.35$)	0.035780879206619
4.6006e-09,4.8105e-08	2.00000001756851,0.999999972252682	2.6353e-08	SHEGA ($\lambda = 0.40$)	0.034509424138501
9.0164e-08,2.8446e-08	1.99999997942747,1.00000000423288	5.9305e-08	SHEGA ($\lambda = 0.45$)	0.037425326021257

G'' means maximal generation

TABLE 2. The comparison results of the best function value F for benchmark 2.

The proposed algorithm: The best solutions obtained in 20 runs			EGA and SHEGA [33]: The average of the best function (in 20 runs) value F with $G = 300$	
f_1,f_2,f_3	x_1,x_2,x_3	F		F
8.1293e-08,1.7983e-07,2.3139e-07	-0.0327591321066124,1.26462879567440,1.40064387735038	1.6417e-07	EGA	0.0081958187235953
2.2096e-07,5.7394e-08,1.2367e-08	-0.0327589695693218,1.26462870306312,1.40064399020112	9.6908e-08	SHEGA ($\lambda = 0.05$)	0.0021219384775699
2.0579e-07,9.5554e-08, 2.7042e-07	-0.0327592529919166,1.26462874497731,1.40064372941569	1.9059e-07	SHEGA ($\lambda = 0.10$)	0.0019286950245614
1.5265e-07, 8.4477e-08,1.3548e-07	-0.0327591379828040,1.26462879184554,1.40064379164548	1.2420e-07	SHEGA ($\lambda = 0.15$)	0.0018367719544782
1.5161e-07,2.3268e-08, 1.5710e-08	-0.0327589961888146,1.26462868481683,1.40064395551739	6.3529e-08	SHEGA ($\lambda = 0.20$)	0.0022816080103967
7.0813e-08, 2.4517e-07, 1.1791e-07	-0.0327589424808460,1.26462881499237,1.40064389505317	1.4463e-07	SHEGA ($\lambda = 0.25$)	0.0023297925904943
3.8327e-08,3.4307e-08, 2.4373e-07	-0.0327591699133487,1.26462872233391,1.40064384846897	1.0545e-07	SHEGA ($\lambda = 0.30$)	0.0023318357433983
3.4337e-08,1.3396e-07, 2.2496e-07	-0.0327588585095305,1.26462874342104,1.40064396540968	1.3109e-07	SHEGA ($\lambda = 0.35$)	0.0021392510790106
3.3092e-09, 1.0484e-07, 2.5082e-07	-0.0327591894675818,1.26462870219270,1.40064382875580	1.1965e-07	SHEGA ($\lambda = 0.40$)	0.0022381534380744
4.0356e-08, 1.4050e-07, 1.9419e-07	-0.0327588959827691,1.26462876218205,1.40064392425333	1.2502e-07	SHEGA ($\lambda = 0.45$)	0.0025798012930550

TABLE 3. The comparison results of the best function value F for benchmark 3.

The present study: The best solution obtained in 20 runs			EGA and SHEGA [33]: The average of the best function (in 20 runs) value F with $G = 500$	
f_1,f_2	$\boldsymbol{x_1,x_2}$	F		F
1.0356e-08,2.7202e-08	1.15936084839974,2.36182434809133	1.8779e-08	EGA	0.1668487275323187
8.5255e-08,6.5579e-08	1.15936081271468,2.36182433628177	7.5417e-08	SHEGA ($\lambda = 0.05$)	0.0752619855962109
5.3446e-09,4.9676e-08	-1.25195282517727,2.81760286238918	2.7510e-08	SHEGA ($\lambda = 0.10$)	0.0500864062405815
2.5529e-08, 2.3128e-08	1.15936083869689,2.36182433968804	2.4329e-08	SHEGA ($\lambda = 0.15$)	0.0358268275921585
9.9194e-08,4.0646e-08	-1.25195278608611,2.81760285401479	6.9920e-08	SHEGA ($\lambda = 0.20$)	0.0257859494269335
3.8038e-08, 4.2611e-08	1.15936083910945,2.36182435319996	4.0325e-08	SHEGA ($\lambda = 0.25$)	0.0239622084932336
8.0422e-08,1.6479e-08	1.15936088198772,2.36182433892625	4.8451e-08	SHEGA ($\lambda = 0.30$)	0.0247106452514721
4.6369e-08, 2.1640e-08	-1.25195280637207,2.81760285423760	3.4004e-08	SHEGA ($\lambda = 0.35$)	0.0171980128114993
1.1263e-08, 2.7873e-08	-1.25195281722414,2.81760284750974	1.9568e-08	SHEGA ($\lambda = 0.40$)	0.0179659124369376
5.4002e-08, 3.9527e-08	1.15936087377521,2.36182434539194	4.6764e-08	SHEGA ($\lambda = 0.45$)	0.0158999282064303

D. BENCHMARK 4: EXPERIMENT TEST

In this benchmark, the following system of nonlinear equations has been considered [71]:

$$\begin{cases} f_1(x_1, x_2) = \cos(2x_1) - \cos(2x_2) - 0.4 = 0, \\ f_2(x_1, x_2) = 2(x_2 - x_1) + \sin(2x_2) - \sin(2x_1) - 1.2 = 0. \end{cases}$$

$$x_1 \in [-10, 10],$$

$$x_2 \in [-10, 10]. \tag{13}$$

Table 4 summarizes the results of this nonlinear system by five different methods [71]: Newton's method, Secant's method, Multi objective optimization's method, GA and the proposed algorithm. Figure 8 show the convergence curve of the best F that obtained by the proposed algorithm in three different runs. While the methods (Newton's method, Secant's method and multi objective optimization's method) found only one solution for the problem and GA found two solutions for the problem, three solutions were obtained by the proposed algorithm. Moreover, in the proposed algorithm, the obtained values for the functions are sufficiently close to zero in comparison with the other methods. In addition, the quick convergence of the algorithm is noticeable as shown in Figure 8.

TABLE 4. **The comparison results of the best function value *F* for benchmark 4, experiment test.**

Method	(x_1,x_2)	(f_1,f_2)
Newton's method [71]	(0.15,0.49)	(0.00168,0.01497)
Secant's method [71]	(0.15,0.49)	(0.00168,0.01497)
Multi objective optimization's method [71]	(0.15722,49458)	(0.001264,0.000969)
GA [71]	(0.156522,0.49338)	(4.8606e-06, 3.7164e-06)
	(0.68021,2.26002)	(6.9752e-06, 1.0970e-04)
Present study	(0.680235945188233,2.25999176017399)	(2.2840e-06,1.2967e-06)
	(0.680236281982930,2.25999123973785)	(1.9211e-06,3.5933e-07)
	(0.680235245449499,2.25999126133352)	(6.3875e-08,2.1815e-06)

TABLE 5. **The comparison results of the best function value *F* for benchmark 5, arithmetic application obtained by three studies.**

Present Study				GA [71]				Grosan and Abraham [32]			
x_1	0.257833393701793 0.257833393702522	f_1	1.2656e-12 1.9790e-12	x_1	2.5783339e-01	f_1	-7.3844e-10	x_1	0.2077500302	f_1	0.0464943
x_2	0.381097154602943 0.381097154599575	f_2	7.9096e-14 3.2443e-12	x_2	3.8109715e-01	f_2	-1.1684e-12	x_2	0.0299198492	f_2	0.3489889
x_3	0.278745017344737 0.278745017345920	f_3	1.7517e-12 5.3920e-13	x_3	2.7874502e-01	f_3	1.7931e-09	x_3	-0.0339491324	f_3	0.3058418
x_4	0.200668964229936 0.200668964227426	f_4	4.5315e-12 2.0764e-12	x_4	2.0066896e-01	f_4	-8.8837e-10	x_4	-0.2027950317	f_4	0.4012915
x_5	0.445251424842223 0.445251424844883	f_5	1.1361e-12 3.8675e-12	x_5	4.4525142e-01	f_5	-4.5866e-10	x_5	0.2131771707	f_5	0.2284027
x_6	0.149183919971614 0.149183919968370	f_6	2.2230e-12 9.9666e-13	x_6	1.4918391e-01	f_6	-5.270e-09	x_6	0.0568458067	f_6	0.0886970
x_7	0.432009698985234 0.432009698984241	f_7	1.4795e-12 5.4015e-13	x_7	4.3200969e-01	f_7	-6.3852e-09	x_7	0.2267650517	f_7	0.2024745
x_8	0.073402777776967 0.073402777775704	f_8	6.5123e-13 5.5117e-13	x_8	7.3402777e-02	f_8	-9.7362e-10	x_8	-0.0977041236	f_8	0.1687259
x_9	0.345966826872031 0.345966826876406	f_9	3.5476e-12 8.6115e-13	x_9	3.4596683e-01	f_9	-6.0389e-11	x_9	-0.0339921200	f_9	0.3787652
x_{10}	0.427326275993876 0.427326275994709	f_{10}	5.5468e-13 1.4100e-12	x_{10}	4.2732628e-01	f_{10}	3.0841e-10	x_{10}	0.2532921324	f_{10}	0.1741025

E. BENCHMARK 5: ARITHMETIC APPLICATION

This benchmark problem consists of complex set of nonlinear equations as follows [32]:

$$\begin{cases} f_1 = x_1 - 0.25428722 - 0.18324757 x_4 x_3 x_9 = 0, \\ f_2 = x_2 - 0.37842197 - 0.16275449 x_1 x_{10} x_6 = 0, \\ f_3 = x_3 - 0.27162577 - 0.16955071 x_1 x_2 x_{10} = 0, \\ f_4 = x_4 - 0.19807914 - 0.15585316 x_7 x_1 x_6 = 0, \\ f_5 = x_5 - 0.44166728 - 0.19950920 x_7 x_6 x_3 = 0, \\ f_6 = x_6 - 0.14654113 - 0.18922793 x_8 x_5 x_{10} = 0, \\ f_7 = x_7 - 0.42937161 - 0.21180486 x_2 x_5 x_8 = 0, \\ f_8 = x_8 - 0.07056438 - 0.17081208 x_1 x_7 x_6 = 0, \\ f_9 = x_9 - 0.34504906 - 0.19612740 x_{10} x_6 x_8 = 0, \\ f_{10} = x_{10} - 0.42651102 - 0.21466544 x_4 x_8 x_1 = 0. \end{cases}$$

$$-10 \le x_1, x_2, \ldots, x_{10} \le 10 \quad (14)$$

Table 5 shows the solution of this system using the proposed algorithm, GA [71] and Grosan and Abraham [32]. Figure 9 show the convergence curve of the best F that obtained by the proposed algorithm in two different runs. In addition, this problem is solved in [66]; where the best obtained fitness function value F equal to 0.040217. We can see that the values that obtained for the functions by the present study is closer to zero in comparison with the values of the other three methods. In addition, the algorithm have the same convergence in the two different runs which mean that it is stable for the different conditions of optimizations as shown in Figure 9.

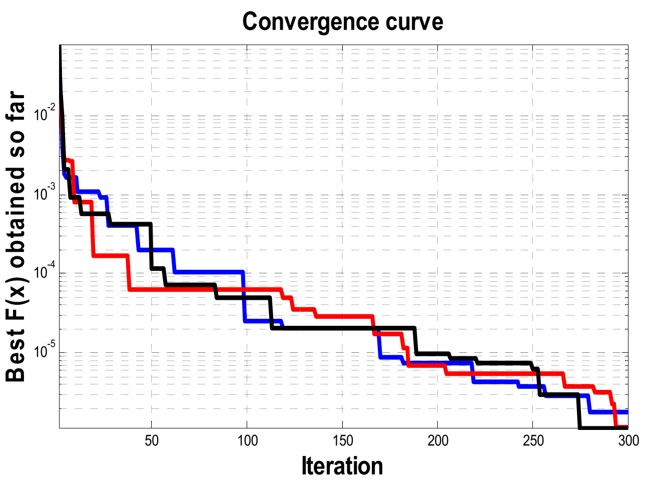


FIGURE 8. **The convergence curves of the best *F* obtained so far by the proposed algorithm in 3 different runs for benchmark 4, experiment test.**

F. BENCHMARK 6: FLUID MECHANICS APPLICATION

The following nonlinear equation, was developed by Colebrook–White equation [78], is used to determine the

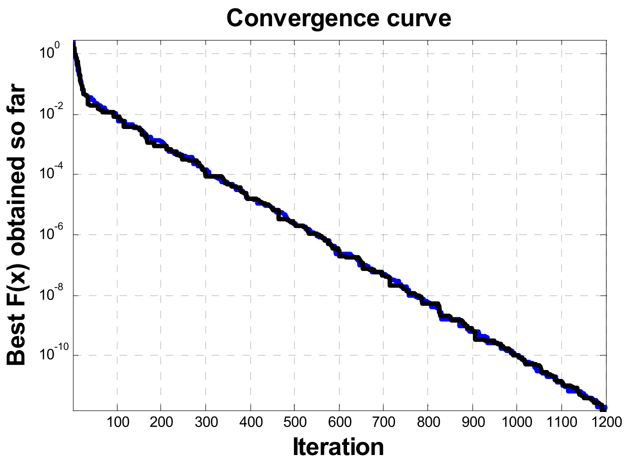


FIGURE 9. The convergence curves of the best F obtained so far by the proposed algorithm in 2 different runs for benchmark 5, Arithmetic application.

friction coefficient (f) in a straight pipeline as:

$$F(f) = \frac{1}{\sqrt{f}} + 2\log_{10}\left(\frac{\varepsilon/D}{3.7} + \frac{2.51}{\mathrm{Re}\sqrt{f}}\right); \qquad (15)$$

where ε is the pipe surface roughness, D is the pipe diameter and Re is a dimensionless Reynolds number. The problem was solved for $\varepsilon/D = 0.001$, $Re = 10^{\wedge}n$ for $n = 5, 6, 7$. Table 6 shows the values of friction coefficient and the function that obtained by our algorithm. While Figure 10 show the convergence curve of the best F that obtained by the proposed algorithm in three ceases $n = 5, 6, 7$. We can see that, our results have good agreement with the data in the Moody chart [78] and our algorithm can easily get on friction coefficient for a wide range of Re and ε/D. In addition, our results are better than that obtained by GA [71].

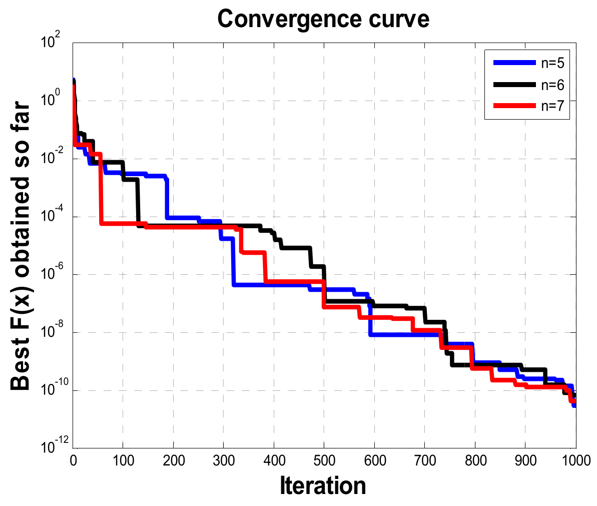


FIGURE 10. The convergence curves of the best F obtained so far by the proposed algorithm in 3 different runs for benchmark 6, Fluid mechanics application.

TABLE 6. The comparison results of the best function value F for benchmark 6, Fluid mechanics obtained by two studies.

	GA [71]		The present study	
n	f	$F(f)$	f	$F(f)$
5	0.02218	-8.6832e-04	0.022174535944694	2.8511e-11
6	0.01994	6.2002e04	0.019943465840535	1.0405e-11
7	0.01967	-5.3472e04	0.019667052432323	4.1059e-11

G. BENCHMARK 7: COMBUSTION APPLICATION

This benchmark problem consists of a complex set of non-linear equations which governs a combustion problem as follows [32]:

$$\begin{cases} f_1 = x_2 + 2x_6 + x_9 + 2x_{10} - 10^{-5} = 0, \\ f_2 = x_3 + x_8 - 3.10^{-5} = 0, \\ f_3 = x_1 + x_3 + 2x_5 + 2x_8 + x_9 + x_{10} - 5.10^{-5} = 0, \\ f_4 = x_4 + 2x_7 - 10^{-5} = 0, \\ f_5 = 0.5140437.10^{7}x_5 - x_1^2 = 0, \\ f_6 = 0.1006932.10^{-6}x_6 - 2x_2^2 = 0, \\ f_7 = 0.7816278.10^{-15}x_7 - x_4^2 = 0, \\ f_8 = 0.1496236.10^{-6}x_8 - x_1x_3 = 0, \\ f_9 = 0.6194411.10^{-7}x_9 - x_1x_2 = 0, \\ f_{10} = 0.2089296.10^{-14}x_{10} - x_1x_2^2 = 0. \end{cases}$$
$$-10 \le x_1, x_2, \ldots, x_{10} \le 10 \qquad (16)$$

Some obtained solutions by the proposed algorithm, GA [71] and Grosan and Abraham [32] as well as the function values are listed in Table 7. In addition, this problem is solved in [66]; where the best obtained fitness function value F equal to 0.024030224. Moreover, the convergence curves of the best F that obtained by the proposed algorithm in two different runs are depicted in Figure 11. We can be seen that the results of our algorithm are very close to zero and has more convergence than other algorithms.

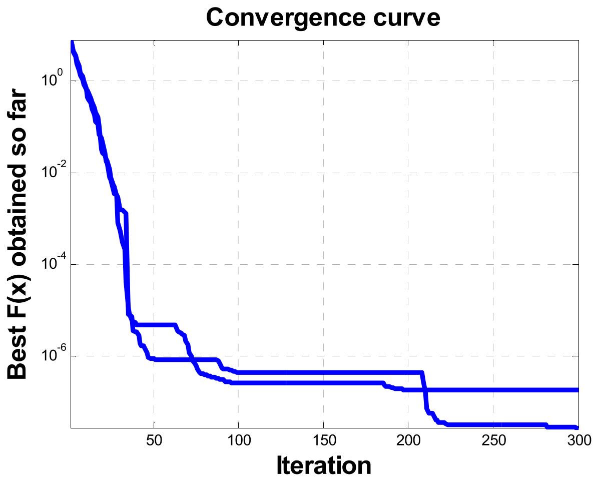


FIGURE 11. The convergence curves of the best F obtained so far by the proposed algorithm in 2 different runs for benchmark 7, Combustion application.

TABLE 7. The comparison results of the best function value *F* for benchmark 7, Combustion obtained by three studies.

Present Study				GA [71]				Grosan and Abraham [32]			
x_1	5.21779475364814e-06	f_1	9.3077e-12	x_1	7.7944699e-005	f_1	-9.0000000e-005	x_1	2.8724570e-004	f_1	-9.0156756e-005
	1.55416640952108e-09		8.5611e-12								
x_2	1.40880551532908e-07	f_2	1.0051e-07	x_2	2.3453123e-004	f_2	-4.7433845e-020	x_2	4.6449359e-004	f_2	-3.3881318e-021
	4.67103882154831e-06		1.2440e-08								
x_3	2.28073956343815e-05	f_3	2.0002e-11	x_3	5.6870072e-008	f_3	-5.5091023e-018	x_3	-3.8722475e-006	f_3	-5.9848143e-008
	2.98520197109892e-05		1.9449e-14								
x_4	9.67928615027979e-06	f_4	1.6723e-12	x_4	-5.1124010e-004	f_4	-9.0000000e-005	x_4	5.7046411e-005	f_4	-9.0000000e-005
	1.72396380792863e-10		6.6138e-12								
x_5	1.01563226859111e-06	f_5	2.7173e-11	x_5	1.1665683e-001	f_5	-7.8705351e-011	x_5	1.2033492e+000	f_5	-2.0652682e-008
	9.83322254293174e-06		5.0547e-13								
x_6	5.59198086961396e-10	f_6	3.9638e-14	x_6	3.6717284e-001	f_6	-7.3037986e-008	x_6	3.2144041e+000	f_6	-1.0783996e-007
	2.50296479375058e-06		4.3385e-11								
x_7	1.60356088720311e-07	f_7	9.3689e-11	x_7	2.6062005e-004	f_7	-2.6136644e-007	x_7	-2.3523205e-005	f_7	-3.2542930e-009
	4.99991049490908e-06		2.5812e-20								
x_8	7.29311644063544e-06	f_8	1.1791e-10	x_8	2.9943130e-005	f_8	4.7478263e-014	x_8	3.3872248e-005	f_8	1.1173545e-009
	1.35540003851492e-07		2.6115e-14								
x_9	8.56574022358758e-07	f_9	6.8203e-13	x_9	2.6776713e-001	f_9	-1.6938693e-009	x_9	1.6152635e+000	f_9	-3.3367727e-008
	9.47790670982597e-08		1.3886e-15								
x_{10}	4.50071816882287e-06	f_{10}	9.4156e-20	x_{10}	-5.0116867e-001	f_{10}	-4.2883872e-012	x_{10}	-4.0222631e+000	f_{10}	-6.1982897e-011
	1.14121981385416e-07		3.3671e-20								

TABLE 8. The comparison results of the best function value *F* for benchmark 8, Neurophysiology obtained by three studies.

Present Study				GA [71]				Grosan and Abraham [32]			
x_1	0.0949210611770202	f_1	6.1760e-11	x_1	3.2484137e-001	f_1	1.5105117e-010	x_1	7.0148122e-001	f_1	1.1532022e-009
	0.0820223613267075		6.9593e-11								
x_2	-0.0949210609568043	f_2	7.5751e-12	x_2	3.2484137e-001	f_2	1.5114510e-010	x_2	7.5925767e-001	f_2	2.6058267e-011
	-0.138287000903135		3.1647e-11								
x_3	0.995484802599612	f_3	1.3914e-12	x_3	9.4576852e-001	f_3	-1.2749912e-011	x_3	-7.1268794e-001	f_3	-6.5553074e-010
	-0.996630489354999		3.3110e-12								
x_4	-0.995484802585785	f_4	2.9057e-12	x_4	9.4576852e-001	f_4	4.6365863e-012	x_4	6.5079013e-001	f_4	1.1783451e-009
	0.990392197774631		9.6123e-12								
x_5	0.490886141709622	f_5	1.0662e-10	x_5	-5.6887875e-001	f_5	1.0181522e-011	x_5	2.4122542e-009	f_5	1.1134504e-009
	4.48130330622387e-09		2.5478e-10								
x_6	0.490886141728666	f_6	2.0320e-11	x_6	5.6887875e-001	f_6	8.4981744e-012	x_6	7.8977724e-010	f_6	-5.4967453e-010
	4.56992671931472e-09		5.6505e-11								

H. BENCHMARK 8: NEUROPHYSIOLOGY APPLICATION

This benchmark problem consists of a complex set of nonlinear equations that concerns problem of a neurophysiology as follows [32]:

$$\begin{cases} f_1 = x_1^2 + x_3^2 - 1 = 0, \\ f_2 = x_2^2 + x_4^2 - 1 = 0, \\ f_3 = x_5 x_3^3 + x_6 x_4^3 = 0, \\ f_4 = x_5 x_1^3 + x_6 x_2^3 = 0, \\ f_5 = x_5 x_1 x_3^2 + x_6 x_4^2 x_2 = 0, \\ f_6 = x_5 x_1^2 x_3 + x_6 x_2^2 x_4 = 0, \end{cases}$$
$$-10 \leq x_1, x_2, \ldots, x_6 \leq 10 \quad (17)$$

Table 8 contains some results of this benchmark with the functions values that obtained by the proposed algorithm, GA [71] and Grosan and Abraham [32]. Furthermore, Figure 12 show the convergence curve of the best F that obtained by the proposed algorithm in two different runs. The nearness of the objective functions values to zero in the results of the current study are noticeable. In addition, the proposed algorithm is stable for the different conditions of optimizations in the two runs as shown in Figure 12.

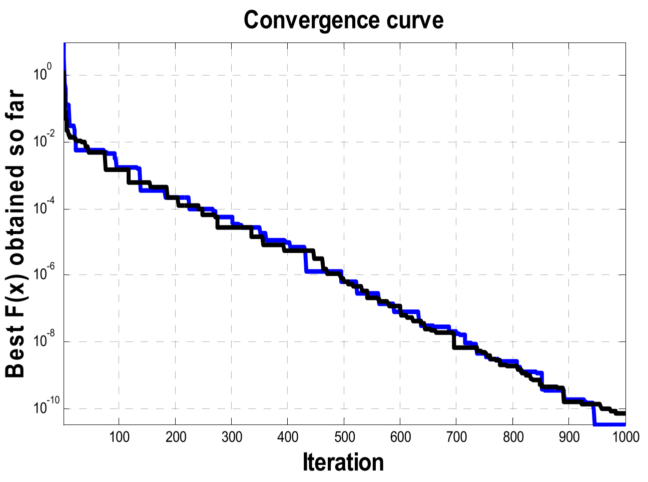


FIGURE 12. The convergence curves of the best *F* obtained so far by the proposed algorithm in 2 different runs for benchmark 8, Neurophysiology application.

V. DISCUSSIONS

The research proposed is divided into three parts. In the first part, The system of non-linear equations (SNLEs) is transformed into an optimization problem. The hybrid-GOA-GA was used to solve this optimization problem.

TABLE 9. Results of benchmark 1 at different initial intervals.

Initial Interval		F	f_1,f_2	x_1,x_2
D	$[-3.5, 2.5]$	4.2378e-08	3.7190e-08,4.7567e-08	1.00000001816148,1.99999999654106
$10D$	$[-35, 25]$	4.3387e-07	5.3831e-07,3.2942e-07	0.999999697364968,2.00000028924366
$100D$	$[-350, 250]$	3.3879e-06	3.8015e-06,2.9743e-06	2.00000225860121,0.999997502826225
$1000D$	$[-3500, 2500]$	9.3583e-05	1.0195e-04, 8.5214e-05	1.99999442038899,0.999975315038051
$10000D$	$[-35000, 25000]$	2.7607e-04	3.5477e-04,1.9738e-04	2.00018401322862,0.999811543217344

TABLE 10. Results of benchmark 2 at different initial intervals.

Initial Interval		F	f_1,f_2,f_3	x_1,x_2,x_3
D	$[-5,5]\times[-1,3]\times[-5,5]$	9.4171e-08	4.0128e-08,1.8040e-09,2.4058e-07	-0.0644172605927118,2.09043964511622,-1.37047251766814
$10D$	$[-50,50]\times[-10,30]\times[-50,50]$	1.1325e-06	1.9006e-06,2.1402e-07, 1.2829e-06	-0.0327586609343846,1.26462902019859,1.40064334669972
$100D$	$[-500,500]\times[-100,300]\times[-500,500]$	4.8602e-06	4.5121e-07,5.5416e-06, 8.5879e-06	-0.0327529639159027,1.26462995552519,1.40064634530864
$1000D$	$[-5000,5000]\times[-1000,3000]\times[-5000,5000]$	1.6810e-04	3.3626e-05,1.7982e-04,2.9087e-04	-0.0329513166986744,1.26458110757298,1.40058324946997
$10000D$	$[-50000,50000]\times[-10000,30000]\times[-50000,50000]$	0.0014	0.00183487,0.0011117094,0.0013537162	-0.0330615295572167,1.26485510090264,1.40118100641021

TABLE 11. Results of benchmark 3 at different initial intervals.

Initial Interval		F	f_1,f_2	x_1,x_2
D	$[-2,2]\times[1,6]$	3.0942e-08	3.1947e-08,2.9938e-08	-1.25195281287198,2.81760285681280
$10D$	$[-20,20]\times[10,60]$	45.5924	6.4903e-10, 91.1847	-2.92638597817765,10.0000000013121
$100D$	$[-200,200]\times[100,600]$	5.0070e+03	1.5488e-07,1.0014e+04	9.90005581164752,100.000000009041
$1000D$	$[-2000,2000]\times[1000,6000]$	5.0004e+05	1.3137e-08,1.0001e+06	-31.5496967821878,1000.00000001065
$10000D$	$[-20000,20000]\times[10000,60000]$	5.0001e+07	1.5498e-06,1.0000e+08	-99.9389041753464,10000.0000000165

In general, the proposed algorithm quickly converges without being influenced by the difficulty or complexity of the equation system. The second part was to solve four benchmarking problems and compare the results with other methods. Compared to other approaches, the precision of results is obvious especially with conventional methods (Benchmark 4, Experiment test). We can see that the suggested methodology is very fast and easy to implement and hence we can recommend using it to solve a set of nonlinear equations with any number of variables.

Finally, a complex system of nonlinear equations (Benchmark 5, Arithmetic applications) and three examples of different applications have been resolved. It is noted that the objective functions values that obtained by the proposed algorithm approaching to zero and the proposed algorithm is very stable and able to solve this kind of problems. In order to investigate the ability of the proposed algorithm to solve nonlinear systems thoroughly, Effect of changing initial intervals and computational cost are discussed in the coming subsections.

A. EFFECT OF CHANGING INITIAL INTERVALS

In this subsection, we study the sensitivity of the proposed algorithm when the initial intervals are changed. All benchmark problems are resolved at the initial interval *D*, *10D*, *100D*, *1000D* and *10000D*. At each interval, the objective function F is calculated for each benchmark problem as shown in tables 9-16. In addition, Figures 13-20 show the convergence curve of the objective function for all benchmark problems at each interval.

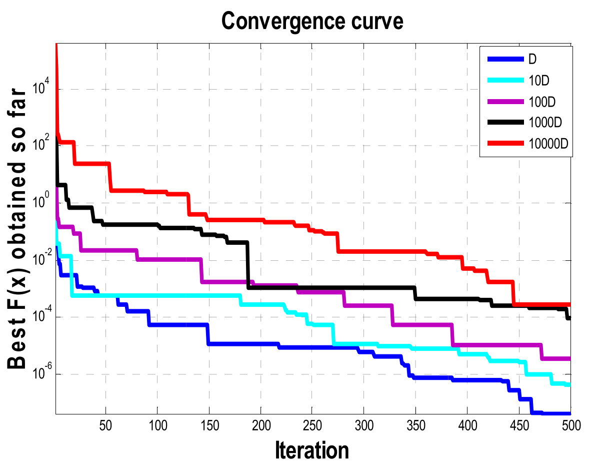


FIGURE 13. Convergence curves of the best F obtained so far by the proposed algorithm for benchmark 1 at different initial intervals.

TABLE 12. Results of benchmark 4 at different initial intervals.

	Initial Interval	F	f_1, f_2	x_1, x_2
D	$[-2,2]\times[1,6]$	3.3559e-06	5.6302e-06,1.0817e-06	-2.98507598872599,-2.64822091066403
$10D$	$[-20,20]\times[10,60]$	6.2575e-05	9.5824e-05, 2.9326e-05	28.4307999579039,28.7676328922980
$100D$	$[-200,200]\times[100,600]$	4.0729e-04	1.5614e-04, 6.5845e-04	917.501230937581,917.838210767026
$1000D$	$[-2000,2000]\times[1000,6000]$	0.0019	0.0026,0.0011	-9977.54308762156, -977.20693907882
$10000D$	$[-20000,20000]\times[10000,60000]$	0.0409	0.0610,0.0208	45041.1213024372,45041.4532870304

TABLE 13. Results of benchmark 5 at different initial intervals.

	Initial Interval	F
D	$[-10,10]\forall x$	1.7220e-12
$10D$	$[-100,100]\forall x$	1.9066e-11
$100D$	$[-1000,1000]\forall x$	2.2150e-10
$1000D$	$[-10000,10000]\forall x$	1.7218e-09
$10000D$	$[-100000,100000]\forall x$	1.8091e-08

TABLE 14. Results of benchmark 6 at different initial intervals.

Initial Interval	$n=5$	Friction coefficient (f)	$F(f)$
D	$[-10,10]\forall f$	0.022174535944086	6.8212e-11
$10D$	$[-100,100]\forall f$	0.022174535954141	1.5301e-09
$100D$	$[-1000,1000]\forall f$	0.022174535873613	1.127e-08
$1000D$	$[-10000,10000]\forall f$	0.022174537210046	2.0115e-07
$10000D$	$[-100000,100000]\forall f$	0.022174541344562	8.5831e-07

TABLE 15. Results of benchmark 7 at different initial intervals.

	Initial Interval	F
D	$[-10,10]\forall x$	5.6726e-09
$10D$	$[-100,100]\forall x$	6.3915
$100D$	$[-1000,1000]\forall x$	523.4762
$1000D$	$[-10000,10000]\forall x$	4.3847e+03
$10000D$	$[-100000,100000]\forall x$	4.7574e+04

TABLE 16. Results of benchmark 8 at different initial intervals.

	Initial Interval	F
D	$[-10,10]\forall x$	3.9846e-11
$10D$	$[-100,100]\forall x$	1.2428e-10
$100D$	$[-1000,1000]\forall x$	4.0382e-09
$1000D$	$[-10000,10000]\forall x$	2.3714e-08
$10000D$	$[-100000,100000]\forall x$	4.4574e-07

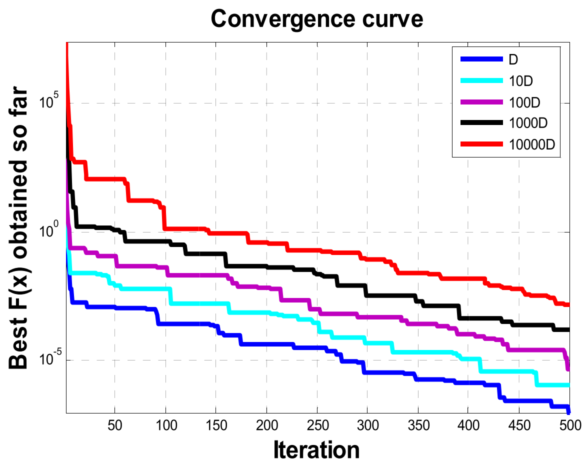


FIGURE 14. Convergence curves of the best F obtained so far by the proposed algorithm for benchmark 2 at different initial intervals.

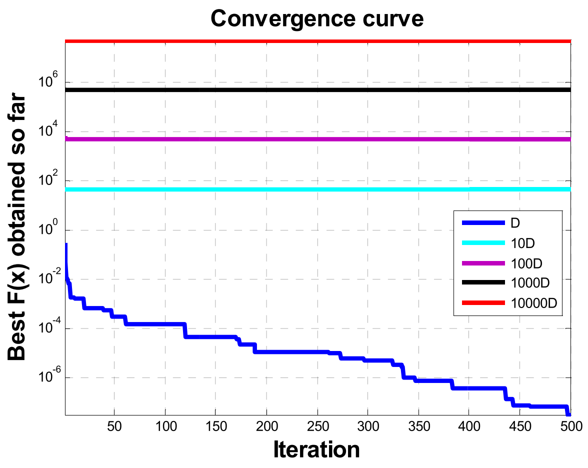


FIGURE 15. Convergence curves of the best F obtained so far by the proposed algorithm for benchmark 3 at different initial intervals.

From these results, it is obvious that for benchmarks 1, 5, 6 and 8 that the proposed algorithm doesn't affect by changing the initial intervals and give better results than other algorithms (Tables 9, 13, 14 and 16). For Benchmark No. 3, changing the initial interval led to bad results and this is probably because of that the objective functions are non-differentiable. Also, For Benchmark No. 7, changing the initial interval led to bad results. Finally, for Benchmarks 2 and 4, the proposed algorithm yields a satisfactory result compared to other methods.

B. COMPUTATIONAL COST

The computational cost of the proposed algorithm is analyzed in this sub-section and contrasted with other approaches. Table 17 shows the CPU time for the eight benchmarks for the three studies: Grosan and Abraham [32] and GA [71] and the present study hybrid-GOA-GA. It should be noted in the present study that the CPU time is the average of

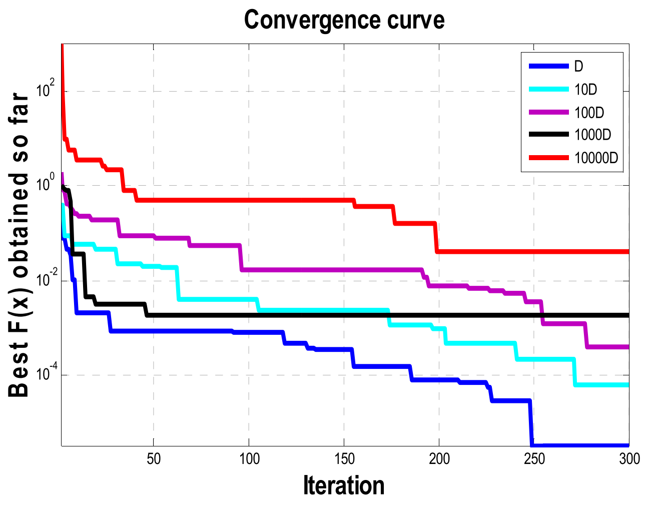


FIGURE 16. Convergence curves of the best *F* obtained so far by the proposed algorithm for benchmark 4 at different initial intervals.

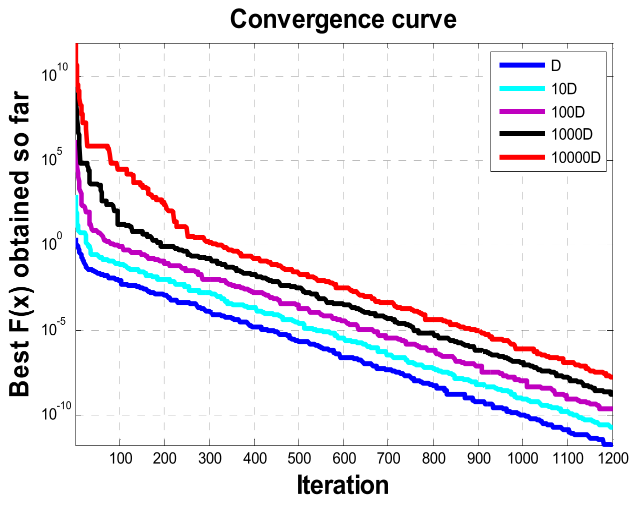


FIGURE 17. Convergence curves of the best *F* obtained so far by the proposed algorithm for benchmark 5 at different initial intervals.

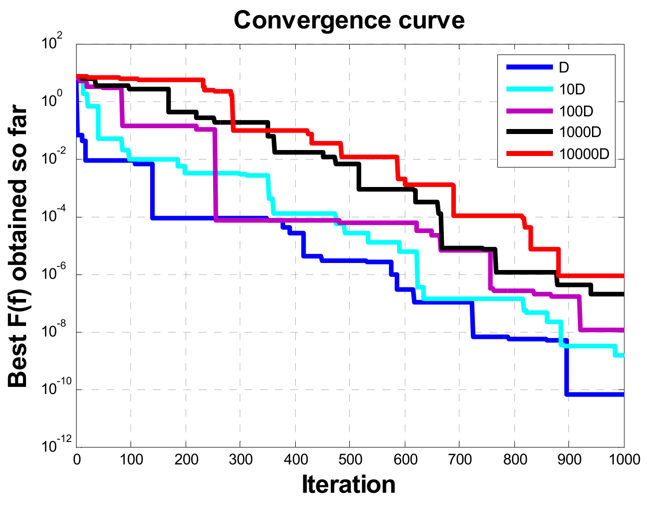


FIGURE 18. Convergence curves of the best *F* obtained so far by the proposed algorithm for benchmark 6 at different initial intervals.

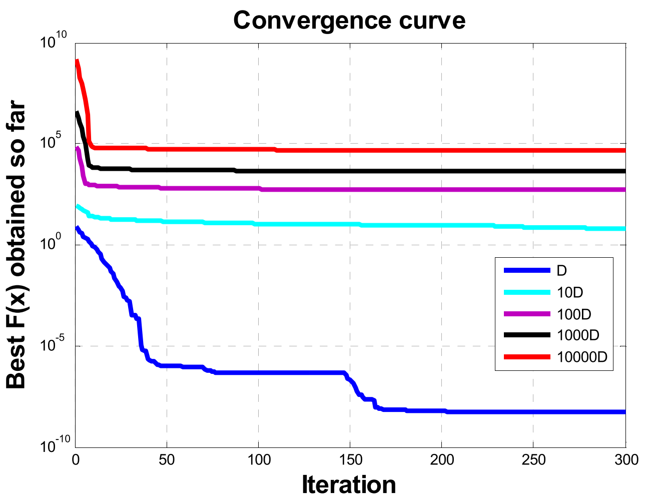


FIGURE 19. Convergence curves of the best *F* obtained so far by the proposed algorithm for benchmark 7 at different initial intervals.

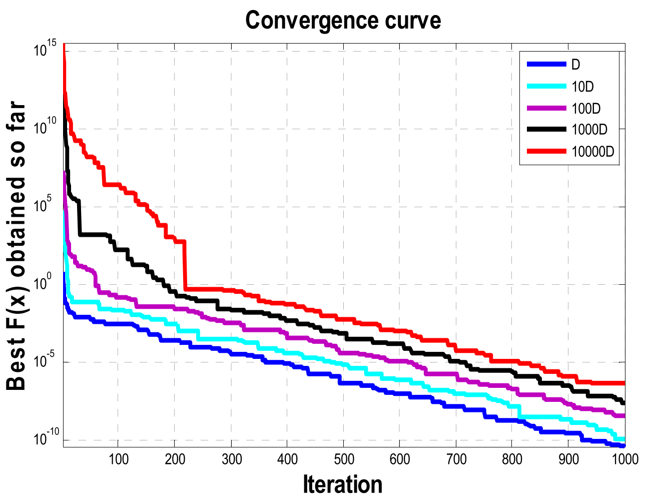


FIGURE 20. Convergence curves of the best *F* obtained so far by the proposed algorithm for benchmark 8 at different initial intervals.

100 independent runs for all the test problems. The table shows the comparison between the proposed algorithm and other methods in terms of time, population size and number of generations (Iterations). As shown in Table 17, the difference of running times between the three studies is clear. This is attributed to the number of generations and population size considered in these studies. The more population/generation the more the objective function evaluation, the higher the computational costs. So, we can say that our algorithm outperforms the two algorithms in terms of accuracy of results. But in terms of time, computational cost of the proposed algorithm is acceptable.

Finally, For fair comparisons and prove our motivation to propose this hybrid algorithm, a comparison between Hybrid-GOA-SA, GA, and GOA is achieved under the same conditions (number of population and number of iterations and the same equipment). Table 18 shows the comparison between Hybrid-GOA-GA, GA, and GOA according to the best function value F for all benchmarks problems, while Figure 21 shows the convergence curves for the

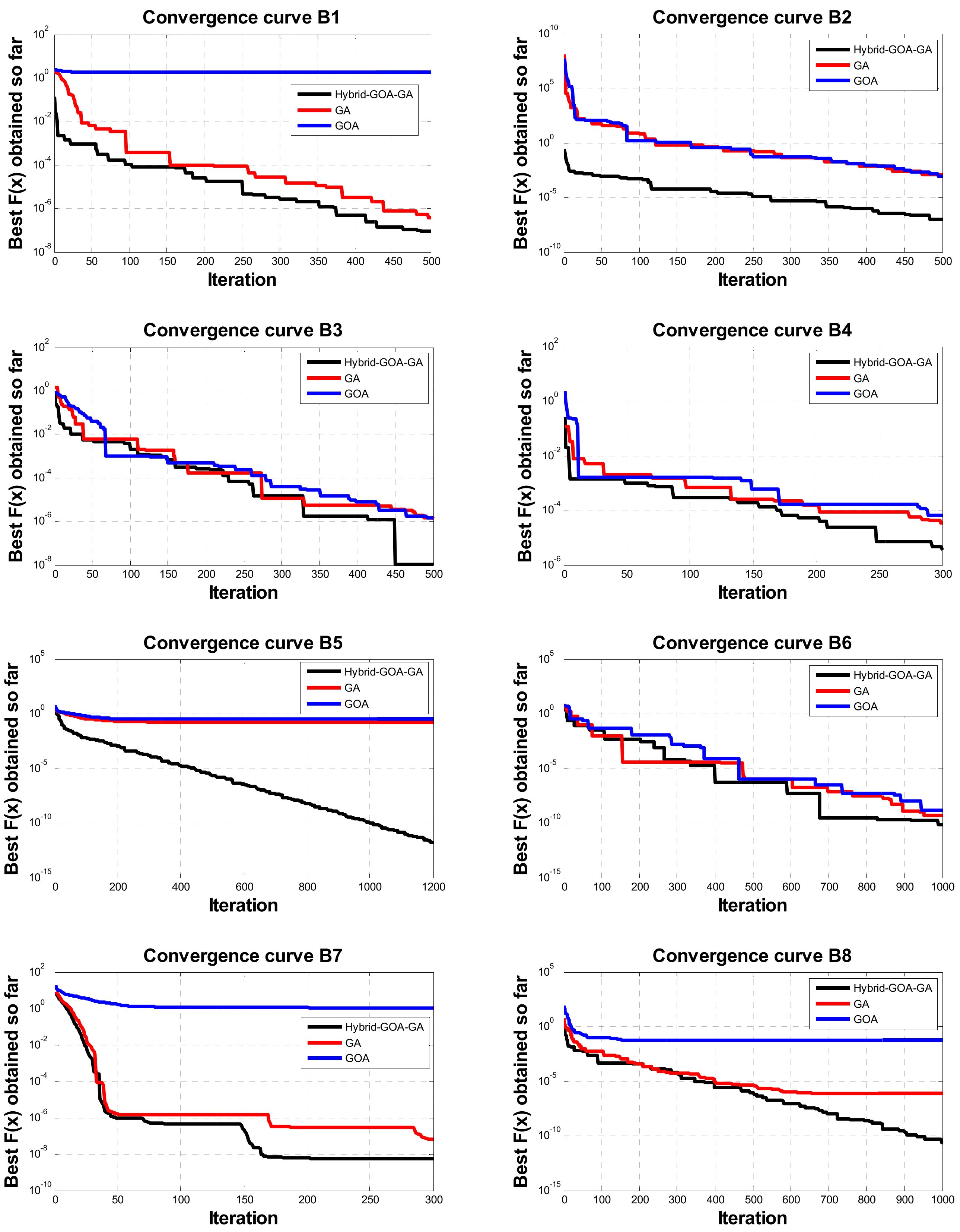


FIGURE 21. The convergence curves of the best F obtained by Hybrid-GOA-GA, GA, and GOA for all benchmark problems under the same conditions.

TABLE 17. **Comparison of required computational cost between two studies and Hybrid-GOA-GA.**

Benchmark	Grosan and Abraham [32]			Constrained GA [71]			Hybrid-GOA-GA		
	Population	Generation (Iteration)	CPU time(s)	Population	Generation	CPU time(s)	Population	Generation (Iteration)	CPU time(s)
Benchmark 1	NA	NA	NA	NA	NA	NA	10	500	1.22E-002
Benchmark 2	NA	NA	NA	NA	NA	NA	10	500	3.98E-002
Benchmark 3	NA	NA	NA	NA	NA	NA	10	500	2.49E-002
Benchmark 4	250	150	5.14	10	10	9.7E-002	10	300	1.54E-002
Benchmark 5	500	300	39.07	20	10	2.02	10	1200	2.008
Benchmark 6: Fluid mechanics	NA	NA	NA	10	10	9.97E-002	10	1000	7.73E-002
Benchmark 7: Combustion	500	300	151.12	40	70	41.18	10	300	13.25
Benchmark 8: Neurophysiology	300	200	28.9	10	20	2.28	10	1000	1.49
Configuration	2.4-GHz Intel Duo Core CPU with 2-GB RAM			2.6-GHz Intel Duo Core i7 with 3-GB RAM			2.27 GHz Intel(R) core (TM) i5 with 6-GB RAM		

NA'' means that the result is not available

TABLE 18. **The comparison results between Hybrid-GOA-GA, GA, and GOA OF the best function value *F* for all benchmarks problem under the same conditions.**

	Hybrid-GOA-GA		Original GA		Original GOA	
	Best function value F	CPU time(s)	Best function value F	CPU time(s)	Best function value F	CPU time(s)
Benchmark 1	9.335964490375659e-08	1.15E-002	3.869776437070982e-07	10.77	1.780852528919779	12.56
Benchmark 2	9.15141820101439e-08	3.53E-002	1.29985183119567e-03	9.23	8.73143984527536e-04	8.45
Benchmark 3	1.04105449261205e-08	2.75E-002	1.34198781612249e-06	8.36	1.50415595218112e-06	9.03
Benchmark 4	3.53232353555999e-06	1.68E-002	3.31995714273892e-05	7.25	6.73262630843108e-05	6.98
Benchmark 5	1.70197794659849e-12	2.15	0.180984099759073	8.65	0.322954187877703	7.08
Benchmark 6	6.44879705191670e-11	7.85E-002	4.94866370104319e-10	2.54	1.51797330261161e-09	3.14
Benchmark 7	5.672647019819165e-09	12.87	6.55986316446494e-08	20.06	1.10674631250539	5.36
Benchmark 8	2.46962954485145e-11	1.97	7.01249556436421e-07	9.63	0.0536470576751199	13.47

three algorithms. We can see that Hybrid-GOA-GA outperformed the original GA and original GOA concerning the absolute value function F, convergence to the best solution of SNLEs, and CPU time. Also, both GA and GOA are stuck in a local optimum in some cases. For GOA, it is trapped in local optima in benchmarks No. 1, 5, 7, and 8. While, for GA, it is trapped in local optima in Benchmark No. 5. These results reinforce our motivation to introduce a hybrid algorithm between GOA and GA.

Comparative studies have been performed in this section to determine the proposed hybrid algorithm efficiency of the solution. First of all, PBAs are suffering from the consistency of the solution. Therefore, the proposed hybrid algorithm was used to increase the quality of the solutions by integrating the merits of two PBAs. On the other hand, unlike deterministic algorithms our method searches by a population of points, not a single point. So, it can provide a globally optimal solution. In addition, our algorithm uses only objective function information, not derivatives or other auxiliary knowledge. So, it can deal with the problems of non-smooth, non-continuous, and non-differentiable optimization that currently occur in real-life applications. Also, the convergence of deterministic algorithms and their performance properties can be highly sensitive to the initial guess of the solution provided. However, it is very difficult to pick a good initial estimate for most SNLEs. Another positive observation is that the results of the simulation show that the proposed hybrid is superior to those stated in the literature, as it is better than both GA, GOA, and other PBAs in terms of accuracy, the effect of changing initial intervals, and computational cost. The reason for this is due to the integration between GOA (exploration ability) and GA (exploitation ability) which makes solutions sufficiently diverse. Moreover, due to its simplicity, the new hybrid algorithm tackles higher-order non-linear equations that we mostly face in the engineering field such as weather forecasting, geological oil exploration, computational mechanics and control fields, etc. Finally, it can be said that Hybrid-GOA-GA is competitive and able to solve SNLEs efficiently.

VI. CONCLUSION

In this article, we proposed a hybrid grasshopper algorithm (GOA) with a genetic algorithm (GA) to solve the system of non-linear equations (SNLEs). Hybrid-GOA-GA integrates the merits of both GOA and GA; where it combines the exploitation capability for GA and exploration capability for GOA. In the hybrid-GOA-GA, a population of random solutions is initialized. These solutions, by GOA, are searching in the domain of the optimization problem to obtain an optimal solution for SNLE. During this process, an evolution of these solutions is performed by GA. The optimization problem is configured from a system of non-linear equations (SNLEs). Then, this optimization problem is solved by the hybrid algorithm. Eight benchmarks problems with different applications are considered. The main research findings of the proposed algorithm mention as follows:

1) Using GOA with GA strikes a good balance between exploration and exploitation capabilities, improving the performance of the proposed algorithm.
2) Combining GOA with GA avoids the trapping into local minima, accelerates the seeking operation, and speeds the convergence to the best solution of SNLEs.
3) Hybrid-GOA-GA can be used to handle large-scale SNLEs due to its simple procedures.
4) Numerical results have proven the superiority of the proposed algorithm over those reported in the literature, as it is significantly better than other comparison methods and finds the best solution with high accuracy for the eight benchmark problems.
5) The proposed algorithm is not affected by changing the initial intervals of the SNLEs and gives better results than other algorithms in most cases.
6) In terms of time, the proposed algorithm is acceptable.

The potential weakness of the proposed algorithm, as all population-based approaches (PBAs), is that the guarantee of improvement in the computational speed or accuracy is not guaranteed when solving any optimization problem. This due to that PBAs are random approaches. So, in future works, other large-scale and more complex SNLEs can be considered to test the ability of the algorithm and know its potential weakness. Also, if there are any potential weaknesses, the proposed hybrid algorithm can be improved by using a chaotic local search. It is also suggested that other optimization approaches, such as sine cosine algorithm (SCA), salp swarm algorithm (SSA), gradient-based optimiser (GBO), slime mould algorithm and harris hawks optimization (HHO) be used to solve SNLEs.

ACKNOWLEDGMENT

The authors would like to thank the referees for their positive comments and recommendations, which have helped to clarify the arguments and improve the structure of this article.

REFERENCES

[1] A. A. M. Cuyt, "Computational implementation of the multivariate halley method for solving nonlinear systems of equations," *ACM Trans. Math. Softw.*, vol. 11, no. 1, pp. 20–36, 1985.

[2] J. J. More and M. Y. Cosnard, "Numerical solution of nonlinear-equations," *ACM Trans. Math. Softw.*, vol. 5, no. 1, pp. 64–85, 1979.

[3] A. Eiger, K. Sikorski, and F. Stenger, "A bisection method for systems of nonlinear equations," *ACM Trans. Math. Softw.*, vol. 10, no. 4, pp. 367–377, Dec. 1984.

[4] P. Deuflhard, *Newton Techniques for Highly Nonlinear Problems Theory and Algorithms*. New York, NY, USA: Academic, 1996.

[5] T. A. Jeeves, "Secant modification of Newton's method," *Commun. ACM*, vol. 1, no. 8, pp. 9–10, Aug. 1958.

[6] P. Van Hentenryck, D. McAllester, and D. Kapur, "Solving polynomial systems using a branch and prune approach," *SIAM J. Numer. Anal.*, vol. 34, no. 2, pp. 797–827, Apr. 1997.

[7] P.-Y. Nie, "An SQP approach with line search for a system of nonlinear equations," *Math. Comput. Model.*, vol. 43, nos. 3–4, pp. 368–373, Feb. 2006.

[8] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Philadelphia, PA, USA: SIAM, 1996.

[9] A. R. Conn, N. I. M. Gould, and P. Toint, "A globally convergent augmented lagrangian algorithm for optimization with general constraints and simple bounds," *SIAM J. Numer. Anal.*, vol. 28, no. 2, pp. 545–572, Apr. 1991.

[10] J. D. Hoffman, *Numerical Methods for Engineers and Scientists*, 2nd ed. New York, NY, USA: Marcel Dekker, 2001.

[11] A. A. Mousa, M. A. El-Shorbagy, and M. A. Farag, "K-means-clustering based evolutionary algorithm for multi-objective resource allocation problems," *Appl. Math. Inf. Sci.*, vol. 11, no. 6, pp. 1–12, 2017.

[12] A. M. Abdelsalam and M. A. El-Shorbagy, "Optimization of wind turbines siting in a wind farm using genetic algorithm based local search," *Renew. Energy*, vol. 123, pp. 748–755, Aug. 2018.

[13] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA, USA: MIT Press, 2004.

[14] M. A. El-Shorbagy and A. E. Hassanien, "Particle swarm optimization from theory to applications," *Int. J. Rough Sets Data Anal.*, vol. 5, no. 2, pp. 1–24, 2018.

[15] M. A. E. Shorbagy and A. A. Mousa, "Chaotic particle swarm optimization for imprecise combined economic and emission dispatch problem," *Rev. Inf. Eng. Appl.*, vol. 4, no. 1, pp. 20–35, 2017.

[16] A. A. A. Mousa and M. A. El-Shorbagy, "Enhanced particle swarm optimization based local search for reactive power compensation problem," *Appl. Math.*, vol. 03, no. 10, pp. 1276–1284, 2012.

[17] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Dept. Comput. Eng., Erciyes Univ., Kayseri, Turkey, Tech. Rep. TR06, 2005, vol. 200, pp. 1–10.

[18] W. Zhao and L. Wang, "An effective bacterial foraging optimizer for global optimization," *Inf. Sci.*, vol. 329, pp. 719–735, Feb. 2016.

[19] S.-C. Chu, P.-W. Tsai, and J.-S. Pan, "Cat swarm optimization," in *Proc. Pacific Rim Int. Conf. Artif. Intell.* Berlin, Germany: Springer, 2006, pp. 854–858.

[20] M. Marinaki and Y. Marinakis, "A glowworm swarm optimization algorithm for the vehicle routing problem with stochastic demands," *Expert Syst. Appl.*, vol. 46, pp. 145–163, Mar. 2016.

[21] M. A. Elsisy, D. A. Hammad, and M. A. El-Shorbagy, "Solving interval quadratic programming problems by using the numerical method and swarm algorithms," *Complexity*, vol. 2020, pp. 1–11, Sep. 2020.

[22] Y. Zhou, X. Chen, and G. Zhou, "An improved monkey algorithm for a 0-1 knapsack problem," *Appl. Soft Comput.*, vol. 38, pp. 817–830, Jan. 2016.

[23] A. L. Bolaji, M. A. Al-Betar, M. A. Awadallah, A. T. Khader, and L. M. Abualigah, "A comprehensive review: Krill herd algorithm (KH) and its applications," *Appl. Soft Comput.*, vol. 49, pp. 437–446, Dec. 2016.

[24] Y. Abo-elnaga and M. A. El-Shorbagy, "Multi-sine cosine algorithm for solving nonlinear bilevel programming problems," *Int. J. Comput. Intell. Syst.*, vol. 13, no. 1, pp. 421–432, 2020.

[25] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017.

[26] M. Shehab, A. T. Khader, M. Laouchedi, and O. A. Alomari, "Hybridizing cuckoo search algorithm with bat algorithm for global numerical optimization," *J. Supercomput.*, vol. 75, no. 5, pp. 2395–2422, May 2019.

[27] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm: A bio-inspired optimizer for engineering design problems," *Adv. Eng. Softw.*, vol. 114, pp. 163–191, Dec. 2017.

[28] I. Ahmadianfar, O. Bozorg-Haddad, and X. Chu, "Gradient-based optimizer: A new Metaheuristic optimization algorithm," *Inf. Sci.*, vol. 540, pp. 131–159, Nov. 2020.

[29] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: A new method for stochastic optimization," *Future Gener. Comput. Syst.*, vol. 111, pp. 300–323, Oct. 2020.

[30] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: Algorithm and applications," *Future Gener. Comput. Syst.*, vol. 97, pp. 849–872, Aug. 2019.

[31] W.-D. Chang, "An improved real-coded genetic algorithm for parameters estimation of nonlinear systems," *Mech. Syst. Signal Process.*, vol. 20, no. 1, pp. 236–246, Jan. 2006.

[32] C. Grosan and A. Abraham, "A new approach for solving nonlinear equations systems," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 38, no. 3, pp. 698–714, May 2008.

[33] H. Ren, L. Wu, W. Bi, and I. K. Argyros, "Solving nonlinear equations system via an efficient genetic algorithm with symmetric and harmonious individuals," *Appl. Math. Comput.*, vol. 219, no. 23, pp. 10967–10973, Aug. 2013.

[34] Y. Mo, H. Liu, and Q. Wang, "Conjugate direction particle swarm optimization solving systems of nonlinear equations," *Comput. Math. Appl.*, vol. 57, nos. 11–12, pp. 1877–1882, 2009.

[35] M. Jaberipour, E. Khorram, and B. Karimi, "Particle swarm algorithm for solving systems of nonlinear equations," *Comput. Math. Appl.*, vol. 62, no. 2, pp. 566–576, 2011.

[36] R. Jia and D. He, "Hybrid artificial bee colony algorithm for solving nonlinear system of equations," in *Proc. 8th Int. Conf. Comput. Intell. Secur.*, 2012, pp. 56–60.

[37] R. H. Zhou and Y. G. Li, "An improve cuckoo search algorithm for solving nonlinear equation group," *Appl. Mech. Mater.*, vols. 651–653, pp. 2121–2124, Sep. 2014.

[38] M. K. A. Ariyaratne, T. G. I. Fernando, and S. Weerakoon, "Solving systems of nonlinear equations using a modified firefly algorithm (MODFA)," *Swarm Evol. Comput.*, vol. 48, pp. 72–92, Aug. 2019.

[39] M. A. El-Shorbagy, A. A. Mousa, and W. Fathi, *Hybrid Particle Swarm Algorithm for Multiobjective Optimization: Integrating Particle Swarm Optimization With Genetic Algorithms for Multiobjective Optimization*. Saarbrücken, Germany: Lambert Academic, 2011.

[40] M. A. El-Shorbagy, "Hybrid particle swarm algorithm for multi-objective optimization," M.S. thesis, Dept. Basic Eng. Sci., Menoufia Univ., Al Minufiyah, Egypt, 2010.

[41] R. Goel and R. Maini, "A hybrid of ant colony and firefly algorithms (HAFA) for solving vehicle routing problems," *J. Comput. Sci.*, vol. 25, pp. 28–37, Mar. 2018.

[42] A. Al Malki, M. M. Rizk, M. A. El-Shorbagy, and A. A. Mousa, "Hybrid genetic algorithm with K-means for clustering problems," *Open J. Optim.*, vol. 5, no. 2, pp. 8–71, 2016.

[43] S. Nasr, M. El-Shorbagy, I. El-Desoky, Z. Hendawy, and A. Mousa, "Hybrid genetic algorithm for constrained nonlinear optimization problems," *Brit. J. Math. Comput. Sci.*, vol. 7, no. 6, pp. 466–480, Jan. 2015.

[44] S. S. Jadon, R. Tiwari, H. Sharma, and J. C. Bansal, "Hybrid artificial bee colony algorithm with differential evolution," *Appl. Soft Comput.*, vol. 58, pp. 11–24, Sep. 2017.

[45] B. Turanoğlu and G. Akkaya, "A new hybrid heuristic algorithm based on bacterial foraging optimization for the dynamic facility layout problem," *Expert Syst. Appl.*, vol. 98, pp. 93–104, May 2018.

[46] V. I. Skoullis, I. X. Tassopoulos, and G. N. Beligiannis, "Solving the high school timetabling problem using a hybrid cat swarm optimization based algorithm," *Appl. Soft Comput.*, vol. 52, pp. 277–289, Mar. 2017.

[47] X. Chen, Y. Zhou, Z. Tang, and Q. Luo, "A hybrid algorithm combining glowworm swarm optimization and complete 2-opt algorithm for spherical travelling salesman problems," *Appl. Soft Comput.*, vol. 58, pp. 104–114, Sep. 2017.

[48] İ. B. Aydilek, "A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems," *Appl. Soft Comput.*, vol. 66, pp. 232–249, May 2018.

[49] M. K. Marichelvam, Ö. Tosun, and M. Geetha, "Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time," *Appl. Soft Comput.*, vol. 55, pp. 82–92, Jun. 2017.

[50] L. M. Abualigah, A. T. Khader, E. S. Hanandeh, and A. H. Gandomi, "A novel hybridization strategy for krill herd algorithm applied to clustering techniques," *Appl. Soft Comput.*, vol. 60, pp. 423–435, Nov. 2017.

[51] M. A. El-Shorbagy, M. A. Farag, A. A. Mousa, and I. M. El-Desoky, "A hybridization of sine cosine algorithm with steady state genetic algorithm for engineering design problems," in *Proc. Int. Conf. Adv. Mach. Learn. Technol. Appl. (AMLTA), Adv. Intell. Syst. Comput.*, vol. 921. Cham, Switzerland: Springer, 2020, pp. 143–155.

[52] D. Wang, H. Chen, T. Li, J. Wan, and Y. Huang, "A novel quantum grasshopper optimization algorithm for feature selection," *Int. J. Approx. Reasoning*, vol. 127, pp. 33–53, Dec. 2020.

[53] P. Singh and S. Prakash, "Optimizing multiple ONUs placement in fiber-wireless (FiWi) access network using grasshopper and harris hawks optimization algorithms," *Opt. Fiber Technol.*, vol. 60, Dec. 2020, Art. no. 102357.

[54] S. Dwivedi, M. Vardhan, and S. Tripathi, "An effect of chaos grasshopper optimization algorithm for protection of network infrastructure," *Comput. Netw.*, vol. 176, Jul. 2020, Art. no. 107251.

[55] F. Raeesi, B. F. Azar, H. Veladi, and S. Talatahari, "An inverse TSK model of MR damper for vibration control of nonlinear structures using an improved grasshopper optimization algorithm," *Structures*, vol. 26, pp. 406–416, Aug. 2020.

[56] A. A. Ewees, M. A. Elaziz, Z. Alameer, H. Ye, and Z. Jianhua, "Improving multilayer perceptron neural network using chaotic grasshopper optimization algorithm to forecast iron ore price volatility," *Resour. Policy*, vol. 65, Mar. 2020, Art. no. 101555.

[57] R. Purushothaman, S. P. Rajagopalan, and G. Dhandapani, "Hybridizing gray wolf optimization (GWO) with grasshopper optimization algorithm (GOA) for text feature selection and clustering," *Appl. Soft Comput.*, vol. 96, Nov. 2020, Art. no. 106651.

[58] A. A. Ewees, M. Abd Elaziz, and E. H. Houssein, "Improved grasshopper optimization algorithm using opposition-based learning," *Expert Syst. Appl.*, vol. 112, pp. 156–172, Dec. 2018.

[59] M. M. A. Alphonsa and N. MohanaSundaram, "A reformed grasshopper optimization with genetic principle for securing medical data," *J. Inf. Secur. Appl.*, vol. 47, pp. 410–420, Aug. 2019.

[60] J. H. Holland, *Adaptation in Natural and Artificial Systems*, 1st ed. Cambridge, MA, USA: MIT Press, 1975.

[61] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA, USA: Addison-Wesley, 1989.

[62] M. A. Farag, M. A. El-Shorbagy, I. M. El-Desoky, A. A. El-Sawy, and A. A. Mousa, "Binary-real coded genetic algorithm based k-means clustering for unit commitment problem," *Appl. Math.*, vol. 6, no. 11, pp. 1873–1890, 2015.

[63] I. M. El-Desoky, M. A. El-Shorbagy, S. M. Nasr, Z. M. Hendawy, and A. A. Mousa, "A hybrid genetic algorithm for job shop scheduling problems," *Int. J. Advancement Eng., Technol. Comput. Sci.*, vol. 3, no. 1, pp. 6–17, 2016.

[64] M. A. El-Shorbagy, A. Y. Ayoub, A. A. Mousa, and I. M. El-Desoky, "An enhanced genetic algorithm with new mutation for cluster analysis," *Comput. Statist.*, vol. 34, no. 3, pp. 1355–1392, Sep. 2019.

[65] M. A. El-Shorbagy, A. A. Mousa, and M. A. Farag, "An intelligent computing technique based on a dynamic-size subpopulations for unit commitment problem," *OPSEARCH*, vol. 56, no. 3, pp. 911–944, Sep. 2019.

[66] C. Mangla, H. Bhasin, M. Ahmad, and M. Uddin, "Novel solution of nonlinear equations using genetic algorithm," in *Industrial Mathematics and Complex Systems* (Industrial and Applied Mathematics), P. Manchanda, R. Lozi, A. Siddiqi, Eds. Singapore: Springer, 2017.

[67] A. Rovira, M. Valdés, and J. Casanova, "A new methodology to solve non-linear equation systems using genetic algorithms. Application to combined cycle gas turbine simulation," *Int. J. Numer. Methods Eng.*, vol. 63, pp. 1424–1435, 2005.

[68] Z. Ji, Z. Li, and Z. Ji, "Research on genetic algorithm and data information based on combined framework for nonlinear functions optimization," *Procedia Eng.*, vol. 23, pp. 155–160, 2011.

[69] G. Joshi and M. B. Krishna, "Solving system of non-linear equations using genetic algorithm," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2014, pp. 1302–1308.

[70] N. E. Mastorakis, "Solving non-linear equations via genetic algorithms," in *Proc. 6th WSEAS Int. Conf. Evol. Comput.*, Lisbon, Portugal, 2005, pp. 24–28.

[71] A. Pourrajabian, R. Ebrahimi, M. Mirzaei, and M. Shams, "Applying genetic algorithms for solving nonlinear algebraic equations," *Appl. Math. Comput.*, vol. 219, no. 24, pp. 11483–11494, Aug. 2013.

[72] M. A. Z. Raja, Z. Sabir, N. Mehmood, E. S. Al-Aidarous, and J. A. Khan, "Design of stochastic solvers based on genetic algorithms for solving nonlinear equations," *Neural Comput. Appl.*, vol. 26, no. 1, pp. 1–23, Jan. 2015.

[73] P.-Y. Nie, "A null space method for solving system of equations," *Appl. Math. Comput.*, vol. 149, no. 1, pp. 215–226, Feb. 2004.

[74] L. Chen, J. McPhee, and W. G. Yeh, "A diversified multi objective GA for optimizing reservoir rule curves," *Adv. Water Resour.*, vol. 30, no. 5, pp. 1082–1093, 2007.

[75] M. A. E. Shorbagy, A. A. Mousa, and M. Farag, "Solving nonlinear single-unit commitment problem by genetic algorithm based clustering technique," *Rev. Comput. Eng. Res.*, vol. 4, no. 1, pp. 11–29, 2017.

[76] N. Soni and T. Kumar, "Study of various crossover operators in genetic algorithms," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 6, pp. 7235–7238, 2014.

[77] N. Soni and T. Kumar, "Study of various mutation operators in genetic algorithms," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 3, pp. 4519–4521, 2014.

[78] F. M. White, *Fluid Mechanics*, 4th ed. New York, NY, USA: McGraw-Hill, 2001.

M. A. EL-SHORBAGY was born in Menoufia, Egypt, in 1982. He received the B.Sc. degree in electrical engineering, the M.Sc. degree in engineering mathematics, and the Ph.D. degree in engineering mathematics from the Faculty of Engineering, Menoufia University, Shebin El-Kom, Egypt, in 2004, 2010, and 2013, respectively. He is currently an Assistant Professor with the Department of Mathematics, College of Science and Humanities in Al-Kharj, Prince Sattam bin Abdulaziz University, Al-Kharj, Saudi Arabia. His current research interests include swarm intelligence, artificial intelligence, evolutionary algorithms, numerical optimization and engineering optimization, and cluster analysis.

ADEL M. EL-REFAEY was born in Menoufia, Egypt, in 1975. He received the B.Sc. degree in electrical engineering, the M.Sc. degree in engineering mathematics, and the Ph.D. degree in engineering mathematics from the Faculty of Engineering, Menoufia University, Shebin El-Kom, Egypt, in 1998, 2006, and 2011, respectively. He is currently an Assistant Professor and the Head of the Department of Basic and applied Science (Smart Village), Arab Academy for Science & Technology (AASTMT), Cairo, Egypt. His current research interests include swarm intelligence, artificial intelligence, evolutionary algorithms, numerical optimization, and engineering optimization.