This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3448295

IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.Doi Number

Chaotic Noise-Based Particle Swarm Optimization Algorithm for Solving System of Nonlinear Equations

M.A. El-Shorbagy

Department of Mathematics, College of Science and Humanities in Al-Kharj, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia.
Department of Basic Engineering Science, Faculty of Engineering, Menoufia University, Shebin El-Kom 32511, Egypt.

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

This work was supported by Prince Sattam bin Abdulaziz University under Project PSAU/2024/01/28963.

ABSTRACT The effective solution of system of nonlinear equations (SNEs) is crucial for the creation of engineering and scientific models. SNEs can be represented and analyzed as an optimization problem. The objective of this research is to introduce a new optimization technique, called Chaotic Noise-Based Particle Swarm Optimization Algorithm (CN-BPSOA), to effectively address the SNEs. CN-BPSOA is a particle swarm optimization algorithm (PSOA) that utilizes chaotic noise to enhance its performance. The configuration of CN-BPSOA will involve the utilization of a novel definition, namely chaotic noise, to address the limitations associated with optimization methods. These limitations include the insufficient variety of solutions, the imbalance between exploiting current solutions and exploring new ones, and the sluggish convergence towards the optimal solution. The objective of chaotic noise is to minimize the occurrence of duplicated solutions and iterations in order to accelerate the rate of convergence. The chaotic logistic map is employed due to its widespread adoption by researchers and its demonstrated efficacy in enhancing solution quality and optimizing performance. CN-BPSOA is evaluated utilizing numerous renowned SNEs. The performance of the proposed method is compared with five other algorithms that also solve the same benchmark issues and with the results of the original PSOA to emphasize the significance of the modifications made in CN-BPSOA.CN-BPSOA's effectiveness in addressing SNEs was demonstrated by the promising findings that obtained, where the best solution obtained by CN-BPSOA is less than that obtained by all algorithms by improvement percentage (percentage drop (PD%)) more than or equal to 93.30% in all benchmark's problems. Ultimately, while comparing the outcomes of CN-BPSOA with those of prior investigations, the application of statistical analysis through Friedman and Wilcoxon's tests conclusively revealed its better performance and efficacy in resolving this kind of problem.

INDEX TERMS Particle Swarm Optimization, Chaotic Noise, System of Nonlinear Equations, Optimization.

I. INTRODUCTION

The solution of system of nonlinear equations (SNEs) is crucial for the advancement of engineering and science, as many models in these domains rely on them. SNEs can be directly present in some applications, or they can be indirectly derived from practical models [1]. Discovering a resilient and efficient solution for the SNEs may pose a challenging endeavor in theory.

Solving SNEs typically involves the use of Muller's method, the false-position method, the bisection technique, the Levenberg-Marquardt algorithm, steepest descent methods, the Broyden method, Halley's method, the branch and prune approach, Newton/damped Newton methods, and the Secant method[2]. The methods of choice for solving SNEs in general are secant and Newton methods. Alternatively, several methods convert the SNEs into an optimization problem [3], which is then resolved using the enhanced Lagrangian method [4]. These methods have some limitations such as time-consuming, may diverge, lack of efficiency in solving a collection of nonlinear equations, necessity for a laborious procedure to compute

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3448295

partial derivatives for constructing the Jacobian matrix, and its sensitivity to the initial conditions [5].

Due to these limitations, the researchers employed evolutionary algorithms (EAs) to solve SNEs. Evolutionary algorithms (EAs) are commonly employed to tackle optimization issues that are deemed too challenging for conventional approaches. Evolutionary algorithms, such as the genetic algorithm (GA) [6-8], particle swarm optimization algorithm (PSOA) [9,10], artificial bee colony (ABC) [11], cuckoo search algorithm (CSA) [12], and firefly algorithm (FA) [13], have been employed for the resolution of SNEs. In [6], Chang introduced a real-coded genetic algorithm (GA) as a solution for the SNEs. In [7], Grosan and Abraham proposed a new method using GA to address the challenge of complicated SNEs, where they achieved this by reformulating the problem as a multiobjective optimization problem. The researchers in [8] employed a highly effective GA that utilized symmetric and harmonic individuals to successfully address SNEs. Mo and Liu [9] introduced a conjugate direction approach to PSOA in order to tackle SNEs, where they incorporated the conjugate direction method (CDM) into PSOA, resulting in an improved algorithm capable of efficiently optimizing high-dimensional problems. CDM assists PSOA in circumventing local minima by transforming the problem of optimizing high-dimensional functions into a lower-dimensional space. Jaberipour et al. proposed an innovative approach of updating the location and velocity of each particle in a new version of PSOA for solving SNEs [10]. Also, to address the limitations of the traditional PSOA, such as being stuck in local minimums and experiencing slow convergence, the researchers modified the manner in which each particle was updated. In addition, Jia and He introduced a hybrid ABC technique in [11], which integrated the ABC and PSO algorithms to solve SNEs. The hybrid algorithm resolves the issue of becoming stuck in a premature or local optimum by combining the advantages of both algorithms. Zhou and Li introduced an enhanced CSA in [12] to address the SNEs, where they utilized an innovative encoding method that guarantees the attainability of the proposed solution without necessitating any alteration to the evolution of the cuckoo. In [13], Ariyaratne et al. were present an improved version of the Firefly Algorithm (FA) that tackles SNEs as an optimization problem, where this approach offers various benefits such as that no need to the initial conditions, differentiation, and the requirement for function continuity. Additionally, it allows for the generation of several root estimates simultaneously.

Based on swarm intelligence, the particle swarm optimization algorithm (PSOA) is a population-based, stochastic computer method. Based on social psychology concepts, swarm intelligence contributes to engineering applications as well as offering insights into social behavior. Russell C. Eberhart and James Kennedy first described the PSO approach in 1995 [14]. PSOA, like other evolutionary algorithms, starts with a population of individuals described as random guesses for problem-solving. Individuals from this population are considered potential solutions. They are also referred to as particles, hence the term particle swarm. An iterative procedure for improving these candidate solutions is initiated. The particles iteratively assess the fitness of the candidate solutions and remember where they had the most success. The particle best, also known as the local best, is an individual's best solution. Each particle makes this information known to its neighbors. They can also notice where their neighbors have found success. These successes serve as a guide for population movements in the search space, and at the end of a trial, the population typically converges to the solution of the problem. PSOA was proposed and has been applied to various application fields. The PSOA is a commonly used technique and has been used for hybrid models [15,16].

While PSOA can quickly identify good solutions, it may become stuck in local optimum and not reach the global optimum. As a result, PSOA has numerous disadvantages, including extreme slowness and difficulty in finding the global optimal solution because of the high number of iterations or lengthy search time. Motivated by this, this paper presents an algorithm that addresses a major limitation of PSOA and other evolutionary algorithms (EAs), namely the repetition of solutions during the optimization process, resulting in time wastage. The optimization algorithm being developed is called a Chaotic Noise-Based Particle Swarm Optimization Algorithm (CN-BPSOA). Chaotic is a mathematical method that has demonstrated the ability to enhance the efficiency of many optimization algorithms. The subject has garnered significant interest and has been implemented in various fields, such as optimization [17-20]. The suggested CN-BPSOA is a hybridization of PSOA with chaotic noise. Chaotic noise is employed in the optimization process of a PSOA to introduce random and unpredictable changes to the positions of the solutions when they are repeated. This combination seeks to improve the performance of PSOA by addressing its limitations, including the limited variety of solutions, the imbalance between using known solutions and exploring new ones, and the sluggish convergence towards the optimal solution. This paper's main contributions encompass:

1. In order to solve SNEs, this work presents a novel method called the chaotic noise-based particle swarm optimization algorithm (CN-BPSOA), which combines both PSOA and chaotic noise.
2. Providing a sufficient diversity of solutions and avoiding time wastage by eliminating repeating solutions during the optimization process.

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3448295

3. Striving to achieve continuous progress with each iteration in the optimization, ultimately reaching optimal solutions.
4. Evaluating the effectiveness of CN-BPSOA through multiple well-known SNEs.
5. Applying statistical tests to assess the significance of the CN-BPSOA findings.
6. Demonstrating the competitiveness and superiority of CN-BPSOA over alternative optimization techniques.

The structure of the paper is as follows: The definition of systems of nonlinear equations is presented in Section 2. The proposed algorithm is fully presented in Section 3. Section 4 contains the numerical results and discussions. A summary of the results and conclusions concludes Section 5.

II. SYSTEMS OF NONLINEAR EQUATIONS

The mathematical description of a system of nonlinear equations (SNEs) is as follows:

$$\text{SNEs} = \begin{cases} f_1(y) = 0 \\ f_2(y) = 0 \\ \vdots \\ f_Q(y) = 0 \end{cases}; \tag{1}$$

where $y = (y_1, y_2, \dots, y_n)$ is a vector of n components subset of $\mathbb{R}^n$, and $f_q \forall q = 1,2, \dots, Q$ are the nonlinear functions that translate the n-dimensional space $\mathbb{R}^n$'s vector $y = (y_1, y_2, \dots, y_n)$ to the real line. Certain functions may be linear, whereas others may not be. Finding a solution where each of the aforementioned Q functions equals zero is required to resolve SNEs [21].

Definition 1: If the Q functions satisfy the condition $f_q(y) = 0\ \forall q = 1, \dots, Q$, then the solution $y = (y_1^*, y_2^*, \dots, y_n^*)$ that is called the optimal solution of the SNEs.

Various approaches [22, 23] transform the SNEs into an optimization issue by incorporating the left-hand side of every equation and subsequently employing the absolute value function as:

$$F(y) = \text{abs}\left(f_1(y) + f_2(y) + \cdots + f_Q(y)\right)$$
$$\text{Subject to: } \begin{cases} f_1(y) = 0 \\ f_2(y) = 0 \\ \vdots \\ f_Q(y) = 0 \end{cases}; \tag{2}$$

where $F(y)$ represents the objective function. The objective function $F(y)$ in Equation (2) obtains a global optimal solution if all the nonlinear equations $\left(f_q = 0\ \forall q = 1, \dots, Q\right)$ are equal to zero.

III. THE PROPOSED ALGORITHM

This section offers a concise introduction to the concepts of particle swarm optimization algorithm (PSOA) and chaos theory. Then, the proposed chaotic noise-based particle swarm optimization algorithm (CN-BPSOA) will be presented in a comprehensive manner.

A. The particle swarm optimization algorithm (PSOA)

Bird flow's collective activity serves as an inspiration for PSOA [14]. Each particle (solution) uses two essential kinds of information when making decisions. The first is based on their personal experience; in other words, they have made the choices and are aware of which location has so far proven to be superior and how it was good. The second is the experience of other particles; in other words, they are aware of the actions of other particles in their vicinity. Every particle in the PSO system bases his decision on his personal experiences as well as those of other particles. There is initially a population of random solutions in the system. Every possible resolution, referred to as a particle (solution), is assigned an arbitrary speed and flies through the search domain. Because of their memory, the particles remember their prior best position (called the $p^{best,t}$) and the associated fitness. The particles in the swarm have a many $p^{best,t}$, and the particle that has the highest fitness at iteration t is referred to as the global best ($g^{best,t}$) particle of the swarm. In an n-dimensional space, every particle is handled as a single point. The representation of the i-th particle is $y_i = (y_i^1, y_i^2, \dots, y_i^n)$. $p_i^{best,t} = (p_i^1, p_i^2, \dots, p_i^n)$. represents the best previous position of the i-th particle that yields the best fitness value at iteration t. The best particle in the population at iteration t is denoted by $g^{best,t} = \left(g^1, g^2, \dots, g^n\right)$. The velocity, or rate of position change for particle i at iteration t, is written as $v_i^t = (v_i^1, v_i^2, \dots, v_i^n)$. The particles are managed using the following equations (the superscripts indicate the iteration):

$$v_i^{t+1} = w \times v_i^t + c_1 \times r_1 \times \left(p_i^{best,t} - y_i^t\right) + c_2 \times r_2 \times \left(g^{best,t} - y_i^t\right), \tag{3}$$

$$y_i^{t+1} = y_i^t + v_i^{t+1}; \tag{4}$$

where w is the inertia weight, $i = 1,2, \dots, N$ is the population size, c_1 and c_2 are two positive constants that represent the social and cognitive parameters, respectively, and r_1 and r_2 are random values that are uniformly distributed throughout the interval [0,1]. Equation (4) gives the new position of the i-th particle, y_i^{t+1}, by adding its new velocity, v_i^{t+1}, to its present position, y_i^t. Equation (3) is used to find the i-th particle's new velocity, v_i^{t+1}, at each iteration. Figure I displays the basic PSOA's pseudo code, while Figure II displays the general PSOA flowchart.

Initialize each particle's position and velocity at random.
Do:
  Set $p_i^{best} \forall i = 1, \dots, N$ and g^{best}.
  Particle velocity is computed using equation (3).
  Particle position is updated using equation (4).
  The fitness value is evaluated.
While: A satisfactory solution is identified.

FIGURE 1. The generic PSOA's pseudo-code

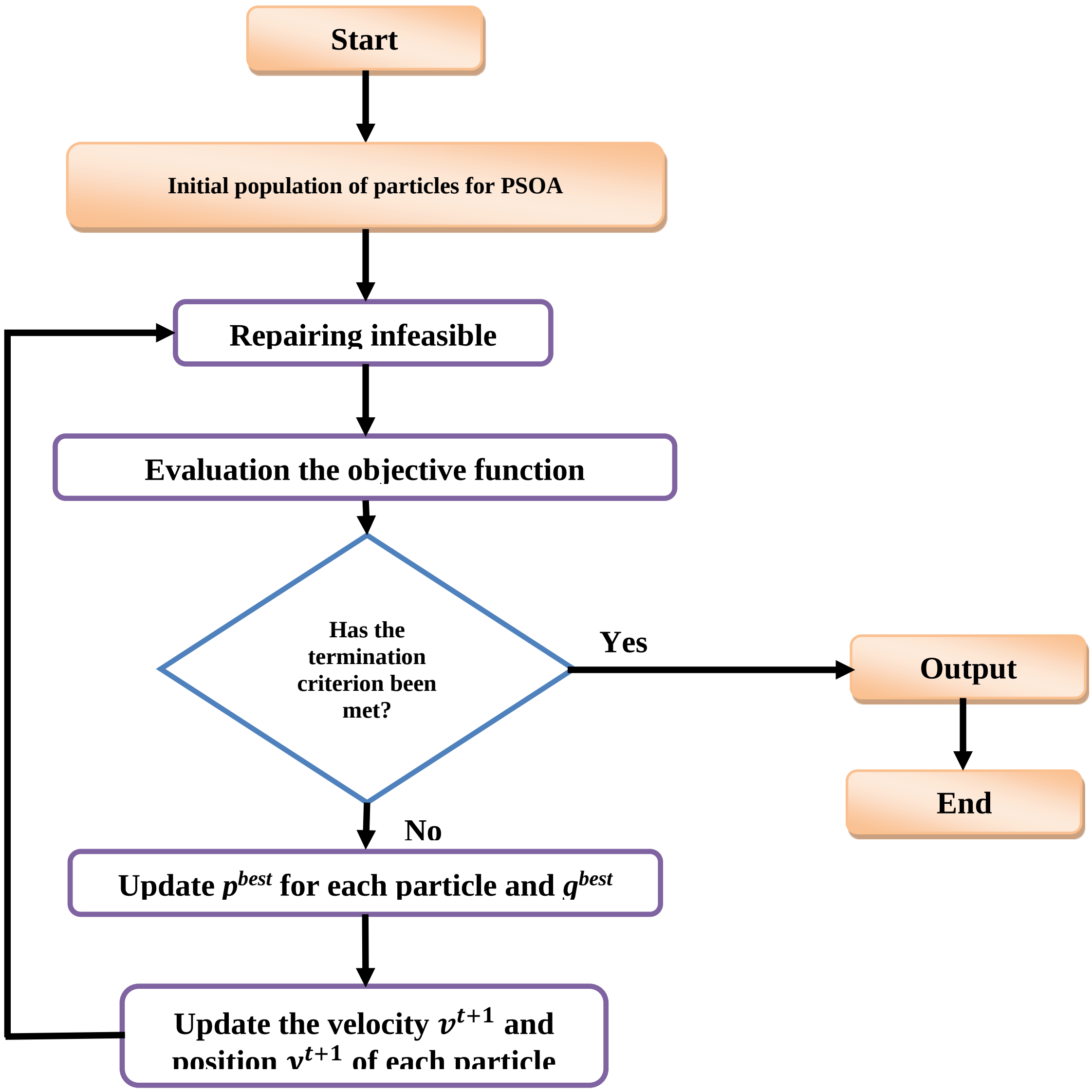

FIGURE 2. The generic PSOA's pseudo-code

B. CHAOS THEORY

The dynamics of systems that obey deterministic laws but appear random and unpredictable are the subject of chaos theory. Optimization sciences have benefited greatly from the applications of chaos theory mathematics in many different areas. Chaos optimization algorithms have gained significant interest as a good approach to global optimization due to their utilization of various chaotic maps. These techniques can improve optimization by making it easier to escape from local solutions and accelerating the convergence towards the global solution because of the intrinsic characteristics of chaotic maps. To enhance the quality of the solution, numerous academics have recommended the integration of chaos theory and optimization algorithms [24,25]. Mathematical functions that behave chaotically are called chaotic maps, and they are typically written as iterated functions. Numerous well-known chaotic maps, including the sinusoidal, Chebyshev, singer, tent, sine, circle, Gauss, and logistic maps, may be found in the literature [26].

C. Chaotic Noise-Based Particle Swarm Optimization

This section will provide a description of the proposed Chaotic Noise-Based Particle Swarm Optimization Algorithm (CN-BPSOA), which combines Particle Swarm Optimization Algorithm (PSOA) with chaos theory. CN-BPSOA is set to utilize chaotic noise to address certain constraints that may arise during optimization using PSOA. These shortcomings include a lack of alternative solutions, an unequal distribution of exploration and exploitation, and a slow convergence to the best solution. There are two main phases of CN-BPSOA. To solve the SNEs efficiently in the first step, the PSOA is applied as a global optimization technique. Chaotic noise is used as the second phase in the

PSOA optimization method if the best solution is repeated. By avoiding the repetition of the optimal solution and reducing the number of iterations, chaotic noise seeks to exhibit a broad variety of solutions without taking an excessive amount of time throughout the optimization process. An extensive description of the suggested algorithm may be found below:

Step 1. Initialization

- **Initialization:** At iteration *t*, initialize a population of particles on *n*-dimensions of the problem space with random positions and velocities.
- **Repair of infeasible particles:** To make the population's particles feasible, repair its unfeasible particles. Using this method, the population of unfeasible particles co-evolves until they are feasible. We produce new feasible particles (NF) on a segment delineated by two points: feasible particle (*F*) and infeasible particle (*INF*) (see [27]). New feasible particle NF is expressed as: $NF = \gamma \cdot INF + (1-\gamma) \cdot F,\ NF = \gamma \cdot F + (1-\gamma) \cdot INF;$ where $\gamma = (1+2\mu)\delta - \mu$, μ is a user specified parameter used to extend the segment equally on both sides of *F* and *INF*, and $\delta \in [0,1]$ is a random generated number.
- **Evaluation:** The fitness function $F(y)$ of *n* variables is evaluated for every particle
- **Setting $p^{best,t}$ and $g^{best,t}$:** Assign the current location and objective value of each particle as $p^{best,t}$, and set the position and objective value of the best particle among the whole population as $g^{best,t}$.

Step 2. Updating the particles

- **Updating the velocity and position:** Using Equations 3 and 4, update each particle's position and velocity.
- **Repair of infeasible particles**
- **Evaluation**
- **Updating $p^{best,t}$ and $g^{best,t}$:** Compare each particle's current objective value to its $p^{best,t-1}$ objective value. Update $p^{best,t}$ and its objective value with the current location and value if the current value is better. Find the current swarm particle with the best objective value. Update $g^{best,t}$ and its objective value with the position and objective value of the current best particle if the objective value is better than $g^{best,t-1}$. If $g^{best,t}$ repeated *M* times applied chaotic noise (step 3). Otherwise go to step 4.

Step 3. Chaotic Noise phase

- **Chaotic noise**: Chaotic noise is implemented on the PSOA optimization process when the best solution is repeatedly *M* times. It aims to demonstrate a sufficient diversity of solutions while minimizing the time required for optimization by avoiding the repeat of the best solution and decreasing the number of iterations. During this stage, the particles at iteration *t* is subjected to chaotic noise, resulting in alterations for particles' positions as:

$$y_i^t)_{\text{chaotic}} = \xi \cdot y_i^t \forall i = 1, \dots, N; \quad (5)$$

where the chaotic random number, denoted as ξ, is generated using the logistic map using the following equation:

$$\xi_k = 4\xi_{k-1}(1-\xi_{k-1}), \xi_0 \in (0,1),\ \xi_0 \notin \{0.0, 0.25, 0.50, 0.75, 1.0\}; \quad (6)$$

where *k* is iterations of chaotic noise phase.

- **Evaluation:** For each particle $y_i^t)_{\text{chaotic}}\ \forall i = 1, \dots, N$, $F(y)$ is evaluated to find the best new position $g^{best})_{\text{chaotic}}$.
- **Updating g^{best}**
  1. If the best new position $g^{best})_{\text{chaotic}}$ is better than the best position $g^{best,t}$, updating the best position $g^{best,t}$ as $g^{best,t} = g^{best})_{\text{chaotic}}$ and go to step 2. Otherwise, repeat step 3 *k* times.
  2. If the best new position $g^{best})_{\text{chaotic}}$ is not better than the best position $g^{best,t}$, chaotic noise is applied again with a new chaotic random number ξ that generated by Eq. (6).
  3. If $g^{best,t}$ is not improved after all chaotic noise k iterations, chaotic noise phase should be stopped, $g^{best,t}$ should be shown as the best solution and go to step 4.

Step 4. Termination criteria

CN-BPSOA terminates either when it reaches the maximum number of iterations t_{max} or if $g^{best,t}$ is not improved after all chaotic noise k iterations. Additionally, CN-BPSOA terminates if $\delta = \left| \|F_{optimum}\| - \|F_t\| \right| \le \varepsilon = 1e-20$; here, $\|F_{optimum}\|$ denotes the objective function's optimal value, which is 0 in all nonlinear systems benchmark problems. At each iteration *t*, the calculated objective function is denoted by $\|F_t\|$. In the end, determine that the best particle position, $g^{best,t}$, is the optimal solution. If not, proceed to step 2. Figure III shows the main flowchart for CN-BPSOA.

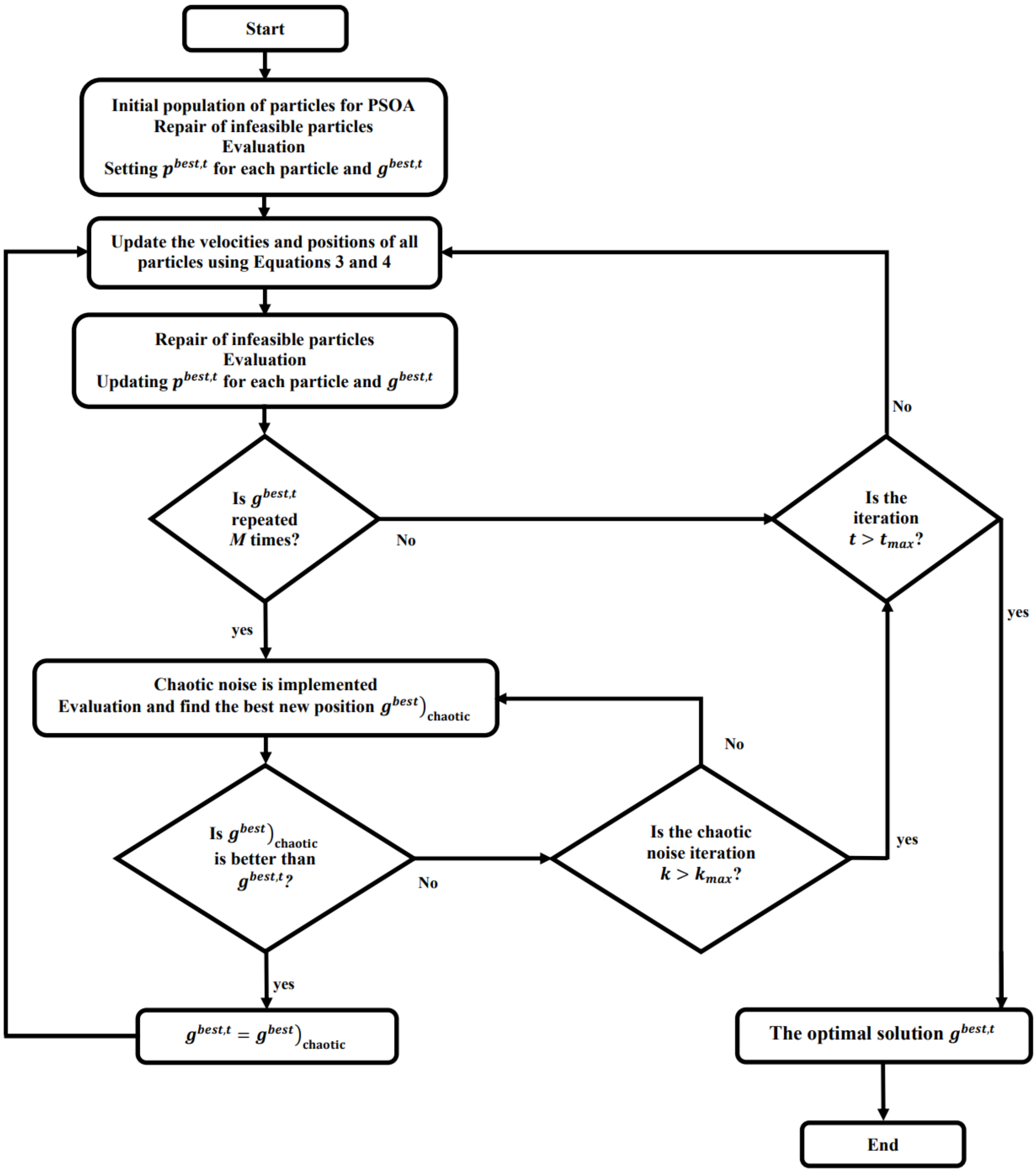

FIGURE 3. **The generic PSOA's pseudo-code**

IV. NUMERICAL RESULTS

Four systems of nonlinear equations are solved to assess the recommended approach. These four test systems, which have been extensively examined by other researchers, are known as benchmarks. On a PC with an Intel(R) Core (TM) i7-6600U CPU operating at 2.60GHz, the proposed CN-BPSOA is implemented in MATLAB R2012b. The computer is running Windows 10 and has 16 GB of RAM. The outcomes will be juxtaposed with the findings of the original PSOA to demonstrate the benefits of the suggested modifications and their impact on achieving the best possible outcome. The results of the proposed algorithm will also be compared with other algorithms that solved the same benchmark problems. The parameters used to execute the CN-BPSOA are displayed in Table I.

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3448295

TABLE I
THE PARAMETERS USED IN THE EXECUTION OF CN-BPSOA

Phase	Parameter	Notation	value
PSOA phase	Number of iterations	t_{max}	300-1200
	Population size	N	100 - 700
	Acceleration coefficients	c_1	2.8
		c_2	1.3
	The inertia weight	w	0.5
Chaotic noise phase	Number of chaotic noise iteration	k_{max}	30
	Initial chaotic random number	ξ_0	0.9
	The number of times $g^{best,t}$ is repeated	M	5

It is important to mention that both algorithms, the original PSOA and the proposed CN-BPSOA, have the same maximum number of iterations. Additionally, all data are recorded starting from the first run. Moreover, once one of them fulfills the termination condition, the calculations cease, and the count of iterations used is recorded. Also, the CN-BPSOA is statistically evaluated in comparison to other algorithms using the Friedman test and Wilcoxon rank-sum test.

A. *Benchmark problems for a SNEs*

The 4 Benchmark problems for a SNEs can be characterized as follows [7]:

1) A. BENCHMARK 1: EXPERIMENT TEST

$$\begin{cases} f_1(y_1,y_2) = cos(2y_1) - cos(2y_2) - 0.4 = 0, \\ f_2(y_1,y_2) = 2(y_2 - y_1) + sin(2y_2) - sin(2y_1) - 1.2 = 0, \\ y_1 \in [-10,10], \\ y_2 \in [-10,10]. \end{cases} \tag{7}$$

2) B. BENCHMARK 2: ARITHMETIC APPLICATION

$$\begin{cases} f_1(y) = y_1 - 0.254287220 - 0.18324757 \times y_4 y_3 y_9 = 0, \\ f_2(y) = y_2 - 0.378421970 - 0.16275449 \times y_1 y_{10} y_6 = 0, \\ f_3(y) = y_3 - 0.271625770 - 0.16955071 \times y_1 y_2 y_{10} = 0, \\ f_4(y) = y_4 - 0.198079140 - 0.15585316 \times y_7 y_1 y_6 = 0, \\ f_5(y) = y_5 - 0.441667280 - 0.19950920 \times y_7 y_6 y_3 = 0, \\ f_6(y) = y - 0.146541130 - 0.18922793 \times y_8 y_5 y_{10} = 0, \\ f_7(y) = y_7 - 0.429371610 - 0.21180486 \times y_2 y_5 y_8 = 0, \\ f_8(y) = y_8 - 0.070564380 - 0.17081208 \times y_1 y_7 y_6 = 0, \\ f_9(y) = y_9 - 0.345049060 - 0.19612740 \times y_{10} y_6 y_8 = 0, \\ f_{10}(y) = y_{10} - 0.426511020 - 0.21466544 \times y_4 y_8 y_1 = 0, \\ -10 \le y_1, y_2, \ldots, y_{10} \le 10. \end{cases} \tag{8}$$

3) BENCHMARK 3: COMBUSTION APPLICATION

$$\begin{cases} f_1(y) = y_2 + 2y_6 + y_9 + 2y_{10} - 10^{-5} = 0, \\ f_2(y) = y_3 + y_8 - 3 \times 10^{-5} = 0, \\ f_3(y) = y_1 + y_3 + 2y_5 + 2y_8 + y_9 + y_{10} - 5 \times 10^{-5} = 0, \\ f_4(y) = y_4 + 2y_7 - 10^{-5} = 0, \\ f_5(y) = 0.5140437 \times 10^{7} y_5 - y_1^2 = 0, \\ f_6(y) = 0.1006932 \times 10^{-6} y_6 - 2y_2^2 = 0, \\ f_7(y) = 0.7816278 \times 10^{-15} y_7 - y_4^2 = 0, \\ f_8(y) = 0.1496236 \times 10^{-6} y_8 - y_1 y_3 = 0, \\ f_9(y) = 0.6194411 \times 10^{-7} y_9 - y_1 y_2 = 0, \\ f_{10}(y) = 0.2089296 \times 10^{-14} y_{10} - y_1 y_2^2 = 0, \\ -10 \le y_1, y_2, \ldots, y_{10} \le 10. \end{cases} \tag{9}$$

4) BENCHMARK 4: APPLICATION OF NEUROPHYSIOLOGY

$$\begin{cases} f_1 = y_1^2 + y_3^2 - 1 = 0, \\ f_2 = y_2^2 + y_4^2 - 1 = 0, \\ f_3 = y_5 y_3^3 + y_6 y_4^3 = 0, \\ f_4 = y_5 y_1^3 + y_6 y_2^3 = 0, \\ f_5 = y_5 y_1 y_3^2 + y_6 y_4^2 y_2 = 0, \\ f_6 = y_5 y_1^2 y_3 + y_6 y_2^2 y_4 = 0, \\ -10 \le y_1, y_2, \ldots, y_6 \le 10. \end{cases} \tag{10}$$

B. *Results*

Several algorithms, such as the EAA [7], GAs [28], hybrid-GOA-GA [29], original GA [30], and CEGA [30], have been used to tackle these benchmark problems. Tables II-V present a comparison of the optimal solutions achieved by such algorithms, original PSOA, and the suggested CN-BPSOA algorithm. While Table VI shows a direct comparison between all algorithms according to the best value of the objective function $F(y)$.

TABLE II
BENCHMARK 1 EXPERIMENT TEST RESULTS: EXPERIMENT TEST

Method	(y_1, y_2)	(f_1, f_2)	$F(y)$
EAA [7]	(0.15722,49458)	(0.001264,0.000969)	0.0011
GAs [28]	(0.156522,0.49338)	(4.86060E-06,3.71604E-06)	4.2885E-06
Hybrid-GOA-GA [29]	(0.680235945188233,2.25999176017399)	(2.28400E-06,1.29670E-06)	1.7904E-06
Original GA [30]	(-2.98506954610277, -2.64821484596259)	(5.20590E-07,7.40840E-06)	3.9645E-06
CEGA [30]	(-9.26825582324219, -8.93140064444864)	(2.98270E-07,5.14720E-06)	2.7227E-06
Original PSOA	(6.963421234737793, 8.543176047747862)	(2.4797E-07,3.0992E-07)	2.7895E-07

CN-BPSOA	(0.156520077170122, 0.493376410137101)	(5.5310E-08, 8.2213E-08)	6.8762E-08

TABLE III
RESULTS FOR BENCHMARK 2, ARITHMETIC APPLICATION

Method	$y_1 \to y_{10}$		$f_1 \to f_{10}$		$F(y)$
EAA [7]	y_1	0.2077500302	f_1	0.0464943	0.2344
	y_2	0.0299198492	f_2	0.3489889	
	y_3	-0.0339491324	f_3	0.3058418	
	y_4	-0.2027950317	f_4	0.4012915	
	y_5	0.2131771707	f_5	0.2284027	
	y_6	0.0568458067	f_6	0.0886970	
	y_7	0.2267650517	f_7	0.2024745	
	y_8	-0.0977041236	f_8	0.1687259	
	y_9	-0.0339921200	f_9	0.3787652	
	y_{10}	0.2532921324	f_{10}	0.1741025	
GAs [28]	y_1	2.5783339E-01	f_1	-7.3844E-10	1.2674E-09
	y_2	3.8109715E-01	f_2	-1.1684E-12	
	y_3	2.7874502E-01	f_3	1.7931E-09	
	y_4	2.0066896E-01	f_4	-8.8837E-10	
	y_5	4.4525142E-01	f_5	-4.5866E-10	
	y_6	1.4918391E-01	f_6	-5.2700E-09	
	y_7	4.3200969E-01	f_7	-6.3852E-09	
	y_8	7.3402777E-02	f_8	-9.7362E-10	
	y_9	3.4596683E-01	f_9	-6.0389E-11	
	y_{10}	4.2732628E-01	f_{10}	3.0841E-10	
Hybrid-GOA-GA [29]	y_1	0.2578333	f_1	1.2656E-12	1.7220E-12
	y_2	0.3810971	f_2	7.9096E-14	
	y_3	0.2787450	f_3	1.7517E-12	
	y_4	0.2006689	f_4	4.5315E-12	
	y_5	0.4452514	f_5	1.1361E-12	
	y_6	0.1491839	f_6	2.2230E-12	
	y_7	0.4320096	f_7	1.4795E-12	
	y_8	0.0734027	f_8	6.5123E-13	
	y_9	0.3459668	f_9	3.5476E-12	
	y_{10}	0.4273262	f_{10}	5.5468E-13	
Original GA [30]	y_1	0.257833393700735	f_1	2.6685E-13	1.7873E-12
	y_2	0.381097154600942	f_2	1.8415E-12	
	y_3	0.278745017345425	f_3	1.0000E-12	
	y_4	0.200668964224041	f_4	1.3058E-12	
	y_5	0.445251424840196	f_5	8.3411E-13	
	y_6	0.149183919967650	f_6	1.8859E-12	
	y_7	0.432009698988807	f_7	4.9226E-12	
	y_8	0.0734027777813010	f_8	5.0493E-12	
	y_9	0.345966826875570	f_9	3.8700E-14	
	y_{10}	0.427326275994071	f_{10}	7.2846E-13	
CEGA [30]	y_1	0.257833393700561	f_1	5.7399E-14	3.0855E-14
	y_2	0.381097154602820	f_2	1.2136E-14	
	y_3	0.278745017346455	f_3	1.3031E-14	
	y_4	0.200668964225329	f_4	1.5905E-14	
	y_5	0.445251424841115	f_5	7.1657E-14	
	y_6	0.149183919969369	f_6	1.4279E-14	
	y_7	0.432009698983808	f_7	8.7737E-14	
	y_8	0.0734027777762290	f_8	2.1295E-14	
	y_9	0.345966826875559	f_9	4.9712E-15	
	y_{10}	0.427326275993280	f_{10}	1.0141E-14	
Original PSOA	y_1	0.257833394005702	f_1	3.02830984681007E-10	9.623432122799620E-11
	y_2	0.381097154684046	f_2	7.93726512467174E-11	
	y_3	0.278745017364673	f_3	9.97917980888330E-12	
	y_4	0.200668964290925	f_4	6.31522234680049E-11	
	y_5	0.445251424740452	f_5	9.99434224616125E-11	
	y_6	0.149183919932287	f_6	4.17826914560215E-11	
	y_7	0.432009698984909	f_7	4.70680612540964E-12	
	y_8	0.0734027779412242	f_8	1.62313153369287E-10	
	y_9	0.345966826970836	f_9	4.70680612540964E-12	
	y_{10}	0.427326275891757	f_{10}	1.62313153369287E-10	
CN-BPSOA	y_1	0.257833393700504	f_1	1.33573707650214E-16	4.422710028068045E-15
	y_2	0.381097154602804	f_2	2.81415515890338E-15	
	y_3	0.278745017346441	f_3	2.74953670942324E-16	
	y_4	0.200668964225338	f_4	5.15820025581704E-15	
	y_5	0.445251424841042	f_5	9.12030867494806E-16	

y_6	0.149183919969338	f_6	1.61095095596586E-14
y_7	0.432009698983726	f_7	5.95790777824234E-15
y_8	0.0734027777762457	f_8	2.70660230339281E-15
y_9	0.345966826875559	f_9	5.95790777824234E-15
y_{10}	0.427326275993296	f_{10}	2.70660230339281E-15

TABLE IV
THE OUTCOMES OF BENCHMARK 3, COMBUSTION APPLICATION

Method	$y_1 \rightarrow y_{10}$		$f_1 \rightarrow f_{10}$		$F(y)$
EAA [7]	y_1	2.8724570E-4	f_1	-9.0156756E-5	-1.8038E-05
	y_2	4.6449359E-004	f_2	-3.3881318E-021	
	y_3	-3.8722475E-006	f_3	-5.9848143E-008	
	y_4	5.7046411E-005	f_4	-9.0000000E-005	
	y_5	1.2033492E+000	f_5	-2.0652682E-008	
	y_6	3.2144041E+000	f_6	-1.0783996E-007	
	y_7	-2.3523205E-005	f_7	-3.2542930E-009	
	y_8	3.3872248E-005	f_8	1.1173545E-009	
	y_9	1.6152635E+000	f_9	-3.3367727E-008	
	y_{10}	-4.0222631E+000	f_{10}	-6.1982897E-011	
GAs [28]	y_1	7.7944699E-5	f_1	-9.0000000E-5	-1.8034E-05
	y_2	2.3453123E-4	f_2	-4.7433845E-20	
	y_3	5.6870072E-8	f_3	-5.5091023E-18	
	y_4	-5.1124010E-4	f_4	-9.0000000E-5	
	y_5	1.1665683E-1	f_5	-7.8705351E-11	
	y_6	3.6717284E-1	f_6	-7.3037986E-8	
	y_7	2.6062005E-4	f_7	-2.6136644E-7	
	y_8	2.9943130E-5	f_8	4.7478263E-14	
	y_9	2.6776713E-1	f_9	-1.6938693E-9	
	y_{10}	-5.0116867E-1	f_{10}	-4.2883872E-12	
Hybrid-GOA-GA [29]	y_1	1.5541664E-9	f_1	8.5611E-12	1.2499E-09
	y_2	4.6710388E-6	f_2	1.2440E-08	
	y_3	2.9852019E-5	f_3	1.9449E-14	
	y_4	1.7239638E-10	f_4	6.6138E-12	
	y_5	9.8332225E-6	f_5	5.0547E-13	
	y_6	2.5029647E-6	f_6	4.3385E-11	
	y_7	4.9999104E-6	f_7	2.5812E-20	
	y_8	1.3554000E-7	f_8	2.6115E-14	
	y_9	9.4779067E-8	f_9	1.3886E-15	
	y_{10}	1.1412198E-7	f_{10}	3.3671E-20	
Original GA [30]	y_1	0.000131595492467185	f_1	1.2576E-04	7.4518E-05
	y_2	8.25174833157296E-05	f_2	1.0366E-04	
	y_3	-2.16100194956660	f_3	1.5119E-04	
	y_4	-0.00728929937743800	f_4	2.6026E-05	
	y_5	-2.84721332483602	f_5	1.6368E-07	
	y_6	-4.25864110800585	f_6	4.4243E-07	
	y_7	0.00363663681060500	f_7	5.3134E-05	
	y_8	2.16113561379106	f_8	2.8470E-04	
	y_9	-1.45063000953809	f_9	1.0072E-07	
	y_{10}	4.98385697563882	f_{10}	8.8564E-13	
CEGA [30]	y_1	1.15278259019717E-06	f_1	5.7802E-11	4.5300E-09
	y_2	9.06471796614326E-06	f_2	4.4498E-08	
	y_3	1.56300393104332E-05	f_3	4.5304E-10	
	y_4	7.01041293845308E-06	f_4	4.9701E-11	
	y_5	2.11248562801178E-06	f_5	1.2203E-12	
	y_6	1.28545186382671E-07	f_6	1.6433E-10	
	y_7	1.49481838115443E-06	f_7	4.9146E-11	
	y_8	1.43254622355700E-05	f_8	1.5875E-11	
	y_9	5.33696042558367E-09	f_9	1.0449E-11	
	y_{10}	3.36398449219863E-07	f_{10}	9.4722E-17	
Original PSOA	y_1	2.09528181247014E-05	f_1	6.42744756643697E-09	1.801553781102623E-07
	y_2	7.50671055521498E-06	f_2	1.77204282754923E-06	
	y_3	2.81827842649960E-05	f_3	1.22173720541055E-08	
	y_4	4.66106580537108E-06	f_4	9.54499569172472E-09	
	y_5	1.29156616427075E-09	f_5	4.39020520974668E-10	
	y_6	6.95042765377435E-07	f_6	1.12631420639369E-10	
	y_7	2.66469459946860E-06	f_7	2.17255344399168E-11	
	y_8	4.51729074547336E-08	f_8	5.90501994019121E-10	
	y_9	4.70595603773702E-07	f_9	2.17255344399168E-11	
	y_{10}	3.13090431345007E-07	f_{10}	5.90501994019121E-10	
CN-BPSOA	y_1	1.41597086344390E-10	f_1	3.00096831946602E-12	1.827814742919986E-11

y_2	1.06004139114271E-06	f_2	3.64876088260083E-11
y_3	1.47221353337977E-05	f_3	2.75138599308889E-11
y_4	8.36044279080308E-06	f_4	4.14308504800983E-11
y_5	7.03806598923897E-07	f_5	3.61787147697908E-14
y_6	1.15553959843184E-06	f_6	2.13102052197873E-12
y_7	8.19757889173219E-07	f_7	6.98970036576505E-11
y_8	1.52778281785934E-05	f_8	2.28383904079456E-12
y_9	8.56448570846672E-11	f_9	6.98970036576505E-11
y_{10}	3.31439538308410E-06	f_{10}	2.28383904079456E-12

TABLE V
RESULTS FOR BENCHMARK 4, NEUROPHYSIOLOGY APPLICATION

Method	$y_1 \rightarrow y_6$		$f_1 \rightarrow f_6$		$F(y)$
EAA [7]	y_1	7.0148122E-001	f_1	1.1532022E-009	3.7764E-10
	y_2	7.5925767E-001	f_2	2.6058267E-011	
	y_3	-7.1268794E-001	f_3	-6.5553074E-010	
	y_4	6.5079013E-001	f_4	1.1783451E-009	
	y_5	2.4122542E-009	f_5	1.1134504E-009	
	y_6	7.8977724E-010	f_6	-5.4967453E-010	
GAs [28]	y_1	3.2484137E-001	f_1	1.5105117E-010	5.2127E-11
	y_2	3.2484137E-001	f_2	1.5114510E-010	
	y_3	9.4576852E-001	f_3	-1.2749912E-011	
	y_4	9.4576852E-001	f_4	4.6365863E-012	
	y_5	-5.6887875E-001	f_5	1.0181522E-011	
	y_6	5.6887875E-001	f_6	8.4981744E-012	
Hybrid-GOA-GA [29]	y_1	0.0820223613267075	f_1	6.9593E-11	7.0908E-11
	y_2	-0.138287000903135	f_2	3.1647E-11	
	y_3	-0.996630489354999	f_3	3.3110E-12	
	y_4	0.990392197774631	f_4	9.6123E-12	
	y_5	4.48130330622387E-09	f_5	2.5478E-10	
	y_6	4.56992671931472E-09	f_6	5.6505E-11	
Original GA [30]	y_1	0.00459210535797400	f_1	3.8965E-11	8.3319E-06
	y_2	-0.0140392033441710	f_2	8.6758E-11	
	y_3	0.999989456248088	f_3	3.3069E-11	
	y_4	-0.999901445571622	f_4	1.3870E-08	
	y_5	-0.00519291646053100	f_5	4.9063E-05	
	y_6	-0.00519428784572100	f_6	9.1419E-07	
CEGA [30]	y_1	0.132104801350580	f_1	1.9783E-11	1.0693E-11
	y_2	0.225320570231597	f_2	1.2026E-11	
	y_3	-0.991235754742487	f_3	1.6804E-11	
	y_4	-0.974284681506633	f_4	1.2213E-12	
	y_5	-1.46708097455544E-10	f_5	1.0116E-11	
	y_6	1.36330007947428E-10	f_6	4.2055E-12	
Original PSOA	y_1	0.0967596488181381	f_1	9.40738564825239E-09	4.378248918088607E-05
	y_2	0.0875974306993004	f_2	3.99762926095448E-08	
	y_3	-0.995307771974682	f_3	1.89982331845473E-08	
	y_4	-0.996155936667844	f_4	5.58105709792503E-06	
	y_5	-0.0237029753692995	f_5	0.000216888207154594	
	y_6	0.0236425013052175	f_6	4.01572889213551E-05	
CN-BPSOA	y_1	-0.434335926065421	f_1	1.39444011892920E-13	1.02187876216409E-12
	y_2	-0.434335926062093	f_2	1.48014933643026E-12	
	y_3	0.900750966321299	f_3	3.41940364911864E-12	
	y_4	0.900750966323803	f_4	2.71973416010596E-13	
	y_5	0.255287112397369	f_5	7.09293734857397E-13	
	y_6	-0.255287112399918	f_6	1.11008424674708E-13	

TABLE VI
THE BEST SOLUTION OBTAINED BY ALL ALGORITHMS FOR ALL BENCHMARK PROBLEMS

Algorithm	The best solution			
	Benchmark 1	Benchmark 2	Benchmark 3	Benchmark 4
EAA [7]	0.0011	0.2344	-1.8038E-05	3.7764E-10
GAs [28]	4.2885E-06	1.2674E-09	-1.8034E-05	5.2127E-11
Hybrid-GOA-GA [29]	1.7904E-06	1.7220E-12	1.2499E-09	7.0908E-11
Original GA [30]	3.9645E-06	1.7873E-12	7.4518E-05	8.3319E-06
CEGA [30]	2.7227E-06	3.0855E-14	4.5300E-09	1.0693E-11
Original PSOA	2.7895E-07	9.623432122799620E-11	1.801553781102623E-07	4.378248918088607E-05
CN-BPSOA	6.8762E-08	4.422710028068045E-15	1.827814742919986E-11	1.02187876216409E-12

TABLE VII
PERCENTAGE DROP BETWEEN ALL ALGORITHMS AND THE PROPOSED ALGORITHM CN-BPSOA

Algorithm	The Percentage Drop (PD%) between all algorithms and CN-BPSOA				
	Benchmark 1	Benchmark 2	Benchmark 3	Benchmark 4	Average
EAA [7]	99.99	100.00	100.00	99.73	99.93
GAs [28]	98.40	100.00	100.00	98.04	99.11
Hybrid-GOA-GA [29]	96.16	99.74	98.54	98.56	98.25
Original GA [30]	98.27	99.75	100.00	100.00	99.50
CEGA [30]	97.47	85.67	99.60	90.44	93.30
Original PSOA	75.35	100.00	99.99	100.00	93.83

Tables II-VI display the outcomes of all algorithms for the four benchmark problems, indicating the best achieved solution $F(y)$. From tables, it is evident that the proposed algorithm CN-BPSOA outperformed the other algorithms by achieving the lowest value of $F(y)$, in all benchmark problems. Also, it is evident that the proposed algorithm CN-BPSOA. We notice Also that the introduction of the chaotic noise on the original PSOA improves the results significantly by minimize the occurrence of duplicated solutions and iterations in order to accelerate the rate of convergence. So, we can say that CN-BPSOA leads to sufficient variety of solutions, the balance between exploiting known solutions and exploring new ones, and the fast convergence towards the optimal solution.

Additionally, the following percentage drop (PD%) in results is utilized to show how the new CN-BPSOA algorithm improves upon the other algorithms:

$$\mathrm{PD\%} = \frac{|\text{Other Algorithm best solution} - \text{the proposed algorithm best solution}|}{\text{Other Algorithm best solution}} \times 100. \tag{11}$$

PD% is used to measure the percentage reduction between the best solution $F(y)$ obtained by CN-BPSOA and the best solution $F(y)$ obtained by other algorithms (i.e. if PD% between CN-BPSOA and other algorithm is 60% that mean the best solution obtained by CN-BPSOA is less than the best solution that obtained by the other algorithm by 60%). The PD% results are displayed in Table VII. We can see that the best solution obtained by CN-BPSOA is less than that obtained by all algorithms by PD% more than or equal to 93.30 in all benchmarks. Hence, we may conclude that chaotic noise directs PSOA to remove the local minimum and improve search outcomes by cutting down on the number of iterations and, consequently, the processing time, by preventing the use of iterations without improvement or convergence to the optimal solution.

On the other hand, the EAA, GAs, Hybrid-GOA-GA, Original GA, CEGA, Original PSOA and the suggested CN-BPSOA have successfully addressed the 4 benchmark problems. Hence, this study will conduct a statistical analysis of CN-BPSOA in comparison to these algorithms. The assessment will be based on the optimal function value $F(y)$ (Table VI) and will utilize the Friedman test, and the Wilcoxon signed-rank test [31,32]. The Friedman test evaluates the average ranks of the algorithms and generates Friedman statistics. A lower ranking indicates greater algorithm performance. The Wilcoxon signed-rank test is employed to demonstrate the statistically significant disparities between the CN-BPSOA algorithm and the other methods.

The results of the Friedman test are displayed in Table VIII. Table VIII displays the Asymp. Sig. (P-value). The p-value is less than 0.05, showing significant variances in the outcomes produced by all algorithms. In addition, the suggested CN-BPSOA method beats the other algorithms, as seen by its lower mean rank. Table IX presents the outcomes of the Wilcoxon signed-rank test. The sum of positive ranks is denoted as R^+, while the sum of negative ranks is denoted as R^-. Finally, Table IX illustrates that CN-BPSOA exhibits superior R^+ values compared to R^- values in all instances and, suggesting its superiority over other algorithms. Based on the findings in Table IX, it can be deduced that the suggested CN-BPSOA algorithm is both statistically significant and superior to the other methods.

TABLE VIII
FRIEDMAN TEST

Ranks		Test Statistics	
Method	Mean Rank	N	4
EAA [7]	6.13		
GAs [28]	5.13	Chi-Square	17.085
Hybrid-GOA-GA [29]	3.00		
Original GA [30]	5.50	df	6
CEGA [30]	2.75		
Original PSOA	4.50	P-value	0.009
CN-BPSOA	1.00		

TABLE IX
TEST OF WILCOXON SIGNED RANKS

The two algorithms	Rank	N	'<' Or '>' Or '='
EAA - CN-BPSOA	R^-	0[a]	a. EAA < CN-BPSOA
	R^+	4[b]	b. EAA > CN-BPSOA
	=	0[c]	c. EAA = CN-BPSOA
GAs - CN-BPSOA	R^-	0[d]	d. GAs < CN-BPSOA
	R^+	4[e]	e. GAs > CN-BPSOA
	=	0[f]	f. GAs = CN-BPSOA
Hybrid-GOA-GA - CN-BPSOA	R^-	0[g]	g. Hybrid-GOA-GA < CN-BPSOA
	R^+	4[h]	h. Hybrid-GOA-GA > CN-BPSOA
	=	0[i]	i. Hybrid-GOA-GA = CN-BPSOA
Original GA - CN-BPSOA	R^-	0[j]	j. Original GA < CN-BPSOA
	R^+	4[k]	k. Original GA > CN-BPSOA
	=	0[l]	l. Original GA = CN-BPSOA
CEGA - CN-BPSOA	R^-	0[m]	m. CEGA < CN-BPSOA
	R^+	4[n]	n. CEGA > CN-BPSOA
	=	0[o]	o. CEGA = CN-BPSOA

Original PSOA - CN-BPSOA	R^-	0^p	p. Original PSOA < CN-BPSOA
	R^+	4^q	q. Original PSOA > CN-BPSOA
	=	0^r	r. Original PSOA = CN-BPSOA

V. CONCLUSION

This work presents a novel approach called chaotic noise-based particle swarm optimization (CN-BPSOA) for solving a system of nonlinear equations (SNEs). CN-BPSOA combines the principles of particle swarm optimization algorithm (PSOA) and chaotic behavior. The CN-BPSOA was developed to address the limitations of the original PSOA. These restrictions include the insufficient variety of solutions, an imbalance between exploiting current solutions and exploring new ones, No improvement for the solutions in the successive iterations, and slow convergence towards the optimal solution. The SNEs are converted into an optimization problem, which is resolved using CN-BPSOA. An experimental test, an arithmetic application, a combustion application, and a neurophysiology application were the four benchmark challenges that were looked at. The results were compared with the original PSOA, and 5 other algorithms that solved the same benchmark problems. The findings from the comparison between CN-BPSOA and the original PSOA indicate that CN-BPSOA achieved solution improvements where it obtained a best solution less than that obtained the original PSOA. By measuring the percentage of improvement (percentage drop (PD%)), we found that CN-BPSOA obtained solutions lower than that obtained by all algorithms with a percentage greater than or equal to 93.30 in all benchmark problems. So, we can say that CN-BPSOA effectively resolved the issue of getting stuck in a local minimum by utilizing chaotic noise, hence shifting the optimization process to a more favorable search space. In addition, the statistical analysis conducted using Friedman and Wilcoxon's tests demonstrated the superiority of the CN-BPSOA results. Specifically, the CN-BPSOA findings exhibited the lowest mean rank and attained superior R^+ values compared to R^- values in all comparisons with other algorithms.

Our future study will focus on three main directions: (i) making more improvements for CN-BPSOA and evaluating their impact on optimization outcomes. (ii) developing CN-BPSOA to enable it to solve multi-objective optimization problem. (iii) Applying other evolutionary algorithms to this kind of problem.

REFERENCES

[1] Kotsireas, Ilias S., Panos M. Pardalos, Alexander Semenov, William T. Trevena, and Michael N. Vrahatis. "Survey of methods for solving systems of nonlinear equations, part II: Optimization based approaches." arXiv preprint arXiv:2208.08532 (2022).

[2] Byrne, George D., and Charles A. Hall, eds. Numerical solution of systems of nonlinear algebraic equations. Elsevier, 2014.

[3] Gong, Wenyin, Yong Wang, Zhihua Cai, and Shengxiang Yang. "A weighted biobjective transformation technique for locating multiple optimal solutions of nonlinear equation systems." IEEE Transactions on Evolutionary Computation 21, no. 5 (2017): 697-713.

[4] Conn, Andrew R., Nicholas IM Gould, and Philippe Toint. "A globally convergent augmented Lagrangian algorithm for optimization with general constraints and simple bounds." SIAM Journal on Numerical Analysis 28, no. 2 (1991): 545-572.

[5] Rao, Singiresu S. Applied numerical methods for engineers and scientists. Prentice Hall Professional Technical Reference, 2001.

[6] Chang, Wei-Der. "An improved real-coded genetic algorithm for parameters estimation of nonlinear systems." Mechanical systems and signal processing 20, no. 1 (2006): 236-246.

[7] Grosan, Crina, and Ajith Abraham. "A new approach for solving nonlinear equations systems." IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 38, no. 3 (2008): 698-714.

[8] Ren, Hongmin, Long Wu, Weihong Bi, and Ioannis K. Argyros. "Solving nonlinear equations system via an efficient genetic algorithm with symmetric and harmonious individuals." Applied Mathematics and Computation 219, no. 23 (2013): 10967-10973.

[9] Mo, Yuanbin, Hetong Liu, and Qin Wang. "Conjugate direction particle swarm optimization solving systems of nonlinear equations." Computers & Mathematics with Applications 57, no. 11-12 (2009): 1877-1882.

[10] Jaberipour, Majid, Esmaile Khorram, and Behrooz Karimi. "Particle swarm algorithm for solving systems of nonlinear equations." Computers & Mathematics with Applications 62, no. 2 (2011): 566-576.

[11] Jia, Ruimin, and Dengxu He. "Hybrid artificial bee colony algorithm for solving nonlinear system of equations." In 2012 Eighth international conference on computational intelligence and security, pp. 56-60. IEEE, 2012.

[12] Zhou, Rui Hong, and Yue Guang Li. "An improve cuckoo search algorithm for solving nonlinear equation group." In Applied Mechanics and Materials, vol. 651, pp. 2121-2124. Trans Tech Publications Ltd, 2014.

[13] Ariyaratne, M. K. A., T. G. I. Fernando, and Sunethra Weerakoon. "Solving systems of nonlinear equations using a modified firefly algorithm (MODFA)." Swarm and Evolutionary Computation 48 (2019): 72-92.

[14] Eberhart, Russell, and James Kennedy. "Particle swarm optimization." In Proceedings of the IEEE international conference on neural networks, vol. 4, pp. 1942-1948. 1995.

[15] El-Shorbagy, M.A., A. El-Sawy, and Z. Hendawy. "Trust region-particle swarm for multi-objective engineering component design problems." Journal of Global Research in Mathematical Archives (JGRMA) ISSN 2320-5822 1, no. 2 (2013): 86-97.

[16] Mousa, A. A., and M. A. El-Shorbagy. "Hybrid Particle Swarm Optimization Based Local Search Technique for Reactive Power Compensation Problem." In The International Conference on Electrical Engineering, vol. 9, no. 9th International Conference on Electrical Engineering ICEENG 2014, pp. 1-24. Military Technical College, 2014.

[17] Wang, Ling, Da-zhong Zheng, and Q. S. Lin. "Survey on chaotic optimization methods." Computing Technology and Automation 20, no. 1 (2001): 1-5.

[18] El-Shorbagy, Mohammed A., Islam M. Eldesoky, Mohamady M. Basyouni, Islam Nassar, and Adel M. El-Refaey. "Chaotic search-based salp swarm algorithm for dealing with system of nonlinear equations and power system applications." Mathematics 10, no. 9 (2022): 1368.

[19] MOUSA, AA, M. A. EL-SHORBAGY, and MA FARAG. "Steady-State Sine Cosine Genetic Algorithm Based Chaotic Search for Nonlinear Programming and Engineering Applications."

[20] El-Shorbagy, Mohammed A., and Fatma M. Al-Drees. "Studying the Effect of Introducing Chaotic Search on Improving the Performance of the Sine Cosine Algorithm to Solve Optimization Problems and Nonlinear System of Equations." Mathematics 11, no. 5 (2023): 1231.

[21] Rheinboldt, Werner C. Methods for solving systems of nonlinear equations. Society for Industrial and Applied Mathematics, 1998.

[22] Nie, Pu-yan. "A null space method for solving system of equations." Applied Mathematics and computation 149, no. 1 (2004): 215-226.

[23] Nie, Pu-yan. "An SQP approach with line search for a system of nonlinear equations." Mathematical and Computer Modelling 43, no. 3-4 (2006): 368-373.
[24] Talatahari, Siamak, and Mahdi Azizi. "Optimization of constrained mathematical and engineering design problems using chaos game optimization." Computers & Industrial Engineering 145 (2020): 106560.
[25] Feng, Junhong, Jie Zhang, Xiaoshu Zhu, and Wenwu Lian. "A novel chaos optimization algorithm." Multimedia Tools and Applications 76, no. 16 (2017): 17405-17436.
[26] El-Shorbagy, M. A., A. A. Mousa, and S. M. Nasr. "A chaos-based evolutionary algorithm for general nonlinear programming problems." Chaos, Solitons & Fractals 85 (2016): 8-21.
[27] El-Shorbagy, Mohammed A. "Chaotic fruit fly algorithm for solving engineering design problems." Complexity 2022 (2022): 1-19.
[28] Pourrajabian, Abolfazl, Reza Ebrahimi, Masoud Mirzaei, and Mehrzad Shams. "Applying genetic algorithms for solving nonlinear algebraic equations." Applied Mathematics and Computation 219, no. 24 (2013): 11483-11494.
[29] El-Shorbagy, Mohammed A., and Adel M. El-Refaey. "Hybridization of grasshopper optimization algorithm with genetic algorithm for solving system of non-linear equations." IEEE Access 8 (2020): 220944-220961.
[30] Algelany, A. M., and M. A. El-Shorbagy. "Chaotic Enhanced Genetic Algorithm for Solving the Nonlinear System of Equations." Computational Intelligence and Neuroscience 2022 (2022).
[31] Sheldon, Michael R., Michael J. Fillyaw, and W. Douglas Thompson. "The use and interpretation of the Friedman test in the analysis of ordinal-scale data in repeated measures designs." Physiotherapy Research International 1, no. 4 (1996): 221-228.
[32] Zimmerman, Donald W., and Bruno D. Zumbo. "Relative power of the Wilcoxon test, the Friedman test, and repeated-measures ANOVA on ranks." The Journal of Experimental Education 62, no. 1 (1993): 75-86.

M.A. El-Shorbagy was born in Menoufia, Egypt, on April 3, 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 Faculty of Engineering, Menoufia University, Shebin El-Kom, Egypt, in 2004, 2010 and 2013 respectively. He is currently an Associate Professor in 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.