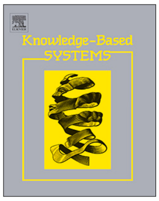
Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys



Homonuclear Molecules Optimization (HMO) meta-heuristic algorithm

Amin Mahdavi-Meymand [a], Mohammad Zounemat-Kermani [b,*]

[a] Institute of Hydro-Engineering, Polish Academy of Sciences, Poland
[b] Water Engineering Department, Shahid Bahonar University of Kerman, Kerman, Iran

ARTICLE INFO

Article history:
Received 7 April 2021
Received in revised form 7 August 2022
Accepted 13 October 2022
Available online 20 October 2022

Keywords:
Search algorithm
Non-linear optimization
Artificial intelligence
Swarm algorithm
Electron cloud

ABSTRACT

This study introduces a novel meta-heuristic algorithm known as Homonuclear Molecules Optimization (HMO) for optimizing complex and nonlinear problems. HMO is inspired by the arrangement of electrons around atoms given the Bohr atomic model and the structure of homonuclear molecules. This algorithm is based on creating the initial population of a set of atoms in the search space and the electrons associated with each atom (searching agents) given the quantum numbers. In each iteration, the best electron of each atom is selected as the new location of the nucleus, and a number of atoms move toward the atom with the best solution to form a homonuclear molecule. The results of applying the MHO algorithm were evaluated in comparison with three classical optimization algorithms of PSO, GA, and DE along with a novel algorithm called Equilibrium Optimizer (EO). The HMO was able to precisely solve unimodal functions and find global and local solutions for multimodal functions. The outcomes of Wilcoxon's rank-sum test demonstrated that there is a significant difference between the final results of HMO and those of the other algorithms ($a = 0.05\%$). Overall, it was concluded that the HMO outperforms the classical algorithms and can compete with new and efficient algorithms such as EO.

1. Introduction

In the past, traditional mathematical methods such as linear programming (LP), non-linear programming (NLP), and dynamic programming (DP) were widely used to solve optimization problems. Using these methods, the global optima of the simple and ideal optimization problems could be found. Nevertheless, all the aforementioned methods have certain drawbacks. For example, LP considerably diverges from the solution when the problem transforms from the ideal linear state to the non-linear one in practical problems. In DP, with an increase in the number of variables, the assessments of recursive functions are increased and, in NLP, if the used functions are not recognizable, the algorithm is not likely to find the optimal solution for the problem.

Nowadays, the spread of computers has developed the random and meta-heuristic methods to solve the optimization problems and the problems of the above mathematical methods. A random method that is proposed by Robins and Monro [1] was used in the following years in various forms for solving optimization problems. In meta-heuristic algorithms, searching for optimal results is random and many iterations are required to complete the problem-solving process. In such methods, no information from the gradient of the objective function is required and a relationship between the problem variables and objective function is sufficient for the solution process; this feature has attracted great attention to the random methods for solving complex problems. In general, the population based meta-heuristic algorithms can be divided into two main branches including evolutionary and swarm-intelligence algorithms [2]. The genetic algorithm (GA) and particle swarm optimization (PSO) are the well-known evolutionary and swarm-based algorithms, respectively [3,4]. The GA was proposed by Holland [5] and is inspired by Darwinian evolutionary theory, which has so far been used in many practical applications and its appropriate performance has been confirmed [6,7]. The PSO is a collective search algorithm that models the collective behavior of the shoals of fish or flocks of birds [8]. In the two past decades, the PSO has been successfully applied in almost all of science branches [3]. In recent years, many meta-heuristic algorithms have been inspired by the behavior of natural phenomena [9,10]. Some of these algorithms, such as firefly algorithm (FA), gravitational search algorithm (GSA), and butterfly optimization algorithm (BOA) are categorized as swarm-based algorithms. In these algorithms, each member of the population looks for a solution by performing specific operations in the search space and shares its information with other

* Corresponding author.
E-mail addresses: amin.mahdavi1990@gmail.com, a.mahdavi@ibwpan.gda.pl (A. Mahdavi-Meymand), zounemat@uk.ac.ir (M. Zounemat-Kermani).

https://doi.org/10.1016/j.knosys.2022.110032
0950-7051/© 2022 Elsevier B.V. All rights reserved.

members of the population. The local interactions between the members of the population creates a global result for the system. These conditions allow the system to solve the desired problem without using a central controller.

Two important concepts regarding population-based meta-heuristic algorithms are exploration and exploitation. Exploration is a procedure for expanding search space and exploitation is the ability of an algorithm to approach an optimal solution in the vicinity of a good solution. A meta-heuristic algorithm that has both of these properties explores the search space in the initial population iterations to find the proper solution. This property prevents the algorithm from being trapped in the local solutions. After the algorithm iterations, the exploration property gradually fades and the exploitation property around semi-optimal points begins to reach the optimal solution. Most of the proposed meta-heuristic algorithms have both of these properties, but their methods and operators are different from each other. Subsequently, the history of meta-heuristic algorithms that have been proposed by the researchers will be reviewed.

Simulated annealing algorithm (SA) is a simple and effective meta-heuristic algorithm in solving optimization problems in large search spaces, which is inspired by thermodynamic effects [11]. Differential evaluation (DE) algorithm is a simple random evolutionary-based algorithm similar to the genetic algorithm, which was proposed by Storn and Price [12]. Some of the DE modifications are including self-adaptive DE (SaDE) [13], JADE [14], success-history based adaptive DE (SHADE) [15], and chaotic-local-search-based JADE (CJADE) [16]. The harmony search algorithm (HS) is a search algorithm that is inspired by the music process [17]. The cultural evolutionary algorithm that was proposed by Reynolds [18] is a population-based algorithm, in which there is another component called "belief" in addition to the population. Ant colony optimization (ACO) is another algorithm inspired by nature where the behavior of ants for nutrition is inspired [9]. The firefly algorithm (FA) is inspired by the optical communication between the fireflies and an algorithm with proper performance is presented [19]. The artificial bee colony (ABC) proposes an important method for optimizing problems by simulating the intelligent behavior of bees [20]. Via formulating the law of gravity between objects, the gravitational search algorithm (GSA) provides a structure for making a movement in the population and approaching the optimal solution [21]. Some robust modifications of GSA such as hierarchical GSA (HGSA) [22], and hierarchy and distributed framework (DGSA) [23], have been also developed by researches. The gases Brownian motion optimization is inspired by the law of Brownian motion of atoms of gases and its formulation is inspired by physical phenomena [24]. Water cycle algorithm (WCA) and is inspired by a hydrology phenomenon [25]. The Fruit fly optimization algorithm (FOA) and its modified version namely joint search strategies FOA (JS-FOA) are inspired by the behavior of fruit flies in searching for foods [26,27]. Zhang [28] introduced specular reflection learning (SRL) for enhancing the meta-heuristic algorithm performance. The results confirmed that the SRL improves the backtracking search algorithm (BSA) [29]. Monarch butterfly optimization (MBO) algorithm [30] was inspired by the behavior of a type of North American butterfly.

An improved simulated annealing algorithm is a meta-heuristic method base on the SA algorithm. In this algorithm, two new operators including folding and reheating, that inspired by the ancient Japanese Swordsmithing method, have been used to handle exploitation and exploration [31]. Henry Gas Solubility Optimization (HGSO) is another meta-heuristic approach that mimics the Henry's law in gases [32]. The side-blotched lizard algorithm (SBLA) emulates the polymorphic population of the lizard. The population of SBLA moves dynamically by applying three operators that allows the coexistence of multiple morphs [33]. Black widow optimization (BWO) is a new developed evolutionary based meta-heuristic algorithm inspired by the black widow spider's life cycle [34]. Hayyolalam et al. [34] compared the performance of BWO with the PSO, GA, and ABC in solving several benchmark functions and reported the outperformance of the BWO in global solution fining. However, an application study conducted by Memar et al. revealed that the BWO is more suitable for low dimensions problems [35]. Another novel meta-heuristic algorithm namely Equilibrium optimizer (EO) was proposed based on the control volume mass physical law. Faramarzi et al. [36] compared the EO performance with some state-of-the-art optimization algorithms and its performance in several benchmark functions especially EC2017 has been confirmed. Moe recently, the reptile search algorithm (RSA) – as a nature-inspired MHA that mimics the crocodiles hunting behavior – has been introduced [37]. The RSA performance in benchmark functions has been reported acceptable.

2. Objectives and contribution

Considering the fact that it is not impossible to develop an algorithm to solve all classes of problems [36,38], this study intends to introduce a more accurate and effective meta-heuristic algorithm. The main objectives and contribution of HMO are as follows:

(1) In contrast to most of recent and new algorithms, which have been developed based on the collective behaviors of living organisms, HMO is based on the quantum behavior of atoms to generate an optimal molecule. In other words, the main issue in the development of this algorithm is not biological, but physio-chemical inspiration.

(2) The structure of HMO is straightforward and users can easily understand the theory and implement it in real problems.

(3) HMO benefits from unique strategies for updating the candidate solutions. This strategy makes HMO a robust technique for solving high dimensional optimization problems.

(4) HMO has a quick convergence rate and can avoid being trapped in local solutions.

(5) Most of the state-of-the-art meta-heuristic algorithms have presented proper operators for exploration and exploitation properties and, consequently, the determination of global optima. An optimization problem may have several global solutions and a number of local solutions. Hence, in some cases finding all these solutions is important for the user(s). By providing a unique structure, the HMO algorithm is capable of specifying other solutions equal to the global solution or local solutions while determining the global optima for high dimensional complex problems.

In the following, the structure of this algorithm will be fully explained. In addition, to evaluate the performance of the HMO algorithm, the findings of this algorithm will be compared to those of three cases of the most well-known classical meta-heuristic algorithms such as the GA, PSO, and DE. Moreover, the performance of the HMO will be compared with the equilibrium optimizer (EO) algorithm [36] which has been developed recently.

3. Homonuclear molecules algorithm (HMO)

The basis of this algorithm is inspired by the structure and arrangement of electrons around the nucleus of atoms. In order to establish the structure of this algorithm, it is first essential to review the nuclear arrangement of the elements found in nature, which is summarized below.

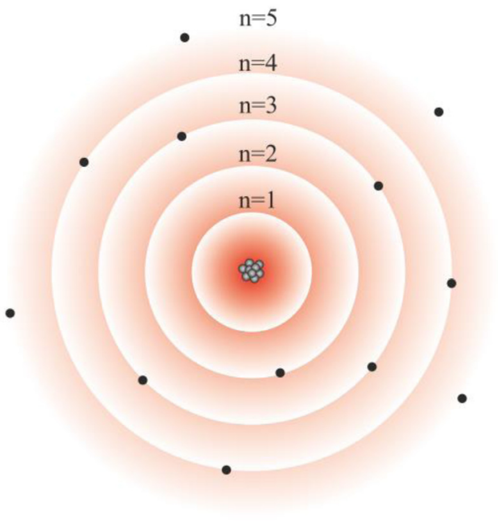


Fig. 1. The electron cloud around the nucleus of atom based on Bohr's theory.

3.1. Electron arrangement of atoms

Atoms are composed of protons, neutrons, and electrons [39]. The existence of atoms was first discussed around 400 BC by two great Greek philosophers, Democritus and Leucippus, without any conclusive evidence. There was no convincing laboratory evidence to confirm the existence of atoms for 20 centuries. The information on chemical reactions was first quantitatively measured by Lavoisier et al. but the existence of atoms was not mentioned. However, John Dalton presented the first atomic theory using this information [40]. Thomson discovered the existence of electrons in 1897through experimentation, and he later proposed his atomic model as a positively charged homogeneous space, in which electrons were uniformly dispersed [41]. By radiating the alpha particles to a thin gold sheet, Rutherford (1906) observed that some alpha particles were deviating from their path (2%), despite the fact that in accordance with Thomson's atomic model, they should have passed without deviation. In 1911, Rutherford introduced the planetary model. In 1913, Niels Bohr concluded that the rotation of electrons around the nucleus in accordance with Rutherford's theory would lead to loss of energy and the spiral collapse of electrons around the nucleus [40].

Bohr presented the quantum model for the hydrogen atom. According to this model, the electrons around the nucleus move in certain circuits and each electron could have a certain amount of energy. Each of these energy values is referred to as the energy level or energy layer and is specified with values of n (1, 2, 3, 4, …) [42]. The Bohr atomic model shows that the atomic radius increases in proportion to n^2.

$$r = n^2 \left(\frac{4\pi \varepsilon_0 h^2}{m_0 e^2} \right) = n^2 \alpha_1 \tag{1}$$

where e stands for the elementary charge, m_0 is the electron mass, ε_0 denotes the permittivity of free space, h is the Planck's constant, and π is a constant coefficient (the ratio of the circumference of a circle to its diameter).

Fig. 1 shows a schematic of the electron cloud around the nucleus based on Bohr's atomic model.

Subsequently, Schrödinger presented his atomic model, which was based on the dual behavior of electrons with an emphasis on their wave behavior. In this model, instead of limiting the electron to the circuit, the presence of the electron in a 3D space called orbital was suggested. In wave mechanics, in atoms with multiple electrons, electrons are distributed in layers. Each of these layers consists of one or two sublayers and each sublayer has one or several orbitals. Each of the atom's electrons is specified by four quantum numbers, which correspond to its main layer, sublayer, orbital, and spin. The principal quantum number indicates the layer to which the electron belongs, where there is a high possibility of finding electrons. This number is shown by n and is an integer [42].

$$n{=}1,\ 2,\ 3,\ 4,\ \ldots \tag{2}$$

The number of sublayers is equal to the principal quantum number. Each sublayer has an angular momentum quantum number, l, and an integer from 0 to n-1, which is shown by the following letters:

$$\begin{array}{lcccccccc} \text{Letter Used} & s & p & d & f & g & h & \ldots \\ \quad l & 0 & 1 & 2 & 3 & 4 & 5 & \ldots \end{array} \tag{3}$$

Each sublayer consists of a certain number of orbitals, which is calculated by the following equation:

$$\text{Number of orbitals} = 2n + 1 \tag{4}$$

Each of the sublayer's orbitals is identified by the magnetic quantum number (ml) and is composed of integers between 1, −1, and 0. Each orbital is filled with two electrons with different electron spin quantum numbers ($m_s = \frac{1}{2}, -\frac{1}{2}$). For example, the arrangement of the 36 electrons of this element around the nucleus is $1s^2$, $2s^2$, $2p^6$, $3s^2$, $3p^6$, $3d^{10}$, $4s^2$, $4p^6$ [40].

3.2. Modeling the HMO algorithm

In the previous section, it was noted that the electrons around the nucleus of an atom are placed in layers with different energy levels. In the HMO algorithm, the electrons are in charge of searching for a solution for the desired problem. Given that the velocity of the electrons is very high and close to the speed of light, in the HMO algorithm, the electrons are randomly distributed as searching agents around the nucleus considering the maximum possible distance.

Each electron of an atom that has the best solution is selected as the new location for the nucleus of the reciprocal atom and the process is reiterated until the nucleus approaches the solution (exploration phase). This algorithm is designed in a way that can find several solutions for problems with many local extrema. In order to enable the algorithm to find a solution for a problem with high precision, similar to most of the other swarm-based algorithms, atoms must move toward the atom with the best solution (paramount atom), which finally leads to the formation of the homonuclear molecule. Therefore, in this algorithm, the structure of a homonuclear molecule is determined, and accordingly, the number of atoms required to form the molecule is specified. It is assumed that in the exploitation procedure, a number of adjacent atoms travel toward the paramount atom for the formation of a molecule and the global optima is the location of the molecule and the location of the nucleus of the best atom. It should be noted that the type of population movement toward the best representative in most algorithms is almost random. Inspired by some quantum principles for electrons, this algorithm is based on the assumptions below:

1. The electrons belonging to a layer with more energy (layers away from the nucleus) can be transferred to a layer with less energy, but the electrons of a layer with less energy (layers closer to the nucleus) cannot be transferred to the layer with more energy.
2. The geometric location of the electrons around the nucleus is different from the real model. For example, this geometric location is square in 2D space and cubic in 3D space.
3. If necessary (given the iterations of the algorithm and convergence rate), the distance of layers from the nucleus can be reduced in each iteration.
4. The effect of the collision of atoms with each other is ignored.
5. A number of atoms move toward the atom that has the best solution (paramount atom). This movement creates the bond and molecule.

To further describe the performance of the proposed HMO algorithm, the optimization process is divided into the following five phases (Fig. 3).

- **Phase I: Initial phase**

In this phase, the initial population consisting of several atoms (N) with atomic number z is randomly generated to form a homonuclear molecule composed of m atoms in the search space. In other words, selecting the number of initial parameters with respect to the dimensions and the type of problem is the responsibility of the user. When selecting the proposed values, the following factors should be considered:

1. Use a heavier atom for problems with high dimensions and multiple local extrema.
2. If there is no need to find the local solutions, fewer atoms can be considered in the search space.
3. If the goal is to find the local extrema and the precise solution is not very important, molecules with few atoms can be used.
4. It is suggested that the atomic number (z) of the selected atom be larger than the problem dimension (D). On this basis, one can write:

$$D < z,\ \ 2 \leq m,\ \ m \leq N \leq 2m \tag{5}$$

After the placement of atoms in the search space, the electrons are randomly distributed around atoms. The atom that has the best solution is chosen as the paramount atom. The arrangement of the electron cloud around the atom's nucleus is based on the distance of the electrons from the nucleus. Hence, the principal quantum number (n) is used. At the beginning of the algorithm for atom i, these distances are calculated based on the following equation:

$$R_i^j = \alpha_1 j^2 \tag{6}$$

where R_i^j refers to the maximum possible distance of the electrons of layer j of atom i from the nucleus, and α_1 refers to the Bohr radius related to Eq. (1), for which a value should be considered based on the search space of the problem. In this research, the following equation was proposed for calculating α.

$$a = \frac{\lambda}{2(n_{\max})^2},\quad \lambda = \frac{|R_{\min}| + |R_{\max}|}{\beta} \tag{7}$$

In this equation, $R_{\max}$ refers to the maximum bound of the search space along an axis, $R_{\min}$ refers to the minimum bound of the search domain along the axis, $n_{\max}$ refers to the maximum quantum number related to the desired element, and β is the distance control coefficient. If an electron falls out of the border of the search domain, it will be positioned at the edge of the border. Fig. 2 illustrates the extraction of Eq. (7) as well as the motion of atoms around the borders under the assumption that $\beta = 2$.

The arrangement of electrons can be done based on the elements of the periodic table or a hypothetical element (which is absent in nature). In each iteration, the position of the electrons is calculated using the following equation:

$$Xelectron_i^t = Xnucleus_i^t + r_i^j \tag{8}$$

where $Xelectron_i^t$ refers to the position of the electron of atom i in iteration t, $Xnucleus_i^t$ refers to the position of the nucleus of atom i in iteration t, and r_j^i is a random vector in the interval $[-R_i^j, R_i^j]$. In the schematic representation of the performance of the algorithm (Fig. 3), five oxygen atoms ($z = 8$) are distributed in the search space to form an ozone molecule (O_3) as the location of global extrema. In this figure, it is also assumed that the search space has two similar optimal solutions.

- **Phase II: Exploration phase**

In phase II, according to the position of atoms and their distributed electrons, the fitness value of all the electrons is calculated given that the cost function and the best electron of each atom are also determined. Using the cost function, the electron that has the best solution is specified for each atom $Xelectron_i^{best}$ and the new position of the nucleus of each atom is calculated using the following equation:

$$Xnucleus_i^{t+1} = \left[Xelectron_i^{best}\right]^t \tag{9}$$

where t represents the iteration step. Finally, the location of the paramount atom is updated. In a new iteration, if another atom offers a better solution than the selected paramount atom, their location will be switched.

- **Phase III: Exploitation phase**

In order to enable the algorithm to find the precise solution for the problems and determine the local extrema, a number of atoms should move toward the paramount atom to form a homonuclear molecule. Therefore, by considering a molecule and the number of its atoms, several atoms can be moved toward the paramount atom. In this algorithm, the atoms that are geometrically closer to the paramount atom will move toward it. Hence, the Euclidean distance is used to determine the adjacent atoms.

$$d_i^t = \left\| Xnucleus\_paramount^t, Xnucleus_i^t \right\|_2,\ \ i = 1, \ldots, N-1 \tag{10}$$

where d_i^t is equal to the Euclidean distance between the nucleus of the paramount atom and the position of the nucleus of the *ith* atom ($Xnucleus_i^t$), and $Xnucleus\_paramount^t$ denotes the position of the nucleus of the paramount atom at iteration t. With this assumption, m-1 number of adjacent atoms will be able to increase their learning through the paramount atom and the exploitation property of the algorithm is established by approaching a better atom. The movement of $m - 1$ adjacent

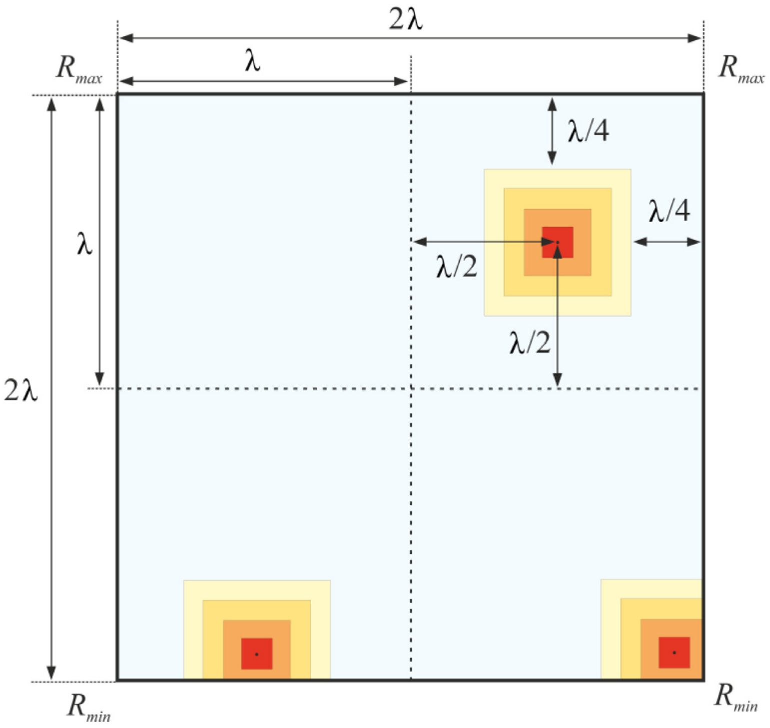


Fig. 2. Calculating Bohr radius in the HMO algorithm.

atoms to form a homonuclear molecule is as follows:

$$Xnucleus_i^{t+1} = Xnucleus_i^t + \gamma \times \left(Xnucleus\_paramount^t - Xnucleus_i^t\right),\ i = 1, \ldots, m-1 \quad (11)$$

where γ is the distance reduction factor and is a random vector between 0 and 1.

The smallest amount of γ increases the effect of the best electron of atoms in motion and decreases the effect of the homonuclear molecule. In this study, γ was between 0 and 0.1.

- **Phase IV: Iteration and shrinkage phase**

In this phase, the calculation process in phases II and III is repeated in order to achieve an optimal solution. For faster convergence as well as finding a more precise solution for the problem, it is suggested to reduce the size of the electron cloud (shrinkage) around the nucleus in each iteration if and only if the best electron is positioned in the first layer. This process is another type of exploitation property that is considered for all atoms. In this research, the following function is used for the shrinkage of the atom radius:

$$\text{IF (the best electron of } i\text{th atom} \in \text{layer1) THEN } \left[R_i^j\right]^{\boldsymbol{t+1}} = \left[R_i^j\right]^{\boldsymbol{t}} \times \exp\left(\frac{-10 \times t}{Max_t}\right) \quad (12)$$

where Max_t refers to the maximum number of iterations (epochs), t refers to the current iteration step, and $\left[R_i^j\right]^{\boldsymbol{t+1}}$ denotes the new value of the maximum possible distance of the electron in layer j of atom i (dash lines around cores in Fig. 2).

- **Phase V: Termination phase (composing the homonuclear molecule)**

The fifth phase (the termination phase) is the final step of the algorithm in which the algorithm solutions are determined. In this phase, the calculations end and the solution is converged when either the solution process reaches the final epoch number ($t = Max_t$) or the final error of the objective function E_{final} is less than the convergence rate ($e.g.\ \ \varepsilon = 1 \times 10^{-6}$).

$$E_{final} < \varepsilon \quad (13)$$

The error measurement criterion can be selected by the user (e.g. mean squared error, MSE) with respect to the objective function. In this study, the value of the test functions is the error. In this case, the set of paramount atoms and their adjacent atoms that are located next to each other during exploitation compose a homonuclear molecule. The position of the nucleus of the paramount atom of the composed molecule is considered to be the global optima. It is important to note that the position of other atoms determines the other local optimal solutions. For example, according to Fig. 2, in this phase, the position of the ozone molecule is the global solution and the locations of other atoms are the other solutions for the problem. The flowchart of the HMO algorithm described in five phases is shown in Fig. 4.

4. A review of the other applied meta-heuristic methods in this study

In the present study, to evaluate and challenge the competence of the HMO algorithm, three well-known algorithms of genetic algorithm (GA), particle swarm optimization (PSO), and differential evolution (DE) are employed that have been confirmed by the researchers in many practical works, which will be briefly described below.

4.1. Genetic algorithm (GA)

The GA uses the Darwinian evolutionary theory (using the operators of crossover, mutation, and selection) to reach the optimal solution [5] and begins to work by selecting a set of solutions as the initial population of the algorithm [43]. The selection of the initial population can be either random selection or random by regions, and the random selection is usually the most widely used. In this population, two solutions are selected

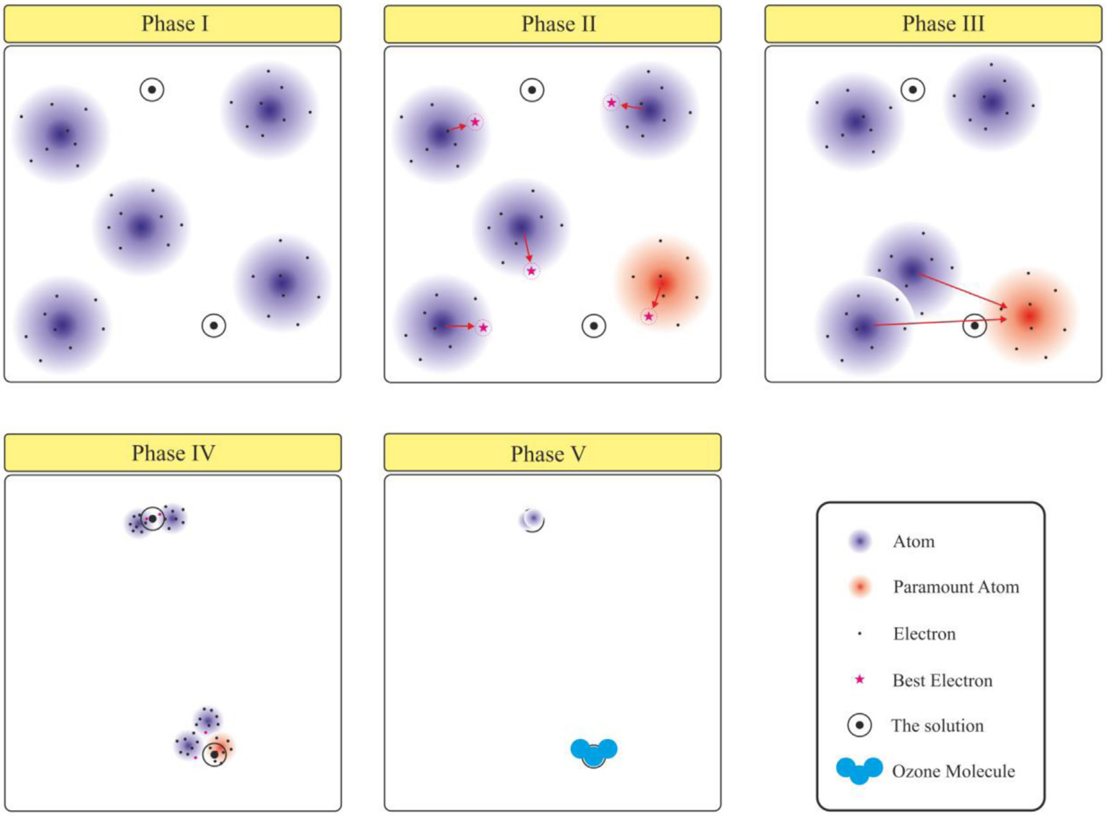


Fig. 3. Schematic view of the various stages of the HMO algorithm; *Phase I*-Initiation: random distribution of five oxygen atoms each with eight electrons in a search space with two extrema; *Phase II*-Exploration: movement of each atom toward the best position of its best electron; determining the paramount atom (it is specified by the red color); *Phase III*-Exploitation: movement of adjacent atoms toward the paramount atom; *Phase IV*-Iteration: repetition until reaching a determined convergence criterion; *Phase V*-Termination: finding the best solution by forming a homonuclear molecule.

as parents and, then, the children are generated by combination and mutation. Through generating the offspring, a population composed of main population and population of the offspring is created. As the objective function is defined for the algorithm, the precision of each individual of the population will be identified. In the population, those who have a better solution are selected as the new population. This process is repeated until the algorithm reaches the optimal solution.

4.2. Particle swarm algorithm (PSO)

The PSO algorithm is the collective behavior simulation which is used to indicate the group movement such as flocks of birds and school of fish. The PSO algorithm, similar to other evolutionary computational methods, uses a population that is the potential solution of the studied problem. As the initial population in this algorithm, the particles are randomly distributed in the specified search space. For each particle, a velocity vector and a location vector are defined [8]. In each iteration of the algorithm, the velocity and location of the particles are updated given the personal experience and experience of the best member. This process is repeated until reaching the optimal solution.

4.3. Differential evolution algorithm (DE)

DE is a population-based optimization algorithm which uses the operators of crossover, mutation, and selection, similar to the genetic algorithm. The main difference between these two algorithms is that the genetic algorithm relies on the crossover, but the DE algorithm relies on the mutation operator [12]. This algorithm uses the mutation operator as a search structure and uses the selection operator to conduct the search process toward the future regions. One of the advantages of this algorithm is having a memory that preserves the information of proper solutions in the current population. Another advantage of this algorithm is the selection operator. In this algorithm, all the members of the population have equal chance for being selected as the parent. In this algorithm, after generation of the new population by parents using the defined objective function, the best members enter the next step of the algorithm as the new population.

5. Implementation

The performance assessment of the proposed homonuclear molecules algorithm (HMO) as well as genetic algorithm (GA), particle swarm algorithm (PSO), and differential evolution algorithm (DE) is accomplished by using three types of bench mark functions. These functions are generally divided into three classes of unimodal high-dimensional functions, multimodal high-dimensional functions, and multimodal low-dimensional functions (Table 1). For the unimodal high-dimensional functions that have an extremum point, the convergence rate of the algorithm will be evaluated. The multimodal high-dimensional functions also have several local extrema points and the capability of the algorithm in avoiding from being trapped in local points as well as finding the global solution will be tested. The multimodal 2D functions can also be used for the capability of escaping from the local points and finding several solutions for the problem.

In this research, MATLAB codes were developed for all of the three algorithms. Besides, a core i7 processor hardware system

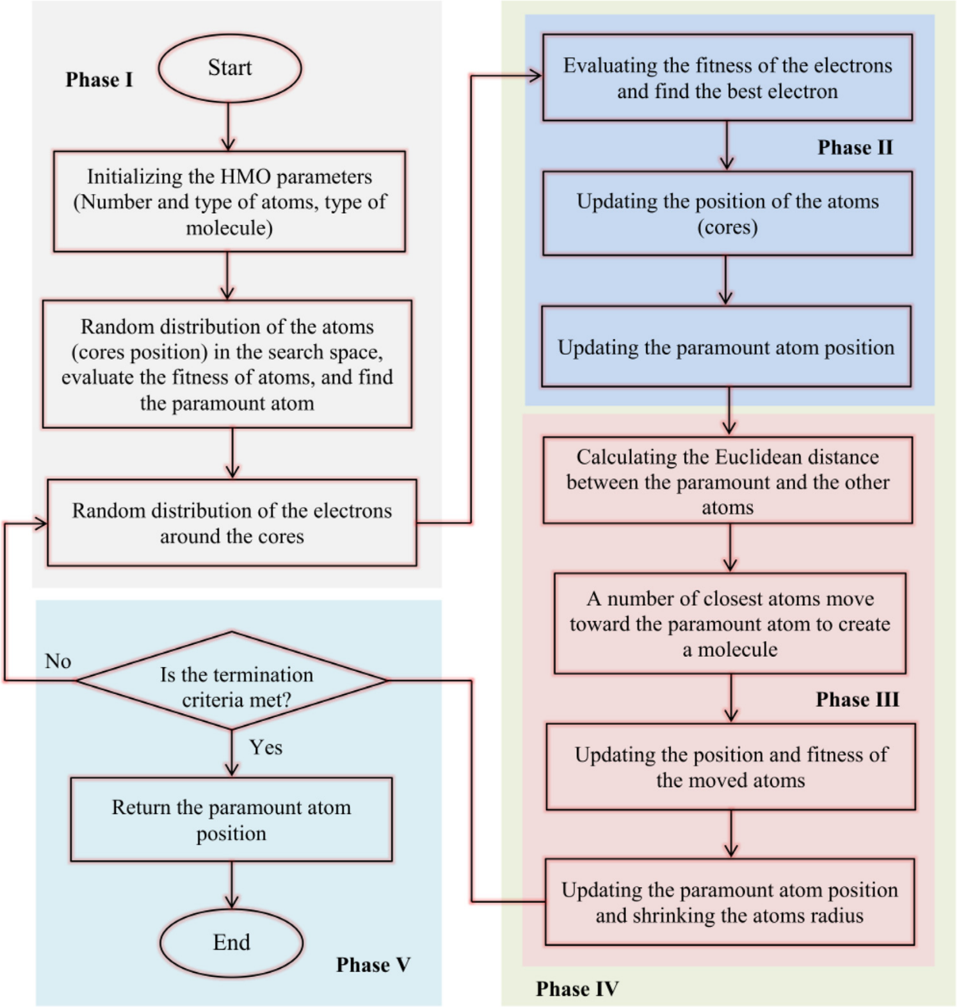

Phase I
Start
Initializing the HMO parameters (Number and type of atoms, type of molecule)
Random distribution of the atoms (cores position) in the search space, evaluate the fitness of atoms, and find the paramount atom
Random distribution of the electrons around the cores
Evaluating the fitness of the electrons and find the best electron
Phase II
Updating the position of the atoms (cores)
Updating the paramount atom position
Calculating the Euclidean distance between the paramount and the other atoms
A number of closest atoms move toward the paramount atom to create a molecule
Phase III
Updating the position and fitness of the moved atoms
Updating the paramount atom position and shrinking the atoms radius
Phase IV
No
Is the termination criteria met?
Yes
Return the paramount atom position
End
Phase V

Fig. 4. Process flowchart of the general structure of the HMO algorithm.

equipped with 8 GB RAM was used for the execution processes. In order to solve the equations of Table 1 using the HMO method, for the unimodal and multimodal high-dimensional functions, twelve Se (Selenium) atoms ($N = 12$) were considered in the search space. The element Se had an atomic number of 34 ($z = 24$) and, thus, it contained 34 electrons [40]. Considering the Se_8 molecule, 8 of these atoms would generate this molecule [40]. This implies that 408 electrons ($N \times z = 408$) as searching agents are chosen for the solving process. So as to reaching a uniform initial condition for all the meta-heuristic algorithms (HMO, GA, PSO & DE) the same number of searching agents is considered for the other algorithms (PSO, GA & DE) as the initial population (population = 408).

For the multimodal low-dimensional functions, the phosphorus (P) atom ($z = 15$) was considered. A number of 30 atoms of this element were distributed in the search space [40]. Four of these atoms would generate the P_4 molecule. For the other algorithms (PSO, GA & DE), the initial population was considered equal to the same number as the number of electrons of 30 phosphorus atoms (population = 450). In Fig. 5, the structure of the molecules used in 3D space is shown.

In Table 2, the initial values and coefficients used to optimize the above cases are shown for the proposed HMO method and other optimization methods. These parameters were chosen based on the previous studies for the classical algorithms (PSO, GA, and DE [6,21,44]), whilst a trial-and-error procedure was applied for the HMO.

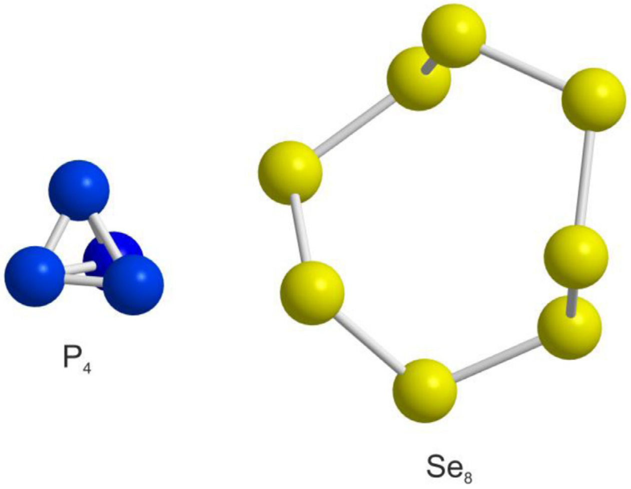


Fig. 5. Schematic structure of (a) Se8 molecule to be composed in the termination phase of unimodal high-dimensional functions and multimodal high-dimensional functions and, (b) P_4 molecule to be composed in the termination phase of multimodal low-dimensional functions in Table 1.

Table 1
Selected bench mark functions and the applied dimension(s) for assessing the performance of HMO, GA, PSO and DE algorithms.

Category	Test function	Bounds	Dimension
Unimodal	$f_1(\mathrm{x}) = \sum_{i=1}^{D} x_i^2$	[−100, 100]	D
	$f_2(\mathrm{x}) = \sum_{i=1}^{D} i x_i^4 + random[0, 1)$	[−1.28, 1.28]	D
	$f_3(\mathrm{x}) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	[−30, 30]	D
	$f_4(\mathrm{x}) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	[−100, 100]	D
Multimodal high-dimensional	$f_5(\mathrm{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + \exp(1)$	[−32, 32]	D
	$f_6(\mathrm{x}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{i^{0.5}}\right) + 1$	[−600, 600]	D
	$f_7(\mathrm{x}) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2\left[1 + \sin^2(3\pi x_1 + 1)\right] + (\mathrm{x}_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\}$	[−50, 50]	D
	$+ \sum_{i=1}^{D} g(x_i, 5, 100, 4)$		
	$g\ (x_i, a, b, c) = \begin{cases} b(x_i - a)^c & x_i > a \\ 0 - a < x_i < a & \\ b(-x_i - a)^c & x_i < -a \end{cases}$		
Multimodal low-dimensional	$f_8(x) = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$		2
	$f_9(\mathrm{x}_1, \mathrm{x}_2) = \left[\sum_{i=1}^{5} i\cos((i+1)x_1 + i)\right] \times \left[\sum_{i=1}^{5} i\cos((i+1)x_2 + i)\right]$	[−5, 5]	2
	$f_{10}(x_1, x_2) = a\left(x_2 - bx_1^2 + cx_1 - d\right)^2 + e(1 - f)\cos(x_1) + e$ $a = 1, b = \frac{5.1}{4\lambda^2},\ c = \frac{5}{\pi},\ d = 6, e = 10, f = \frac{1}{8\pi}$	[−15, 15]	2

Table 2
The initial values and optimization coefficients in HMO, PSO, DE, and GA algorithms in the present research.

Method	General tuning parameters	Parameters for low-dimensional functions[a]	Parameters for high-dimensional functions[b]
PSO	Initial inertia weight = 1 Inertia Weight Damping Ratio = 0.99 Cognitive acceleration (C_1) = 2 Social acceleration (C_2) = 2	Population = 450	Population = 408

GA	Mutation = 0.8
	Crossover = 0.7
	Selection Pressure = 8
		Population = 450	Population = 408

DE	Lower Bound of Scaling Factor = 0.2 Upper Bound of Scaling Factor = 0.8 Crossover = 0.7	Population = 450	Population = 408
HMO	Distance Reduction Factor (γ) = [0, 0.1] Distance Control Coefficient (β) = 2	Population = 450 Atom type = P ($z = 15$) Number of atoms (N) = 30 Molecule type = P_4 ($m = 4$)	Population = 408 Atom type = P ($z = 34$) Number of atoms (N) = 12 Molecule type = Se_8 ($m = 8$)

[a] Unimodal high-dimensional functions and multimodal high-dimensional functions in Table 1.
[b] Multimodal low-dimensional functions.

6. Results and discussion

In this study, the ability of homonuclear molecules algorithm (HMO) evaluated in minimizing the mathematical test functions. After determination of the objective function (Table 1) and establishment of algorithms (Table 2), the performance of the HMO algorithm was evaluated and compared with the three algorithms of genetic algorithm (GA), particle swarm algorithm (PSO), and differential evolution algorithm (DE) based on the value of the test functions in the best and weakest performance and total average of optimization cases in 30 iterations of optimization. The results of solving unimodal functions using 4 algorithms for dimensions of 30 and 15 for 30 consecutive runs are presented in Table 3.

The results of Table 3 show that the HMO algorithm had acceptable results in the optimization of the used functions and could compete with well-known algorithms such as PSO, GA, and DE. The performance of the HMO algorithm in the final solution of the used functions was in the way that it had more precise solution than the other algorithms in 3 functions in the dimensions of 15 and 30. It should be noted that the existence of better performance in these functions did not necessarily mean the superiority of this algorithm to the other algorithms. In order to consider the performance of the algorithms in the rate of

Table 3
Comparing the results of various algorithms in unimodal functions.

Function	Dimension	Result type	HMO	PSO	DE	GA
F_1	15	Best	$\mathbf{3.07 \times 10^{-124}}$	2.23×10^{-87}	6.12×10^{-28}	8.74×10^{-18}
		Worst	$\mathbf{1.23 \times 10^{-121}}$	1.46×10^{-59}	6.15×10^{-27}	8.17×10^{-16}
		Mean	$\mathbf{1.6 \times 10^{-122}}$	1.45×10^{-60}	2.45×10^{-27}	4.64×10^{-16}
F_2	15	Best	3.06×10^{-4}	4.85×10^{-4}	1.53×10^{-3}	$\mathbf{4.86 \times 10^{-5}}$
		Worst	1.55×10^{-3}	1.71×10^{-3}	8.42×10^{-3}	$\mathbf{5.37 \times 10^{-4}}$
		Mean	8.1×10^{-4}	9.27×10^{-4}	5.82×10^{-3}	$\mathbf{2.45 \times 10^{-4}}$
F_3	15	Best	**2.95**	4.024	10.66	10.10
		Worst	**8.10**	25.09	18.02	72.19
		Mean	**6.35**	9.64	13.80	30.06
F_4	15	Best	$\mathbf{2.67 \times 10^{-31}}$	3.01×10^{-10}	134.07	0.34
		Worst	$\mathbf{2.24 \times 10^{-27}}$	19.8×10^{-4}	313.37	5.68
		Mean	$\mathbf{3.85 \times 10^{-28}}$	2.07×10^{-5}	195.59	2.22
F_1	30	Best	$\mathbf{2.07 \times 10^{-75}}$	5.77×10^{-18}	3.12×10^{-12}	2.34×10^{-5}
		Worst	$\mathbf{9.47 \times 10^{-75}}$	2.54×10^{-12}	1.42×10^{-11}	1.85×10^{-4}
		Mean	$\mathbf{3.65 \times 10^{-76}}$	5.01×10^{-13}	7.01×10^{-12}	7.92×10^{-5}
F_2	30	Best	1.73×10^{-3}	5.28×10^{-3}	12.68×10^{-2}	$\mathbf{3.85 \times 10^{-4}}$
		Worst	5.59×10^{-3}	2.80×10^{-3}	2.61×10^{-2}	$\mathbf{2.07 \times 10^{-3}}$
		Mean	3.19×10^{-3}	1.53×10^{-3}	2.08×10^{-2}	$\mathbf{1.46 \times 10^{-3}}$
F_3	30	Best	**20.60**	22.01	26.53	21.11
		Worst	**28.92**	100.12	80.41	89.72
		Mean	**24.30**	54.59	37.07	70.03
F_4	30	Best	$\mathbf{7.50 \times 10^{-5}}$	17.05	18538.81	60.55
		Worst	$\mathbf{3.20 \times 10^{-4}}$	162.07	25772.54	243.12
		Mean	$\mathbf{1.69 \times 10^{-4}}$	70.27	23180.34	130.67

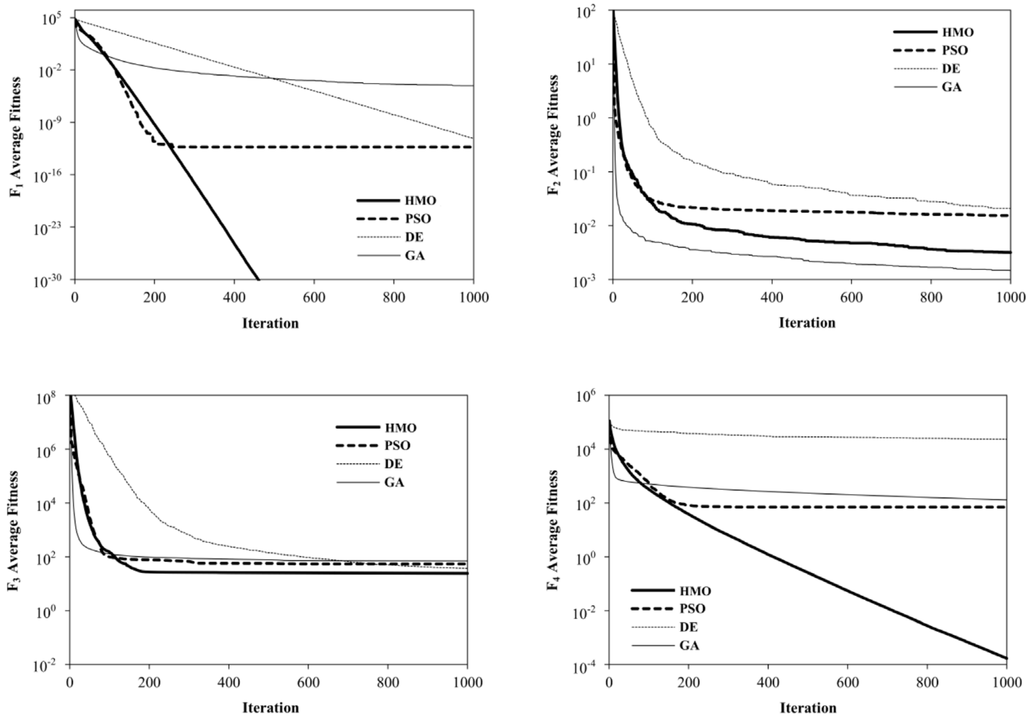


Fig. 6. Comparison of the convergence rate of the HMO algorithm with other algorithms ($D = 30$).

convergence to the solution, these diagrams are depicted for dimension of 30 for all 4 unimodal functions (see Fig. 6).

The depicted diagrams indicate the fair performance of the HMO algorithm. Taking into account the results of three functions of F_1, F_3, and F_4, the superiority of the HMO algorithm to other algorithms is evident. In the F_2 function, the GA algorithm has better performance than other algorithms and the performance of the PSO and DE is close to each other. The scalability analysis of the HMO and other algorithms is shown in Fig. 7.

Fig. 7 shows that the HMO's performance, especially in high dimension problems, is better than the other applied algorithms. Multimodal functions have many local extrema, hence, reaching a global solution in these functions will be more difficult for algorithms. The results of these algorithms for 30 dimensions and 30 runs are presented in Table 4.

The results of Table 4 show that all three used algorithms approached the solution. The HMO and PSO algorithms approached the solution with more precision than the other two algorithms.

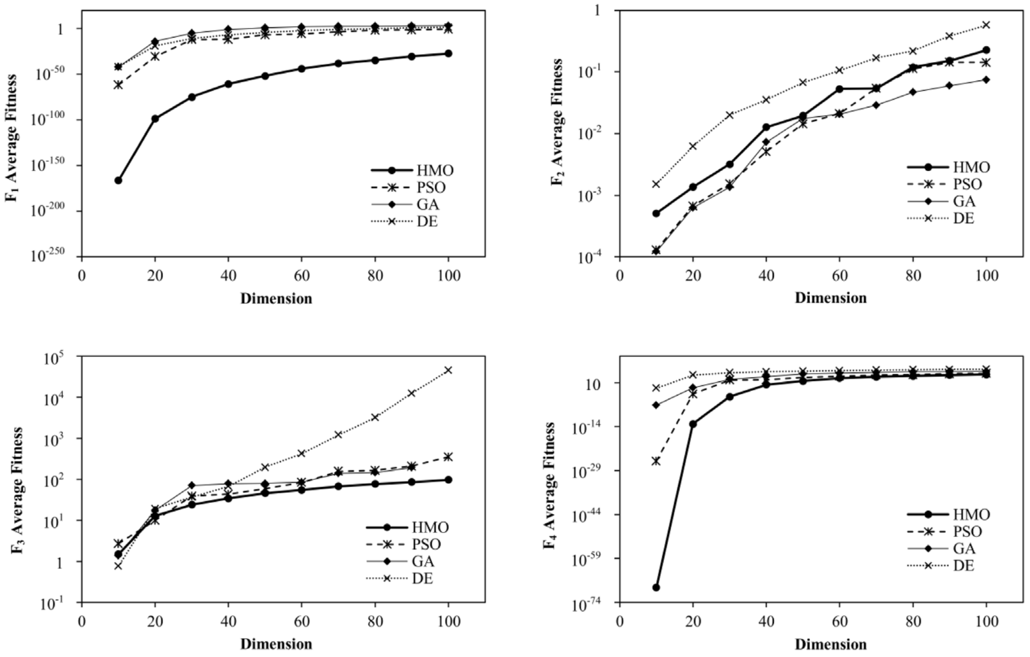


Fig. 7. Scalability results of the HMO, PSO, GA, and DE for Unimodal functions.

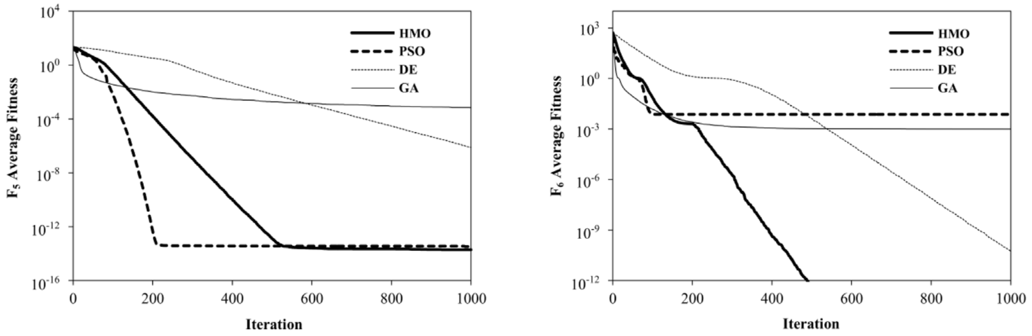


Fig. 8. Comparison of convergence rates of various algorithms in multimodal functions.

Table 4
Results of various algorithms for multimodal functions.

Function	Dimension	Result type	HMO	PSO	DE	GA
F_5	30	Best	7.99×10^{-15}	1.51×10^{-14}	7.14×10^{-7}	3.97×10^{-4}
		Worst	1.51×10^{-14}	4.35×10^{-14}	8.81×10^{-7}	1.04×10^{-3}
		Mean	$\mathbf{1.29 \times 10^{-14}}$	3.39×10^{-14}	7.81×10^{-7}	6.97×10^{-4}
F_6	30	Best	1.11×10^{-15}	0	4.06×10^{-11}	2.7×10^{-6}
		Worst	4.21×10^{-15}	2.22×10^{-2}	9.97×10^{-11}	9.86×10^{-3}
		Mean	$\mathbf{2.91 \times 10^{-15}}$	7.4×10^{-3}	5.6×10^{-11}	9.96×10^{-4}
F_7	30	Best	3.57×10^{-32}	1.47×10^{-32}	4.15×10^{-12}	6.38×10^{-8}
		Worst	6.03×10^{-32}	2.33×10^{-31}	6.29×10^{-12}	1.10×10^{-6}
		Mean	$\mathbf{4.7 \times 10^{-32}}$	5.83×10^{-32}	5.49×10^{-12}	2.68×10^{-7}

These results demonstrated that the proposed algorithm of the electron cloud in unimodal functions was also capable of optimization. In Fig. 8, the convergence diagrams are shown for two cases of the used functions.

The results of Fig. 8 illustrate that the HMO algorithm shows a proper convergence rate compared to the other methods, and could properly optimize the studied functions with high precision. The convergence diagrams show that the PSO algorithm demonstrates a very good performance in the F_5 function, but the performance of this algorithm in the F_6 function was not as good as the other applied algorithms. This point notifies that if an algorithm has a good convergence rate in some functions, the performance of that algorithm might not be considered as an efficient algorithm in practical problems or complicated mathematical functions. In Fig. 9, the scalability diagrams are shown for the two cases of multimodal functions.

One of the strength points of the HMO algorithm is to find local extrema or different solutions for practical functions and problems. To indicate this property of HMO algorithm, three functions having multiple solutions were selected, and the performance of various algorithms was investigated. As noted earlier, 30 phosphorus atoms were used to disperse electrons in the search space. If the goal was only to determine the optimal solution, all the algorithms would be able to find it with much smaller

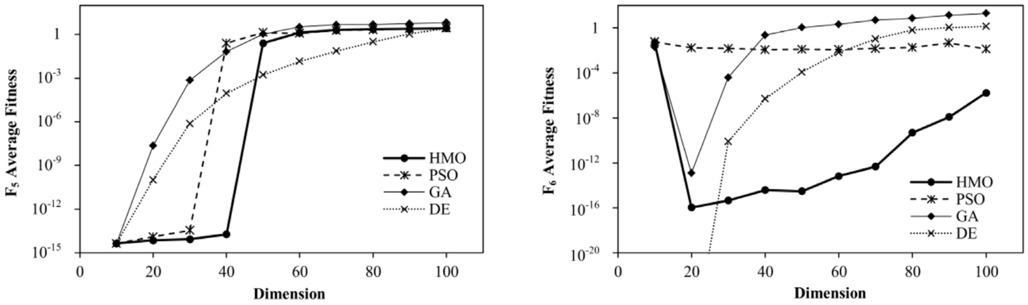


Fig. 9. Scalability results of the HMO and other algorithms for multimodal functions.

Table 5
Results of various algorithms for multimodal functions with fixed dimensions ($D = 2$).

Function	Result type	Method HMO	PSO	GA	DE
F_8	Best fitness	−1.0316	−1.0316	−1.0316	−1.0316
	Solutions	(0.0898, 0.7127) (−0.0898, 0.7127) (−1.6071, −0.5687) (1.6067, 0.5679)	(0.0898, 0.7127)	(0.0898, 0.7127)	(0.0898, 0.712)
F_9	Best fitness	−186.7309	−186.7309	−186.7309	
	Solutions	(−1.4251, 5.4829) (−7.0835, −1.4251) (5.4829, 4.8581) (4.8581, −7.0835) (−0.8003, −7.7083)	(5.4829, 4.8581)	(−0.8003, −1.4251)	(5.4829, 4.858) (7.0835, 4.858) (4.8581, 0.800)
F_{10}	Best fitness	0.3979	0.3979	0.3979	0.3979
	Solutions	(3.1416, 1.4058) (−3.1416, 10.5942) (9.4248, −7.7825)	(9.4248, −7.7825)	(−3.1416, 10.5942)	(3.1416, 1.405)

population; however, the goal was to examine the performance of the algorithms in finding several solutions for functions and, thus, a large initial population was considered. The results of one execution are presented in Table 5.

The results of Table 5 show that all the algorithms were capable of finding the optimal solution. The superiority of the HMO algorithm to the other algorithms is that this algorithm is capable of finding most of these solutions in a run for the functions having several optimal solutions, and if the population number of the initial designated atoms is large, most of these extrema will be identifiable without making more computational cost for HMO compared to the other meta-heuristic algorithms. In order to have a diagnostic perspective and observation for the problem-solving process in the HMO algorithm, the convergence procedure of the problem during different iteration is shown in Fig. 10.

Fig. 10 shows that the desired function has 6 extrema points, two of which are global. In the first iteration, the paramount atom that has the best solution is shown in red. Three atoms near this atom are shown in blue, which are approaching to generate the desired molecule. It is observed that in subsequent iterations, all the atoms are located on the extrema points, and while determining the global solutions, the local solutions are also extracted.

Another point that specifies the capability of an algorithm in solving the optimization problems is the stability of an algorithm in achieving the desired solution in different executions. To examine this issue, the boxplots of final results for 4 functions for the dimensions of 10, 20, and 30 for the results of 30 consecutive executions are depicted in Fig. 11.

The boxplots depicted in Fig. 11 show that in the lower dimensions, all the algorithms had good stability in reaching the final solution. As the dimensions of the problem increase, two algorithms of GA ($20 \leq$ D) and PSO ($30 \leq$ D) lost their stability, but the DE algorithm worked better than these two algorithms. The diagrams of HMO algorithm had the best performance and very good stability in achieving the final solution. This problem indicates that the proper exploration operators that are selected for the HMO algorithm helps this algorithm not be caught in the trap of local solutions.

To analyze the exploration and exploitation processes of the applied algorithms, the graphs of these processes for multimodal functions are depicted based on Morales-Castañeda et al. [45] in Fig. 12.

Based on Fig. 12, the exploration effect is very short for all algorithms. The average exploration and exploitation rate of all four algorithms are 1.151 and 98.848 percent, respectively. The DE exploration and exploitation rates are far from other algorithms. The GA and PSO have better exploration and exploitation rates than the HMO and DE. The reason is that the PSO and GA candidate solutions are at the end of the iterations located at the same position (global solution); whereas the HMO and DE candidate solutions are located at different positions (local solutions and global solution). The HMO shows an appropriate exploration and exploitation rates.

To find out if there is a significant difference between the results of the HMO and the other applied meta-heuristic algorithms, the final results of 30 continues executions of different test functions for $D = 30$ were tested using the Wilcoxon's rank-sum test (Table 6).

Table 6 indicates that in most test functions there is a significant difference between the final results of the HMO and the other algorithms ($a = 0.05\%$). Based on the p-values, in two functions (F_3 and F_6) there is no a significant difference between the final results of PSO and HMO.

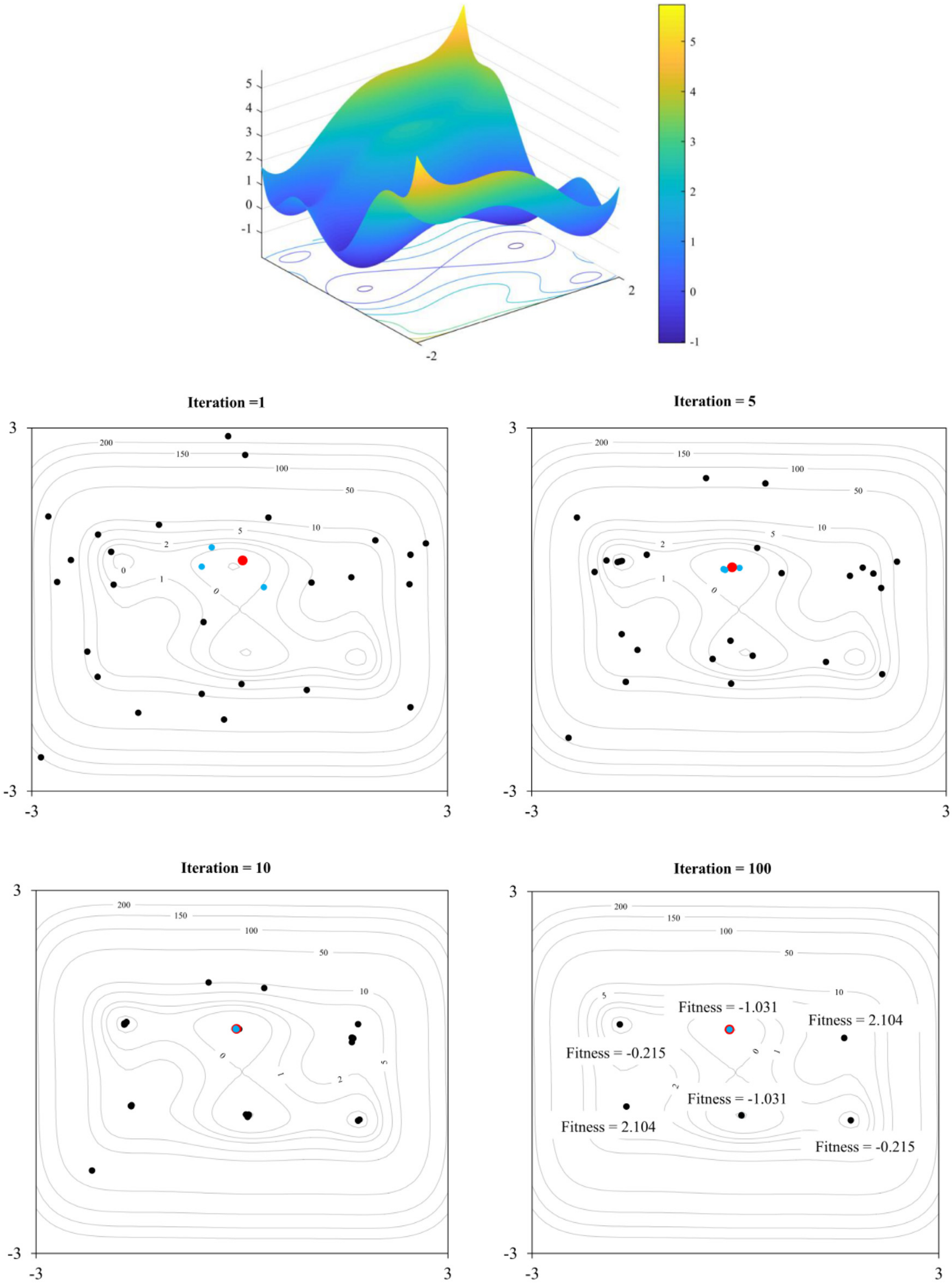


Fig. 10. Schematic view of the historical behavior of the optimization process of HMO algorithm in F_8 function (with 2 global extrema and 4 relative extrema) in different iterations of 1, 5, 10, and 100.

7. Discussion

The performance of HMO in optimizing 10 different benchmark functions was compared with three well-known and classical algorithms, including GA, PSO, and DE. The results revealed that the HMO algorithm outperforms GA, PSO, and DE. Moreover, the ability of HMO to find local solutions was confirmed (Table 5 and Fig. 10). The exploration and exploitation rates of HMO during the optimization processes are acceptable (Fig. 12). The Wilcoxon rank sum test results showed that there is a significant difference between the final results of HMO and those of the other algorithms, which confirms its effectiveness (Table 6). The scalability analysis indicated that the higher dimension problems are more difficult to optimize for all algorithms, which is in line with the results reported by Gao et al. [46]. However, HMO's performance in high-dimension problems is acceptable.

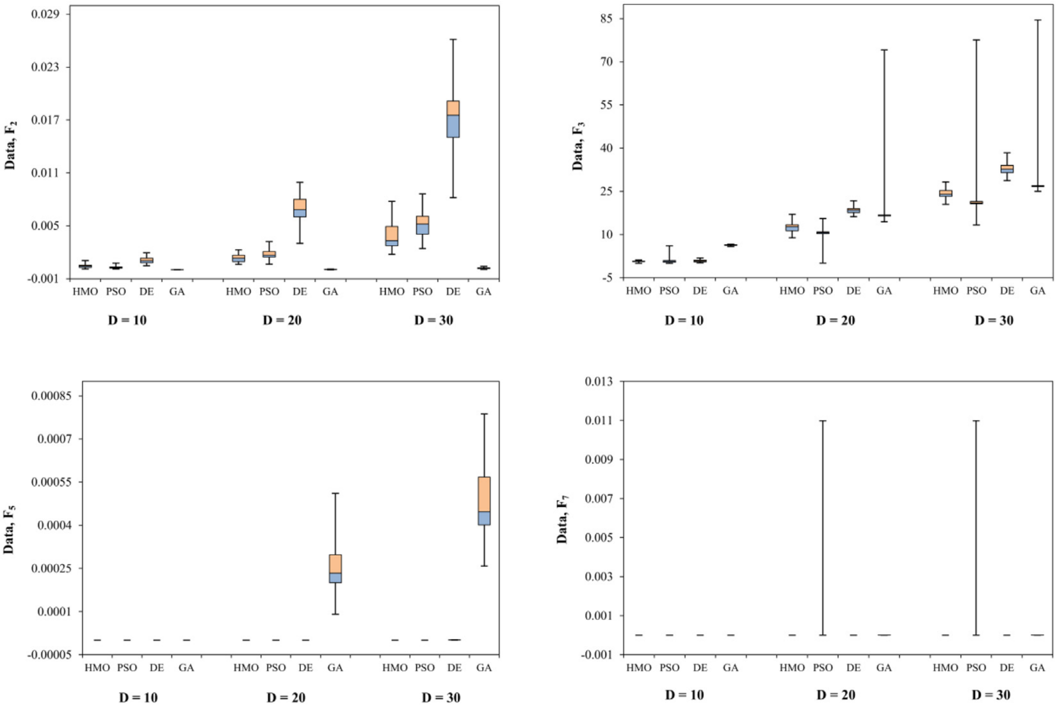


Fig. 11. Boxplot for 4 functions in 3 different dimensions.

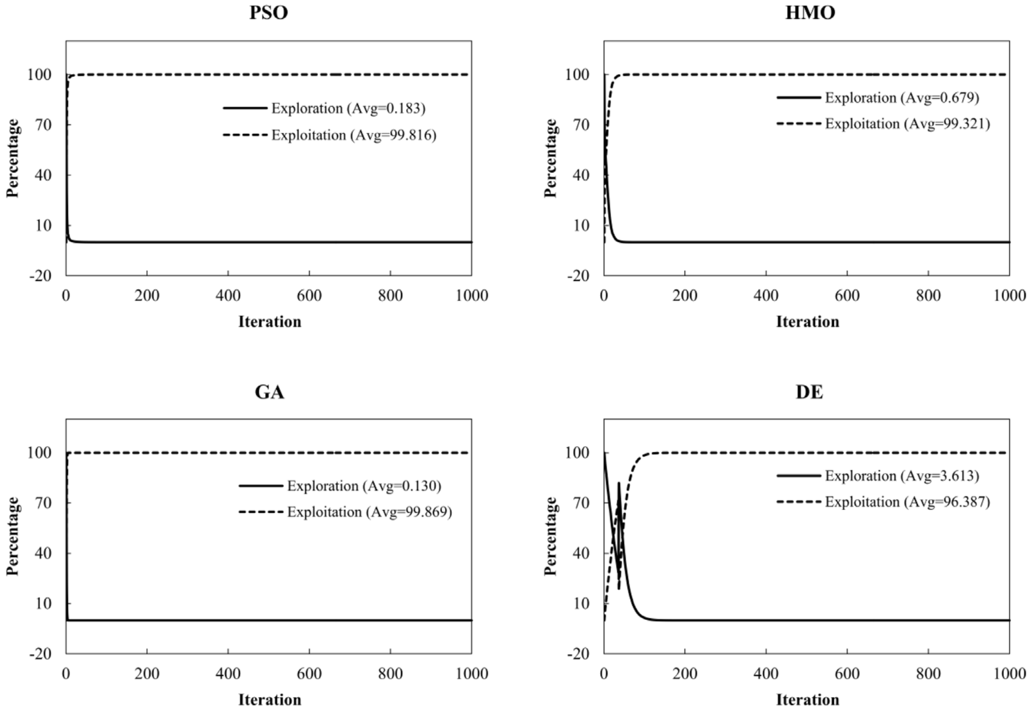


Fig. 12. Exploration and exploitation graph for 6 multimodal functions.

Recently, many new algorithms or modified versions of conventional algorithms have been developed by researchers. EO is a new optimization algorithm that has been developed by Faramarzi et al. [36]. Faramarzi et al. [36] compared EO's performance with several state-of-the-art algorithms (classical and new algorithms) in optimizing CEC2017 [47] functions and verified its superiority. In this regard, the performance of HMO in solving hybrid functions of CEC2017 was compared with EO. Table 7 presents the results of the EO and HMO algorithms in 30 continuous runs under the same conditions (population = 408, $D = 30$).

The results presented in Table 7 demonstrate that the performances of both HMO and EO are satisfactory. Although EO is a reliable algorithm, HMO can compete with it; as can be seen in Table 7, HMO surpassed in half of the applied functions, e.g. CEC'17$_{10}$, CEC'17$_{13}$, CEC'17$_{14}$, CEC'17$_{16}$, and CEC'17$_{18}$.

Table 6
The Wilcoxon rank sum test results between the HMO and the other applied algorithms ($a = 0.05\%$).

Function	Algorithm	p-value	Significantly different (95%)
F$_1$	GA	1.83×10^{-4}	Yes
	PSO	1.83×10^{-4}	Yes
	DE	1.83×10^{-4}	Yes
F$_2$	GA	4.40×10^{-4}	Yes
	PSO	2.46×10^{-4}	Yes
	DE	1.823×10^{-4}	Yes
F$_3$	GA	0.0028	Yes
	PSO	0.0639	No
	DE	3.28×10^{-4}	Yes
F$_4$	GA	1.83×10^{-4}	Yes
	PSO	1.83×10^{-4}	Yes
	DE	1.82×10^{-4}	Yes
F$_5$	GA	1.28×10^{-4}	Yes
	PSO	3.10×10^{-4}	Yes
	DE	1.28×10^{-4}	Yes
F$_6$	GA	1.78×10^{-4}	Yes
	PSO	0.47	No
	DE	1.78×10^{-4}	Yes
F$_7$	GA	1.82×10^{-4}	Yes
	PSO	0.025	Yes
	DE	1.82×10^{-4}	Yes

The most efficient features of the HMO algorithm can be summarized as follows:

1- Finding local solutions: HMO is able to find local and global solutions with suitable accuracy. Local solutions are also important for users. For example, in the training process of machine learning models, the global solution may be introduced by an overfitting problem. Some local solutions may have better performance in the testing dataset.

2- Self-adaption of hyper-parameters: Similar to an aggregative learning gravitational search algorithm [48], HMO has the self-adaptive ability to tune parameters such as the maximum possible distance of the electrons on the layer from the nucleus (R, Eq. (6)).

3- Population evolution: The HMO population is evaluated based on two strategies. The first strategy is set up according to the best electron of atoms, whereas the second strategy considers the paramount atom.

4- Exploitation and exploration: The results indicated that HMO has suitable exploitation and exploration rates.

HMO needs some initial parameters before starting the solving process, such as the number of searching agents in the initial population, the type of atoms and molecules, and the distance control coefficient. In this study, these parameters have been initialized based on the trial-and-error procedure. As for the future directions, more studies can be conducted to optimize these values and parameters according to the given problem and even improve HMO's performance by applying strategies such as chaotic local search to this algorithm [16,49].

8. Conclusion

In this research, the novel meta-heuristic HMO algorithm was proposed to optimize complex problems. This algorithm is inspired by the structure of electrons around the nucleus of atoms based on Bohr's atomic model in order to generate a homonuclear molecule.

Different functions were considered to investigate the performance of this algorithm, and its results were compared with three well-known algorithms of GA, PSO, and DE, as well as EO as a new and state-of-the-art algorithm. The results showed that in most of these functions, the HMO algorithm had more favorable performance than the other algorithms. The HMO algorithm had a successful performance in determining the final solution as well as the convergence rate. In this research, various functions were optimized in different dimensions and the results demonstrated that all the methods had appropriate stability in achieving the solution in small dimensions. However, with an increase in the dimensions of the problem, the stability of the two methods of HMO and DE was shown to be higher than the GA and PSO algorithms. The HMO algorithm had better precision in high dimensions than classical algorithms. Both HMO and EO are efficient algorithms and showed suitable performance in optimizing the CEC2017 functions. One of the innovations of the HMO algorithm is finding several solutions for the problem. A number of functions with several solutions were considered to examine this matter and the results showed that this algorithm could find most of these solutions. In addition, the boxplots demonstrated that this algorithm had appropriate stability in achieving the desired solution in consecutive iterations.

Notations

n	Principal quantum number
a_1	Coefficient of Bohr radius formula
s, p, d, f, g, h	Sublayer letter
ml	Magnetic quantum number
m_s	Quantum numbers
D	Problem dimension
z	Atomic number
N	Number of atoms
m	Number of homonuclear molecule atoms
R_i^j	Maximum possible distance of the electrons of layer j of atom i from nucleus
$R_{\max}$	Maximum bound of the search space along an axis
$R_{\min}$	Minimum bound of the search domain along the axis
$n_{\max}$	Maximum quantum number related to the desired element
β	Distance control coefficient
$Xelectron$	Position of the electron of
$Xnucleus_i^t$	Position of nucleus of atom
r_j^i	A random vector in the interval [-R_i^j, R_i^j]
$Xelectron^{best}$	Position of the electron that has the best solution
d	Euclidean distance between the nucleus of paramount atom and other nucleuses
$Xnucleus\_paramount$	Position of the nucleus of the paramount atom
γ	Distance reduction factor
Max_t	Maximum number of iterations (epochs)
E_{final}	Final error of the objective function
ε	Convergence rate
C_1	Cognitive acceleration
C_2	Social acceleration
t	Iteration
π	Constant coefficient (the ratio of the circumference of a circle to its diameter)

Table 7
The results of the HMO and EO algorithms in solving hybrid functions of CEC2017.

Function	Dimension	Result type	HMO	EO	Winner
CEC'17$_{10}$	30	Best	$3.25 \times 10^{+3}$	$2.64 \times 10^{+3}$	HMO
		Worst	$4.95 \times 10^{+3}$	$5.89 \times 10^{+3}$	
		Mean	$4.19 \times 10^{+3}$	$4.42 \times 10^{+3}$	
CEC'17$_{11}$	30	Best	$1.15 \times 10^{+3}$	$1.11 \times 10^{+3}$	EO
		Worst	$1.27 \times 10^{+3}$	$1.2 \times 10^{+3}$	
		Mean	$1.22 \times 10^{+3}$	$1.14 \times 10^{+3}$	
CEC'17$_{12}$	30	Best	$1.19 \times 10^{+5}$	$0.25 \times 10^{+5}$	EO
		Worst	$7.83 \times 10^{+5}$	$10.82 \times 10^{+5}$	
		Mean	$3.77 \times 10^{+5}$	$2.29 \times 10^{+5}$	
CEC'17$_{13}$	30	Best	$1.70 \times 10^{+4}$	$0.41 \times 10^{+4}$	HMO
		Worst	$1.70 \times 10^{+4}$	$6.37 \times 10^{+4}$	
		Mean	$2.92 \times 10^{+4}$	$3.23 \times 10^{+4}$	
CEC'17$_{14}$	30	Best	$1.71 \times 10^{+3}$	$1.98 \times 10^{+3}$	HMO
		Worst	$2.78 \times 10^{+3}$	$20.68 \times 10^{+3}$	
		Mean	2.1910^{+3}	$8.43 \times 10^{+3}$	
CEC'17$_{15}$	30	Best	$1.19 \times 10^{+4}$	$0.15 \times 10^{+4}$	EO
		Worst	$3.82 \times 10^{+4}$	$4.23 \times 10^{+4}$	
		Mean	$2.29 \times 10^{+4}$	$1.35 \times 10^{+4}$	
CEC'17$_{16}$	30	Best	$1.93 \times 10^{+3}$	$2.13 \times 10^{+3}$	HMO
		Worst	$2.69 \times 10^{+3}$	$2.91 \times 10^{+3}$	
		Mean	$2.45 \times 10^{+3}$	$2.49 \times 10^{+3}$	
CEC'17$_{17}$	30	Best	$1.94 \times 10^{+3}$	$1.74 \times 10^{+3}$	EO
		Worst	$2.29 \times 10^{+3}$	$2.13 \times 10^{+3}$	
		Mean	$2.13 \times 10^{+3}$	$1.85 \times 10^{+4}$	
CEC'17$_{18}$	30	Best	$2.68 \times 10^{+4}$	$1.08 \times 10^{+4}$	HMO
		Worst	$11.10 \times 10^{+4}$	$20.02 \times 10^{+4}$	
		Mean	$5.05 \times 10^{+4}$	$10.88 \times 10^{+4}$	
CEC'17$_{19}$	30	Best	$1.23 \times 10^{+4}$	$0.2 \times 10^{+4}$	EO
		Worst	$37.69 \times 10^{+4}$	$2.91 \times 10^{+4}$	
		Mean	$14.48 \times 10^{+4}$	$0.64 \times 10^{+4}$	

CRediT authorship contribution statement

Amin Mahdavi-Meymand: Conception and design of study, Analysis and/or interpretation of data, Drafting the manuscript. **Mohammad Zounemat-Kermani:** Conception and design of study, Revising the manuscript critically for important intellectual content.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors are indebted to the reviewers for valuable comments.

References

[1] H. H. Robbins, S. S. Monro, A stochastic approximation method, Ann. Math. Stat. 22 (3) (1951) 400–407.

[2] S. Katoch, S.S. Chauhan, V. Kumar, A review on genetic algorithm: past, present, and future, Multimedia Tools Appl. 80 (2021) 8091–8126.

[3] E.H. Houssein, A.G. Gad, K. Hussain, P.N. Suganthan, Major advances in particle swarm optimization: Theory, analysis, and application, Swarm Evol. Comput. 63 (2021) 100868.

[4] Z.Z. Wang, A. Sobey, A comparative review between genetic algorithm use in composite optimisation and the state-of-the-art in evolutionary computation, Compos. Struct. 233 (2020) 111739.

[5] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, USA, 1975.

[6] M. Zounemat-Kermani, A. Mahdavi-Meymand, Hybrid meta-heuristics artificial intelligence models in simulating discharge passing the piano key weirs, J. Hydrol. 569 (2019) 12–21.

[7] R. Koker, A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization, Inform. Sci. 222 (2013) 528–543.

[8] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, Vol. 4, 1995, pp. 1942–1948.

[9] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. 26 (1996) 29–41.

[10] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, Soft Comput. 23 (3) (2019) 715–734.

[11] S. Kirkpatrick, C.D. Gelatto, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.

[12] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (1997) 341–359.

[13] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417.

[14] J. Zhang, A.C. Sanderson, JADE: Adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.

[15] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: Proc. IEEE Congr. Evol. Comput, CEC, 2013, pp. 71–78.

[16] S. Gao, Y. Yu, Y. Wang, J. Wang, J. Cheng, M. Zhou, Chaotic local search-based differential evolution algorithms for optimization, IEEE Trans. Syst. Man Cybern. Syst. 51 (6) (2021) 3954–3967.

[17] K. Lee, Z. Geem, A new structural optimization method based on the harmony search algorithm, Comput. Struct. 82 (2004) 781–798.

[18] R. Reynolds, An introduction to cultural algorithms, in: Proceedings of the 3rd Annual on Evolutionary Programming, World Scientific, River Edge, NJ, 1994, pp. 131–139.

[19] X.S. Yang, Firefly algorithms for multimodal optimization, in: International Symposium on Stochastic Algorithms, Springer, Berlin, Heidelberg, 2009, pp. 169–178.

[20] D. Karaboga, S. Okdem, A simple and global optimization algorithm for engineering problems: differential evolution algorithm, Turk. J. Electron. Eng. 12 (1) (2004).

[21] E. Rashedi, S. Nezamabadi, S. Saryazdi, GSA: a gravitational search algorithm, Inf. Sci. (NY) 179 (13) (2009) 2232–2248.

[22] Y. Wang, Y. Yu, S. Gao, H. Pan, G. Yang, A hierarchical gravitational search algorithm with an effective gravitational constant, Swarm Evol. Comput. 46 (2019) 118–139.

[23] Y. Wang, S. Gao, Y. Yu, Z. Cai, Z. Wang, A gravitational search algorithm with hierarchy and distributed framework, Knowl.-Based Syst. 218 (2021) 106877.

[24] M. Abdechiri, M.R. Meybodi, H. Bahrami, Gases Brownian motion optimization: an algorithm for optimization (GBMO), Appl. Soft Comput. 13 (2013) 2932–2946.

[25] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm—a novel metaheuristic optimization method for solving constrained engineering optimization problems, Comput. Struct. 110 (2012) 151–166.

[26] W.T. Pan, A new fruit fly optimization algorithm: taking the financial distress model, Knowl. Based Syst. 26 (2012) 69–74.

[27] L. Wang, Y. Xiong, S. Li, Y.R. Zeng, New fruit fly optimization algorithm with joint search strategies for function optimization problems, Knowl. Based Syst. 176 (2019) 77–96.

[28] Y. Zhang, Backtracking search algorithm with specular reflection learning for global optimization, Knowl. Based Syst. 212 (2021) 106546.

[29] Z. Wang, Y.R. Zeng, S. Wang, L. Wang, Optimizing echo state network with backtracking search optimization algorithm for time series forecasting, Eng. Appl. Artif. Intell. 81 (2019) 117–132.

[30] G.G. Wang, X. Zhao, S. Deb, A novel monarch butterfly optimization with greedy strategy and self-adaptive, in: Second International Conference on Soft Computing and Machine Intelligence, ISCMI, 2015, pp. 45–50.

[31] B. Morales-Castañeda, D. Zaldívar, E. Cuevas, O. Maciel-Castillo, I. Aranguren, F. Fausto, An improved simulated annealing algorithm based on ancient metallurgy techniques, Appl. Soft Comput. 84 (2019) 105761.

[32] F.A. Hashim, E.H. Houssein, M.S. Mabrouk, W. Al-Atabany, S. Mirjalili, Henry gas solubility optimization: A novel physics-based algorithm, Future Gener. Comput. Syst. 101 (2019) 646–667.

[33] O. Maciel-Castillo, E. Cuevas, M.A. Navarro, D. Zaldívar, S. Hinojosa, Side-blotched lizard algorithm: A polymorphic population approach, Appl. Soft Comput. 88 (2020) 106039.

[34] V. Hayyolalam, A.A. Pourhaji Kazem, Black widow optimization algorithm: A novel meta-heuristic approach for solving engineering optimization problems, Eng. Appl. Artif. Intell. 87 (2020) 103249.

[35] S. Memar, A. Mahdavi-Meymand, W. Sulisz, Prediction of seasonal maximum wave height for unevenly spaced time series by black widow optimization algorithm, Mar. Struct. 78 (10) (2021) 103005.

[36] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, Knowl.-Based Syst. 191 (2020) 105190.

[37] L. Abualigah, M.A. Elaziz, P. Sumari, Z.W. Geem, A.H. Gandomi, Reptile search algorithm (RSA): A nature-inspired meta-heuristic optimizer, Expert Syst. Appl. (2021).

[38] X.S. Yang, S. Deb, Y.X. Zhao, S. Fong, X. He, Swarm intelligence: past, present and future, Soft Comput. 22 (2018) 5923–5933.

[39] P. Thagard, E. Toombs, Handbook of Categorization in Cognitive Science, second ed., Elsevier, 2005.

[40] S.S. Zumdahl, S.A. Zumdahl, Chemistry, seventh ed., Houghton Mifflin Harcourt, 2007.

[41] E. Cantore, Atomic Order: An Introduction to the Philosophy of Microphysics, MIT Press, Cambridge, MA, 1969.

[42] C.E. Mortimer, Chemistry, fifth ed., Wadsworth Pub. Co, 1983.

[43] D. Whitley, A genetic algorithm tutorial, Stat. Comput. 4 (2) (1994) 65–85.

[44] R. Ugolotti, Y.S.G. Nashed, P. Mesejo, S. Ivekovi˘c, L. Mussi, S. Cagnoni, Particle swarm optimization and differential evolution for model-based object detection, Appl. Soft Comput. 13 (2013) 3092–3105.

[45] B. Morales-Castañeda, D. Zaldívar, E. Cuevas, F. Fausto, A. Rodríguez, A better balance in meta-heuristic algorithms: Does it exist? Swarm Evol. Comput. 54 (2020) 100671.

[46] Z.M. Gao, J. Zhao, Y.R. Hu, H.F. Chen, The challenge for the nature-inspired global optimization algorithms: Non-symmetric benchmark functions, IEEE Access 9 (2021) 106317-106339.

[47] N. Awad, M. Ali, J. Liang, B. Qu, P. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization, Technical Report, NTU, Singapore, 2016.

[48] Z. Lei, S. Gao, S. Gupta, J. Cheng, G. Yang, An aggregative learning gravitational search algorithm with self-adaptive gravitational constants, Expert Syst. Appl. 152 (2020) 113396.

[49] Y. Yu, S. Gao, S. Cheng, Y. Wang, S. Song, F. Yuan, CBSO: a memetic brain storm optimization with chaotic local search, Memet. Comput. 10 (2018) 353–367.