Arabian Journal for Science and Engineering (2021) 46:3889–3911
https://doi.org/10.1007/s13369-020-05228-5

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

A New Fusion of ASO with SA Algorithm and Its Applications to MLP Training and DC Motor Speed Control

Erdal Eker[1] · Murat Kayri[2] · Serdar Ekinci[3] · Davut Izci[4]

Received: 13 August 2020 / Accepted: 7 December 2020 / Published online: 2 February 2021

Abstract
An improved version of atom search optimization (ASO) algorithm is proposed in this paper. The search capability of ASO was improved by using simulated annealing (SA) algorithm as an embedded part of it. The proposed hybrid algorithm was named as hASO-SA and used for optimizing nonlinear and linearized problems such as training multilayer perceptron (MLP) and proportional-integral-derivative controller design for DC motor speed regulation as well as testing benchmark functions of unimodal, multimodal, hybrid and composition types. The obtained results on classical and CEC2014 benchmark functions were compared with other metaheuristic algorithms, including two other SA-based hybrid versions, which showed the greater capability of the proposed approach. In addition, nonparametric statistical test was performed for further verification of the superior performance of hASO-SA. In terms of MLP training, several datasets were used and the obtained results were compared with respective competitive algorithms. The results clearly indicated the performance of the proposed algorithm to be better. For the case of controller design, the performance evaluation was performed by comparing it with the recent studies adopting the same controller parameters and limits as well as objective function. The transient, frequency and robustness analysis demonstrated the superior ability of the proposed approach. In brief, the comparative analyses indicated the proposed algorithm to be successful for optimization problems with different nature.

Keywords Atom search optimization · Simulated annealing · Multilayer perceptron · DC motor speed control

1 Introduction

Optimization can be described as the process of achieving optimal parameters of a given system with a lower cost. Development of optimization algorithms has gained an incredible attention since the optimization problems can be encountered in a variety of fields such as engineering, science, economics and business [1–3]. A real-world optimization problem may be solved if it can be formulated in terms of mathematical form. Various deterministic algorithms are available to solve such problems. However, a considerable amount of those problems has specific characteristics such as non-continuous and non-differentiable nature, too many decision variables and objective functions and thus cannot be solved effectively using conventional mathematical programming approaches [4]. Therefore, alternative methods are required in the case of such problems instead of conventional techniques.

Metaheuristic algorithms have gained an incredible attention among alternative techniques due to their flexible and simple structure along with the ability of random search and

✉ Serdar Ekinci
serdar.ekinci@batman.edu.tr

Erdal Eker
e.eker@alparslan.edu.tr

Murat Kayri
muratkayri@yyu.edu.tr

Davut Izci
davut.izci@batman.edu.tr

[1] Department of Marketing and Advertising, Muş Alparslan University, Muş, Turkey

[2] Department of Computer and Instructional Technology, Yüzüncü Yıl University, Van, Turkey

[3] Department of Computer Engineering, Batman University, Batman, Turkey

[4] Vocational School of Technical Sciences, Batman University, Batman, Turkey

avoidance of local optima. Therefore, metaheuristic algorithms have been studied extensively as an alternative and effective way of tackling such problems [5–8] since they are powerful tools to handle previously mentioned problems. Those types of problems are inspired from real-world physical phenomena or biological behavior of species and can be categorized into three classes such as physics based, evolution based and swarm based [9]. In metaheuristic algorithms, the problem is considered as a black box (although those algorithms are derived from nature) and the algorithm attempts to solve the problem without concerning the nature of the problem. Therefore, they can easily be implemented to real-world problems. It is worth to note that some real-world optimization problems are subjected to constraints of inequality and/or equality and thus known as constraint optimization problems and require constraint handling techniques. There are different available constraint handling strategies, such as famous penalty and multi-objective approaches, that can be found in the literature. The readers are referred to Refs. [10] and [11] for more details.

Some of the examples of metaheuristic algorithms can be listed as squirrel search [12], the ant lion optimizer [13], artificial ecosystem-based optimization [14], slime mould [15], Henry gas solubility optimization [16], Harris hawks optimization [4], Manta ray foraging optimization [9], butterfly optimization [17], symbiotic organisms search [18], artificial bee colony [19], emperor penguins colony [20], sine cosine [21] and kidney inspired [22] algorithms. The reason of having a variety of metaheuristic algorithms derives from No Free Lunch theorem [23]. According to this theorem, there is not any optimization algorithm capable of finding the optimal solution for every optimization problem. Therefore, no algorithm can solve all optimization problems, with different type and nature, effectively than any other option. Instead, each of them can be quite successful for specific set of problems. Atom search optimization (ASO) [24] algorithm is one of those algorithms that have been developed for tackling specific optimization problems as stated by No Free Lunch theorem.

ASO is a recently developed population-based metaheuristic algorithm that was inspired from molecular dynamics [24] and proposed for dealing a variety of optimization problems [25]. It considers the potential function along with the interaction force and geometric constraint. ASO has already been successfully implemented to solve a variety of problems such as hydrogeologic parameter estimation [24], feature selection [26–28], centralized thermoelectric generation system in heterogeneous temperature difference [29], dispersion coefficient estimation in groundwater [25], peak sidelobe level reduction [30], automatic voltage regulator [31], optimal power flow [32], modular multilevel converters [33] and modeling fuel cells [34] along with line loss and cost minimization of shunt capacitors [35].

Despite the popularity of metaheuristic algorithms, there are drawbacks such as local minima stagnation and immature convergence. Those are two critical problems in metaheuristics which are caused by their randomized exploration and exploitation operators and thus need to be addressed. Several strategies have been proposed to overcome the respective weaknesses of metaheuristic algorithms. Hybridization is an outstanding method among the proposed approaches since it provides more effective results via synthesizing the best aspects of the algorithms which helps exhibiting a more robust behavior and greater flexibility against difficult problems [36]. Therefore, it has found a place as a demanding trend [37]. Similar to many other metaheuristics, original ASO algorithm also suffers from premature convergence and local optima stagnation [38] and thus requires improvement in order to balance the exploration and exploitation. Developing an improved version of ASO is feasible although the successful implementation of it to the problems listed in the previous paragraph is a good indication of its ability. Further improvement in terms of its capability can be obtained by achieving a balance between exploration and exploitation stages. The latter would help the algorithm to perform better compared to its implementation with the original version. To do so, simulated annealing (SA) algorithm [39], a well-known algorithm that has good local search capabilities, can be used for hybridization. The latter is one of the algorithms that was recently hybridized with other metaheuristic algorithms to solve different types of optimization problems [40–49].

SA [39] is a stochastic and single solution-based algorithm that simulates the metallurgical process of annealing in which high temperature molecules with high energy levels move against other molecules relatively easy and temperature is decreased slowly to reach a steady state condition with minimum energy level. Similar to ASO, several applications of SA algorithm are also available in the literature. Some those applications can be listed as solution of clustering problem [50], multiple non-consecutive processing of parts on a machine [51], placement of virtual machine for optimum power consumption in data centers [52], optimal estimation of solar cell model parameters [53], optimization of pressure-swing distillation process [54], minimization of the fuel cost and the gas emissions [55], solution for a green vehicle routing problem with fuel consumption [56] and optimal design of supercritical carbon dioxide compressor [57] along with several structural optimization problems [58–61]. SA is an easy to implement metaheuristic algorithm that is strong in terms of local search and requires less computation time [62]. Therefore, SA algorithm can be interconnected in such a way that a new hybrid model can be constructed. In this way, the solution quality of ASO algorithm can be improved.

A crucial idea in the SA, which makes it to be considered as a hill-climbing technique, is the acceptance of lower quality solutions to escape from local optima using a probability function. Also, SA has only one control parameter to set, which is called temperature and usually reduced monotonically over the search. Therefore, SA algorithm requires minimum number of evaluations for finding optimal solutions due to having a simple structure with easy implementation and a strong local search ability [63]. This is also the motivation of this paper which adopts SA to overcome the previously mentioned drawbacks of ASO algorithm and thus obtain a better structure to solve the optimization problems of various types.

In light of the fact mentioned above, this paper proposes a novel hybrid ASO and SA (hASO-SA) algorithm by considering the lack of balance between exploration and exploitation stages of ASO and incredible local search ability of SA algorithm. The developed hybrid algorithm adopts SA algorithm as an embedded part of the ASO algorithm instead of running both algorithms one by one. That helps SA to operate for worse solutions so that the potential of neighborhood solutions is not neglected. As stated above, the aim of this paper is to achieve an improved version of ASO so that it can be implemented to various optimization problems with greater capability. To do so, eight well-known classical and four CEC2014 benchmark functions of unimodal, multimodal, hybrid and composition types, five classification data sets for nonlinear multilayer perceptron (MLP) training system, and proportional-integral-derivative (PID) controller design for linearized DC motor speed control system were used as different optimization problems for performance evaluation of the proposed hybrid algorithm.

In terms of test functions, the performance evaluations were carried out using classical benchmark functions of Sphere, Rosenbrock, Step, Quartic, Schwefel, Rastrigin, Ackley and Griewank [64] as well as CEC2014 functions [65]. The obtained results were compared with six stochastic algorithms such as particle swarm optimization (PSO), gravitational search (GSA), wind driven optimization (WDO) and genetic algorithm (GA) along with SA and original ASO algorithms. Moreover, hybrid versions of PSO and cuckoo search (CS) algorithms, merged with SA, were also used for performance comparison. The proposed hASO-SA algorithm was run with the same swarm size and the maximum number of iterations for a fair comparison with the stated algorithms. The statistical results obtained for the adopted test functions showed that the best results were achieved via the proposed algorithm. Further performance validation was also carried out using Wilcoxon signed-rank test [66] has been used to prove that the capability of the proposed hybrid hASO-SA algorithm was not by chance.

Likewise, datasets of Iris, Balloon, XOR, Breast cancer and Heart [67] were adopted for MLP training to observe the performance of the proposed algorithm for nonlinear optimization problems. The obtained results for the latter case were compared with the MLP training results that were achieved by using grey wolf optimization (GWO), ant colony optimization (ACO), probability-based incremental learning (PBIL), particle swarm optimization (PSO) and evolutionary strategies (ES) algorithms along with original ASO algorithm. All algorithms were run under similar conditions for a fair comparison and the lower average and standard deviation of mean square error were achieved via the proposed approach which is an indication of better performance.

Similar to benchmark function and MLP training cases, PID controller design for DC motor was also performed by comparing the obtained results with grey wolf-based PID (GWO/PID), sine cosine-based PID (SCA/PID) and atom search optimization-based PID (ASO/PID) controllers. The reason of using the latter algorithms was because of similar set of parameters for both the controller and motor, in addition to the same objective function, so that a fair comparison can be performed. Transient and frequency responses showed the proposed method helps in achieving a better performing system along with a better robustness. The DC motor system also proved the ability of the proposed algorithm to be considerably successful than its counterparts for real-world engineering problems. In summary, the comparisons for all adopted systems have demonstrated that the proposed hybrid hASO-SA algorithm has better performance for a variety of problems having different nature.

1.1 Previous Works on MLP Training

Artificial neural networks mimic human brain via computational models and broadly used for complex nonlinear problems [68]. The MLP is part of the hidden layered feed forward neural networks [69]. It is also an extensively adopted neural network and requires training on particular application [70]. Deterministic approaches can be found in the literature in terms of algorithms used for neural network training [71]; however, slow convergence and local optima stagnation are the issues that the training process suffers from. Therefore, training such structure requires a better algorithm in order to overcome the latter issues. To do so, several metaheuristic algorithms have been proposed so far. Some of those algorithms can be listed as grey wolf [72, 73] and improved grey wolf optimization [74], ant lion optimization [75], chimp optimization [76], grasshopper optimization [77], salp swarm [78] and multiple leader salp swarm [79], hybrid Nelder-Mead and dragonfly [80], hybrid particle swarm optimization and gravitational search [81], magnetic optimization [82] and biogeography-based optimization [83] along with hybrid monarch butterfly and artificial bee colony optimization [84] algorithms. Novel algorithms that can provide further improvement for the MLP training is

feasible despite the presented promise of the methods listed above. The latter may also be achieved through improvement of existing algorithms instead of development of new ones from the scratch. In light of the above fact, this study attempts to achieve such a novel algorithm that can perform better compared to other available approaches. Therefore, the hybridization of ASO algorithm with SA technique is proposed in this study to deal with the training MLP which can help to achieve a better algorithm for such a purpose.

1.2 Previous Works on Controller Design for DC Motor

The use of DC motors can be found in almost all of the industrial applications [85] due to their lower price and maintenance cost along with easier control. Robotics, paper mills, machine tools and textile industry are a few to name the industrial applications of DC motors. Several examples of controllers such as PID, FOPID, fuzzy logic or neural networks can be found in the literature [86]. Since DC motors provide an observable test bed for performance evaluations and comparisons, their speed control has been an application area for many metaheuristics algorithms as a real-world engineering application. There are various examples in terms of metaheuristic optimization algorithms for controlling DC motors. Some of those examples can be listed as stochastic fractal search [87], kidney-inspired [88], teaching–learning-based optimization [89], particle swarm optimization [90], swarm learning process [91], ant colony optimization [92], Harris hawks optimization [86], sine cosine [93], grey wolf optimization [94], chaotic atom search optimization [95], flower pollination [96] and improved sine cosine [97] along with genetic [98] and improved genetic [99] algorithms. As part of this study, we have implemented the new proposed approach for similar purpose as well as to assess the performance quality of the proposed hybrid algorithm for such a real-world engineering problem. Similar to the motivation explained in the previous section, this study aims to develop a novel approach that can achieve a more stable structure, compared to existing techniques, for the stated system in terms of transient and frequency responses as well as robustness.

2 Overview of ASO, SA and Proposed hASO-SA Algorithms

2.1 ASO Algorithm

ASO is a population-based global optimization technique inspired by molecular dynamics [25]. Basically, it is a mathematical representation of atomic motion which behaves according to classical mechanics [100]. According to Newton's second law, relationship of an atomic system can be written as in Eq. (1):

$$a_i = \frac{F_i + G_i}{m_i} \tag{1}$$

where F_i and G_i represent interaction and constraint forces that act on ith atom together. The acceleration and the mass of atom i is denoted by a_i and m_i, respectively. In dimension d and at time t, the interaction force that acts on ith atom due to jth atom can be expressed as in Eq. (2). The latter one is a revised version of The Lennard–Jones (L–J) potential [101] to prevent the atoms from the case where they cannot converge to a specific point.

$$F'_{ij}(t) = -\eta(t)\left[2\left(h_{ij}(t)\right)^{13} - \left(h_{ij}(t)\right)^{7}\right] \tag{2}$$

$\eta(t)$ is called the depth function and is defined as in Eq. (3) where α represents the depth weight and T denotes the maximum number of iterations. This function is used for arrangement of the repulsion or attraction regions.

$$\eta(t) = \alpha\left(1 - \frac{t-1}{T}\right)^{3} e^{-\frac{20t}{T}} \tag{3}$$

$h_{ij}(t)$ is expressed as given in Eq. (4) where r is the distance between two atoms, $h_{\min}$ is the lower bound, and $h_{\max}$ is the upper bound. The latter function helps repulsion, attraction or equilibrium to occur.

$$h_{ij}(t) = \begin{cases} h_{\min}, & \frac{r_{ij}(t)}{\sigma(t)} < h_{\min} \\ \frac{r_{ij}(t)}{\sigma(t)}, & h_{\min} \le \frac{r_{ij}(t)}{\sigma(t)} \le h_{\max} \\ h_{\max}, & \frac{r_{ij}(t)}{\sigma(t)} > h_{\max} \end{cases} \tag{4}$$

The exploration is improved by having lower limit of repulsion ($h = 1.1$) and upper limit of attraction ($h = 1.24$). To represent the limits as explained previously, the terms of g_0 and u, provided in Eq. (5), are equal to 1.1 and 1.24, respectively.

$$\begin{aligned} h_{\min} &= g_0 + g(t) \\ h_{\max} &= u \end{aligned} \tag{5}$$

Drift factor is expressed by g which is used to allow the algorithm to drift from exploration to exploitation and given as in Eq. (6).

$$g(t) = 0.1 \times \sin\left(\frac{\pi}{2} \times \frac{t}{T}\right) \tag{6}$$

$\sigma(t)$, given in Eq. (4), denotes the length scale, which represents the collision diameter, and is defined as follows

$$\sigma(t) = \left\| x_{ij}(t), \frac{\sum_{j \in K\text{best}} x_{ij}(t)}{K(t)} \right\|_2 \tag{7}$$

where Kbest denotes an atom population that includes the best function values of first K atoms. The function F' behavior with respect to values of $\eta(t)$ (corresponding to h values) is illustrated in Fig. 1.

The sum of components having random weights in dth dimension that act on ith atom (due to other atoms) can be expressed as total force and is given as in Eq. (8) where rand_j represents a random number in [0, 1].

$$F_i^d(t) = \sum_{j \in K\text{best}} \text{rand}_j F_{ij}^d(t) \tag{8}$$

In molecular dynamics, atomic motion is greatly affected from the geometric constraint. In ASO, this is simplified by supposing a covalent bond between each atom and the best atom. Therefore, the constraint of atom i can be written as

$$\theta_i(t) = \left[\left| x_i(t) - x_{\text{best}}(t) \right|^2 - (b_{i,\text{best}})^2 \right] \tag{9}$$

where $x_{\text{best}}(t)$ represents the best atom position at iteration t, whereas $b_{i,\text{best}}$ denotes the fixed bond length between the best and the ith atoms. Thus, the constraint force can be acquired as

$$G_i^d(t) = \lambda(t)\left(x_{\text{best}}^d(t) - x_i^d(t)\right) \tag{10}$$

where $\lambda(t)$ is the Lagrangian multiplier and defined as in Eq. (11).

$$\lambda(t) = \beta e^{-\frac{20t}{T}} \tag{11}$$

In the latter, β is the multiplier weight. The acceleration of atom i at time t can be written as in Eq. (12) where $m_i(t)$ is the mass of atom i at time t.

$$a_i^d(t) = \frac{F_i^d(t)}{m_i^d(t)} + \frac{G_i^d(t)}{m_i^d(t)} \tag{12}$$

As the latter equation express, an atom with a bigger mass provides a better function fitness value, thus causing a reduced acceleration. The mass of atom i can be computed as in Eq. (13).

$$m_i(t) = \frac{M_i(t)}{\sum_{j=1}^{N} M_j(t)} \tag{13}$$

$$M_i(t) = e^{-\frac{\text{Fit}_i(t) - \text{Fit}_{\text{best}}(t)}{\text{Fit}_{\text{worst}}(t) - \text{Fit}_{\text{best}}(t)}} \tag{14}$$

$\text{Fit}_{\text{best}}(t)$ represents the atom with minimum fitness value and $\text{Fit}_{\text{worst}}(t)$ denotes the maximum fitness value at iteration t. The latter fitness values are expressed as given in Eqs. (15) and (16), respectively. $\text{Fit}_i(t)$ is a representation of function fitness value of atom i at iteration t.

$$\text{Fit}_{\text{best}}(t) = \min_{i \epsilon (1,2,\ldots N)} \text{Fit}_i(t) \tag{15}$$

$$\text{Fit}_{\text{worst}}(t) = \max_{i \epsilon (1,2,\ldots N)} \text{Fit}_i(t) \tag{16}$$

The velocity and the position of atom i at iteration $(t + 1)$ can be expressed as follows in order to simplify the algorithm.

$$v_i^d(t+1) = \text{rand}_i^d v_i^d(t) + a_i^d(t) \tag{17}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{18}$$

Each atom requires to have higher possible interactions with many atoms having better fitness values as its K neighbors in order to improve the exploration at the beginning of iterations. On the contrary, each atom requires to have fewer

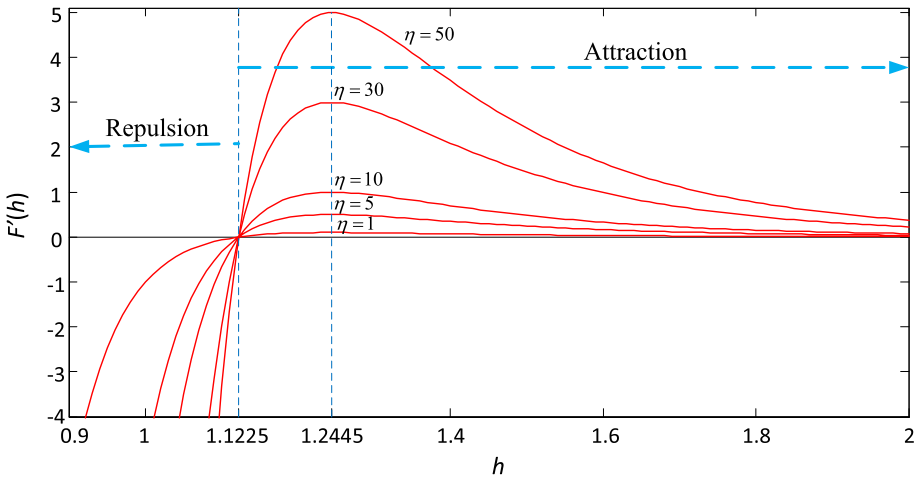

Fig. 1 Corresponding F' function behavior with different η values

possible interactions with atoms having better fitness values as its K neighbors to improve the exploitation. Here, K represents a time-dependent function and is calculated as in Eq. (19) in order to show the gradual decrease with respect to number of iterations.

$$K(t) = N - (N-2) \times \sqrt{\frac{t}{T}} \tag{19}$$

2.2 SA Algorithm

This algorithm mimics the annealing process in metallurgy and classified as a single-based solution method [39]. Basically, the process is performed by heating and cooling stages which consequently helps in generating uniform crystals with less defects. The SA starts with an initial value set for random solution of X_i and determines a neighborhood solution of X_i'. Then, it computes the fitness value for X_i and X_i'. If the fitness value of $X_i^{'}$ ($F(X_i')$) is smaller than that of X_i ($F(X_i)$), then SA sets $X_i = X_i'$. Meanwhile, SA may replace the solution of X_i by solution of X_i' even if the fitness values do not have the latter relationship. The replacement for such a case depends on the probability p as defined in Eq. (20):

$$p = e^{-\frac{\Delta F}{T_k}};\ \Delta F = F\left(X_i'\right) - F\left(X_i\right) \tag{20}$$

where F and T denote control parameters of fitness function and temperature, respectively. The algorithm will not replace X_i by X_i' if $p <$ rand(0, 1); however, a replacement will happen on the contrary case. The SA algorithm later reduces the value of the temperature using the following equation where μ denotes the cooling coefficient, which is a random constant between 0 and 1.

$$T_{k+1} = \mu T_k \tag{21}$$

2.3 Proposed Hybrid hASO-SA Algorithm

The proposed hASO-SA algorithm is a hybrid version of ASO and SA algorithms. The SA is local search metaheuristic algorithm, and it is widely used to solve continuous and discrete optimization problems [39]. The ability of escaping local minimum via hill-climbing moves is one of the main benefits of SA which is useful in terms of searching for a global solution. Therefore, a hybrid approach is proposed with this work by introducing SA is to assist the ASO in terms of avoiding local minimum. Also, it helps increasing the level of diversity while searching for optimum solution in the search space. The novel hybrid hASO-SA algorithm exploits the fast-optimal search capability and hill-climbing property of ASO and SA algorithms, respectively, and proposed to solve various optimization problems.

A flowchart of the proposed hASO-SA is illustrated in Fig. 2. As can be seen from the flowchart, the proposed hybrid algorithm starts with defining the parameters of ASO and SA algorithms first along with initializing a random set of atoms with their velocities and a fitness value set to infinity. Once this achieved, the iterations begin by calculating the fitness value for each atom and then the obtained fitness value is compared with the best fitness value. In the case of better values, the algorithm updates the best solution and fitness value and the rest of the steps in the flow chart are executed. However, the proposed hybrid algorithm gives a chance the current solution even if the current fitness value is not better than the best one. In such a case, the algorithm generates a new solution in a neighborhood of current solution and evaluates the newly generated solution based on the justification of probability. That means the SA behaves as an embedded part of the ASO and operates to justify the neighborhood solution in the case of current solution without better fitness values.

Based on the justification, the best solution may or may not be updated by the algorithm. In such a scenario, SA is nicely operating as part of the ASO only for fitness values that are worse than the best one so that any potential neighborhood is not passed directly by just looking at the fitness value. It is also worth to note that the hybrid algorithms have a disadvantage of requiring more computational time despite their better performing ability. However, in the proposed algorithm, the fundamental steps of SA technique have been embedded into ASO algorithm which consequently reduced the computational time considerably than expected.

3 Experimental Setup and Results

3.1 Classical and CEC2014 Benchmark Functions

Eight well-known classical and four CEC2014 benchmark functions were employed to achieve an extensive performance evaluation of the proposed hASO-SA algorithm. Those benchmark functions can be assessed under four main types such as unimodal and multimodal, hybrid and composition functions. Therefore, the performance of various optimization algorithms can effectively be measured using them. A summary of employed benchmark functions is provided in Table 1.

The functions from of $F_1(x) - F_4(x)$ are unimodal functions. They have one global optimum and no local optimum. On the other hand, the functions from $F_5(x)$ to $F_8(x)$ are multimodal functions. These ones have considerable number of local optima. In addition to the above, hybrid functions of $F_9(x)$ and $F_{10}(x)$ were also adopted. The variables of the latter functions are separated into different subdivisions randomly and either unimodal or multimodal functions are

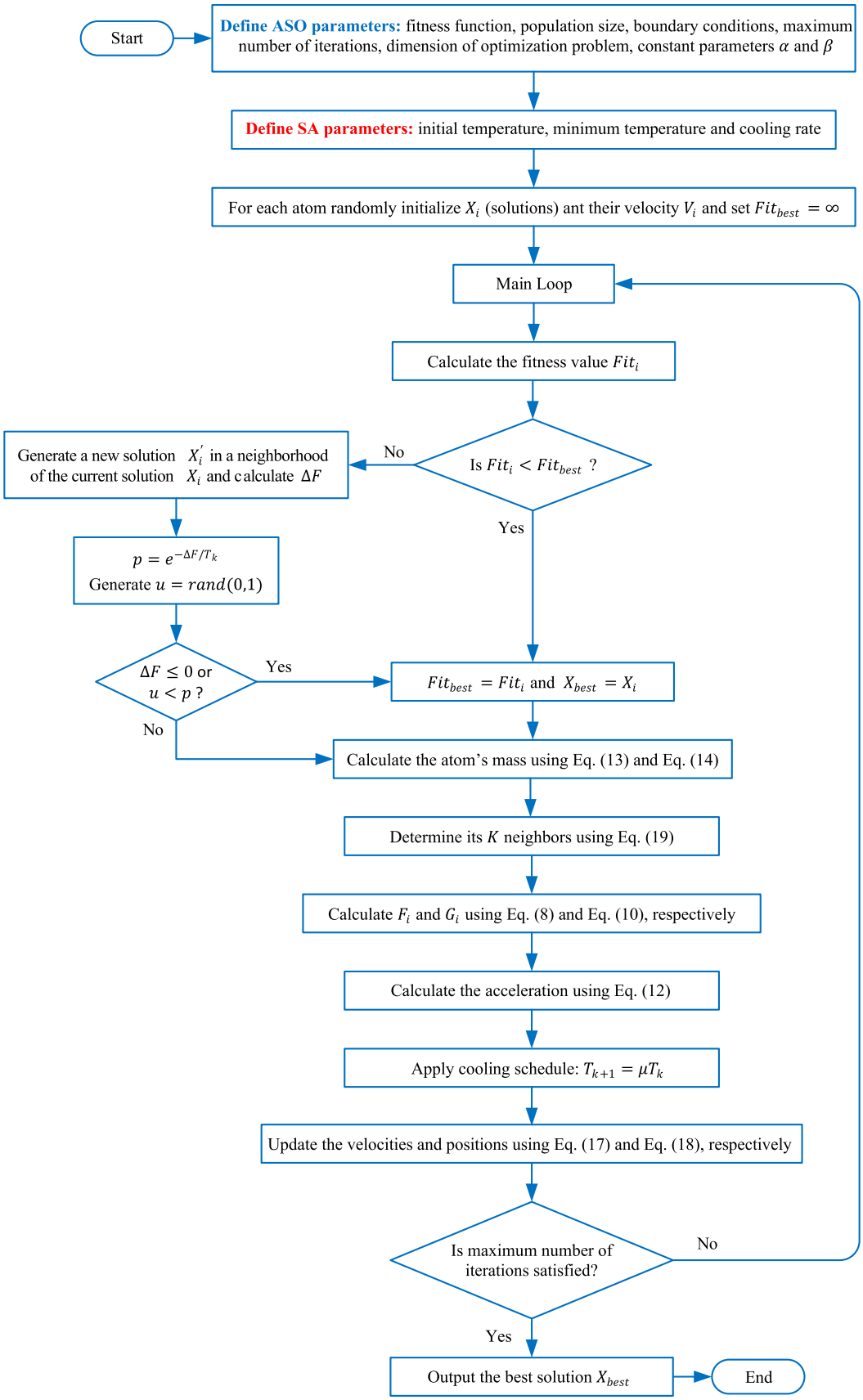

Fig. 2 Flowchart for the hASO-SA algorithm

Table 1 Details of used classical and CEC2014 benchmark functions

Name	Test function	D	Range	Optimum
Sphere	$F_1(x) = \sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0
Rosenbrock	$F_2(x) = \sum_{i=1}^{D-1} \left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right)$	30	$[-30, 30]^D$	0
Step	$F_3(x) = \sum_{i=1}^{D} \left(x_i + 0.5\right)^2$	30	$[-100, 100]^D$	0
Quartic	$F_4(x) = \sum_{i=1}^{D} ix_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]^D$	0
Schwefel	$F_5(x) = -\sum_{i=1}^{D} \left(x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)\right)$	30	$[-500, 500]^D$	$-12,569.5$
Rastrigin	$F_6(x) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right]$	30	$[-5.12, 5.12]^D$	0
Ackley	$F_7(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi x_i\right)\right) + 20 + e$	30	$[-32, 32]^D$	0
Griewank	$F_8(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]^D$	0
Hybrid function 3 ($N = 4$)	$F_9(x)$	30	$[-100, 100]^D$	1900
Hybrid function 6 ($N = 5$)	$F_{10}(x)$	30	$[-100, 100]^D$	2200
Composition function 4 ($N = 5$)	$F_{11}(x)$	30	$[-100, 100]^D$	2600
Composition function 5 ($N = 5$)	$F_{12}(x)$	30	$[-100, 100]^D$	2700

used to replace those subdivisions. Moreover, composition functions are represented by $F_{11}(x)$ and $F_{12}(x)$ in the table. Similar to the case in the hybrid functions, the variables of those composition functions are also randomly separated into different subdivisions; however, those subdivisions are constructed by using the basic and hybrid functions. Hybrid and composition benchmark functions are more complex and challenging than the basic unimodal and multimodal benchmark functions, thus presenting more challenging optimization problems. The latter two types of benchmark functions are specifically suitable for testing the potential performance of the algorithms for solving real-world problems. References [38, 64, 65] provide a detailed description of all employed functions.

3.2 Compared Algorithms

The comparisons were carried out using eight stochastic algorithms by utilizing aforementioned test functions. The algorithms used for comparison include three popular algorithms such as PSO, GA and SA, along with three of recently proposed algorithms such as GSA, WDO and original ASO. In addition, further comparative performance evaluations were carried out using two additional hybrid algorithms which were developed based on SA by using particle swarm optimization (hPSO-SA) and cuckoo search (hCS-SA) algorithms.

SA is inspired from the certain rate of heating and cooling used in metallurgy for heating materials [39]. This is a probabilistic approach that seeks for the global optimum of a search space in a fixed amount of time. This algorithm has parameters of initial temperature, temperature reduction rate and mutation rate and those parameters were set to have values of 0.1, 0.98 and 0.5, respectively [24].

GA is an evolutionary algorithm that is inspired from biological evolutionary theory [102]. This algorithm tends to exceptional in terms of finding good global solutions. It adopts selection, crossover and mutation operations to achieve high quality solutions. The latter parameters for this study were chosen to be Roulette wheel for selection, 0.8 for crossover and 0.4 for mutation [24].

PSO is an algorithm that mimics the flocking behavior of birds in the sky [103]. It adopts individual and social group learning to update the velocities and positions of a population. In this way, it searches for the desired goal. This algorithm has a good ability in terms of local search. In this study, the PSO parameters of cognitive and social constants were chosen to be 2 for each. The inertia constant was set to decrease linearly from 0.8 to 0.2 [24].

GSA is another algorithm used for comparison which is a competitive search algorithm [64]. This algorithm is based on the gravitational law and thus makes the agents to interact using the motional law. The attraction can lead to generation of attractive force and this helps in facilitating all agents to move toward the agent with the heavier mass. The parameters of this algorithm were set to 100 and 20 for initial gravitational constant and decreasing coefficient, respectively [24].

The last algorithm used for comparison is WDO which is inspired from the motion of earth's atmosphere [104]. In

here, each small parcel of air moves by following the Newton's second law. This algorithm has a good global search ability and approximates the global optimum by updating the velocity and position of each parcel using gradient, Coriolis, gravitational and friction forces. The parameter values for this study were set to 3, 0.2 and 0.4 for RT coefficient, gravitational constant and Coriolis effect, respectively. The maximum allowable speed was set to 0.3, whereas the constant in the update equations was 0.4 [24].

In addition of above algorithms, hybrid structures of PSO and CS algorithms (hPSO-SA and hCS-SA, respectively) were also merged with SA and used for comparison. The PSO algorithm used in hPSO-SA is already mentioned briefly in one of the above paragraphs. CS algorithm in hCS-SA is a population-based optimization algorithm and simulates the parasitic breeding behavior of some cuckoo species [105]. Those species lay their eggs on the nests of host birds. Depending on the discovery of the replacement of the eggs by the host bird, the eggs may be thrown, or the nest may be abandoned. In this study, the parameters of CS algorithm were chosen as $\beta = 1.5$ for Lévy flight and $P_a = 0.25$ for mutation probability [105].

3.3 Statistical Results and Discussion

The proposed hASO-SA algorithm was run 50 times along with a chosen swarm size of 50 and the maximum number of iterations of 1000 in order to achieve a fair comparison with SA, GA, PSO, GSA, WDO and ASO algorithms [24]. The statistical results obtained for the adopted test functions using listed algorithms are presented in Table 2 by highlighting the best mean results in bold.

Considering the presented numerical values in the table, the proposed hybrid hASO-SA algorithm provided the best statistical values compared to other competitive algorithms, including original ASO, for functions of $F_2(x)$, $F_4(x)$, $F_5(x)$, $F_9(x)$, $F_{10}(x)$, $F_{11}(x)$ and $F_{12}(x)$. In addition, it has also found the same values as its other closest competitors in functions of $F_3(x)$, $F_6(x)$, $F_7(x)$ and $F_8(x)$. The proposed hASO-SA algorithm is behind the WDO algorithm only for the function of $F_1(x)$. The results in Table 2 demonstrate the good optimizing performance of the proposed hASO-SA compared to its competitive algorithms (including the basic ASO) on the benchmark functions, including unimodal, multimodal, hybrid and composition functions.

3.4 Nonparametric Test Analysis

The superiority of an algorithm may generally occur by chance, due to stochastic nature, if the comparison is performed based on statistical criteria such as best, mean and standard deviation. Because of 50 independent runs, the probability of the latter case is low for this study. However, a nonparametric statistical test was also performed to compare the results of each run and decide on the significance of the results. In this work, a nonparametric test named Wilcoxon signed-rank test [66] has been used to prove the superiority of the proposed hybrid hASO-SA algorithm. This test is performed at 5% significant level for the hASO-SA versus other competitive algorithms, and the obtained p values are provided in Tables 3 and 4. The p values less than 0.05 indicate significant difference between the algorithms. The column W (winner) in Table 3 and 4 reveals the results of the test where the sign of '=' is an indication of no significant difference between hASO-SA and the competitive algorithm, whereas the signs of '+' and '−' are of significantly better and worse performances of hASO-SA, respectively, compared to its competitive algorithms.

The results of Tables 3 and 4 showed that the hASO-SA algorithm presents better performance with great effectiveness with respect to other algorithms. Moreover, the corresponding statistical results for each function in 50 runs are listed in Table 5. This table shows the proposed hASO-SA algorithm outperforming all other algorithms significantly for unimodal, multimodal, hybrid and composition benchmark functions.

4 Application of hASO-SA in Training MLP

4.1 MLP

MLP can be regarded as a distinctive class of feedforward neural networks. Neurons are organized in one-direction in MLPs. MLPs has a layered structure where data transition occurs. The structure of MLPs can be imagined as parallel layers which are named as input layer, hidden layer and output layer. Figure 3 illustrates an MLP with those three layers where n denotes the number of input nodes, h is hidden layer, and m shows output nodes. The MLP output is calculated in few steps. Firstly, the weighted sums are calculated using Eq. (22) where W_{ij} denotes the connection weight from input layer's ith node to the hidden layer's jth node, X_i represents the ith input, and θ_j is the bias of the jth hidden node.

$$s_j = \sum_{i=1}^{n} \left(W_{ij} X_i\right) - \theta_j, \quad j = 1, 2, \ldots, h \tag{22}$$

Secondly, each hidden node's output is calculated as in Eq. (23).

$$S_j = \text{sigmoid}\left(s_j\right) = \frac{1}{1 + e^{-s_j}}, \quad j = 1, 2, \ldots, h \tag{23}$$

After calculating the outputs of hidden nodes, the final outputs are defined as in Eqs. (24) and (25) where ω_{jk}

Table 2 Comparisons of results for unimodal, multimodal, hybrid and composition functions

Function	Metric	SA	GA	PSO	GSA	WDO	ASO	hPSO-SA	hCS-SA	hASO-SA (proposed)
$F_1(x)$	Best	7.76E−14	2.98E−03	9.56E−06	1.06E−17	0.00E+00	3.52E−22	1.23E−15	1.12E−17	8.36E−99
	Mean	2.04E−13	1.02E−02	1.46E−04	2.11E−17	**0.00E+00**	2.68E−21	3.64E−14	2.19E−17	5.57E−96
	SD	6.14E−14	5.07E−03	1.19E−04	6.67E−18	0.00E+00	3.65E−21	6.07E−14	6.38E−18	3.02E−95
$F_2(x)$	Best	2.32E+01	9.70E+00	2.62E+01	2.58E+01	2.80E+01	1.66E+01	1.72E+01	2.58E+01	4.12E−05
	Mean	1.06E+03	9.71E+01	1.34E+02	2.81E+01	2.82E+01	2.48E+01	1.92E+01	2.67E+01	**2.03E−02**
	SD	2.05E+03	1.29E+02	1.29E+02	1.13E+01	1.69E−01	5.16E−01	5.47E−01	2.72E+00	2.84E−02
$F_3(x)$	Best	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
	Mean	5.67E−01	**0.00E+00**	1.33E−01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SD	7.28E−01	0.00E+00	3.46E−01	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
$F_4(x)$	Best	6.25E−02	1.32E−02	2.98E−02	7.01E−03	1.43E−06	3.61E−02	3.84E−06	1.37E−02	2.77E−12
	Mean	1.24E−01	5.05E−02	6.63E−02	2.08E−02	6.10E−05	3.56E−02	6.32E−05	3.09E−02	**1.19E−10**
	SD	3.97E−02	2.18E−02	1.95E−02	7.72E−03	4.49E−05	1.95E−02	6.83E−05	7.94E−03	7.93E−11
$F_5(x)$	Best	−1.01E+04	−8.15E+03	−6.79E+03	−3.49E+03	−7.59E+03	−5.56E+03	−9.03E+03	−1.19E+04	−1.25E+04
	Mean	−9.29E+03	−6.80E+03	−5.20E+03	−2.65E+03	−5.84E+03	−7.43E+03	−8.69E+03	−1.15E+04	**−1.24E+04**
	SD	4.02E+02	6.33E+02	5.29E+02	3.42E+02	8.57E+02	4.22E+02	2.36E+02	3.09E+02	1.53E+02
$F_6(x)$	Best	2.69E+01	6.09E+00	1.73E+01	7.96E+00	1.56E+01	0.00E+00	0.00E+00	1.60E+01	0.00E+00
	Mean	5.44E+01	1.25E+01	2.93E+01	1.51E+01	5.77E+01	**0.00E+00**	**0.00E+00**	3.08E+01	**0.00E+00**
	SD	1.38E+01	2.92E+00	6.88E+00	4.44E+00	2.12E+01	0.00E+00	0.00E+00	8.89E+00	0.00E+00
$F_7(x)$	Best	8.29E−08	1.07E−02	5.49E−04	2.96E−09	8.88E−16	1.13E−11	8.88E−16	1.81E−08	8.88E−16
	Mean	3.44E−01	2.12E−02	7.43E−03	3.69E−09	**8.88E−16**	3.00E−11	**8.88E−16**	5.43E−08	**8.88E−16**
	SD	4.47E−01	4.55E−03	1.42E−02	3.96E−10	0.00E+00	2.15E−11	0.00E+00	2.62E−08	3.68E−31
$F_8(x)$	Best	3.16E−06	6.12E−03	5.88E−05	1.94E+00	0.00E+00	0.00E+00	7.96E−15	0.00E+00	0.00E+00
	Mean	1.24E−02	1.84E−02	2.28E−02	4.47E+00	9.89E−03	**0.00E+00**	2.05E−03	**0.00E+00**	**0.00E+00**
	SD	1.03E−02	9.77E−03	2.74E−02	2.05E+00	2.19E−02	0.00E+00	3.88E−03	0.00E+00	0.00E+00
$F_9(x)$	Best	1.91E+03	1.91E+03	1.91E+03	1.93E+03	1.91E+03	1.91E+03	1.91E+03	1.91E+03	1.90E+03
	Mean	1.91E+03	1.93E+03	1.95E+03	2.05E+03	1.93E+03	1.91E+03	1.91E+03	1.91E+03	**1.90E+03**
	SD	2.04E+00	2.40E+01	3.31E+01	4.03E+01	3.07E+01	7.43E−01	4.92E+00	1.47E+00	2.74E−01
$F_{10}(x)$	Best	2.28E+03	2.35E+03	2.37E+03	2.63E+03	2.43E+03	2.36E+03	2.31E+03	2.35E+03	2.24E+03
	Mean	2.46E+03	2.83E+03	2.83E+03	3.27E+03	2.87E+03	2.76E+03	2.64E+03	2.68E+03	**2.28E+03**
	SD	1.16E+02	2.04E+02	2.66E+02	2.93E+02	1.83E+02	2.04E+02	1.72E+02	1.96E+02	3.72E+01
$F_{11}(x)$	Best	2.70E+03	2.70E+03	2.70E+03	2.75E+03	2.70E+03	2.70E+03	2.70E+03	2.70E+03	2.70E+03
	Mean	2.71E+03	2.79E+03	2.71E+03	2.79E+03	2.77E+03	2.71E+03	2.71E+03	2.72E+03	**2.70E+03**
	SD	2.56E+01	2.53E+01	1.81E+01	1.70E+01	4.17E+00	1.83E+00	8.16E+00	3.59E+01	5.37E−01
$F_{12}(x)$	Best	3.11E+03	3.10E+03	3.15E+03	3.06E+03	3.11E+03	3.10E+03	3.11E+03	3.07E+03	3.02E+03
	Mean	3.21E+03	3.47E+03	3.38E+03	4.37E+03	3.61E+03	3.13E+03	3.18E+03	3.15E+03	**3.06E+03**
	SD	8.08E+01	3.45E+02	2.08E+02	4.11E+02	3.10E+02	3.51E+01	7.25E+01	6.14E+01	2.98E+01

Table 3 Wilcoxon signed-rank test results for hASO-SA versus SA, GA, PSO and GSA

Function	hASO-SA versus SA		hASO-SA versus GA		hASO-SA versus PSO		hASO-SA versus GSA	
	p value	*W*	*p* value	*W*	*p* value	*W*	*p* value	*W*
$F_1(x)$	4.42E−10	+	5.29E−10	+	7.30E−10	+	6.87E−10	+
$F_2(x)$	7.25E−10	+	7.36E−10	+	7.35E−10	+	7.38E−10	+
$F_3(x)$	2.36E−10	+	1	=	2.84E−08	+	1	=
$F_4(x)$	6.87E−10	+	7.02E−10	+	6.98E−10	+	6.92E−10	+
$F_5(x)$	2.89E−10	+	7.80E−10	+	7.49E−10	+	7.41E−10	+
$F_6(x)$	5.84E−10	+	5.67E−10	+	7.73E−10	+	7.00E−10	+
$F_7(x)$	5.30E−10	+	2.28E−10	+	1.24E−10	+	4.40E−10	+
$F_8(x)$	4.59E−10	+	2.01E−10	+	3.98E−10	+	5.95E−10	+
$F_9(x)$	2.29E−05	+	2.88E−07	+	4.31E−09	+	9.44E−08	+
$F_{10}(x)$	3.68E−05	+	7.89E−07	+	1.25E−07	+	3.27E−08	+
$F_{11}(x)$	7.41E−04	+	1.99E−06	+	9.13E−04	+	1.64E−07	+
$F_{12}(x)$	1.42E−08	+	1.42E−08	+	1.42E−08	+	1.42E−08	+

Table 4 Wilcoxon signed-rank test results for hASO-SA versus WDO, ASO, hPSO-SA and hCS-SA

Function	hASO-SA vs WDO		hASO-SA vs ASO		hASO-SA vs hPSO-SA		hASO-SA vs hCS-SA	
	p value	*W*	*p* value	*W*	*p* value	*W*	*p* value	*W*
$F_1(x)$	6.06E−10	−	6.74E−10	+	6.03E−10	+	6.18E−10	+
$F_2(x)$	6.00E−10	+	7.29E−10	+	7.03E−10	+	7.5058 − 10	+
$F_3(x)$	1	=	1	=	1	=	1	=
$F_4(x)$	7.58E−10	+	7.00E−10	+	6.19E−10	+	6.21E−10	+
$F_5(x)$	7.44E−10	+	7.41E−10	+	6.11E−10	+	6.62E−10	+
$F_6(x)$	5.19E−10	+	1	=	1	=	6.15E−10	+
$F_7(x)$	2.54E−10	−	5.42E−10	+	7.07E−10	−	6.22E−10	+
$F_8(x)$	6.56E−09	+	1	=	6.06E−10	+	1	=
$F_9(x)$	3.52E−07	+	8.23E−05	+	3.17E−05	+	3.57E−05	+
$F_{10}(x)$	3.27E−08	+	2.78E−06	+	1.72E−06	+	1.72E−06	+
$F_{11}(x)$	3.36E−07	+	3.64E−07	+	5.87E−07	+	1.77E−08	+
$F_{12}(x)$	7.47E−10	+	7.54E−10	+	7.55E−10	+	7.49E−10	+

Table 5 Statistical results of Wilcoxon signed-rank test obtained by proposed hASO-SA

Function type	hASO-SA versus SA	hASO-SA versus GA	hASO-SA versus PSO	hASO-SA versus GSA	hASO-SA versus WDO	hASO-SA versus ASO	hASO-SA versus hPSO-SA	hASO-SA versus hCS-SA
	(+/ = /−)	(+/ = /−)	(+/ = /−)	(+/ = /−)	(+/ = /−)	(+/ = /−)	(+/ = /−)	(+/ = /−)
Unimodal	4/0/0	3/1/0	4/0/0	3/1/0	2/1/1	3/1/0	3/1/0	3/1/0
Multimodal	4/0/0	4/0/0	4/0/0	4/0/0	3/0/1	2/2/0	2/1/1	3/1/0
Hybrid	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0
Composition	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0	2/0/0

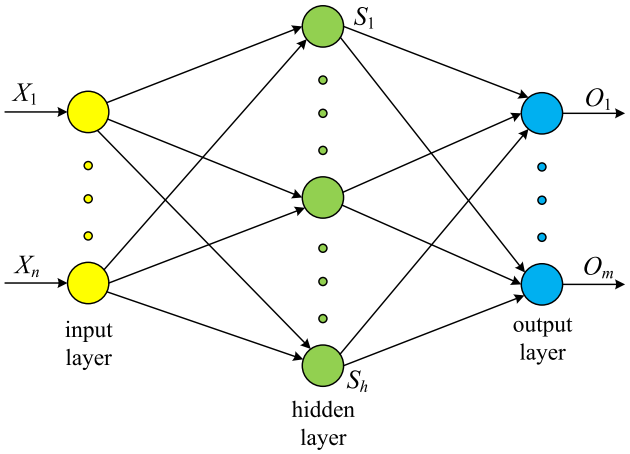

Fig. 3 Structure of MLP neural network

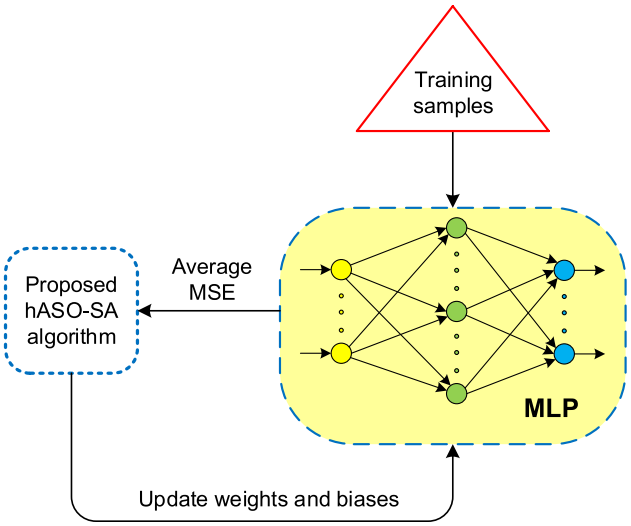

Fig. 4 hASO-SA-based MLP trainer

denotes the connection weight from hidden node j to the output node k.

$$o_k = \sum_{j=1}^{h} \left(\omega_{jk} S_j\right) - \theta'_k, \quad k = 1, 2, \ldots, m \tag{24}$$

$$O_k = \text{sigmoid}\left(o_k\right) = \frac{1}{1 + e^{-o_k}}, \quad k = 1, 2, \ldots, m \tag{25}$$

In MLP training, the biases and connection weights are playing a critical role. The quality of MLP's final output depends on the biases and weights. Therefore, training an MLP means finding optimum values for biases and weights which helps in achieving desirable outputs for defined inputs.

4.2 hASO-SA-Based MLP Trainer

It is feasible to train MLPs in three different methods using metaheuristic methods. The first method includes finding optimal connection weights and biases. In this way, metaheuristics help in achieving minimum error for an MLP. In this method, the MLP architecture remains as it is during the learning process. The second method is about finding an appropriate architecture for an MLP using metaheuristics in the case of a specific problem. Tuning parameters such as learning rate of the gradient-based learning algorithm and momentum is the third method that metaheuristics can be used.

The first method explained above was adopted to implement the proposed hASO-SA since the learning algorithm is required to minimize the MLP error by achieving the convenient weights and biases. An important aspect in MLP training is the representation of biases and weights. Three methods are available to represent them such as binary, matrix and vector [106]. In this paper, the vector method was utilized for representation of biases and weights. The objective function should be defined after representation of biases and weights in vector form in order to evaluate each candidate solution of the algorithm. In this study, the mean square error (MSE) was chosen as objective function which is formulated as:

$$E = \sum_{k=1}^{q} \frac{\sum_{i=1}^{m} \left(o_i^k - d_i^k\right)^2}{q} \tag{26}$$

where m is the number of outputs, q is the number of training samples, d_i^k is the desired output of the ith input unit when the kth training sample is used, and o_i^k is actual output of the ith input unit when the kth training sample appears in the input. Figure 4 represents the overall process of MLP training using proposed hybrid hASO-SA algorithm. As can be seen, the hASO-SA algorithm provides MLP with weights/biases and receives average MSE for all training samples. The hASO-SA algorithm iteratively changes the weights and biases to minimize average MSE of all training samples.

4.3 Experimental Setup and Analysis of Results on Classification Datasets

Five classification datasets (XOR, Balloon, iris, Heart and Breast cancer) was used to benchmark the proposed hASO-SA algorithm. Those datasets were obtained from [67]. Each candidate solution was selected from a range of $[-10, 10]^D$ randomly in the training algorithm. The population size of candidate solutions was chosen to be 200 for Iris, Heart and Breast cancer and 50 for XOR and Balloon classification

Table 6 Classification datasets

Datasets	Number of test samples	Number of attributes	Number of training samples	Number of classes
XOR	8	3	8	2
Balloon	16	4	16	2
Iris	150	4	150	3
Breast cancer	100	9	599	2
Heart	187	22	80	2

Table 7 Experimental results for the XOR classification problem

Training algorithm	MSE (AVE ± STD)	Classification rate (%)
hASO-SA (proposed)	2.31E−04 ± 1.63E−04	100.00
ASO	5.72E−03 ± 2.37E−02	100.00
GWO [72]	9.41E−03 ± 2.95E−02	100.00
PSO [70, 72]	8.40E−02 ± 3.59E−02	37.50
ACO [70, 72]	1.80E−01 ± 2.53E−02	62.50
PBIL [70, 72]	3.02E−02 ± 3.97E−02	62.50
ES [70, 72]	1.19E−01 ± 1.16E−02	62.50

Table 8 Experimental results for the Balloon classification problem

Training algorithm	MSE (AVE ± STD)	Classification rate (%)
hASO-SA (proposed)	4.66E−16 ± 3.71E−15	100.00
ASO	3.47E−08 ± 1.53E−07	100.00
GWO [72]	9.38E−15 ± 2.81E−14	100.00
PSO [70, 72]	5.85E−04 ± 7.49E−04	100.00
ACO [70, 72]	4.85E−03 ± 7.76E−03	100.00
PBIL [70, 72]	2.49E−05 ± 5.27E−05	100.00
ES [70, 72]	1.91E−02 ± 1.70E−01	100.00

Table 9 Experimental results for the Iris classification problem

Training algorithm	MSE (AVE ± STD)	Classification rate (%)
hASO-SA (proposed)	1.67E−02 ± 2.58E−03	91.33
ASO	1.83E−02 ± 2.77E−03	89.33
GWO [72]	2.29E−02 ± 3.20E−03	91.33
PSO [70, 72]	2.29E−01 ± 5.72E−02	37.33
ACO [70, 72]	4.06E−01 ± 5.38E−02	32.66
PBIL [70, 72]	1.16E−01 ± 3.64E−02	86.66
ES [70, 72]	3.14E−01 ± 5.21E−02	46.66

Table 10 Experimental results for the Breast cancer classification problem

Training algorithm	MSE (AVE ± STD)	Classification rate (%)
hASO-SA (proposed)	1.04E−03 ± 4.68E−05	100.00
ASO	3.47E−03 ± 1.64E−03	99.00
GWO [72]	1.20E−03 ± 7.45E−05	99.00
PSO [70, 72]	3.49E−02 ± 2.47E−03	11.00
ACO [70, 72]	1.35E−02 ± 2.14E−03	40.00
PBIL [70, 72]	3.20E−02 ± 3.07E−03	7.00
ES [70, 72]	4.03E−02 ± 2.47E−03	6.00

Table 11 Experimental results for the Heart classification problem

Training algorithm	MSE (AVE ± STD)	Classification rate (%)
hASO-SA (proposed)	8.57E−02 ± 9.46E−03	76.25
ASO	9.52E−02 ± 1.38E−02	73.75
GWO [72]	1.23E−01 ± 7.70E−03	75.00
PSO [70, 72]	1.89E−01 ± 8.94E−03	68.75
ACO [70, 72]	2.28E−01 ± 4.98E−03	0.00
PBIL [70, 72]	1.54E−01 ± 1.82E−02	45.00
ES [70, 72]	1.92E−01 ± 1.52E−02	71.25

problems. Maximum number of iterations (generations) is 250. The datasets were classified as presented in Table 6.

The algorithm was implemented on datasets for 10 times. The results that were obtained from those datasets are shown from Tables 7, 8, 9, 10 and 11 which provide average (AVE), and standard deviation (STD) of the best mean square error (MSE) acquired in the last iteration of the algorithm. Obviously, the lower average and standard deviation of MSE in the last iteration is an indication of better performance. The performance of the proposed hASO-SA was evaluated by comparing with a variety of algorithms such as classical ASO, ACO, GWO, PBIL, PSO and ES algorithms which was adopted to solve these classification problems [70, 72]. Compared algorithms for training an MLP for hASO-SA algorithm were acquired from Refs. [70, 72].

The collected datasets, shown in Table 6, have different difficulty levels, e.g., Heart dataset is considered to be difficult, whereas XOR is simple [67]. The large number of training samples make the problem less difficult, whereas the large number of features causes neural network with larger size and hence increases the difficulty of the problem as more weights need to be determined.

The results of the considered datasets are provided in the following subsections. According to this comprehensive

study, the hASO-SA algorithm can be highly recommended to be used in MLP training due to its high exploratory behavior. The latter specification provides high local optima avoidance while training MLP. In addition, the proposed hybrid hASO-SA algorithm has high exploitative behavior as well which helps hASO-SA-based trainer to be able to converge rapidly toward the global optimum for different datasets.

4.3.1 XOR Dataset

This is a well-known nonlinear benchmark classification problem. Recognizing the number of 1's in the input vector is the objective of this problem. The output is 1 in the case of odd number of input vector consisting of 1 s and 0 for even number of 1 s that form the input vector. The MLP with 3-7-1 structure was used to solve this problem. Numerical results are presented in Table 7 which clearly indicates the proposed hASO-SA algorithm's performance to be better in solving this problem by avoiding the sub-optimal solutions.

4.3.2 Balloon Dataset

The Balloon dataset includes 16 instances having 4 attributes such as color, age, act and size which are in string format. An MLP structure of 4-9-1 was used for classification. The acquired results are presented in Table 8 which shows that the hASO-SA provides the minimum error. The classification rates of all algorithms are the same and is 100%.

4.3.3 Iris Dataset

This dataset includes 150 samples that can be treated under three classes (Virginica, Versicolor and Setosa). Sepal width and length along with petal width and length are the four features that are present in those samples. An MLP having structure of 4-9-3 was used for solving this classification problem. The obtained results are provided in Table 9. The hASO-SA provides better performance to train MLP-based on the average value of the square error and classification rate. The comparative results showed the superior performance of hASO-SA compared to other algorithms.

4.3.4 Breast Cancer Dataset

This dataset consists of 9 attributes and 699 instances. The attributes include marginal adhesion and uniformity of cell shape and size along with clump thickness [107]. The output is 2 for benign cancer, whereas 4 for malignant cancer. The MLP structure of 9-19-1 was adopted for classification. Table 10 provides the results of this problem. As can be seen the hASO-SA provides the best mean square error in terms of average value and classification rate which is a clear indication of better search ability of the proposed algorithm in terms of escaping local optima.

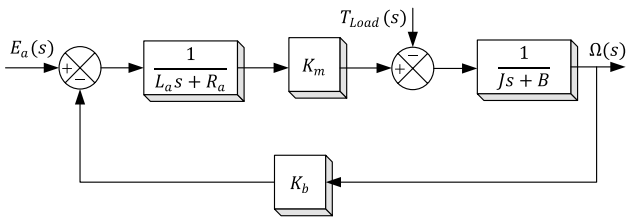

Fig. 5 Block diagram of a DC motor

Table 12 DC motor parameters [93–95]

Parameter/specification	Value
R_a (armature resistance)	$4 \times 10^{-1}\,\Omega$
L_a (armature inductance)	2.7H
J (inertia moment of motor)	4×10^{-4} kg · m^2
B (motor friction constant)	2.2×10^{-3} N · m · s/rad
K_m (motor torque constant)	1.5×10^{-2} N · m/A
K_b (electromotive force constant)	5×10^{-2} V · s/rad

4.3.5 Heart Dataset

This dataset includes 267 images and was created for diagnosing the cardiac tomography images. 22 features were extracted from those images to summarize them. The MLP with 22-45-1 structure was trained by utilizing 80 instances. The condition of a patient can be expressed as normal or not normal using binary form of the dataset. Table 11 lists the results. The table clearly shows that the proposed hASO-SA is capable of providing better results and classification rate than other algorithms.

5 Application of hASO-SA to PID Controller Design in DC Motor Speed Control

5.1 Speed Control of DC Motor System

DC motors are devices that basically convert the electrical energy into mechanical form. The speed control of a DC motor is important to perform a specific work. This can be achieved either manually by an operator or automatically by adopting control devices. Figure 5 illustrates the block diagram of a DC motor system. The relationship between the

speed of the motor (ω) and the applied voltage (E_a) under no load ($T_{Load} = 0$) is provided in Eq. (27).

$$G_{Plant}(s) = \frac{\Omega(s)}{E_a(s)} = \frac{K_m}{(L_a s + R_a)(Js + B) + K_b K_m} \tag{27}$$

The rest of the parameters for DC motor speed control, given in the latter equation, along with the respective values used for this work are listed in Table 12 [93–95].

5.2 DC Motor with PID Controller

PID controllers are the most popular controller type in engineering due to their simple structure and high efficiency. The transfer function of a PID controller is provided in Eq. (28) where K_p, K_i and K_d are gains of proportional, integral and derivative terms, respectively.

$$G_{Controller}(s) = K_p + \frac{K_i}{s} + K_d s \tag{28}$$

The closed loop block diagram of a DC motor with PID controller is given in Fig. 6. The closed loop transfer function of a DC motor having a unit feedback is given as in Eq. (29).

$$T_{closed\text{ - }loop}(s) = \frac{\Omega(s)}{\Omega_{ref}(s)} = \frac{G_{Plant}(s)G_{Controller}(s)}{1 + G_{Plant}(s)G_{Controller}(s)} \tag{29}$$

Using the parameters listed in Table 12 provides the following transfer function.

$$T_{closed\text{ - }loop}(s) = \frac{15\left(K_d s^2 + K_p s + K_i\right)}{1.08s^3 + 6.1s^2 + 1.63s + 15\left(K_d s^2 + K_p s + K_i\right)} \tag{30}$$

5.3 hASO-SA-Based PID Controller Design

The objective function was chosen to be ITAE for this study in order to achieve better speed control. The ITAE objective function is given by [93–95] as:

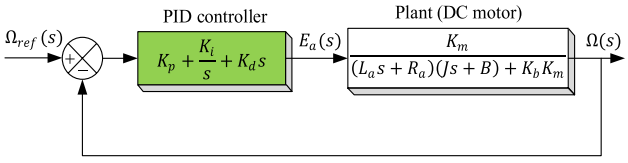

Fig. 6 Block diagram of DC motor with PID controller

$$J_{ITAE} = \int_0^{t_{sim}} t|e(t)|\mathrm{d}t \tag{31}$$

where $e(t)$ denotes the error signal and is equal to $\omega_{ref} - \omega(t)$, whereas t_{sim} is the simulation time and were chosen to be 2s for this study. The upper and lower bounds of the optimization problem are also the limits of the controller parameters and given in Eq. (32) [93–95].

$$10^{-3} \leq K_p, K_i, K_d \leq 20 \tag{32}$$

The application of the proposed hASO-SA algorithm to DC motor speed control system is illustrated in Fig. 7. The stability of DC motor speed control system increases to the highest level after completion of the detailed optimization procedure provided in the figure.

In the optimization module provided in Fig. 7, the swarm size (number of atoms) and the maximum number of iterations (stopping criteria) were set to be 40 and 30, respectively. The value of the ITAE objective function given in Eq. (31) was calculated for each atom in the swarm by integrating the proposed hASO-SA with the DC motor system using MATLAB/Simulink environment. The proposed algorithm was run for 25 times, and the PID parameters corresponding minimum ITAE value were found to be $K_p = 18.4258$, $K_i = 3.3082$ and $K_d = 3.1755$.

5.4 Comparative Simulation Results

The proposed hASO-SA-based controller's performance was evaluated by comparing it with the most recent studies published in prestigious journals using a variety of analysis. The most convenient approaches chosen for comparison were ASO/PID [95], GWO/PID [94] and SCA/PID [93] controllers since those approaches adopted the same DC motor parameters, ITAE objective function and the limits of the controller parameters. PID controller parameters that were optimized with different algorithms are presented in Table 13.

In addition, the closed loop transfer functions of DC motor with the proposed hASO-SA/PID, ASO/PID [95], GWO/PID [94] and SCA/PID [93] are given in Eqs. (33), (34), (35) and (36), respectively. The analyses of time and frequency domain along with robustness were performed using the latter equations.

$$T_{hASO\text{ - }SA}(s) = \frac{47.63s^2 + 276.4s + 49.62}{1.08s^3 + 53.73s^2 + 278s + 49.62} \tag{33}$$

$$T_{ASO}(s) = \frac{36.54s^2 + 179.2s + 30.78}{1.08s^3 + 42.64s^2 + 180.8s + 30.78} \tag{34}$$

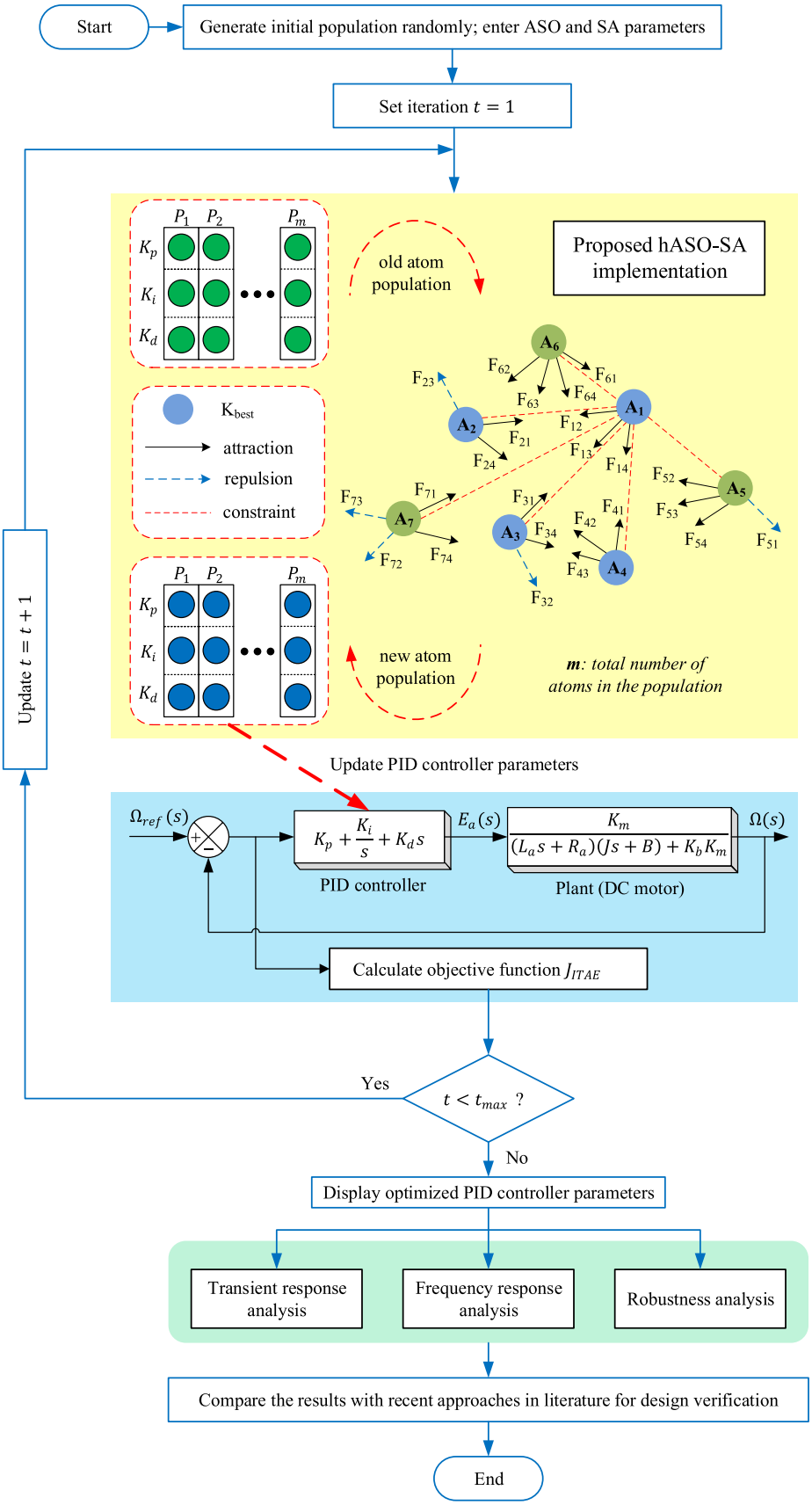

Fig. 7 The procedure of applying the proposed hASO-SA algorithm to DC motor speed control

Table 13 Optimized controller parameters using different algorithms

Algorithm/controller	K_p	K_i	K_d
hASO-SA/PID (proposed)	18.4258	3.3082	3.1755
ASO/PID [95]	11.9437	2.0521	2.4358
GWO/PID [94]	6.8984	0.5626	0.9293
SCA/PID [93]	4.5012	0.5260	0.5302

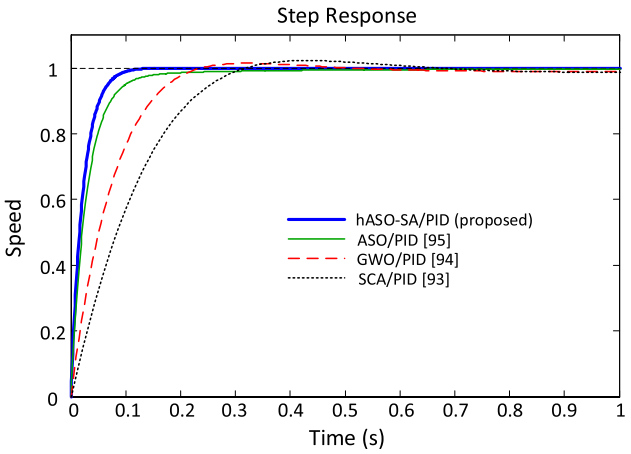

Fig. 8 The comparison of DC motor step responses

$$T_{\mathrm{GWO}}(s) = \frac{13.94s^2 + 103.5s + 8.439}{1.08s^3 + 20.04s^2 + 105.1s + 8.439} \tag{35}$$

$$T_{\mathrm{SCA}}(s) = \frac{7.953s^2 + 67.52s + 7.89}{1.08s^3 + 14.05s^2 + 69.15s + 7.89} \tag{36}$$

Based on the simulation results, it can clearly be seen that the usage of the hASO-SA algorithm provides significantly better transient and frequency responses compared to three algorithms from the recent literature. Furthermore, the suggested hASO-SA/PID controller design is more robust to the variations in the system parameters than the other competitive algorithms-based controller designs.

5.4.1 Transient Response Analysis

The step responses of DC motor (normalized speed responses) using the proposed hASO-SA/PID, ASO/PID [95], GWO/PID [94] and SCA/PID [93] controllers are illustrated in Fig. 8. As can be observed from the figure, the speed of DC motor reaches the steady state value quickly without any overshoot. This behavior confirms the superiority of the proposed hASO-SA algorithm over ASO, GWO and SC algorithms. Also, the comparative results of maximum overshoot, rise time and settling time for a tolerance band of $\pm 2\%$ are provided in Table 14. Moreover, the ITAE objective function values are also given in the table. As can be seen from numerical and graphical outcomes, the speed response of a DC motor with the proposed hASO-SA/PID controller is more stable and has no overshoot.

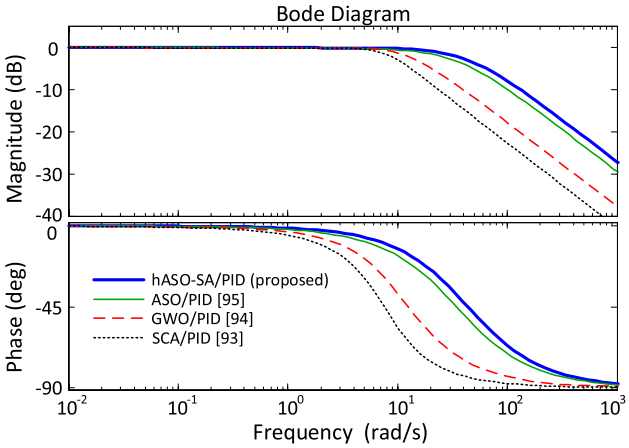

Fig. 9 Comparative Bode diagrams

5.4.2 Frequency Response Analysis

The Bode diagram of a DC motor speed control system provides information about frequency response. The comparative Bode plots of the different approaches are given in Fig. 9. Gain margin, phase margin and bandwidth of the system can be calculated easily from this figure. Important parameters of frequency response such as bandwidth, gain and phase margin, obtained from Fig. 9, are presented in Table 15. The best value is shown in bold. The numerical values presented in the table clearly show that the proposed hASO-SA/PID-based system has the best frequency response.

Table 14 Comparison of transient response analysis results

Algorithm/controller	Overshoot (%)	Rise time (s)	Settling time (s)	ITAE value
hASO-SA/PID (proposed)	0.0000	0.0494	0.0866	0.0036
ASO/PID [95]	0.0000	0.0692	0.1535	0.0075
GWO/PID [94]	1.5062	0.1388	0.2052	0.0223
SCA/PID [93]	2.3056	0.2038	0.4899	0.0307

Table 15 Comparison of frequency response analysis results

Algorithm/controller	Gain margin (dB)	Phase margin (deg.)	Bandwidth (Hz)
hASO-SA/PID (proposed)	∞	180°	**44.1802**
ASO/PID [95]	∞	180°	32.9113
GWO/PID [94]	∞	180°	14.9018
SCA/PID [93]	∞	180°	10.1347

5.4.3 Robustness Analysis

The output of any system may be affected undesirably due to unexpected sudden changes. It is crucial to design the system from having undesired responses. Therefore, a robustness analysis was performed by observing the behavior of the system which were altered separately with ±25% for R_a and with ±20% for K_m. The latter action created four possible operating scenarios. The scenarios and their comparative time domain performance analysis simulation results are presented in Table 16. The best values are shown in bold.

Likewise, the comparative speed step responses for all scenarios is depicted from Figs. 10, 11, 12 and 13. It is clear from the latter figures that despite the changes occurring in the system parameters, the proposed hASO-SA/PID controller has the least rise time and settling time values with no overshoot in all scenarios except Scenario I and Scenario II with a negligible overshoot percentage while compared to other controllers optimized by ASO, GWO and SCA. As can be seen from the results listed in the table and demonstrated in the figures, the proposed hASO-SA/PID controller has

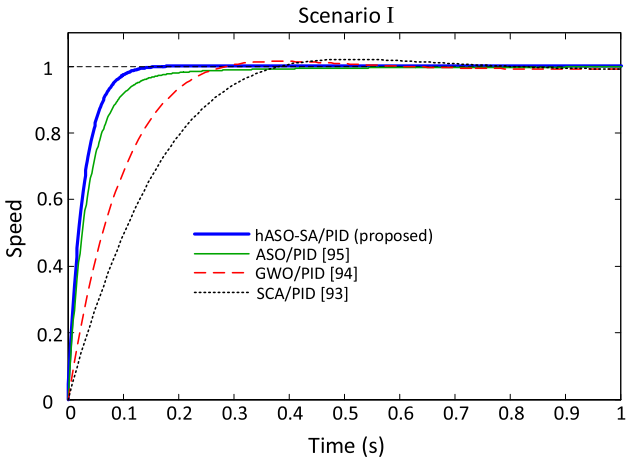

Fig. 10 Comparative speed responses for Scenario I

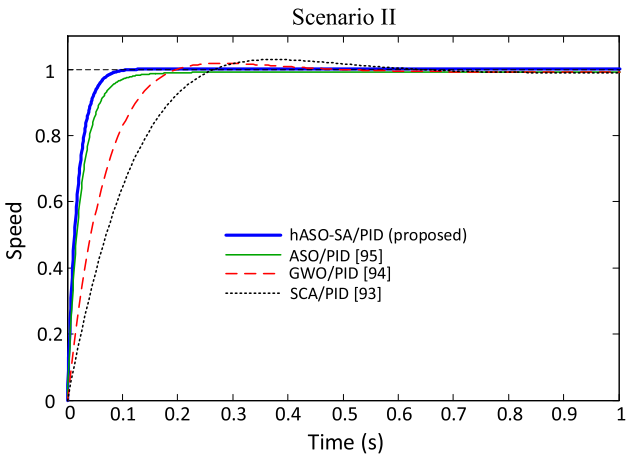

Fig. 11 Comparative speed responses for Scenario II

Table 16 Performance comparisons for parametric uncertainty

Scenario	Parameter changes	Algorithm/Controller	Overshoot (%)	Rise time (s)	Settling time (s)
Scenario I	$R_a = 0.300$ $K_m = 0.012$	hASO-SA/PID (proposed)	0.1083	**0.0614**	**0.1066**
		ASO/PID [95]	**0.0000**	0.0872	0.1936
		GWO/PID [94]	1.5195	0.1683	0.2471
		SCA/PID [93]	2.1514	0.2447	0.5618
Scenario II	$R_a = 0.300$ $K_m = 0.018$	hASO-SA/PID (proposed)	0.0549	**0.0411**	**0.0718**
		ASO/PID [95]	**0.0000**	0.0569	0.1198
		GWO/PID [94]	1.8415	0.1173	0.1731
		SCA/PID [93]	2.9696	0.1733	0.4787
Scenario III	$R_a = 0.500$ $K_m = 0.012$	hASO-SA/PID (proposed)	**0.0000**	**0.0618**	**0.1089**
		ASO/PID [95]	**0.0000**	0.0881	0.2107
		GWO/PID [94]	0.9547	0.1706	0.2547
		SCA/PID [93]	1.3036	0.2492	0.3573
Scenario IV	$R_a = 0.500$ $K_m = 0.018$	hASO-SA/PID (proposed)	**0.0000**	**0.0413**	**0.0729**
		ASO/PID [95]	**0.0000**	0.0573	0.1257
		GWO/PID [94]	1.4479	0.1185	0.1767
		SCA/PID [93]	2.3664	0.1755	0.4326

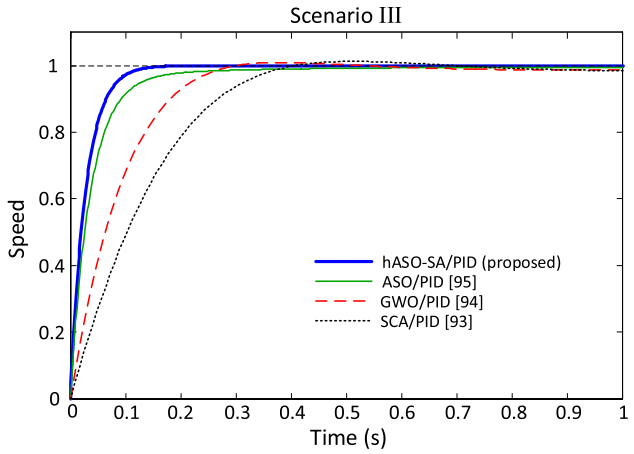

Fig. 12 Comparative speed responses for Scenario III

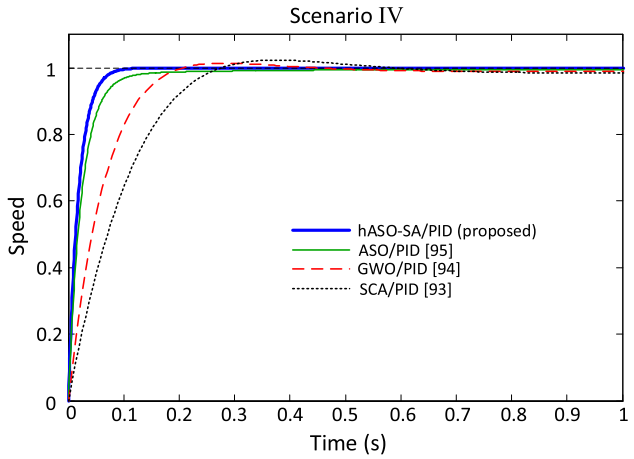

Fig. 13 Comparative speed responses for Scenario IV

not affected from the change of the parameters. Also, the proposed hASO-SA/PID has better robustness performance compared to ASO/PID [95], GWO/PID [94] and SCA/PID [93] controllers.

6 Conclusion

A novel hybrid metaheuristic algorithm based on ASO and SA algorithms was proposed in this work by embedding the SA technique into ASO algorithm. The specific configuration helped improving search capability of ASO without costing considerably longer computational time. Several optimization problems with different nature were utilized to observe the performance of the proposed hybrid hASO-SA algorithm. Classical benchmark functions of Sphere, Rosenbrock, Step, Quartic, Schwefel, Rastrigin, Ackley and Griewank as well as CEC2014 test functions, were adopted for initial assessment of the algorithm. Those functions are of unimodal, multimodal, hybrid and composition types and help effectively measuring the performance of the proposed algorithm. The performance comparisons were carried out with PSO, GA, GSA and WDO along with SA and original ASO, using respective test functions, in order to demonstrate the superiority of the proposed algorithm. Moreover, SA-based hybrid versions of PSO (hPSO-SA) and CS (hCS-SA) algorithms were also used to provide a stronger comparison. The statistical results obtained from those benchmark functions clearly showed the greater capability of the proposed hASO-SA, compared to the algorithms listed above, in terms of achieving values for the metrics of the best, mean and standard deviation. Apart from those statistical values, a nonparametric test named Wilcoxon signed-rank test was also adopted to prove the superiority of the proposed hybrid hASO-SA algorithm was not by chance.

Further assessment was performed by using the proposed algorithm for MLP training, as a nonlinear system, in order to observe the ability of the proposed algorithm for an optimization problem with different nature. To do so, Balloon, Iris, XOR, Heart and Breast cancer datasets were used since they have different difficulty levels, and, similar to the previous case, the performance of the algorithm was compared with other metaheuristics of ASO, GWO, PSO, ACO, ES and PBIL algorithms. The results from those datasets showed the proposed hASO-SA algorithm to be better compared to the listed algorithms since it provided the lowest average and standard deviation of the best mean square error.

A PID controller design for DC motor speed control was also performed as a final evaluation process as it is a widely used test bed for performance assessment of the algorithms. The obtained hASO-SA/PID controller was compared with other algorithms-based PID controllers such as ASO/PID, GWO/PID and SCA/PID controllers since those studies adopted the same parameters for DC motor and the limits of the controller as well as the same objective function. The speed response of DC motor has found to be more stable with no overshoot compared to the same system with different algorithms. In addition, the frequency response was also found to be the best among the competitive algorithms. Moreover, the system presented better robustness, in the case of the changes occurring in the system parameters with the implementation of the proposed hybrid algorithm.

In conclusion, the implementation of the proposed hASO-SA algorithm to problems with different nature showed that this algorithm is a powerful technique for various optimization problems. The proposed hybrid algorithm also has the potential to provide better performance characteristics in several other optimization problems of different types for future studies. Some of those applications can be listed as feature selection and photovoltaic cell parameter estimation

along with controller design for automatic voltage regulator and magnetic ball suspension systems.

Funding 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.

Compliance with Ethical Standards

Conflict of interest All authors declare that they have no conflict of interest.

References

1. Rao, S.S.; Desai, R.C.: Optimization theory and applications. IEEE Trans. Syst. Man Cybern. **10**, 280 (1980). https://doi.org/10.1109/TSMC.1980.4308490
2. Uryasev, S.; Pardalos, P.M.: Stochastic Optimization: Algorithms and Applications. Springer, Berlin (2013)
3. Antoniou, A.; Lu, W.S.: Practical Optimization: Algorithms and Engineering Applications. Springer, Berlin (2007)
4. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H.: Harris Hawks Optimization: Algorithm and Applications. Futur. Gener. Comput. Syst. **97**, 849–872 (2019). https://doi.org/10.1016/j.future.2019.02.028
5. Singh, N.; Son, L.H.; Chiclana, F.; Magnot, J.P.: A new fusion of salp swarm with sine cosine for optimization of non-linear functions. Eng. Comput. **36**, 185–212 (2020). https://doi.org/10.1007/s00366-018-00696-8
6. Mohammed, H.; Rashid, T.: A novel hybrid GWO with WOA for global numerical optimization and solving pressure vessel design. Neural Comput. Appl. (2020). https://doi.org/10.1007/s00521-020-04823-9
7. Das, P.K.: Hybridization of kidney-inspired and sine-cosine algorithm for multi-robot path planning. Arab. J. Sci. Eng. **45**, 2883–2900 (2020). https://doi.org/10.1007/s13369-019-04193-y
8. Zhang, Z.; Ding, S.; Jia, W.: A hybrid optimization algorithm based on cuckoo search and differential evolution for solving constrained engineering problems. Eng. Appl. Artif. Intell. **85**, 254–268 (2019). https://doi.org/10.1016/j.engappai.2019.06.017
9. Zhao, W.; Zhang, Z.; Wang, L.: Manta ray foraging optimization: an effective bio-inspired optimizer for engineering applications. Eng. Appl. Artif. Intell. **87**, 103300 (2020). https://doi.org/10.1016/j.engappai.2019.103300
10. Hasançebi, O.; Erbatur, F.: Constraint handling in genetic algorithm integrated structural optimization. Acta Mech. **139–145**, 15–31 (2000). https://doi.org/10.1007/bf01170179
11. Jordehi, A.R.: A review on constraint handling strategies in particle swarm optimisation. Neural Comput. Appl. **26**, 1265–1275 (2015). https://doi.org/10.1007/s00521-014-1808-5
12. Jain, M.; Singh, V.; Rani, A.: A novel nature-inspired algorithm for optimization: squirrel search algorithm. Swarm Evol. Comput. **44**, 148–175 (2019). https://doi.org/10.1016/j.swevo.2018.02.013
13. Mirjalili, S.: The ant lion optimizer. Adv. Eng. Softw. **83**, 80–98 (2015). https://doi.org/10.1016/j.advengsoft.2015.01.010
14. Zhao, W.; Wang, L.; Zhang, Z.: Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm. Neural Comput. Appl. **32**, 9383–9425 (2020). https://doi.org/10.1007/s00521-019-04452-x
15. Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S.: Slime mould algorithm: a new method for stochastic optimization. Futur. Gener. Comput. Syst. **111**, 300–323 (2020). https://doi.org/10.1016/j.future.2020.03.055
16. Hashim, F.A.; Houssein, E.H.; Mabrouk, M.S.; Al-Atabany, W.; Mirjalili, S.: Henry gas solubility optimization: a novel physics-based algorithm. Futur. Gener. Comput. Syst. **101**, 646–667 (2019). https://doi.org/10.1016/j.future.2019.07.015
17. Arora, S.; Singh, S.: Butterfly optimization algorithm: a novel approach for global optimization. Soft. Comput. **23**, 715–734 (2019). https://doi.org/10.1007/s00500-018-3102-4
18. Cheng, M.Y.; Prayogo, D.: Symbiotic organisms search: a new metaheuristic optimization algorithm. Comput. Struct. **139**, 98–112 (2014). https://doi.org/10.1016/j.compstruc.2014.03.007
19. Karaboga, D.; Akay, B.: A comparative study of Artificial Bee Colony algorithm. Appl. Math. Comput. **214**, 108–132 (2009). https://doi.org/10.1016/j.amc.2009.03.090
20. Harifi, S.; Khalilian, M.; Mohammadzadeh, J.; Ebrahimnejad, S.: Emperor Penguins Colony: a new metaheuristic algorithm for optimization. Evol. Intell. **12**, 211–226 (2019). https://doi.org/10.1007/s12065-019-00212-x
21. Mirjalili, S.: SCA: a Sine Cosine Algorithm for solving optimization problems. Knowl. Based Syst. **96**, 120–133 (2016). https://doi.org/10.1016/j.knosys.2015.12.022
22. Jaddi, N.S.; Alvankarian, J.; Abdullah, S.: Kidney-inspired algorithm for optimization problems. Commun. Nonlinear Sci. Numer. Simul. **42**, 358–369 (2017). https://doi.org/10.1016/j.cnsns.2016.06.006
23. Wolpert, D.H.; Macready, W.G.: No free lunch theorems for optimization. IEEE Trans. Evol. Comput. **1**, 67–82 (1997). https://doi.org/10.1109/4235.585893
24. Zhao, W.; Wang, L.; Zhang, Z.: Atom search optimization and its application to solve a hydrogeologic parameter estimation problem. Knowl. Based Syst. **163**, 283–304 (2019). https://doi.org/10.1016/j.knosys.2018.08.030
25. Zhao, W.; Wang, L.; Zhang, Z.: A novel atom search optimization for dispersion coefficient estimation in groundwater. Futur. Gener. Comput. Syst. **91**, 601–610 (2019). https://doi.org/10.1016/j.future.2018.05.037
26. Too, J.; Abdullah, A.R.: Chaotic atom search optimization for feature selection. Arab. J. Sci. Eng. (2020). https://doi.org/10.1007/s13369-020-04486-7
27. Too, J.; Rahim Abdullah, A.: Binary atom search optimisation approaches for feature selection. Conn. Sci. (2020). https://doi.org/10.1080/09540091.2020.1741515
28. Pham, M.H.; Do, T.H.; Pham, V.M.; Bui, Q.T.: Mangrove forest classification and aboveground biomass estimation using an atom search algorithm and adaptive neuro-fuzzy inference system. PLoS ONE **15**, e0233110 (2020). https://doi.org/10.1371/journal.pone.0233110
29. Yang, B.; Zhang, M.; Zhang, X.; Wang, J.; Shu, H.; Li, S.; He, T.; Yang, L.; Yu, T.: Fast atom search optimization based MPPT design of centralized thermoelectric generation system under heterogeneous temperature difference. J. Clean. Prod. **248**, 119301 (2020). https://doi.org/10.1016/j.jclepro.2019.119301
30. Almagboul, M.A.; Shu, F.; Qian, Y.; Zhou, X.; Wang, J.; Hu, J.: Atom search optimization algorithm based hybrid antenna array receive beamforming to control sidelobe level and steering the null. AEU Int. J. Electron. Commun. **111**, 152854 (2019). https://doi.org/10.1016/j.aeue.2019.152854
31. Ekinci, S.; Demiroren, A.; Zeynelgil, H.; Hekimoğlu, B.: An opposition-based atom search optimization algorithm for automatic voltage regulator system. J. Fac. Eng. Archit. Gazi Univ. **35**, 1141–1158 (2020). https://doi.org/10.17341/gazimmfd.598576
32. Abdel-Rahim, A.M.M.; Shaaban, S.A.; Raglend, I.J.: Optimal Power Flow Using Atom Search Optimization. In: 2019

Innovations in Power and Advanced Computing Technologies, i-PACT 2019. pp. 1–4. IEEE (2019)
33. Diab, A.A.Z.; Ebraheem, T.; Aljendy, R.; Sultan, H.M.; Ali, Z.M.: Optimal design and control of MMC STATCOM for improving power quality indicators. Appl. Sci. **10**, 2490 (2020). https://doi.org/10.3390/app10072490
34. Agwa, A.M.; El-Fergany, A.A.; Sarhan, G.M.: Steady-state modeling of fuel cells based on atom search optimizer. Energies. **12**, 1884 (2019). https://doi.org/10.3390/en12101884
35. Rizk-Allah, R.M.; Hassanien, A.E.; Oliva, D.: An enhanced sitting–sizing scheme for shunt capacitors in radial distribution systems using improved atom search optimization. Neural Comput. Appl. (2020). https://doi.org/10.1007/s00521-020-04799-6
36. Farnad, B.; Jafarian, A.; Baleanu, D.: A new hybrid algorithm for continuous optimization problem. Appl. Math. Model. **55**, 652–673 (2018). https://doi.org/10.1016/j.apm.2017.10.001
37. Mafarja, M.M.; Mirjalili, S.: Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. Neurocomputing. **260**, 302–312 (2017). https://doi.org/10.1016/j.neucom.2017.04.053
38. Sun, P.; Zhang, Y.; Liu, J.; Bi, J.: An improved atom search optimization with cellular automata, a Lévy flight and an adaptive weight strategy. IEEE Access. **8**, 49137–49159 (2020). https://doi.org/10.1109/ACCESS.2020.2979921
39. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P.: Optimization by simulated annealing. Science **220**, 671–680 (1983). https://doi.org/10.1126/science.220.4598.671
40. Nayak, J.R.; Shaw, B.; Sahu, B.K.: Implementation of hybrid SSA-SA based three-degree-of-freedom fractional-order PID controller for AGC of a two-area power system integrated with small hydro plants. IET Gener. Transm. Distrib. **14**, 2430–2440 (2020). https://doi.org/10.1049/iet-gtd.2019.0113
41. Attiya, I.; Abd Elaziz, M.; Xiong, S.: Job scheduling in cloud computing using a modified Harris Hawks optimization and simulated annealing algorithm. Comput. Intell. Neurosci. (2020). https://doi.org/10.1155/2020/3504642
42. Jouhari, H.; Lei, D.; Al-qaness, M.A.A.; Elaziz, M.A.; Ewees, A.A.; Farouk, O.: Sine-cosine algorithm to enhance simulated annealing for unrelated parallel machine scheduling with setup times. Mathematics. **7**, 1120 (2019). https://doi.org/10.3390/math7111120
43. Pan, X.; Xue, L.; Lu, Y.; Sun, N.: Hybrid particle swarm optimization with simulated annealing. Multimed. Tools Appl. **78**, 29921–29936 (2019). https://doi.org/10.1007/s11042-018-6602-4
44. Shang, Y.; Fan, Q.; Shang, L.; Sun, Z.; Xiao, G.: Modified genetic algorithm with simulated annealing applied to optimal load dispatch of the Three Gorges Hydropower Plant in China. Hydrol. Sci. J. **64**, 1129–1139 (2019). https://doi.org/10.1080/02626667.2019.1625052
45. Kurtuluş, E.; Yıldız, A.R.; Sait, S.M.; Bureerat, S.: A novel hybrid Harris hawks-simulated annealing algorithm and RBF-based metamodel for design optimization of highway guardrails. Mater. Test. **62**, 251–260 (2020). https://doi.org/10.3139/120.111478
46. Yu, C.; Heidari, A.A.; Chen, H.: A quantum-behaved simulated annealing algorithm-based moth-flame optimization method. Appl. Math. Model. **87**, 1–19 (2020). https://doi.org/10.1016/j.apm.2020.04.019
47. Shahidul Islam, M.; Rafiqul Islam, M.: A hybrid framework based on genetic algorithm and simulated annealing for RNA structure prediction with pseudoknots. J. King Saud Univ. Comput. Inf. Sci. (2020). https://doi.org/10.1016/j.jksuci.2020.03.005
48. Tavakoli, A.: Multi-criteria optimization of multi product assembly line using hybrid Tabu-SA algorithm. SN Appl. Sci. **2**, 151 (2020). https://doi.org/10.1007/s42452-019-1863-8
49. Al-Rawashdeh, G.; Mamat, R.; Hafhizah Binti Abd Rahim, N.: Hybrid water cycle optimization algorithm with simulated annealing for spam E-mail detection. IEEE Access. **7**, 143721–143734 (2019). https://doi.org/10.1109/ACCESS.2019.2944089
50. Selim, S.Z.; Alsultan, K.: A simulated annealing algorithm for the clustering problem. Pattern Recognit. **24**, 1003–1008 (1991). https://doi.org/10.1016/0031-3203(91)90097-O
51. Elmi, A.; Solimanpur, M.; Topaloglu, S.; Elmi, A.: A simulated annealing algorithm for the job shop cell scheduling problem with intercellular moves and reentrant parts. Comput. Ind. Eng. **61**, 171–178 (2011). https://doi.org/10.1016/j.cie.2011.03.007
52. Wu, Y.; Tang, M.; Fraser, W.: A simulated annealing algorithm for energy efficient virtual machine placement. In: 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 1245–1250 (2012)
53. El-Naggar, K.M.; AlRashidi, M.R.; AlHajri, M.F.; Al-Othman, A.K.: Simulated annealing algorithm for photovoltaic parameters identification. Sol. Energy **86**, 266–274 (2012). https://doi.org/10.1016/j.solener.2011.09.032
54. Wang, Y.; Bu, G.; Wang, Y.; Zhao, T.; Zhang, Z.; Zhu, Z.: Application of a simulated annealing algorithm to design and optimize a pressure-swing distillation process. Comput. Chem. Eng. **95**, 97–107 (2016). https://doi.org/10.1016/j.compchemeng.2016.09.014
55. Ziane, I.; Benhamida, F.; Graa, A.: Simulated annealing algorithm for combined economic and emission power dispatch using max/max price penalty factor. Neural Comput. Appl. **28**, 197–205 (2017). https://doi.org/10.1007/s00521-016-2335-3
56. Karagul, K.; Sahin, Y.; Aydemir, E.; Oral, A.: A Simulated Annealing Algorithm Based Solution Method for a Green Vehicle Routing Problem with Fuel Consumption BT—Lean and Green Supply Chain Management: Optimization Models and Algorithms. Presented at the (2019)
57. Tang, S.; Peng, M.; Xia, G.; Wang, G.; Zhou, C.: Optimization design for supercritical carbon dioxide compressor based on simulated annealing algorithm. Ann. Nucl. Energy **140**, 107107 (2020). https://doi.org/10.1016/j.anucene.2019.107107
58. Hasançebi, O.; Çarbaş, S.; Saka, M.P.: Improving the performance of simulated annealing in structural optimization. Struct. Multidiscip. Optim. **41**, 189–203 (2010). https://doi.org/10.1007/s00158-009-0418-9
59. Hasançebi, O.; Çarbaş, S.; Doğan, E.; Erdal, F.; Saka, M.P.: Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures. Comput. Struct. **87**, 284–302 (2009). https://doi.org/10.1016/j.compstruc.2009.01.002
60. Hasançebi, O.; Çarbaş, S.; Doğan, E.; Erdal, F.; Saka, M.P.: Comparison of non-deterministic search techniques in the optimum design of real size steel frames. Comput. Struct. **88**, 1033–1048 (2010). https://doi.org/10.1016/j.compstruc.2010.06.006
61. Hasançebi, O.; Doğan, E.: Optimizing single-span steel truss bridges with simulated annealing. Asian J. Civ. Eng. (Build. Hous.) **11**, 763–775 (2010)
62. Javidrad, F.; Nazari, M.: A new hybrid particle swarm and simulated annealing stochastic optimization method. Appl. Soft Comput. J. **60**, 634–654 (2017). https://doi.org/10.1016/j.asoc.2017.07.023
63. Alkhateeb, F.; Abed-Alguni, B.H.: A hybrid cuckoo search and simulated annealing algorithm. J. Intell. Syst. **28**, 683–698 (2017). https://doi.org/10.1515/jisys-2017-0268
64. Rashedi, E.; Nezamabadi-pour, H.; Saryazdi, S.: GSA: a gravitational search algorithm. Inf. Sci. (Ny) **179**, 2232–2248 (2009). https://doi.org/10.1016/j.ins.2009.03.004
65. Liang, J.J.; Qu, B.Y.; Suganthan, P.N.: Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session on Single

Objective Real-Parameter Numerical Optimization. Technical Report 201311, Comput. Intell. Lab. Zhengzhou Univ. Nanyang Technol. Univ. **635**, (2013)
66. Woolson, R.F.: Wilcoxon signed-rank test. In: D'Agostino, R.B., Sullivan, L., Massaro, J. (eds.) Wiley Encyclopedia of Clinical Trials (2008). https://doi.org/10.1002/9780471462422.eoct979
67. Blake, C.L.; Merz, C.J.: UCI Repository of machine learning databases. http://archive.ics.uci.edu/ml/
68. Bansal, P.; Kumar, S.; Pasrija, S.; Singh, S.: A hybrid grasshopper and new cat swarm optimization algorithm for feature selection and optimization of multi-layer perceptron. Soft. Comput. (2020). https://doi.org/10.1007/s00500-020-04877-w
69. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall PTR, Upper Saddle River (1999)
70. Gupta, S.; Deep, K.: A novel hybrid sine cosine algorithm for global optimization and its application to train multilayer perceptrons. Appl. Intell. **50**, 993–1026 (2020). https://doi.org/10.1007/s10489-019-01570-w
71. Suratgar, A.A.; Tavakoli, M.B.; Hoseinabadi, A.: Modified Levenberg–Marquardt method for neural networks training. World Acad. Sci. Eng. Technol. **6**, 46–48 (2005)
72. Mirjalili, S.: How effective is the Grey Wolf optimizer in training multi-layer perceptrons. Appl. Intell. **43**, 150–161 (2015). https://doi.org/10.1007/s10489-014-0645-7
73. Faris, H.; Mirjalili, S.; Aljarah, I.: Automatic selection of hidden neurons and weights in neural networks using grey wolf optimizer based on a hybrid encoding scheme. Int. J. Mach. Learn. Cybern. **10**, 2901–2920 (2019). https://doi.org/10.1007/s13042-018-00913-2
74. Zhang, X.; Wang, X.; Chen, H.; Wang, D.; Fu, Z.: Improved GWO for large-scale function optimization and MLP optimization in cancer identification. Neural Comput. Appl. **32**, 1305–1325 (2020). https://doi.org/10.1007/s00521-019-04483-4
75. Heidari, A.A.; Faris, H.; Mirjalili, S.; Aljarah, I.; Mafarja, M.: Ant lion optimizer: theory, literature review, and application in multi-layer perceptron neural networks. In: Mirjalili, S., Song Dong, J., Lewis, A. (eds.) Studies in computational intelligence, pp. 23–46. Springer International Publishing, Cham (2020)
76. Khishe, M.; Mosavi, M.R.: Classification of underwater acoustical dataset using neural network trained by Chimp Optimization Algorithm. Appl. Acoust. **157**, 107005 (2020). https://doi.org/10.1016/j.apacoust.2019.107005
77. Heidari, A.A.; Faris, H.; Aljarah, I.; Mirjalili, S.: An efficient hybrid multilayer perceptron neural network with grasshopper optimization. Soft. Comput. **23**, 7941–7958 (2019). https://doi.org/10.1007/s00500-018-3424-2
78. Khishe, M.; Mohammadi, H.: Passive sonar target classification using multi-layer perceptron trained by salp swarm algorithm. Ocean Eng. **181**, 98–108 (2019). https://doi.org/10.1016/j.oceaneng.2019.04.013
79. Bairathi, D.; Gopalani, D.: Numerical optimization and feedforward neural networks training using an improved optimization algorithm: multiple leader salp swarm algorithm. Evol. Intell. (2019). https://doi.org/10.1007/s12065-019-00269-8
80. Xu, J.; Yan, F.: Hybrid Nelder–Mead algorithm and dragonfly algorithm for function optimization and the training of a multilayer perceptron. Arab. J. Sci. Eng. **44**, 3473–3487 (2019). https://doi.org/10.1007/s13369-018-3536-0
81. Mirjalili, S.; Hashim, S.Z.M.; Sardroudi, H.M.: Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Appl. Math. Comput. **218**, 11125–11137 (2012)
82. Mirjalili, S.; Sadiq, A.S.: Magnetic optimization algorithm for training multi layer perceptron. In: 2011 IEEE 3rd International Conference on Communication Software and Networks. pp. 42–46 (2011)
83. Mirjalili, S.; Mirjalili, S.M.; Lewis, A.: Let a biogeography-based optimizer train your multi-layer perceptron. Inf. Sci. (Ny) **269**, 188–209 (2014). https://doi.org/10.1016/j.ins.2014.01.038
84. Ghanem, W.A.H.M.; Jantan, A.: Training a neural network for cyberattack classification applications using hybridization of an artificial bee colony and monarch butterfly optimization. Neural Process. Lett. **51**, 905–946 (2020). https://doi.org/10.1007/s11063-019-10120-x
85. Sabir, M.M.; Khan, J.A.: Optimal design of PID controller for the speed control of DC motor by using metaheuristic techniques. Adv. Artif. Neural Syst. **2014**, 1–8 (2014). https://doi.org/10.1155/2014/126317
86. Ekinci, S.; Izci, D.; Hekimoglu, B.: PID speed control of DC motor using Harris Hawks optimization algorithm. In: 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE). pp. 1–6 (2020)
87. Bhatt, R.; Parmar, G.; Gupta, R.; Sikander, A.: Application of stochastic fractal search in approximation and control of LTI systems. Microsyst. Technol. **25**, 105–114 (2019). https://doi.org/10.1007/s00542-018-3939-6
88. Hekimoğlu, B.: Speed control of DC motor using PID controller tuned via kidney-inspired algorithm. BEU J. Sci. **8**, 652–663 (2019). https://doi.org/10.17798/bitlisfen.496782
89. Mishra, A.; Singh, N.; Yadav, S.: Design of optimal PID controller for varied system using teaching–learning-based optimization. In: Sharma, H., Govindan, K., Poonia, R., Kumar, S., El-Medany, W. (eds.) Advances in Computing and Intelligent Systems, pp. 153–163. Springer (2020). https://doi.org/10.1007/978-981-15-0222-4_13
90. Qi, Z.; Shi, Q.; Zhang, H.: Tuning of digital PID controllers using particle swarm optimization algorithm for a CAN-Based DC motor subject to stochastic delays. IEEE Trans. Ind. Electron. **67**, 5637–5646 (2020). https://doi.org/10.1109/TIE.2019.2934030
91. Pongfai, J.; Su, X.; Zhang, H.; Assawinchaichote, W.: A novel optimal PID controller autotuning design based on the SLP algorithm. Expert Syst. **37**, e12489 (2020). https://doi.org/10.1111/exsy.12489
92. Kouassi, B.A.; Zhang, Y.; Mbyamm Kiki, M.J.; Ouattara, S.: Speed control of brushless de motor using Ant Colony Optimization. IOP Conf. Ser. Earth Environ. Sci. **431**, 12022 (2020). https://doi.org/10.1088/1755-1315/431/1/012022
93. Agarwal, J.; Parmar, G.; Gupta, R.: Application of sine cosine algorithm in optimal control of DC motor and robustness analysis. Wulfenia J. **24**(11), 77–95 (2017)
94. Agarwal, J.; Parmar, G.; Gupta, R.; Sikander, A.: Analysis of grey wolf optimizer based fractional order PID controller in speed control of DC motor. Microsyst. Technol. **24**, 4997–5006 (2018). https://doi.org/10.1007/s00542-018-3920-4
95. Hekimoğlu, B.: Optimal tuning of fractional order PID controller for DC motor speed control via chaotic atom search optimization algorithm. IEEE Access. **7**, 38100–38114 (2019). https://doi.org/10.1109/ACCESS.2019.2905961
96. Puangdownreong, D.: Fractional order PID controller design for DC motor speed control system via flower pollination algorithm. Trans. Electr. Eng. Electron. Commun. **17**, 14–23 (2019). https://doi.org/10.37936/ecti-eec.2019171.215368
97. Ekinci, S.; Hekimoğlu, B.; Demirören, A.; Eker, E.: Speed Control of DC Motor Using Improved Sine Cosine Algorithm Based PID Controller. In: 2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). pp. 1–7 (2019)
98. El-Deen, A.T.; Hakim Mahmoud, A.A.; El-Sawi, A.R.: Optimal PID tuning for DC motor speed controller based on genetic algorithm. Int. Rev. Autom. Control. **8**, 80–85 (2015). https://doi.org/10.15866/ireaco.v8i1.4839

99. Lotfy, A.; Kaveh, M.; Mosavi, M.R.; Rahmati, A.R.: An enhanced fuzzy controller based on improved genetic algorithm for speed control of DC motors. Analog Integr. Circuits Signal Process. (2020). https://doi.org/10.1007/s10470-020-01599-9
100. Goldstein, H.; Poole, C.; Safko, J.: Classical mechanics, 3rd ed. Am. J. Phys. **70**, 782–783 (2002). https://doi.org/10.1119/1.1484149
101. Lennard-Jones, J.E.: On the determination of molecular fields. Proc. R. Soc. A. **106**, 463–477 (1924)
102. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press, Cambridge (1992)
103. Kennedy, J.; Eberhart, R.: Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks. pp. 1942–1948. IEEE (1995)
104. Bayraktar, Z.; Komurcu, M.; Bossard, J.A.; Werner, D.H.: The wind driven optimization technique and its application in electromagnetics. IEEE Trans. Antennas Propag. **61**, 2745–2757 (2013). https://doi.org/10.1109/TAP.2013.2238654
105. Yang, X.S.; Deb, S.: Cuckoo search via Lévy flights. In: 2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009—Proceedings. pp. 210–214 (2009)
106. Zhang, J.R.; Zhang, J.; Lok, T.M.; Lyu, M.R.: A hybrid particle swarm optimization-back-propagation algorithm for feedforward neural network training. Appl. Math. Comput. **185**, 1026–1037 (2007). https://doi.org/10.1016/j.amc.2006.07.025
107. Mangasarian, O.L.; Wolberg, W.H.: Cancer Diagnosis via Linear Programming. University of Wisconsin-Madison Department of Computer Sciences, Madison (1990)