Sādhanā (2018) 43:2
https://doi.org/10.1007/s12046-017-0780-z

Elephant swarm water search algorithm for global optimization

S MANDAL

Department of Electronics and Communication Engineering, Global Institute of Management and Technology, Krishna Nagar 741102, India
e-mail: sudip.mandal007@gmail.com

MS received 21 March 2017; revised 11 June 2017; accepted 15 June 2017; published online 7 February 2018

Abstract. The rising complexity of real-life optimization problems has constantly inspired computer researchers to develop new efficient optimization methods. Evolutionary computation and metaheuristics based on swarm intelligence are very popular nature-inspired optimization techniques. In this paper, the author has proposed a novel elephant swarm water search algorithm (ESWSA) inspired by the behaviour of social elephants, to solve different optimization problems. This algorithm is mainly based on the water search strategy of intelligent and social elephants during drought. Initially, we perform preliminary parametric sensitivity analysis for our proposed algorithm, developing guidelines for choosing the parameter values in real-life problems. In addition, the algorithm is evaluated against a number of widely used benchmark functions for global optimizations, and it is observed that the proposed algorithm has better performance for most of the cases compared with other state-of-the-art metaheuristics. Moreover, ESWSA performs better during fitness test, convergence test, computational complexity test, success rate test and scalability test for most of the benchmarks. Next, ESWSA is tested against two well-known constrained optimization problems, where ESWSA is found to be very efficient in term of execution speed and best fitness. As an application of ESWSA to real-life problem, it has been tested against a benchmark problem of computational biology, i.e., inference of Gene Regulatory Network based on Recurrent Neural Network. It has been observed that the proposed ESWSA is able to reach nearest to global minima and enabled inference of all true regulations of GRN correctly with less computational time compared with the other existing metaheuristics.

Keywords. Elephant swarm water search algorithm (ESWSA); global optimization; constrained optimization; swarm intelligence; metaheuristic; gene regulatory network; recurrent neural network.

1. Introduction

Metaheuristics [1, 2], which are typically inspired by physical phenomena, animals' behaviours or evolutionary concepts, are becoming very popular over the last few years. In addition to the large number of theoretical works, metaheuristics have been applied to different real-life problems in various fields of study. The reasons behind popularity of metaheuristics are simplicity, flexibility, derivation-free mechanism and local optima avoidance capability [3]. First, metaheuristics are mostly based on very simple concepts of nature and easy to understand. The simplicity of metaheuristics helps researchers to simulate different natural phenomena, propose new metaheuristics, hybridize two or more metaheuristics for better efficiency or modify current metaheuristics. Second, flexibility refers to the quick applicability of metaheuristics to different real-life problems without any special changes in the structure of the algorithm since they presume optimization problems to be black boxes. For a metaheuristic, we need to be concerned only about the input(s) and output(s) of a problem. Normally, all researchers need to know how to represent a problem for metaheuristics by an objective function that will be minimized or maximized. Third, mostly all metaheuristics have derivation-free mechanisms since metaheuristics optimize problems stochastically. The metaheuristics process is initiated with random solution(s) called population, and calculation of derivative of search spaces is not necessary. These characteristics make metaheuristics greatly appropriate for real-life optimization problems with expensive or unknown derivative information. Finally, due to the stochastic nature of metaheuristics, they have superior capability to avoid local optimal point compared with conventional optimization methods and also help search the entire search space extensively. Usually, the search space of real-life optimization problems is very complex and unknown with lots of local optima; hence, metaheuristics are very efficient for optimizing these types of problems.

Although there are different types of metaheuristics, two phases of search process: exploration (global search) and exploitation (local search), are basic characteristics of all.

The exploration phase is the process of finding the promising area(s) of the overall search space. On the other hand, exploitation refers to the searching capability around the promising regions obtained in the exploration phase. Suitable balance between these two types of search phases is a very crucial task for any metaheuristic, which is normally obtained using different stochastic operators.

Generally, metaheuristics can be classified based on number of initial solutions into two groups: single-solution-based and population-based. In case of single solution (as an example: Simulated Annealing [4]) approaches, the optimization starts with a single candidate solution. This initial solution is then modified over the course of iterations. For population-based meta-heuristics, the search process is initialized with a random initial population (a set of solutions), and this population is changed during iterations. However, population-based metaheuristics have some advantages compared with single-solution-based algorithms: greater exploration capability, less chance to stick in local optima and ability to cover all promising search space for sharing information among the population. On the other hand, metaheuristics may be divided into three major classes on the basis of their nature: Evolutionary, Physics-based and Swarm Intelligence (SI) algorithms.

Usually, Evolutionary Algorithms (EA) are inspired by the concepts of evolution in nature, like mutation, etc. Some of the popular EAs are Genetic Algorithm (GA) [5, 6], Differential Evolution (DE) [7], Evolutionary Programming (EP) [8], Biogeography-Based Optimizer (BBO) [9], etc. Use of several operators like selection, crossover, mutation, etc. and computational deficiency are the main disadvantages of EA.

The second branch of metaheuristics is physics-based techniques. Such algorithms typically mimic physical rules, namely gravitational force, ray casting, electromagnetic force, inertia force, weights and so on. Some of the most popular algorithms are Big-Bang Big-Crunch (BBBC) [10], Gravitational Search Algorithm (GSA) [11], Black Hole (BH) algorithm [12], Galaxy-based Search Algorithm (GbSA) [13], etc. Although a large number of physics-based metaheuristics exist, they are not so popular due to the complex nature of these algorithms and search strategies are also not so good.

Finally, the most popular and efficient subclass of metaheuristics is the SI methods. According to Bonabeau *et al* [14], SI is "The emergent collective intelligence of groups of simple agents". These algorithms mostly mimic the social behaviour of swarms, herds, flocks or schools of insects and animals in nature where the search agents navigate using the simulated collective and social intelligence of creatures. Some of the advantages of SI algorithms are the following: easy implementation, fewer parameters to adjust, less operators to be used and storing capability of previous results. Most popular SI techniques are as follows: Particle Swarm Optimization (PSO) [15], Bat Algorithm (BA) [16], Cuckoo Search (CS) [17], Flower Pollination Algorithm (FPA) [18], Firefly Algorithm (FA) [19], Ant Colony Optimization (ACO) [20], Artificial Bee Colony (ABC) Optimization [21], Social Spider Algorithm (SSA) [22], etc. For example, PSO algorithm simulates the social behaviour or movement of particles such as bird flocking or fish schooling. It uses particle best and global best position to update the velocity and position of the particles, i.e., solutions. PSO is very popular for its good convergence speed and exploration capability. Moreover, it has few optimization parameters that need to be tuned properly. Many types of modified PSO [23–26] have already been proposed with advanced variants. BA searches the optimal value of a function on the basis of foraging of prey using echolocation behaviour of bats by updating frequency, velocity and position. However, BA requires many variables to be properly tuned for better search [16]. CS is based on brood parasitism of cuckoo birds, which reproduce their eggs by utilizing nests of others host birds, where the highest quality nests with eggs (i.e., best solutions) selected move over to the next generation. Due to the use of Levy flights in the algorithm, it is very efficient and performs superior than PSO [17] but requires more computational time. FPA is typically associated with the transfer of pollen for reproduction or flowering of plants by different pollinators such as insects. Due to long-distance pollinators and flower consistency [18], FPA can explore larger search space and converges more quickly. FA is based on gradual movement of fireflies towards brighter light during dark night. The main disadvantage of FA is that its computational complexity is very high and it requires lot of computational time [19]. Moreover, its convergence speed is not up to the mark. The social intelligence of ants in finding the shortest path between the nest and a source of food using pheromone is the main inspiration for ACO [20]. It can be used in several dynamic applications but its theoretical analysis is quite difficult as probability distributions change by iteration and sequences of random decisions. Moreover, time to convergence is also uncertain. On the other hand, ABC [21] mimics the collective behaviour of bees in finding food sources, i.e., honey. The advantages of ABC are robustness and fast calculation; however, it suffers from a major drawback, i.e., search space limited by initial solution. SSA algorithm [6] is based on the foraging behaviour of social spiders and the information-sharing foraging strategy. This is conceptually simple but consists of a large number of probabilistic variants.

The No Free Lunch (NFL) theorem [27] is worth mentioning here. Through the NFL theorem it was logically proved that there is no single metaheuristic that is best suited for solving all kinds of optimization problems; this implies that a particular algorithm might show very promising results on a set of problems, but the same metaheuristic might show worse results on a different set of problems. Therefore, proposing new metaheuristics and modification of current approaches (with good convergence speed, good accuracy, less computational time, less number

of parameters to be tuned, good exploration and exploitation capability) is still a very fascinating field of study to the computer science researchers.

This paper aims to introduce a new SI-based optimization algorithm, namely elephant swarm water search algorithm (ESWSA), which is based on water search strategy of elephants swarm and its application in real-life problems. The rest of the paper is organized as follows. In the next section, we discuss some social behaviour and communication techniques of elephants that would be of help for the proposed new ESWSA optimization technique. In section 3, the ESWSA metaheuristic is proposed and elaborated. Later, we perform experimentation for global optimization of several numerical benchmark functions and statistical analysis to validate the efficiency of the proposed algorithm. The results of ESWSA are also compared to those of several state-of-art techniques such as BA, CS, FPA and PSO. Results regarding the performances of ESWSA for two well-known constrained optimization problem in the field of engineering are given in section 5. Following this, ESWSA is applied to a real-life optimization problem; it is used to reconstruct or infer Gene Regulatory Network (GRN) from time-series gene expression data. Results analysis and discussion are given in section 7. Conclusion is given in section 8 followed by references.

2. Behaviours of elephants

Elephants are the largest living terrestrial animals [28] of the family Elephantidae, which are traditionally categorized in two species, the African and Asian elephants [29]. African elephants are 3–4 m tall and weigh 4,000–7,000 kg while Asian elephants are 2–3.5 m in height and weigh 3,000–5,000 kg [30]. Elephants are recognized by their large ears, long trunk, long tusks, large limbs and huge body structure. Elephants are often found to exist in a "fluid fission–fusion" social environment [31]. In natural habitat, elephants live in herds comprising 3–35 elephants. Number of elephants in one herd can be varied depending on weather and availability of food, water, etc. Living in groups is a social habit that pays an important role in protecting the herd members and finding resource for living [32]. Each elephant herd is guided by the biggest and oldest adult cow, while the adult bull lives only for breeding periodically. Meanwhile, young cows and children keep staying in the group.

Elephants are well known for their good memory [33] and exhibit advanced intelligence [34] like self-recognition, an indication of self-awareness and cognition that has also been demonstrated in some apes and dolphins [35]. One study of a captive female Asian elephant suggested that the animal was capable of learning and distinguishing between several visual and some acoustic discrimination pairs [36]. Elephants can also use tools in real life.

Social mammals like elephants have very advanced sensing and communication systems; they use of all of their senses of hearing, smell, vision, touch and exceptional ability to detect vibrations [37]. Among different communication techniques, seismic, acoustic and chemical communications are used for long-distance communication up to 10–12 km away. On the other hand, visual and tactile communications are considered as short-distance communication. For seismic communication, elephants are able to pick up the seismic signals (i.e., 10–40 Hz vibration on earth surface caused due to rumble and movement of elephants), to orient in the direction the vibrations come from and even to respond to them appropriately using their mechano-receptors [38] in the toes or feet and the tip of an elephant's trunk, which are extremely sensitive to vibrations. Elephants produce a broad range of acoustic (i.e., sound) signal from a low of 5 Hz to a high of over 10,000 Hz (generally called as infrasound) rumbles [38]. These animals use their ears like a parabola, scanning back and forth while remaining still to listen to low-frequency sounds from other distant elephants. The trunk can amplify audible sounds. The use of chemical or olfactory cues is central to communication between elephants [38]. They often raise their trunks up to smell the air, or use the tips of their trunks to explore the ground or tree as well as to sniff other elephants. The trunk is a fusion of nose and upper lip and is made up of millions of olfactory receptor cells. On the other hand, the eyesight of elephants is said to be good in dull light, but considerably reduced in bright light, reaching a maximum range of 46–100 m [38]. Elephants use many different displays and gestures to communicate with one another. In case of visual communication, heads, eyes, mouth, ears, trunk and even their whole body are used for signalling messages to each another or to other species. Elephants are extremely tactile animals. They touch each other purposefully using their trunk, tusks, feet, tail, etc. Tactile communication between elephants occurs to express aggressive, defensive, sexual and exploratory behaviour [38], etc.

During dry weather, lack of water creates an overwhelming problem for all animals, especially for huge animals like elephants. In spite of the ruthless living conditions created during drought, elephants are equipped well to survive by finding water resources. Elephants may utilize one or more communication system or methods to search the water resources, depending on their current conditions. An adult elephant drinks approximate 40–60 gallons/day on average from rivers, water holes, lakes, ponds, etc. If the area is very dry and the duration of drought is very long, elephants can migrate to other surrounding areas where water is available or plentiful. Typically, they migrate as far as required to find food and water. If the size of drought area is small, the elephants do not go far away usually. When the drought area is very large, elephants can travel to more remote areas in search of water, where they stay until the rainy season. They dig with their feet, trunks and tusks

into dry stream earths or beds or other spots to discover ample amount of water source lurking below the surface. Elephants also help others animals by creating or finding new water sources. Elephants show an unselfish and social behaviour during adverse situation like drought. They communicate and share information among different elephant groups for better water searching, which is another secret of their longevity.

3. Elephant swarm water search algorithm

Now we can idealize some characteristics of water search strategies for elephants so as to develop ESWSA. For simplicity, following four idealized rules are used to describe our proposed algorithm.

(i) Elephants roam around in search of water during drought in several groups, which is called elephant swarm. Each group consists of a number of elephants and all groups (i.e., elephant swarm) work together to find water. The leader of each group (oldest cow) is responsible for taking decision about the movement of the group to search the best water resource. For an optimization problem, each elephant group is identified by its particular velocity and position whereas each elephant group of the swarm is similar to the solution of the corresponding problem.

(ii) Whenever an elephant group finds some water resource, the leader communicates (via seismic, acoustic, chemical, visual and tactile communication) with the other groups of the swarm about the quantity and quality of the water. For a maximization problem, the fitness value and the objective function are directly proportional to the quantity and quality of the water resources. Better water level denotes better solution.

(iii) Elephants have very sharp memory. Each elephant group can remember the best location of water supply that was discovered by its own group (local best solution) so far and the best location of water source so far (global best solution), which was discovered by the whole swarm or all groups. Based on these memories (solutions), the elephant group can move from one point to another, i.e., velocity and position of each elephant group are updated gradually during the searching process according to some rules (the rules for global search, local search and position updating are described later). Long-distance and short-distance communication techniques of elephants are dominant for global and local search, respectively.

(iv) Water search in local and global area is controlled by a probabilistic constant called switching probability $p \in [0, 1]$. The leader of the group takes probabilistic decision to switch between local search and global search during water search. Due to the physical proximity and other factors such as attenuation of signal from large distance, local water search can have a significant fraction p in the overall searching activities.

For d-dimensional optimization problem, the position of i-th elephant group of a swarm (consisting of N particles, i.e., number of elephant groups) at t-th iteration is given as $X^t_{i,d} = (x_{i1}, x_{i2}, \ldots, x_{id})$ and the velocity is represented by $V^t_{i,d} = (v_{i1}, v_{i2}, \ldots, v_{id})$. Locally best solution by i-th elephant group at current iteration is given as $P^t_{best,i,d} = (P_{i1}, P_{i2}, \ldots, P_{id})$ and global best solution is denoted by $G^t_{best,d} = (G_1, G_2, \ldots, G_d)$. Initially, the elephant groups (position and velocity) are randomly placed throughout the search space. As iteration proceeds, the velocity and position of the elephants are updated according to some rules.

Most water finding activities can occur at both local and global scales. In practice, adjacent water searches in the not-so-far-away neighbourhood are more likely to be executed by the group than those far away. For this, a constant known as switching probability p is used to switch between global and local water searches. It is assumed that if value of a random variable is greater than p, common global water search will be performed, else intense local water search will be executed. This randomized condition helps reducing the probability of sticking at local optima. Global and local best solutions are updated after each iteration. As iteration proceeds, the velocities of the particles are updated in different ways for global and local search according to following equations depending on the value of parameter p:

$$V^{t+1}_{i,d} = V^t_{i,d}\omega^t + rand(1,d) \odot \left(G^t_{best,d} - X^t_{i,d}\right) \quad \text{if } rand > p[\text{for global search}] \tag{1}$$

$$V^{t+1}_{i,d} = V^t_{i,d}\omega^t + rand(1,d) \odot \left(P^t_{best,i,d} - X^t_{i,d}\right) \quad \text{if } rand \leq p[\text{for local search}] \tag{2}$$

where $rand(1, d)$ generates a d-dimensional array of random values within [0,1]; $\odot$ denotes element-wise multiplication; ω^t is the inertia weight at current iteration to balance between exploration and exploitation. Then, position of an elephant group is modified according to following equation.

$$X^{t+1}_{i,d} = V^{t+1}_{i,d} + X^t_{i,d} \tag{3}$$

t_{max}, X_{max} and X_{min} denote the values of maximum iteration number, upper boundary and lower boundary of the positions, respectively. After completion of all iterations, the elephants gradually update their position and reach the best water resource position, i.e., best solution of the optimization problem. Thus, the pseudo-code of the proposed ESWSA is given as follows.

Start ESWSA
Define $N, d, t_{max}, X_{max}, X_{min}, p$ and objective function f; // **Inputs**

for i=1 to N // **Initializations**
  Initialize $X_{i,d}$ and $V_{i,d}$;
  $P_{best,i,d} = X_{i,d}$;
end;

Evaluate fitness value $f(X_{i,d})$for all N elephant group positions; // **Evaluations and find best**
$G_{\text{best ,d}}$=Min(f);
Assign value of ω^t according to the weight update rules [Eq. (6)]; //**Assignment of ω^t**

for $t = 1$ to t_{max} //**Start the iteration**
  for i=1 to N
    if $rand > p$ //**Global Search**
      global water search or update the elephant velocity $V_{i,d}$ using Eq. (1);
    else //**Local Search**
      local water search or update the elephant velocity $V_{i,d}$ using Eq. (2);
    end if;
    update the position $X_{i,d}$using Eq. (3); //**Update positions**
    evaluate fitness value for $f(X_{i,d})$;
    if $f(X_{i,d}) < f(P^t_{best,i,d})$ //**Update current best**
      $P^t_{best,i,d}$=$X_{i,d}$;
    end if;
    if $f(P^t_{best,i,d}) < f(G^t_{best,d})$ //**Update global best**
      $G^t_{best,d} = P^t_{best,i,d}$;
    end if;
  end for;
  $X^* = G^t_{best,d}$;
end for; //**End Iteration**

Return X^* and $f(G^t_{best,d})$; //**Output**
End ESWSA

Our proposed ESWSA optimization technique is slightly different from standard PSO metaheuristic. In case of PSO, the velocity update formula, i.e., new search direction is always effected by three components namely: current velocity ($v^t_{i,d}$), current particle memory influences ($P^t_{best,i,d}$) and swarm memory influences ($G^t_{best,d}$) [39]. Particle memory influence is associated with current best position of a particle and swarm memory influence is associated with the global best position among all particles. Hence, it may be considered that the particles memory influence is responsible for local search and swarm memory influence is responsible for global search of the optimization. It is interesting to note that random parameters are incorporated along with the global search or local search terms so that chance of sticking at local optima for the metaheuristics can be reduced.

However, in case of ESWSA, the velocity update formula, i.e., new search direction is effected by either current velocity and current elephant memory influences or current velocity and swarm memory influences depending on probabilistic value (p), also known as switching probability.

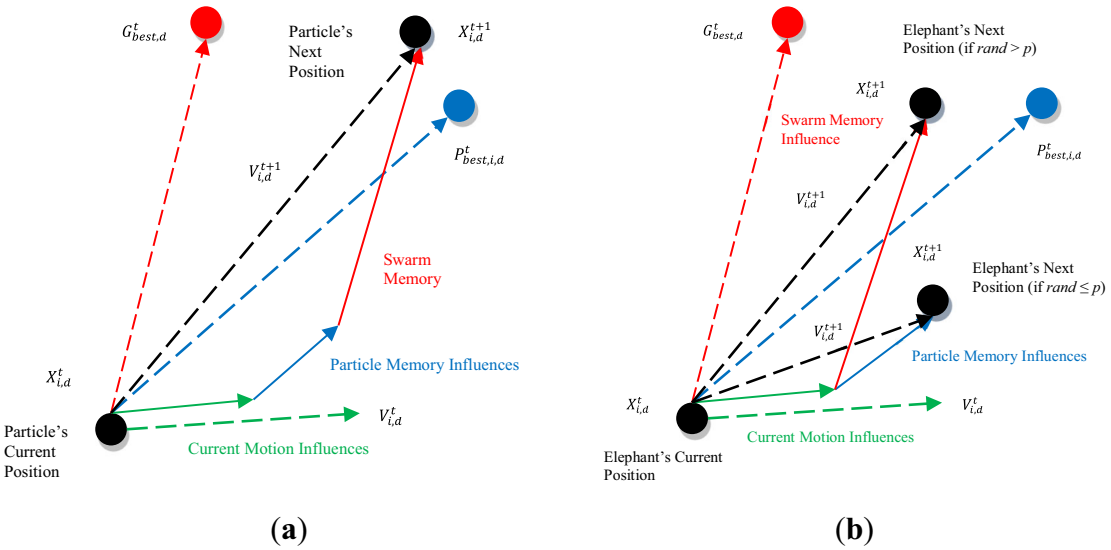

Figure 1. Depiction of velocity and position update in (**a**) PSO [39] and (**b**) proposed ESWSA.

Hence, depending on the value of p, velocity will be updated on the basis of current elephant best or global best solution, i.e., ESWSA switch from local search to global search or vice versa on the basis of switching probability. During local search of ESWSA, velocity is updated according to current elephant best position, i.e., search around the current best solution. During global search of ESWSA, velocity is updated according to global best position of elephants, i.e., search towards the global best solution. However, in case of PSO, the search is simultaneously effected by both current and global best solution. This is the main difference between ESWSA and PSO with respect to formal hypothesis. Figure 1a and b shows the depiction of velocity and position update procedure for PSO and ESWSA, respectively, during iteration that determines the new search direction.

4. Study of ESWSA for global optimization

To validate the proposed ESWSA algorithm, we have chosen 10 different benchmark functions [40, 41], which are De Jong, Rosenbrock, Schwefel P2.22, Noisy Quadric, Rotating hyper-ellipsoid, Ackley, Easom, Griewank, Rastrigin and Alpine. Among these functions, first five are unimodal and the rest are multimodal in nature. The details of these functions, respective search ranges of variables and corresponding global minima points are presented in table 1.

4.1 *Parametric sensitivity of ESWSA*

Selection of suitable parameters of ESWSA is a very important task to achieve best performances for numerical and real-world optimization problems. Use of trial and error scheme or random selection of parameters for a real-life problem may lead to high computational cost and less efficiency. Normally, researchers have used several benchmark functions to tune the parameters of an optimization algorithm to accomplish the best performances and they can also help in revealing some important characteristics about the parametric sensitivity of the algorithm. We carry out extensive simulations on our benchmark functions, which cover a wide range of optimization problems. Thus, the derived rules of thumb can be expected to give generally good performance on unknown problems. In EWSA, following parameters are employed to guide searching behaviour.

- Inertia weight (ω^t): It is a deterministic parameter (use some pre-defined rules to change the parameter values throughout the search) that denotes inertia weight of velocity at current iteration.
- Switching probability (p): It is a fixed parameter (value remains constant throughout the whole search) that denotes the switching probability between local and global water search.
- Maximum iteration number (t_{max}) and number of population (N) are also fixed parameters of ESWSA optimization.

4.1a. *Performance on the basis of inertia weight* (ω^t): Many strategies have been proposed for updating inertia weight during the course of iteration for PSO. However, we consider only three types of inertia update technique for ESWSA. In case of Constant Inertia Weight (CIW) technique [42], value of inertia weight is constant (normally 0.5 is preferred) throughout the iteration. Thus, CIW can be described using the following equation:

$$\omega^t = constant. \tag{4}$$

Table 1. Different benchmark functions for global optimization.

Function name	Function	Range of search	Global minima point
De Jong	$f_1(x) = \sum_{i=1}^{d} x_i^2$	$-5.12 \le x_i \le 5.12$	$f_* = 0$ at (0,0,...0)
Rosenbrock	$f_2(x) = \sum_{i=1}^{d-1}[(x_i - 1)^2 + 100(x_{i+1} - x_i^2)^2]$	$-5 \le x_i \le 5$	$f_* = 0$ at (1,1,...,1)
Schwefel P2.22	$f_3(x) = \sum_{i=1}^{d}\|x_i\| + \prod_{i=1}^{d}\|x_i\|$	$-10 \le x_i \le 10$	$f_* = 0$ at (0,0,...0)
Noisy Quadric	$f_4(x) = \sum_{i=1}^{d} i x_i^4 + rand$	$-1.28 \le x_i \le 1.28$	$f_* = 0$ at (0,0,...0)
Rotating hyper-ellipsoid	$f_5(x) = \sum_{i=1}^{d}(\sum_{j=1}^{i} x_j)^2$	$-100 \le x_i \le 100$	$f_* = 0$ at (0,0,...0)
Ackley	$f_6(x) = -20\exp\left[\frac{1}{5}\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right] - \exp\left[\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right] + 20 + e$	$-30 \le x_i \le 30$	$f_* = 0$ at (0,0,...0)
Easom	$f_7(x) = (-1)^{d+1}\prod_{i=1}^{d}\cos(x_i)\exp\left[-\sum_{i=1}^{d}(x_i - \pi)^2\right]$	$-30 \le x_i \le 30$	$f_* = -1$ at $(\pi, \pi, \ldots, \pi)$
Griewank	$f_8(x) = \frac{1}{400}\sum_{i=1}^{d} x_i^2 - \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$-600 \le x_i \le 600$	$f_* = 0$ at (0,0,...0)
Rastrigin	$f_9(x) = 10d + \sum_{i=1}^{d}[x_i^2 - 10\cos(2\pi x_i)]$	$-5.12 \le x_i \le 5.12$	$f_* = 0$ at (0,0,...0)
Alpine	$f_{10}(x) = -\sum_{i=1}^{d}\|x_i\sin(x_i) + 0.1x_i\|$	$-100 \le x_i \le 100$	$f_* = 0$ at (0,0,...0)

In case of Random Inertia Weight (RIW) [16], the value of inertia weight is selected in a random manner and it is very efficient to find out the optima in a dynamic system. For RIW, the value of inertia weight is assigned using the following equation:

$$\omega^t = 0.5 + rand/2 \tag{5}$$

where *rand* is a function that generates random number within [0, 1].

Linearly Decreasing Inertia Weight (LDIW) [15, 43, 44] is a very popular and efficient technique in improving the fine-tuning characteristics of the PSO, where the value of inertia weight depends linearly on the iteration number. In case of LDIW, the value of ω is linearly decreased from an initial large value (ω_{max}) to a final small value (ω_{min}) according to the following equation:

$$\omega^t = \omega_{max} - \left\{\frac{\omega_{max} - \omega_{min}}{t_{max}}\right\}t \tag{6}$$

where t is iteration index and t_{max} denotes maximum number of iterations.

Now, these three techniques were applied on 10-dimensional benchmark problems to observe the impact of these strategies on ESWSA. For this, a swarm of 50 elephant groups (population) and 1000 maximum iterations are considered. Value of switching probability is set as 0.5. All the techniques were simulated using Matlab 7.6 with 2 GB RAM, a Dual Core processor and Windows7 operating System. Each function is tested 50 times for each of the inertia weight update strategies. Resultant median of best solution and deviation from it for each benchmark function and each strategy are shown in figure 2 using boxplots.

It can be clearly seen that, LDIW performed better than other strategies as the median of best solution for LDIW is always nearer to the global minima and also smaller than the other two. Moreover, the variation in output is minimum for LDIW. Therfore, we shall use LDIW technique for the rest of our experiment and simulation.F

4.1b *Performance on the basis of switching probability (p)*: Now, ESWSA along with LDIW is applied against all afore-mentioned 10-dimensional benchmark functions for different values of p with the same number of iteration and population. The value of p is selected from the set {0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99}. The experiment is performed 50 times for each of the functions. The performance evaluation criteria are minimum fitness value, mean of fitness values, median of fitness values and standard deviation of the same. Table 2 shows the values of p that yield the minimum fitness, mean, median and standard deviation. It is observed that for different functions, the best solution is achieved for different values of p. Therefore, we observed the maximum number of occurrences for a particular value of p such that the best performance can be achieved. For an example, the minimum fitness is obtained for benchmark function number 7 for the value $p = 0.6$. It can be also observed that $p = 0.6$ gives better performance for most of the cases under different evaluation criteria. Therefore, it can be concluded that $p = 0.6$ is the most suitable for numerical and real-life problems of optimization. Moreover, from this point we shall consider the value $p = 0.6$ for further simulation using the proposed ESWSA optimization technique.

4.1c *Performance on the basis of population and maximum iteration number*: Next, performance of ESWSA optimization is observed for different values of population and maximum iteration number of the algorithm. For each function, value of population and maximum iteration number are selected from the sets {25, 50, 75, 100} and {1000, 2000, 3000, 4000, 5000}, respectively. We use the LDIW technique and value of p is 0.6 for 10-dimensional function minimization problems. The experiment is

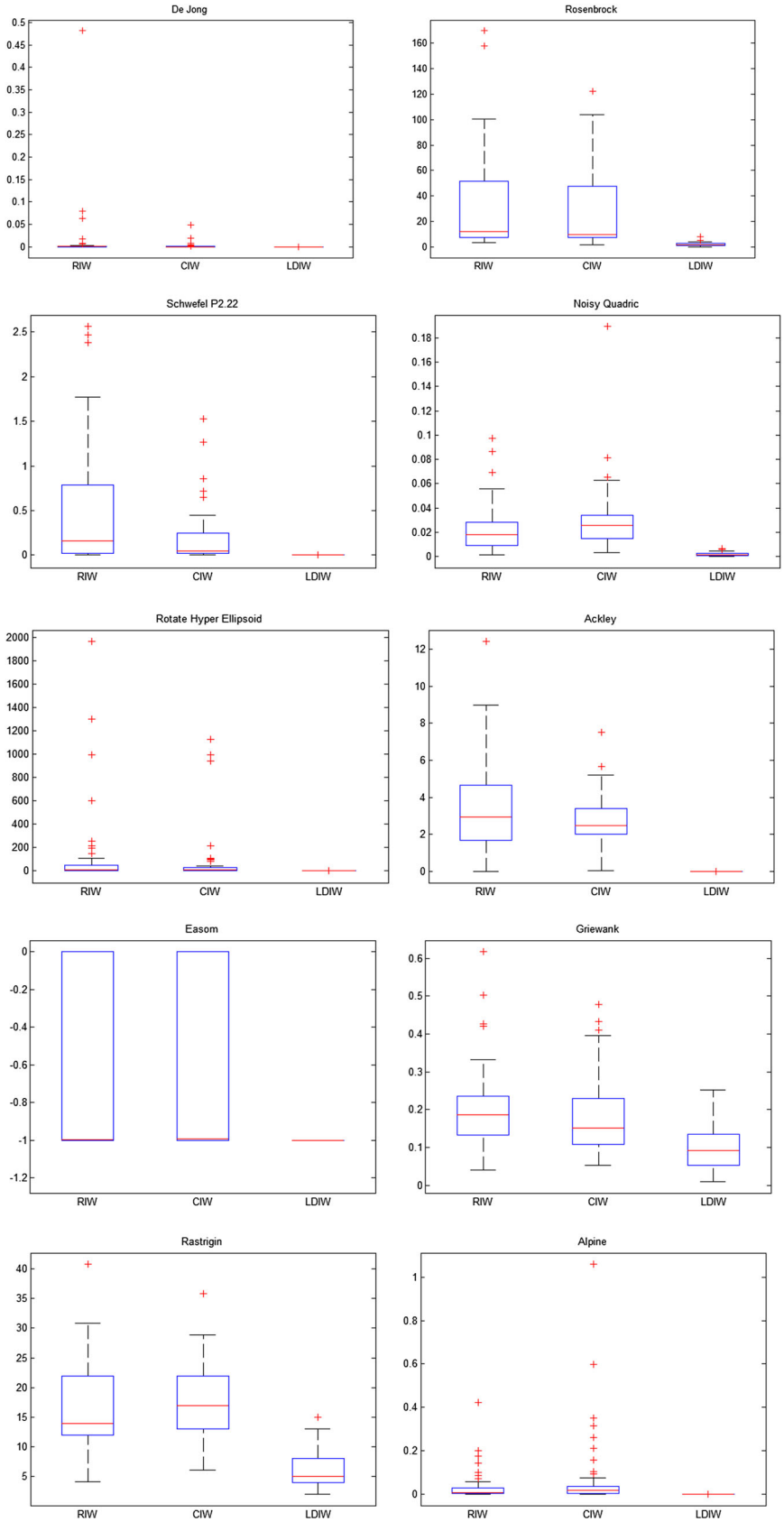

Figure 2. Box plot of raw simulation results for different benchmark functions.

Table 2. Performance of ESWSA for different values of p.

Function	Minimum fitness occurred at	Minimum mean fitness occurred at	Minimum median fitness occurred at	Minimum standard deviation occurred at
De Jong	$p = 0.6$	$p = 0.6$	$p = 0.6$	$p = 0.6$
Rosenbrock	$p = 0.6$	$p = 0.6$	$p = 0.6$	$p = 0.6$
Schwefel 2.22	$p = 0.6$	$p = 0.7$	$p = 0.7$	$p = 0.7$
Noise Quadric	$p = 0.6$	$p = 0.6$	$p = 0.6$	$p = 0.6$
Rotating hyper-ellipsoid	$p = 0.6$	$p = 0.7$	$p = 0.6$	$p = 0.7$
Ackley	$p = 0.6$	$p = 0.6$	$p = 0.6$	$p = 0.6$
Easom	$p = 0.6$	$p = 0.6$	$p = 0.6$	$p = 0.6$
Griewank	$p = 0.1$	$p = 0.6$	$p = 0.6$	$p = 0.7$
Rastrigin	$p = 0.5$	$p = 0.9$	$p = 0.9$	$p = 0.7$
Alpine	$p = 0.5$	$p = 0.7$	$p = 0.8$	$p = 0.7$
Maximum occurrences	7 ($p = 0.6$)	6 ($p = 0.6$)	7 ($p = 0.6$)	5 ($p = 0.6$ or $p = 0.7$)

performed 50 times for each case and table 3 shows the mean value of the best fitness for each function for a set of population and maximum iteration number. It is observed that mean fitness is quite satisfactory for all functions except Rosenbrock and Rastrigin although values of best fitness were very small for these two functions. For all unimodal functions (except Rosenbrock), mean value decreases considerably if population and maximum iteration number increase. Moreover, for multimodal function, increment in maximum iteration number has less effect on mean fitness. However, if we increase population number, the mean fitness value decreases considerably. However, increase in population and iteration number will lead to more computational time. Therefore, observing the following table, $N = 50$ and $t_{max} = 2000$ are preferred in this research work as well as for other real-life optimization problems as it will balance accuracy and computational time.

4.2 *Comparison with other state-of-art optimization techniques*

In this section, the simulation results of ESWSA on the different benchmark functions described in table 1. Moreover, we perform comparison among ESWSA and other algorithms and statistical analysis of the simulation results. For comparison purpose, we have selected some well-known optimization methods, namely BA [16, 45, 46], CS [17, 47, 48], FPA [18, 49] and PSO [15, 43, 44]. For all algorithms, population and maximum iteration number are set to 50 and 2000, respectively. Search space is restricted to 10, i.e., we have considered 10-dimensional function optimization problems. Each algorithm is executed 50 times for each function. The parameters setting for each algorithm in the comparison is described as follows:

1. For ESWSA, value of switching probability (p) is set to 0.6 and the inertia weight factor decreases linearly from 0.9 to 0.4.
2. For BA, loudness decreasing factor (α), pulse rate decreasing factor (γ), minimum and maximum frequency are set to 0.9, 0.9, 0 and 1, respectively, based on earlier work [16, 45, 46].
3. For CS, value of discovery rate of alien eggs (p_a) is set to 0.25, the same as in previous work [17, 47, 48].
4. For FPA, value of probability switch (p) is fixed to 0.8 using the guidelines provided by the reference [18, 49].
5. For PSO, acceleration constants (C_1 and C_2) are both set to 2 and the inertia weight factor decreases linearly from 0.9 to 0.4 as mentioned by the author [15, 43, 50].

In these numerical experimentations, we have tested and compared the efficiency of the proposed algorithm on the basis of some criteria such as fitness test, reliability test, convergence test and computational complexity, which are described in following subsections.

4.2a *Fitness test*: Final output or fitness value of an optimization algorithm is the most important criterion to prove its efficiency. Using the afore-mentioned parameters setting, we optimized each benchmark function. Here, we observed 3 criteria (output), namely best (minimum) fitness, worst (maximum) fitness and median of fitness, which are obtained after 50 program runs. Comparative studies based on these criteria are shown in table 4, where most excellent values are shown in italic letters.

From table 4, it can be observed that ESWSA can reach global minima point for all functions except Rastrigin with satisfactory accuracy. For the Rastrigin function, only PSO was able to reach the global minima point. Moreover, for benchmark function 7, ESWSA is able to give the best (minimum) fitness and these values are far better than those from other algorithms. BA is the least efficient in this case.

Table 3. Performance of ESWSA for different values of population and maximum iteration number.

Function name	Population and iteration	$t_{max} = 1000$	$t_{max} = 2000$	$t_{max} = 3000$	$t_{max} = 4000$	$t_{max} = 5000$
		Mean fitness value (output minima)				
De Jong	$N = 25$	3.75E−23	2.17E−46	4.82E−67	1.80E−84	7.61E−106
	$N = 50$	2.04E−35	6.40E−71	9.63E−105	1.34E−132	8.06E−168
	$N = 75$	4.19E−47	7.84E−89	6.27E−132	2.40E−178	5.95E−210
	$N = 100$	3.28E−57	5.06E−104	1.30E−148	7.56E−196	1.20E−241
Rosenbrock	$N = 25$	4.10E+00	1.74E+00	1.30E+00	6.74E−01	5.70E−01
	$N = 50$	1.83E+00	6.02E−01	5.02E−01	4.63E−01	4.41E−01
	$N = 75$	9.79E−01	6.93E−01	6.06E−01	2.68E−01	1.02E−01
	$N = 100$	1.23E+00	5.23E−01	3.51E−01	2.75E−01	4.09E−01
Shewefel P2.22	$N = 25$	7.99E−09	9.46E−17	1.46E−23	4.72E−28	5.44E−36
	$N = 50$	7.48E−15	9.15E−25	1.19E−38	2.56E−48	6.62E−59
	$N = 75$	1.88E−17	2.14E−34	1.47E−47	3.83E−59	7.34E−80
	$N = 100$	1.16E−18	1.80E−36	4.37E−53	9.38E−76	4.61E−88
Noisy Quadric	$N = 25$	4.40E−03	1.91E−03	1.38E−03	1.06E−03	7.66E−04
	$N = 50$	1.55E−03	8.08E−04	5.24E−04	3.84E−04	3.31E−04
	$N = 75$	1.03E−03	5.16E−04	3.55E−04	2.83E−04	1.95E−04
	$N = 100$	7.56E−04	3.80E−04	2.96E−04	2.17E−04	1.36E−04
Rotating hyper-ellipsoid	$N = 25$	1.56E−20	9.39E−41	2.89E−66	1.78E−83	1.65E−103
	$N = 50$	1.31E−31	5.54E−71	2.57E−97	6.04E−129	9.47E−168
	$N = 75$	9.79E−42	1.26E−113	2.53E−128	8.04E−170	1.57E−202
	$N = 100$	1.38E−50	7.46E−95	1.87E−148	1.36E−198	4.21E−240
Ackley	$N = 25$	2.18E−01	4.62E−02	2.31E−02	2.31E−02	4.58E−15
	$N = 50$	6.43E−15	4.51E−15	4.58E−15	4.30E−15	4.44E−15
	$N = 75$	5.36E−15	4.44E−15	4.30E−15	4.30E−15	4.23E−15
	$N = 100$	4.80E−15	4.44E−15	4.09E−15	4.09E−15	4.23E−15
Easom	$N = 25$	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00
	$N = 50$	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00
	$N = 75$	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00
	$N = 100$	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00	−1.00E+00
Griewank	$N = 25$	1.11E−01	9.92E−02	8.78E−02	9.19E−02	8.25E−02
	$N = 50$	8.11E−02	7.96E−02	7.37E−02	6.46E−02	6.16E−02
	$N = 75$	8.28E−02	6.40E−02	6.01E−02	6.26E−02	5.85E−02
	$N = 100$	6.79E−02	6.43E−02	5.85E−02	5.25E−02	6.59E−02
Rastrigin	$N = 25$	8.02E+00	7.36E+00	5.53E+00	5.05E+00	5.27E+00
	$N = 50$	6.17E+00	4.38E+00	3.90E+00	3.48E+00	3.18E+00
	$N = 75$	4.24E+00	3.88E+00	2.85E+00	2.83E+00	2.41E+00
	$N = 100$	3.90E+00	2.85E+00	2.69E+00	1.79E+00	2.05E+00
Alpine	$N = 25$	1.28E−04	1.10E−09	3.64E−10	2.55E−13	5.21E−15
	$N = 50$	9.79E−08	2.59E−15	3.59E−07	6.66E−15	3.70E−15
	$N = 75$	6.17E−07	2.46E−15	1.84E−15	2.57E−15	1.86E−15
	$N = 100$	2.61E−15	1.83E−15	2.93E−15	1.70E−15	2.83E−15

Next, we have made a comparison based on worst (maximum) fitness, whose value should be as less as possible. From table 5, it can be observed that the proposed ESWSA is able to achieve the minimum value of worst fitness for benchmark function 7. Only CS and PSO are able to achieve minimum value of worst fitness for 2 cases.

Next, we observe the median of 50 best fitness values corresponding to 50 runs. It is clearly observed (table 6) that median values for ESWSA are better than those from other algorithms for all benchmark functions except Rastrigin and Rosenbrock. In case of Rastrigin and Rosenbrock, CS performs better.

4.2b *Reliability test*: An optimization algorithm should always reach nearest to global minima as close as possible, i.e., it should be successful and reliable in every single run. However, due to random nature of the metaheuristic, the output may differ. Therefore, in this subsection, we have tested reliability of ESWSA and also make a comparison with the other algorithms on the basis standard deviation and success rate.

From table 7, it is clearly noticed that ESWSA gives output with very small standard deviation for all functions except Rosenbrock and Rastrigin whereas other algorithms yield comparatively larger deviations than that of ESWSA.

Table 4. Comparative study based on best fitness.

	Minimum fitness				
Function	ESWSA	BA	CS	FPA	PSO
De Jong	*1.45E−94*	2.53E−07	2.36E−33	1.96E−08	3.87E−54
Rosenbrock	*3.77E−04*	3.09E−02	4.88E−08	7.97E−02	7.26E−02
Schwefel 2.22	*2.32E−39*	1.34E−03	5.80E−12	4.44E−02	2.62E−28
Noisy Quadric	*9.98E−05*	2.21E−02	7.81E−04	1.36E−03	4.80E−04
Rotating hyper-ellipsoid	*5.77E−91*	2.62E+01	2.89E−30	3.84E−04	1.25E−51
Ackley	*4.44E−15*	2.01E+00	1.18E−11	2.21E−02	*4.44E−15*
Easom	*−1.00E+00*	*−1.00E+00*	*−1.00E+00*	*−1.00E+00*	*−1.00E+00*
Griewank	1.48E−02	5.16E−02	*8.56E−03*	3.89E−02	1.07E−02
Rastrigin	9.95E−01	8.95E+00	1.12E+00	8.76E+00	*7.50E−10*
Alpine	*4.81E−112*	3.04E−04	1.58E−02	2.27E−01	3.73E−36

Table 5. Comparative study based on worst fitness.

	Maximum fitness				
Function	ESWSA	BA	CS	FPA	PSO
De Jong	*2.69E−69*	8.24E−07	9.35E−31	2.07E−07	6.03E−29
Rosenbrock	5.63E+00	4.25E+00	*3.08E−04*	4.35E+00	8.23E+00
Schwefel 2.22	*4.47E−23*	4.53E+01	1.71E−10	5.20E−01	8.60E−08
Noisy Quadric	*2.05E−03*	5.72E−01	4.23E−03	1.25E−02	8.11E−03
Rotating hyper-ellipsoid	*8.92E−70*	2.61E+04	3.51E−28	2.78E−03	3.35E−25
Ackley	*7.99E−15*	7.67E+00	2.55E−08	2.81E−01	1.16E+00
Easom	*−1.00E+00*	−4.66E−24	*−1.00E+00*	*−1.00E+00*	*−1.00E+00*
Griewank	2.12E−01	1.53E+00	5.64E−02	*1.15E−01*	1.94E−01
Rastrigin	1.29E+01	6.77E+01	*4.33E+00*	2.17E+01	8.95E+00
Alpine	*2.26E−14*	4.13E+00	2.19E−01	1.00E+00	1.27E−02

Table 6. Comparative study based on median fitness.

	Median fitness				
Function	ESWSA	BA	CS	FPA	PSO
De Jong	*1.46E−83*	6.03E−07	2.92E−32	7.10E−08	1.39E−45
Rosenbrock	1.06E−01	3.78E−02	*2.90E−06*	1.98E+00	4.49E+00
Schwefel 2.22	*3.00E−33*	2.16E−03	3.97E−11	9.64E−02	5.52E−21
Noisy Quadric	*7.95E−04*	1.20E−01	2.36E−03	5.87E−03	1.44E−03
Rotating hyper-ellipsoid	*5.79E−79*	5.89E+03	3.17E−29	1.02E−03	1.59E−40
Ackley	*4.44E−15*	5.09E+00	4.47E−10	7.06E−02	*4.44E−15*
Easom	*−1.00E+00*	−6.58*E*−09	*−1.00E+00*	*−1.00E+00*	*−1.00E+00*
Griewank	*7.01E−02*	4.88E−01	2.78E−02	7.24E−02	7.81E−02
Rastrigin	3.98E+00	2.24E+01	*2.79E+00*	1.52E+01	3.98E+00
Alpine	*1.44E−15*	3.49E−01	9.44E−02	5.50E−01	5.85E−10

For 7 cases, it achieves the least deviation where its nearest competitor CS has the least standard deviation for 4 cases. This proves that ESWSA gives output with less variation in output, i.e., it is a more reliable optimization process than other algorithms. Again, BA has the largest standard variation for all benchmark functions.

A simulation is considered as successful if and only if the best-found fitness value is smaller than or equal to the successful threshold. The overall success rate is calculated by the number of successful runs under a specific success threshold divided by the total number of runs. Thus a larger overall success rate implies a more reliable algorithm.

Table 7. Comparative study based on standard deviation.

	Standard deviation				
Function	ESWSA	BA	CS	FPA	PSO
De Jong	*3.8E−70*	1.5E−07	1.4E−31	4.2E−08	8.5E−30
Rosenbrock	1.3E+00	1.4E+00	*5.0E−05*	1.1E+00	2.1E+00
Schwefel 2.22	*6.3E−24*	8.3E+00	3.0E−11	8.7E−02	1.5E−08
Noisy Quadric	*4.5E−04*	1.1E−01	8.6E−04	3.0E−03	1.4E−03
Rotating hyper-ellipsoid	*1.8E−70*	7.4E+03	6.5E−29	6.7E−04	4.7E−26
Ackley	*5.0E−16*	1.2E+00	4.4E−09	6.0E−02	2.3E−01
Easom	*0.0E+00*	1.4E−01	*0.0E+00*	3.8E−06	*0.0E+00*
Griewank	4.0E−02	2.7E−01	*1.1E−02*	1.7E−02	3.5E−02
Rastrigin	2.4E+00	1.2E+01	*8.6E−01*	3.2E+00	2.0E+00
Alpine	*4.2E−15*	1.1E+00	4.5E−02	1.8E−01	2.0E−03

Table 8. Comparative study based on success rate.

	Success rate (%)				
Function	ESWSA	BA	CS	FPA	PSO
De Jong	100	100	100	100	100
Rosenbrock	48	86	100	2	2
Schwefel 2.22	100	74	100	52	100
Noisy Quadric	100	34	100	100	100
Rotating hyper-ellipsoid	100	0	100	100	100
Ackley	100	0	100	64	96
Easom	100	2	100	100	100
Griewank	74	2	100	94	78
Rastrigin	0	0	0	0	4
Alpine	100	20	54	0	100

Here, if the absolute value of best-found fitness is smaller than 0.1 we considered it to be successful for all benchmark functions except the Easom function. In case of Easom function, if the fitness value is less than –0.9 and greater than –1.1, then it will be considered as a successful attempt. From table 8, it is observed that the proposed ESWSA has 100% success rate for all functions except Griewank, Rosenbrock and Rastrigin. It is interesting to note that CS has higher successful rate than ESWSA as they have 85% and 82% average success rate, respectively. For Rastrigin, almost every algorithm has 0% success rate except PSO, which has only 4% success rate. BA is the worst algorithm among these.

4.2c *Convergence test*: The final result comparison cannot completely describe the searching performance of an algorithm. Hence, we further conduct a convergence test on the compared algorithms on each 10-D benchmark function. Next, we have tested the convergence speed of the proposed ESWSA algorithm and compare with others. For this purpose, we have calculated the mean of best-found fitness for all 50 runs at each iteration index. Then, we plot them for all algorithms and for all functions, which are shown in figure 3. It is observed that ESWSA converges faster than other algorithms do except for Rotating hyper-ellipsoid, Griewank and Noisy Quadric. For these 3 functions, CS performs better than ESWSA. PSO has the worst convergence speed. Overall, ESWSA has satisfactory convergence speed, which is highly desirable for an optimization algorithm. For most of the cases, ESWSA converges within 250 iterations.

4.2d *Computational efficiency test*: Besides the previous tests, the computational complexity is also a major factor for evaluating the efficiency of an evolutionary computation algorithm. For this purpose, we studied the average execution time taken by each algorithm for each benchmark function, which indicates the computational efficiency of the algorithm. Table 9 shows a comparative study based on average execution time. It is clearly shown that ESWSA is faster than all others algorithms for all functions except Rosenbrock. PSO and ESWSA have almost same performance regarding computational complexity. CS takes the largest time to execute. This proves the efficiency of our proposed ESWSA optimization technique.

4.2e *Overall performance*: Now we summarize the performance of ESWSA based on afore-mentioned evaluation criteria and compare with other techniques. For this purpose, we assigned a performance score against each algorithm for each criterion. The value of this score is calculated as the ratio of number of cases (functions) where an algorithm achieves the best result (criterion) to the total number of cases. Table 10 shows the comparative study based on these scores to evaluate overall efficiency of our proposed algorithm. It is clearly noticed from table 10 that for all conditions, ESWSA performed better than others. Only in case of success rate, CS and ESWSA have similar performance. Therefore, it can be concluded that ESWSA is one of the most efficient optimization techniques for global optimization.

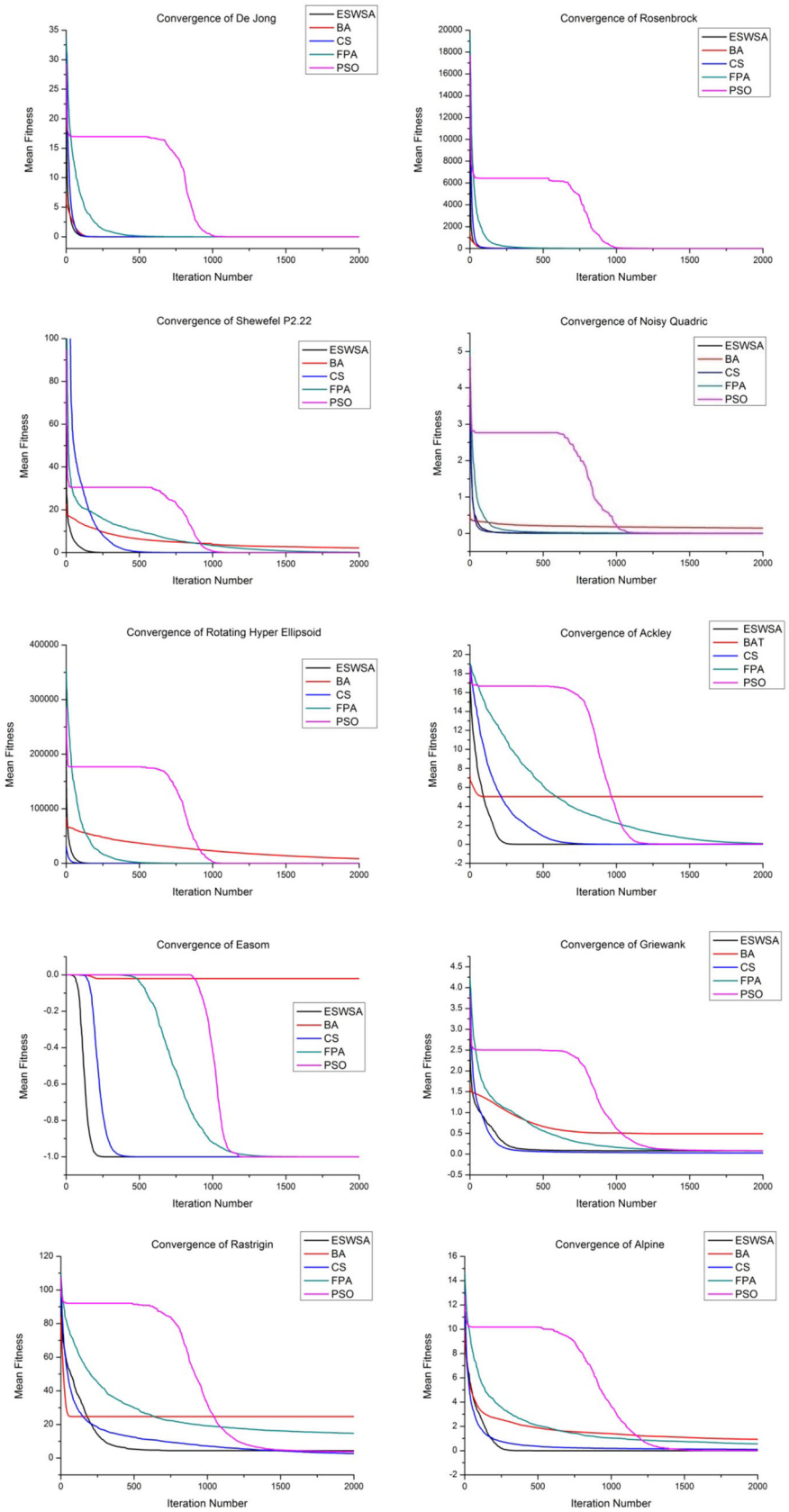

Figure 3. Convergences of different benchmark functions.

Table 9. Comparative study based on average execution time.

Function	Average execution time (s)				
	ESWSA	BA	CS	FPA	PSO
De Jong	*0.740*	2.879	17.115	6.542	0.742
Rosenbrock	0.834	3.124	11.193	6.537	*0.832*
Schwefel 2.22	*0.794*	3.215	7.878	4.561	0.846
Noisy Quadric	*1.030*	3.456	8.374	7.161	1.140
Rotating hyper-ellipsoid	*0.913*	3.308	7.797	4.999	0.983
Ackley	*1.916*	6.178	22.149	9.552	2.438
Easom	*2.131*	4.964	21.030	7.951	2.199
Griewank	*1.958*	4.927	19.960	7.794	2.294
Rastrigin	*0.802*	3.078	7.525	6.627	0.912
Alpine	*0.793*	3.259	24.064	7.592	0.873

Table 10. Comparative study based on score.

Criterion	Score				
	ESWSA	BA	CS	FPA	PSO
Best fitness	0.7	0.1	0.3	0.1	0.3
Worst fitness	0.7	0	0.2	0.1	0.1
Median fitness	0.8	0	0.3	0.1	0.2
Standard deviation	0.7	0	0.4	0.1	0.1
Success rate	0.7	0.1	0.7	0.4	0.5
Convergence	0.7	0	0.3	0	0
Execution time	0.9	0	0	0	0.1

4.3 *Scalability of ESWSA*

In addition to the 10-dimensional benchmark function tests, we also performed a series of simulations on high dimensional (both 30-D and 50-D) benchmarks to test the scalability of ESWSA. Here, we have used $N = 100$ and $t_{max} = 5000$ for simulation and corresponding results are shown in table 11. It has been observed that ESWSA gives very good accuracy and success rate for De Jong, Schwefel 2.22, Noisy Quadric, Rotating hyper-ellipsoid, Griewank and Alpine functions for both cases. In general, it can be stated that performance of ESWSA is degraded for high-dimensional multimodal function optimization. For unimodal and large-dimensional problem, ESWSA can provide satisfactory accuracy. However, if we increase population number (elephant swarm) and iteration number, it is expected that accuracy will be increased but computational time will also increase consequently. Therefore, for a real-life optimization problem, sufficient population and iteration numbers are needed to be considered for better accuracy and efficiency.

5. Application of ESWSA for constrained optimization

In order to evaluate the optimization power of the proposed ESWSA, in the presence of constraints, two engineering design problems are considered and solved, including three-bar truss and tension/compression spring. These two problems have different natures of objective functions, constraints and decision variables. In this paper, for each problem the constraints are directly handled. It indicates that if a solution cannot satisfy all constraints altogether, it will be not considered as a feasible solution and will be abandoned consequently.

Table 11. Scalability test of ESWSA.

Function name	Min.		Success rate (%)	
	$d = 30$	$d = 50$	$d = 30$	$d = 50$
De Jong	1.12E−47	4.78E−18	100	100
Rosenbrock	0.000102	1.534299	14	0
Schwefel 2.22	3.53E−18	2.05E−06	100	92
Noisy Quadric	0.00107	0.00949	100	100
Rotating hyper-ellipsoid	1.61E−44	5.81E−13	100	100
Ackley	1.322669	1.496879	0	0
Easom	−1	−3.1E−09	96	0
Griewank	0	1.24E−14	100	98
Rastrigin	200	400	0	0
Alpine	5.22E−15	1.45E−06	100	94

5.1 *Three-bar truss design problem*

In case of three-bar truss design problem [51, 52], the primary objective is the minimization of the volume of a three-bar truss (statistically loaded) subject to the stress (σ) constraints, length (l) and pressure (P) on every truss member by optimizing the areas of cross section (i.e., x_1 and x_2). Figure 4 shows a schematic of three-bar truss design problem. This constrained optimization problem consists of three nonlinear inequality constraints and two continuous decision variables and corresponding nonlinear fitness function is given as follows:

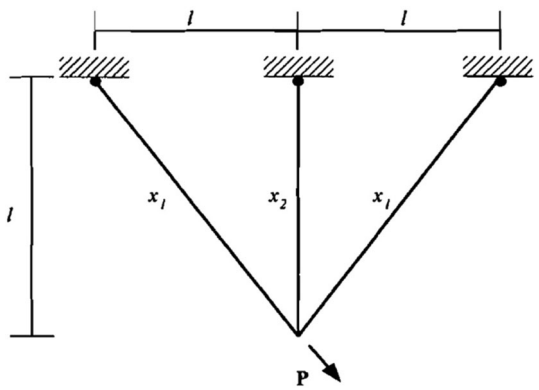

Figure 4. Three-bar truss design problem [52].

Table 12. Comparison for constrained optimization of three-bar truss design.

	Metaheuristics				
Criterion	ESWSA	BA	CS	FPA	PSO
Worst fitness	263.89585	263.89610	263.89584	263.89584	263.94879
Best fitness	263.89584	263.89585	263.89584	263.89584	263.89625
Mean fitness	263.89585	263.89592	263.89584	263.89584	263.91243
Median fitness	263.89584	263.89589	263.89584	263.89584	263.90597
Standard deviation	0.000003	0.000074	0.000000	0.000000	0.017683
Execution time	0.84299	1.56144	3.58392	3.52433	0.85408

$$\begin{aligned}
&\text{Min.} \quad f(x) = \left(2\sqrt{2}x_1 + x_2\right)l \\
&\text{s.t.} \\
&g_1(x) = \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0 \\
&g_2(x) = \frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \le 0 \\
&g_3(x) = \frac{1}{\sqrt{2}x_2 + 2x_1}P - \sigma \le 0 \\
&0 \le x_i \le 1; i = 1, 2 \\
&l = 100\,\text{cm}, P = 2\,\text{kN/cm}^2, \sigma = 2\,\text{kN/cm}^2.
\end{aligned} \tag{7}$$

Table 12 shows the performance of different metaheuristics for constrained optimization of three-bar truss design problem [52]. We have considered six criteria for the comparison. We consider a population of 50 and 2000 iterations for each metaheuristic. The program is executed 10 times and corresponding best, worst, mean, median fitness, standard deviation and execution time are presented in table 12. It is seen that all the metaheuristics can reach close to the optimal point 263.89584 as reported in the paper [52]. Best fitness of ESWSA, CS and FPA are the same and their standard deviation is also very small (for ESWSA it is 3×10^{-6}). However, the standard deviation for PSO and BA is significant whereas CS and FPA have almost zero deviation. This indicates that the output of PSO may not be able to reach close to the optimal point for all cases. One of the most important advantages for ESWSA is that it is the most computationally efficient compared with all other metaheuristic, whereas, the time requirement for CS and FPA is the highest (almost three times).

5.2 *Tension/compression spring design problem*

The objective of tension spring design problem [52] is to minimize the weight of a tension/compression spring (f) with respect to three nonlinear and one linear inequality constraints according to Eq. (8). Figure 5 shows a schematic of tension spring design problem. It has three continuous decision variables, namely, wire diameter (d or x_1), mean coil diameter (D or x_2) and number of active coils (P or x_3).

Figure 5. Schematic of tension/compression spring design problem [52].

$$\begin{aligned}
&\text{Min.} \quad f(x) = (x_3 + 2)x_2x_1^2 \\
&\text{s.t.} \\
&g_1(x) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0 \\
&g_2(x) = \frac{4x_2^2 - x_1x_2}{12566\left(x_2x_1^3 - x_1^4\right)} + \frac{1}{5180x_1^2} - 1 \le 0 \\
&g_3(x) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\
&g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \le 0 \\
&0.05 \le x_1 \le 2, 0.25 \le x_2 \le 1.3, 2 \le x_3 \le 15.
\end{aligned} \tag{8}$$

Table 13 shows the performance of different metaheuristics for constrained optimization of tension/compression spring design problem [52]. For the earlier optimization setting, it has been seen that all the metaheuristics can reach close to the optimal point 0.012665 as reported in the paper [52]. Standard deviation of ESWSA and CS is very small whereas it is almost zero for FPA. However, the standard deviation for PSO and BA is significant, which indicates that the output of PSO and BA may not be able to reach close to the optimal point for all cases. One of the most important advantages for ESWSA is that its computational time is the least compared with others.

6. Application of ESWSA for inference of GRN

Correct inference of genetic regulations [53] inside a cell from the biological database like time-series microarray data [54] (which contains gene expression level of many genes at different time instances) is one of the greatest challenges in post genomic era for the biologist and

Table 13. Results for constrained optimization of tension/compression spring design problem [52].

	Metaheuristics				
Criterion	ESWSA	BA	CS	FPA	PSO
Worst fitness	0.012713	0.016747	0.013596	0.012666	0.013317
Best fitness	0.012666	0.012667	0.012666	0.012665	0.012682
Mean fitness	0.012680	0.013519	0.012985	0.012665	0.013012
Median fitness	0.012674	0.012756	0.012934	0.012665	0.013022
Standard deviation	0.000015	0.001502	0.000306	0.000000	0.000219
Execution time	0.903812	1.649376	3.749864	3.566125	0.909446

computer researchers. Recurrent Neural Network (RNN) [55] is one of the most popular and simple approaches to model the dynamics of gene expression as well as to infer correct dependencies among genes, i.e., GRN. In this paper, the proposed ESWSA is tested against benchmark small-scale artificial genetic network inference problem as an application of ESWSA in the field of computational biology and bioinformatics.

A GRN [56] is represented by a directed graph where nodes of the denote genes and regulatory interactions between genes are denoted by directed edges. In canonical RNN model [57, 58] the gene's regulations are expressed by the following tightly coupled architecture where it is assumed that each of the total N neurons in the output unit $e_i(t+\Delta t)$ is a gene expression value of next time instant, and the neurons in the input units $e_i(t)$ are the gene expression of present state for the same genes; thus they interact with each and everyone:

$$e_i(t+\Delta t) = \frac{\Delta t}{\tau_i} f\left(\sum_{j=1}^{N} w_{i,j} e_j(t) + \beta_i\right) + \left(1 - \frac{\Delta t}{\tau_i}\right) e_i(t), \quad i = 1, 2, \ldots, N \tag{9}$$

where $f()$ is a nonlinear function (usually the sigmoid function is used, where $f(z) = 1/1 + e^{-z}$); $w_{i,j}$ represents the type and strength of the regulatory interaction from j-th gene towards i-th gene. Positive (negative) value of $w_{i,j}$ represents activation (repression) control of gene-j on gene-i; $w_{i,j} = 0$ implies that gene-j has no regulatory control on gene-i; β_i represents the basal expression level and τ_i denotes the decay rate parameter of the i-th gene. Δt is incremental time instance (in this work it is set to 1). Hence, the discrete form of the RNN model for GRN can be described by the following set of $N(N+2)$ unknown parameters $\Omega = \{w_{i,j}, \beta_i, \tau_i\}$, which are called as RNN model parameters, where $i, j = 1, 2, \ldots, N$.

Usually, the inference of RNN-based genetic network from time-series gene expression data is obtained by optimizing the values of RNN parameters with the help of a metaheuristic so that training or learning error is minimized. All metaheuristics use an objective function or a fitness function to measure the goodness of a solution. The most common estimation criterion is squared error [57, 58], which is defined as follows:

$$f = \sum_{k=1}^{M} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(e_{cal,k,i,t} - e_{exp,k,i,t}\right)^2 \tag{10}$$

where N is the number of genes, T is the number of time instances of the gene expression data, M is number of datasets, $e_{cal,k,i,t}$ is numerically calculated gene expression value of k-th dataset at time t of i-th gene using the set of obtained parameters of RNN model and $e_{exp,k,i,t}$ is the actual gene expression level of k-th dataset at time t of i-th gene; f denotes total squared error between the calculated and the observed gene expression data. Therefore, RNN modelling is a nonlinear function optimization problem to discover the optimal RNN parameter by minimizing the fitness function or square error so that calculated gene expression data fit best with the observed gene expression data. Moreover, the genetic network inference problem is decoupled [57], i.e., it is divided into several sub-problems corresponding to each gene to reduce large dimensionality of search space.

6.1 *Experimental results for inference of small-scale GRN*

To explore the effectiveness of ESWSA optimization for inference of RNN-based GRN, initially a benchmark small artificial regulatory network is chosen that contains four genes with simple regulatory dynamics. The parameters of this artificial GRN are presented in table 14 and the network is shown in figure 6. There are totally 8 regulations in the network. The artificial time-series data were generated by solving the set of differential equations (9) using the following RNN parameters and the initial values of these sets were selected randomly. In real life, these time-series data could be obtained by different biological experiments. The number of time instances per dataset is 50 and number of dataset is 4. Hence, for this work, there were totally 200 data points for each gene. For each gene, 6 parameters $\Omega = \{w_{i,j}, \beta_i, \tau_i\}$ need to be identified using ESWSA,

Table 14. Actual RNN model parameters for small artificial genetic network.

$w_{i,j}$	1	2	3	4	β_i	τ_i
1	20	−20	0	0	0	10
2	15	−10	0	0	−5	5
3	0	−8	12	0	0	5
4	0	0	8	−12	0	5

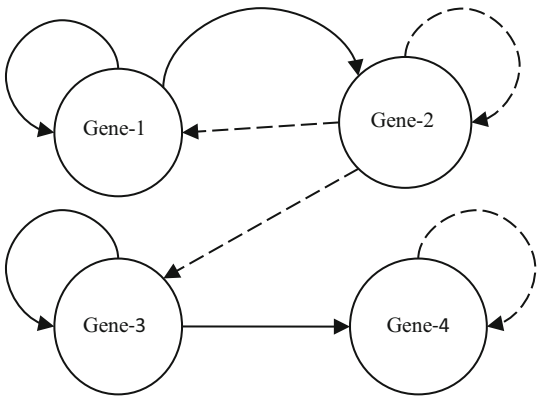

Figure 6. Original small-scale GRN.

where these parameters are considered as the position of elephants groups. The search space was selected as $w_{i,j} \in [-30,30]$, $\beta_i \in [-10,10]$ and $\tau_i \in [0,20]$, the same as in earlier work [48, 50, 57, 58]. Here, we have used ESWSA to get the decoupled RNN model parameters.

First, the most important performance criterion is measured from network structure point of view where inferred network is compared to the original network structure with respect to edge connectivity. Now, the sensitivity (S_n), specificity (S_P), F-score (F) and accuracy (ACC) of the reconstructed network are defined as follows:

$$S_n = \frac{TP}{TP + FN} \tag{11}$$

$$S_p = \frac{TN}{TN + FP} \tag{12}$$

$$F = \frac{2 * TP}{2 * TP + FP + FN} \tag{13}$$

$$ACC = \frac{TP + TN}{TP + FP + FN + TN} \tag{14}$$

where TP (True Positive) denotes the number of correctly predicted regulations, TN (True Negative) represents the number of properly predicted non-regulations, FP (False Positive) denotes the number of incorrectly predicted regulations and FN (False Negative) represents the number of falsely predicted non-regulations by the inference algorithm. Moreover, inferred values of the parameters should not deviate much from the original one as its sign and magnitude may affect the connectivity of reconstructed genetic network. Therefore, another performance measurement parameter, Inferred Parametric Error (IPE), is defined that measures the deviation in the magnitude of obtained parameters from original one:

$$IPE = \sum_{i,j=1}^{N} \left| w_{i,j}^{exp} - w_{i,j}^{cal} \right| + \sum_{i=1}^{N} \left| \beta_i^{exp} - \beta_i^{cal} \right| + \sum_{i=1}^{N} \left| \tau_i^{exp} - \tau_i^{cal} \right| \tag{15}$$

where $w_{i,j}^{exp}, \beta_i^{exp}, \tau_i^{exp}$ are the actual values of RNN parameters and $w_{i,j}^{cal}, \beta_i^{cal}, \tau_i^{cal}$ are the calculated values of the same. However, due to random and stochastic nature of metaheuristic, outputs may be slightly varied for different runs and this may lead to different network topologies. Therefore, in this work, each algorithm is executed 10 times and final regulations of genes are obtained by taking their mean. Other optimization parameters for ESWSA, BA, CS, FPA and PSO remain the same as those in section 4.2.

Table 15 shows a comparative study on the performance of proposed ESWSA algorithm and other state-of-art techniques for inference of small-scale artificial GRN. It can be seen that ESWSA, CS and FPA are able to detect all TPs and do not include any FPs. Therefore, values of S_n, S_P, F and ACC are 1 for all of these cases. However, BA and PSO both include 3 FPs. It can be also seen that PSO has the least computational time whereas FPA has the maximum. Moreover, it is interesting to observe that ESWSA has the least training error, computational time and IPE while PSO has the largest training error. Therefore, overall, ESWSA is preferable if we want to balance accuracy of inference capability and computational time of respective algorithm.

Table 15. Comparative study for inference of small artificial GRN using RNN.

Process	TP	TN	FP	FN	S_n	S_p	F	ACC	Average training error	IPE	Average computational time (s)
ESWSA	8	8	0	0	1	1	1	1	1.09×10^{-11}	0.0040	416.94
BA	6	5	3	2	0.7	0.63	0.71	0.69	0.3253	90542	508.40
CS	8	8	0	0	1	1	1	1	3.63×10^{-10}	0.0087	1104.23
FPA	8	8	0	0	1	1	1	1	0.0061	39	9773.48
PSO	6	5	3	2	0.7	0.63	0.71	0.69	0.2320	6.6×10^{22}	420.63

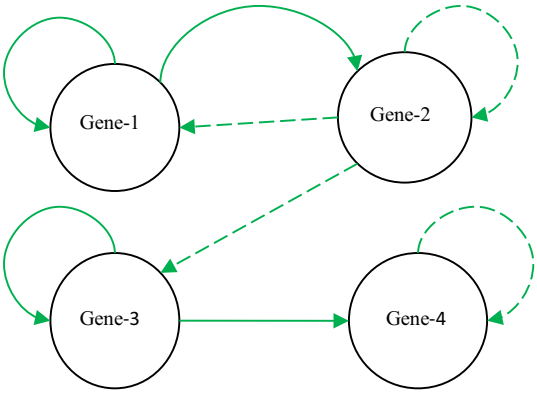

Figure 7. Inferred small-scale GRN by ESWSA, CS and FPA.

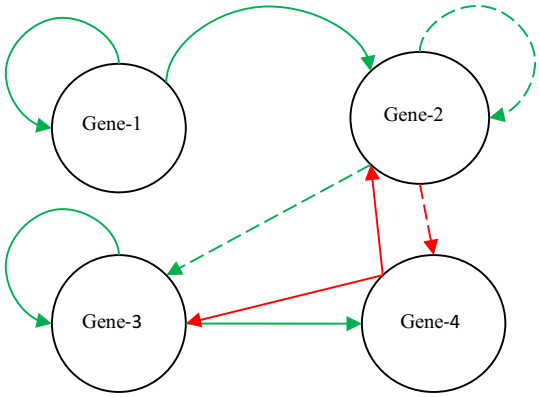

Figure 8. Small-scale GRN inferred by BA.

Figure 6 shows the original small-scale (4 genes) artificial GRN, where solid arrow-lines denote activation and dashed lines denote inhibition. Figure 7 depicts the GRN reconstructed from time-series gene expression data using ESWSA, CS and FPA techniques. It has been observed that all of these three metaheuristics are able to identify all true regulations (both activations and inhibitions, denoted by green coloured arrow) without including any false positive. However, in case of BA and PSO, both metaheuristics are able to identify only 6 true regulations but also include 3 false regulations (shown by red coloured arrow) in the inferred network. Moreover, for each case, BA and PSO have 2 *FN*s. Figures 8 and 9 show the small-scale GRN inferred by BA and PSO, respectively.

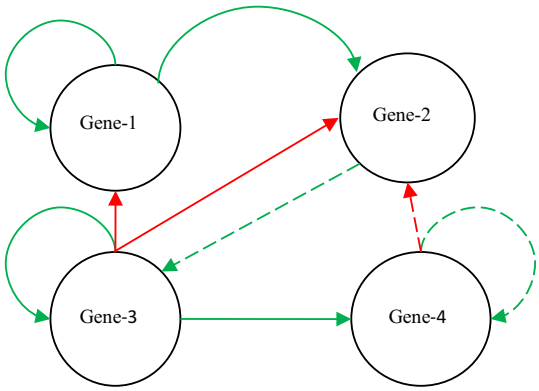

Figure 9. Small-scale GRN inferred by PSO.

7. Results, analysis and discussion

Both PSO and ESWSA are dependent on two pieces of information: current particle or elephant best and swarm or global best. This allows greater diversity and exploration for both PSO and ESWSA. However, due to combined effect of particle best and global best, PSO may not reach as close as the ESWSA can reach to the global optimal point. We can see from tables 4–6 that ESWSA performs the best and PSO holds the second position in terms of accuracy, i.e., minimum, maximum and median fitness of numerical benchmark functions. From the obtained accuracy of different numerical benchmark functions and also GRN inference problem, it can be concluded that ESWSA and PSO used information about particle and global best positions in the search direction, and performed better compared with other existing metaheuristics like CS, FPA and BA. From tables 7 and 8, it is also very interesting to observe that ESWSA is better in term of success rate and standard deviation. It indicates that ESWSA is able to reach nearer to the global optima most of the times. For constrained optimization, ESWSA has similar accuracy as those of CS and FPA (see tables 12 and 13). Table 15 also supports this observation when we tried to infer RNN-based GRN using ESWSA. Training error is the least for ESWSA. However, performance of PSO is not up to mark for constrained optimization.

One of the main advantages of PSO is that the momentum effects (using inertia weight) on particle movement can allow faster convergence (e.g., when a particle is moving in the direction of a gradient) and more variety/diversity in search trajectories [39]. We have used similar approaches for ESWSA, namely LDIW [15], to update the velocity. This strategy leads to better convergence of ESWSA compared with other methods. From figure 2, it can be seen that ESWSA attends faster convergence for 7 cases of numerical functions.

Now, if we consider the case of ESWSA and PSO, both use velocity update formula but in different ways. To use the particle memory or swarm memory or both, we need to perform point-wise multiplication (see Eqs. (1) and (2)). However, to determine the search direction, PSO always use both memories whereas ESWSA uses either particle memory or swarm memory depending on the probability switch. Obviously, the point-wise multiplication for ESWSA is almost half of the required multiplication for PSO during velocity update. Hence, computational efficiency of ESWSA is better than that of PSO. On the other

hand, as CS and FPA both use Levy's Distribution for global search, the required computational time is more for CS and FPA. Moreover, as CS needs to perform more operations like selection of nest using random walk or Levy flight, empty a nest, etc., its computational time is the highest. As BA uses more number of tuning parameters and equations for updating the velocity, position, frequency, etc. during iteration, computational efficiency of BA is better than that of CS and FPA but worse than that of ESWSA and PSO. All these observations are made from table 9 for global optimization of different benchmark functions. In case of constrained optimizations (tables 12 and 13), we have also seen that computational time of ESWSA is lower (comparable or not much difference) than that of PSO whereas other methods required more time to execute. CS and FPA needed almost three times the computational time for constrained optimizations. In case of GRN inference problem, ESWSA needs the least computational time (table 12) whereas PSO required more execution time than that of ESWSA but less than that of other metaheuristics.

It is interesting to note that the *IPE* of GRN inference problem using PSO is very high; this indicates that PSO has been stuck at local minima or values are trying to reach beyond the boundary of the variables. Due to combined effects of particle and swarm memories, large step may be achieved during velocity update and there is always a chance of going beyond limit. Therefore, PSO may sometimes fail to reach nearer to the global optima. For this reason, PSO is not able to learn the exact value of regulatory parameters (i.e., GRN) and consequently parametric error is huge for inference of RNN-based GRN. However, ESWSA achieved the least *IPE* due to the use of particle and swarm memories alternatively (see the pseudo-code of ESWSA). From figures 8 and 9, it can be observed that BA and PSO are not suitable for learning of optimal parameters from time-series data, i.e., GRN inference, due to inclusion of *FP*s in the reconstructed network. CS and FPA have similar performance in terms of accuracy for GRN inference, but their *IPE*, computational time and training error are comparatively higher than those of ESWSA. For constrained optimizations, we have seen similar results, where standard deviation is very significant for PSO. It also indicates that the output of PSO may fluctuate significantly as it is not so robust against sticking at local minima.

Table 10 describes or summarizes overall findings of the exposition of this work. ESWSA is found to be the most efficient optimization in terms of best (minimum) fitness, worst (maximum) fitness, median fitness, standard deviation, success rate, convergence speed and computational time. It is also suitable for large-dimensional optimization problems. In case of constrained optimizations, ESWSA gives best results in terms of computational time and best fitness. As an application of ESWSA, we have tested ESWSA on a benchmark small-scale GRN inference problem, where it also performed better in many ways (accuracy and output training error) than the other state-of-art techniques. Hence, it can be concluded that ESWSA is the most suitable optimization technique compared with the other methods like BA, CS, FPA and PSO.

8. Conclusion

In this paper, the author has proposed a novel ESWSA to solve global optimization problems. This algorithm is based on the water search strategies of social elephants swarm with the help of their different short- and long-distance communication techniques. The velocity and positions of elephants swarms are gradually updated according to the current velocity and either local best or global best position depending on a probabilistic condition. ESWSA is theoretically very simple and relatively easy to implement as there are very few parameters that need to be adjusted during optimization. ESWSA can deal with several different continuous optimization problems and has the potential to be employed to solve real-world problems with satisfactory accuracy.

Initially, in order to evaluate the efficiency of ESWSA, a set of 10 benchmark functions are used, which cover a large variety of different global optimization problem types. Also, compared ESWSA with the state-of-the-art optimization algorithms, namely, BA, CS, FPA and PSO based on different criteria like best-found fitness, worst fitness, mean fitness, median fitness, standard deviation, success rate, convergence speed, computational efficiency, etc. The results showed that the performance of ESWSA is outstanding compared with the afore-listed algorithms and conditions for most of the benchmark functions. This conclusion was validated by both the simulation results and the statistical analysis of the simulation data.

Next, tested ESWSA for two well-known constrained optimization problems, namely three-bar truss and tension spring design problem. It has been observed for both cases that ESWSA performance is outstanding in terms of computational time, best fitness and standard deviation.

As a part of this work, ESWSA has been applied against a benchmark problem of computational biology, i.e., inference of GRN based on RNN. The objective was to learn the parameters of RNN accurately from time-series gene expression dataset by minimizing the training error. It is observed that the proposed ESWSA is able to reach nearest to global minima point and can infer all true regulations of GRN correctly in less computational time compared with the other existing metaheuristics.

In future, research on ESWSA can be carried out in the following areas: scheme, algorithm and real-world application. The local water search and global water search scheme in the current ESWSA may be further improved using advanced and hybrid optimization algorithms. In terms of algorithm research, development of adaptive or self-adaptive strategies for tuning of ESWSA parameters is

also another interesting direction of research, which will reduce the time in tuning parameters. Lastly, it will be very fascinating to test the effectiveness of the proposed algorithm for more different real-world applications.

References

[1] Gandomi A H, *et al* 2013 *Metaheuristic applications in structures and infrastructures*, 1st ed. Elsevier, USA

[2] Bianchi L, *et al* 2009 A survey on metaheuristics for stochastic combinatorial optimization. *Nat. Comput.: Int. J.* 8(2): 239–287

[3] Mirjalili S, Mirjalili S M and Lewis 2014 A grey wolf optimizer. *Adv. Eng. Softw.* 69: 46–61, doi: http://dx.doi.org/10.1016/j.advengsoft.2013.12.007

[4] Kirkpatrick S, Gelatt Jr C D and Vecchi M P 1983 Optimization by simulated annealing. *Science,* 220: 671–680

[5] Lawrence D D1991 *Handbook of genetic algorithms*, 1st ed. New York: Van Nostrand Reinhold

[6] Deb K 1999 An introduction to genetic algorithms. *Sadhana* 24: 293–315, https://doi.org/10.1007/bf02823145

[7] Storn R and Price K 1997 Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.* 11: 341–359

[8] Yao X, Liu Y and Lin G 1999 Evolutionary programming made faster. *IEEE Trans. Evol. Comput.* 3: 82–102

[9] Simon D 2008 Biogeography-based optimization. *IEEE Trans. Evol. Comput.* 12: 702–713

[10] Erol O K and Eksin I 2006 A new optimization method: big bang–big crunch. *Adv. Eng. Softw.* 37: 106–111

[11] Rashedi E, Pour H N and Saryazdi S 2009 GSA: a gravitational search algorithm. *Inf. Sci.* 179: 2232–2248

[12] Hatamlou A 2012 Black hole: a new heuristic optimization approach for data clustering. *Inf. Sci.* 222: 175–184

[13] Hosseini H S 2011 Principal components analysis by the galaxy-based search algorithm: a novel metaheuristic for continuous optimization. *Int. J. Comput. Sci. Eng.* 6: 132–140

[14] Bonabeau E, Dorigo M and Theraulaz G 1999 *Swarm intelligence: from natural to artificial systems*. USA: Oxford University Press

[15] Eberhart R C and Shi Y H 2000 Comparing inertia weights and constriction factors in particle swarm optimization. In: *Proceeding of the IEEE Congress on Evolutionary Computation*, pp. 84–88

[16] Yang X S 2010 A new metaheuristic bat-inspired algorithm. In: *Proceedings of Nature Inspired Cooperative Strategies for Optimization (NISCO 2010)*, vol. 284, pp. 65–74

[17] Yang X S and Deb S 2010 Engineering optimisation by cuckoo search. *Int. J. Math. Model. Num. Optim.* 1(4): 330–343

[18] Yang X S 2012 Flower pollination algorithm for global optimization. *Proceedings of Unconventional Computation and Natural Computation*, *Lecture Notes in Computer Science*, vol. 7445, pp. 240–249

[19] Yang X S 2010 Firefly algorithm, stochastic test functions and design optimization. *Int. J. Bio-Inspired Comput.* 2: 78–84

[20] Dorigo M, Maniezzo V and Colorni A 1996 Ant system: optimization by a colony of cooperating agents. *IEEE Trans. Syst. Man Cybern. Part B* 26(1): 29–41

[21] Karaboga D and Basturk B 2007 A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *J. Global Optim.* 39(3): 459–471

[22] Yu J J Q and Li V O K 2015 A social spider algorithm for global optimization. *Appl. Soft Comput.* 30: 614–627, https://doi.org/10.1016/j.asoc.2015.02.014

[23] De A, Kumar S K, Gunasekaran A and Tiwari M K 2017 Sustainable maritime inventory routing problem with time window constraints. *Eng. Appl. Artif. Intell.* 61: 77–95, doi: https://doi.org/10.1016/j.engappai.2017.02.012

[24] Alexandridis A, Chondrodima E and Sarimveis H 2016 Cooperative learning for radial basis function networks using particle swarm optimization. *Appl. Soft Comput.* 49: 485–497, doi: https://doi.org/10.1016/j.asoc.2016.08.032

[25] De A, Mamanduru V K R, Gunasekaran A, Subramanian N and Tiwari M K 2016 Composite particle algorithm for sustainable integrated dynamic ship routing and scheduling optimization. *Comput. Ind. Eng*. 96(C): 201–215, doi: http://dx.doi.org/10.1016/j.cie.2016.04.002

[26] Soleimani H and Kannan G 2015 A hybrid particle swarm optimization and genetic algorithm for closed-loop supply chain network design in large-scale networks. *Appl. Math. Model.* 39(14): 3990–4012

[27] Wolpert D H and Macready W G 1997 No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* 1(1): 67–82

[28] Wilson E O 2000 *Sociobiology: the new synthesis*, 25th anniversary ed. Cambridge, England: The Belknap Press of Harvard University Press

[29] https://en.wikipedia.org/wiki/Elephant#cite_noteShoshani120135 [Accessed on 26/03/2016]

[30] Shoshani J and Eisenberg J F 1982 Elephas maximus. *Mammal. Species* 182: 1–8 https://doi.org/10.2307/3504045, JSTOR 3504045

[31] Archie E A, Moss C J and Alberts S C 2006 The ties that bind: genetic relatedness predicts the fission and fusion of social groups in wild African elephants. *Proc. R. Soc. B* 273: 513–522

[32] Archie E A and Chiyo P I 2012 Elephant behavior and conservation: social relationships, the effects of poaching, and genetic tools for management. *Mol. Ecol.* 21: 765–778

[33] Adams R *Social behavior and communication in elephants—it's true! elephants don't forget!*. http://www.wildlifepictures-online.com/elephant-communication.html [accessed March 14th 2013]

[34] Krebs J R and Davies N B 1993 *An introduction to behavioral ecology*, 3rd ed. Oxford, UK: Blackwell Publishing

[35] Plotnik J M, de Waal F B M and Reiss D 2006 Self-recognition in an Asian elephant. *Proc. Natl. Acad. Sci.* 103(45): 17053–17057 https://doi.org/10.1073/pnas.0608062103

[36] Rensch B 1957 The intelligence of elephants. *Sci. Am.* 196(2): 44–49 https://doi.org/10.1038/scientificamerican025744

[37] Wyatt T D 2003 *Pheromones and animal behavior—communication by smell and taste*. UK: Cambridge University Press

[38] http://www.elephantvoices.org/elephant-communication.html [Accessed on 16/06/2016]

[39] Hassan R, Cohanim B, Weck O D and Venter G 2005 A comparison of particle swarm optimization and the genetic algorithm. In: *Proceedings of* the *46th AIAA/ASME/ASCE/*

AHS/ASC Structures, Structural Dynamics and Materials Conference, p. 1897

[40] Jamil M and Yang X S 2013 A literature survey of benchmark functions for global optimization problems. *Int. J. Math. Model. Num. Optim.* 4(2): 150–194, https://doi.org/10.1504/ijmmno.2013.055204

[41] Nickabadi A, Ebadzadeh M M and Safabakhsh R 2011 A novel particle swarm optimization algorithm with adaptive inertia weight. *Appl. Soft Comput.* 11: 3658–3670, https://doi.org/10.1016/j.asoc.2011.01.037

[42] Shi Y H and Eberhart R C 1998 A modified particle swarm optimizer. In: *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 69–73

[43] Shi Y H and Eberhart R C 2000 Experimental study of particle swarm optimization. In: *Proceedings of the SCI2000 Conference*, vol. 3, pp. 1945–1950

[44] Xin J, Chen G and Hai Y 2009 A particle swarm optimizer with multistage linearly-decreasing inertia weight. In: *Proceedings of the International Joint Conference on Computational Sciences and Optimization (CSO-2009)*, vol. 1, pp. 505–508

[45] Yang X S 2011 Bat algorithm for multiobjective optimization. *Int. J. Bio-Inspired Comput.* 3(5): 267–274

[46] Mandal S, Khan A, Saha G and Pal R K 2016 Reverse engineering of gene regulatory networks based on S-systems and bat algorithm. *J. Bioinf. Comput. Biol.* 14(3): 1–22, https://doi.org/10.1142/s0219720016500104

[47] Yang X S and Deb S 2014 Cuckoo search: recent advances and applications. *Neural Comput. Appl.* 24(1): 169–174

[48] Mandal S, Khan A, Saha G and Pal R K 2016 Large scale recurrent neural network based modeling of gene regulatory network using cuckoo search-flower pollination algorithm. *Adv. Bioinf.* 2016: 1–9, http://dx.doi.org/10.1155/2016/5283937

[49] Yang X S, Karamanoglu M and He X S 2014 Flower pollination algorithm: a novel approach for multiobjective optimization. *Eng. Optim.* 46(9): 1222–1237

[50] Khan A, Mandal S, Pal R K and Saha G 2016 Construction of gene regulatory networks using recurrent neural networks and swarm intelligence. *Scientifica* 2016: 1–14, https://doi.org/10.1155/2016/1060843

[51] Talaslioglu T 2013 Global stability-based design optimization of truss structures using multiple objectives. *Sadhana* 38(1): 37–68

[52] Askarzadeh A 2016 A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm. *Comput. Struct.* 169: 1–12

[53] Jong H D 2002 Modeling and simulation of genetic regulatory systems: a literature review. *J. Comput. Biol.* 9(1): 67–103

[54] Masys D R 2001 Linking microarray data to the literature. *Nat. Genet.* 28: 9–10

[55] Kolen J F and Kremer S C 2001 *A field guide to dynamical recurrent networks*. Wiley, USA

[56] Frank E S, Matthias D and Benjamin H K 2014 Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Front. Cell Dev. Biol.* 2: 1–7, https://doi.org/10.3389/fcell.2014.00038

[57] Palafox L, Noman N and Iba H 2013 Study on the use of evolutionary technique for inference in gene regulatory networks. In: *Proceedings of Information and Communication Technology (PICT 6)*, pp. 82–92

[58] Palafox L, Noman N and Iba H 2013 Reconstruction of gene regulatory networks from gene expression data using decoupled recurrent neural network model. In: *Proceedings of Information and Communication Technology (PICT 6)*, pp. 93–103