A Review on Swarm Intelligence Techniques and their Application for Solving an Electromagnetic Inverse Design Problem

*Talha Ali Khan and Sai Ho Ling
School of Biomedical Engineering, University of Technology Sydney, Ultimo, 2007, Australia
Talha.khan@uts.edu.au

***Abstract*— This paper encompasses a detailed review of state-of-the-art swarm-based algorithms with a focus on their applications along with a discussion on the merits and limitations of each algorithm. Further, a recently developed Advanced Particle Swarm Optimization (APSO) algorithm was compared with the different state-of-the-art-swarm-based algorithms through solving an electromagnetic inverse problem. Results showed that the APSO algorithm has outperformed the other algorithms. This research provides a scientific guideline for the comparison of different swarm-based algorithms and their utilization regarding specific applications.**

Keywords*— *Artificial Intelligence; Evolutionary computation; Swarm Intelligence; Optimization

I. Introduction

Evolution has been the constant drive in the course of this planet's history which has enabled many animal species to accomplish complicated tasks by learning from their environment, building resilience, and adapting. Examples of such evolutionary capabilities are multiple but some specific ones which revolve around animal social behavior include flocks of birds, colonies of ants, and bees in their hives. These examples profoundly explain the concept of Swarm Intelligence (SI) and stigmergy, where the collective movement of these species improves their mechanism to explore complicated spaces, this is achieved without any central command and just by following local rules by the agents. The results from this technique help the swarm achieve much more as compared to the sum of individual actions. Swarm Intelligence (SI) has been the focal point of numerous researchers belonging to diverse backgrounds of research. SI is defined as "The emergent collective intelligence of groups of simple agents" [1]. SI is the cumulative intelligence demeanour of self-formulated and dispersed systems such as an artificial group of simple agents. Examples of SI include a) nest building, b) food hunting c) unified clustering and d) categorization of the insects. The two principal concepts that are essential parameters of the SI are self-management and labour allocation. Self-management is the capability of an order to independently allocate its resources in a useful manner. Eric et al. established that self-management depends upon four main characteristics i.e.: negative feedback, positive feedback, variations, and frequent communication[2]. The positive and negative feedbacks aid in maintaining equilibrium and expansions. Variations are, however, usually used only for haphazardness. Frequent communication takes place when swarms communicate amongst each other restricting to their search areas. The other important characteristic of SI is the allocation of labour, which is illustrated as carrying out many feasible and simple tasks by entities. This is how individuals grouped as working together through the swarm can deal with intricate problems. The remaining of the paper is structured as follows. In Section II the problems associated with the SI algorithms are examined, section III defines the parameters of an algorithm, section IV explained in detail various SI algorithms. In Section V, 23 test benchmark functions are used to evaluate the performance of the basic SI algorithms. An electromagnetic inverse problem is solved to demonstrate the performance of the APSO algorithm. Section VI summarizes the main points.

II. ONGOING CHALLENGES IN SI COMPUTATION:

Despite the acclamations and accomplishments of SI, some issues remain unaddressed. The focus is on five of these issues: the disparity between practice and theory, categorization, regulating boundaries, large scale problems, and selection of algorithms, which are highlighted in this paper.

A. The disparity between Practise and Theory:

SI computation pertains to a substantial gap when it comes to considering practice and theory. The reason is still not understood why but metaheuristic algorithms, when applied to real-life problems run exquisitely. However, excluding GA, PSO, and simulated annealing, favorable results about metaheuristic algorithms cannot be found. Subsequently, leading to disinclined advancement or real-life application algorithms can be assessed in three crucial ways: dynamical systems, Markov chains, and complexity theory. Contrarily, metaheuristic algorithms, despite being less complex tend to resolve highly intricate problems [3].

B. Categorizations and Terms used for Algorithms:

Various approaches have been employed to categorize optimization techniques. The number of iterations and the number of agents' dependent techniques are the two most widely used approaches. The second approach (number of agents dependent) is further be classified into types: multiple agents and a single agent. Simulated Annealing Algorithm (SA) is an example of the single-agent method having a zigzag trajectory; however, Particle Swarm Optimization (PSO), ant colony, and Cockroach Swarm Optimization (CSW) are population-based techniques. These methods frequently have multiple agents, work together in a nonlinear method, and a subcategory of that is known as SI-based method. PSO and fish swarm, for instance, are swarm-based methods and stimulated by swarming behavior of birds, fish, and/or by SI in common. The other approach for algorithm classification is by classifying the main procedure of the algorithm i.e. how the algorithm works. For instance, deterministic algorithms produce the same output for a given input no matter how many times the computer executes the program. Newton

Raphson and hill-climbing approaches are examples of a deterministic algorithm. Conversely, if randomness is introduced in the algorithm then it is known as the evolutionary, metaheuristic, heuristic, or stochastic method. For instance, PSO is a stochastic method or metaheuristic technique. The other term that has been used more frequently in classifying the algorithms is based on the mobility of the algorithm that is locally or globally search. Local search algorithms usually converge toward a local optimum, not essentially towards the global optimum, these methods are usually deterministic and have no capability of escaping the local optima [4]. Alternatively, for a given problem the usual practice is to find out the global optimum. Local search techniques are incapable to find out a global optimum, therefore, the global search methods are the best choice.

C. Impact on the parameters refinement :

Every metaheuristic method has certain characteristics for obtaining the optimal performance which ultimately defines the efficiency of the method. The most important issue is to set the appropriate value of these parameters along with the tuning of these parameters to get the maximum efficiency of the method. The fine-tuning of these parameters is a difficult optimization problem itself. To solve this issue, two types of methods are available in the literature. The first technique is to use a hit and trial method in which different values are tested one by one for the main parameters. Once an appropriate value is determined it is set for the more test by applying on the same problem or a larger scale problem. The second method is to use one technique to refine the parameters of the other technique. The dependency of one algorithm to another makes this approach an open research area for the researchers.

D. Need for Practical and Large-Scale Problems:

For solving the real-world problems, SI techniques are effective. However, only for those applications having a very few or moderate numbers of design variables. From the current literature, it is revealed that the focus is only on problems having moderate or hundreds of design variables. It is hard to find any application with several hundred variables [4]. On the other hand, linear programming solves problems having around millions of design variables. As a result, it is still an open research area that how to use SI techniques on a large scale as well as practical problems. Along with that issue, another problem is the use of the methodology. Because one algorithm is effective for solving a smaller problem, but it fails to solve a large-scale problem. Other key parameters include computational cost, memory capacity, and computing resources that require special attention as well.

E. Correct selection of the Algorithms

Despite all the literature available it is still hard to decide which algorithm gives the best result for a given problem. There are no clear standards or procedures to choose an algorithm, although there are detailed guidelines on how to use a method and what kinds of problems they can solve. As a result, the problem of choosing an algorithm is still there.

III. EXPLORATION OF THE INCANTATION FOR OPTIMIZATION

A. Basic Principle of an Algorithm

Algorithms are mathematically a method that produces outputs for given inputs. For a given problem all the algorithms generate a solution "a^{t+1}" at the current iteration "t" from a known solution "a^t".

$$a^{t+1} = \alpha(a^t, b(t)) \tag{1}$$

where a^{t+1} is a new solution vector *of* a^t, for a given solution α is a nonlinear mapping, if the algorithm B has "n" parameters $b(t)=(b_1, b_2, \ldots, b_n)$ which is time-dependent and can, therefore, be tuned.

B. What is the Best Algorithm?

For an ideal algorithm, it is anticipated to get the best solution from the initially assumed solution in a single step. Therefore, the minimum computational effort is required. Alternatively, it can be said that the best method is the one that can give the solution of a given problem in a single iteration only. The question arises here that whether any of such a method exists already. The answer is yes for a very precise kind of a problem that is quadratic convex problems. Newton Raphson (NR) approach is used for root finding. NR is used for finding the roots of $f(x) = 0$. For any maximize or minimize the problem the function has to fulfill the main condition $f'(x) = 0$, so it became an optimization problem for finding the roots of $f'(x)$. NR technique gives the following iteration formula:

$$x_{i+1} = x_i - \frac{f'(x_i)}{f''(x_i)} \tag{2}$$

NR is an ideal method for solving the quadratic functions that are also convex. However, the real-world problems are neither quadratic nor convex they are highly nonlinear. Therefore, the search for finding the best algorithm is still attracting researchers.

C. Features of an Algorithm:

There are two main properties of an algorithm which are discussed below:

a) Randomization Approach

One of the most effective methods is to randomly initialize all the population for the metaheuristic algorithms. This approach is easy to implement and proficient for most of the algorithms. Besides that, randomness can be used to the different components of the algorithm and several probability distributions can be used such as Levy distributions, Gaussian, and uniform distributions. Randomization is effective for global search methods. Understandably, it is still an open debate that how to introduce the randomness in an algorithm without reducing the convergence rate of the algorithm.

b) Diversification and Intensification

For any metaheuristic algorithm, diversification and intensification are the two main elements. Diversification is also known as exploration aims to discover the search area more comprehensively and support to produce varied solutions. Intensification is also known as exploitation, and it helps to obtain improved solutions by using the local information in the search process. Determining an equilibrium between exploration and exploitation is the key factor for any metaheuristic algorithm. Since an increase in the exploration helps the algorithm to convergence faster but it leads to the premature convergence to a locally optimal point or even a wrong solution. On the contrary, an increase in the exploitation will enhance the probability of finding the global solution; however, it reduces the convergence rate. Therefore, a steady transition between exploitation and exploration is required. Moreover, only exploration and exploitation are not sufficient. An appropriate strategy is needed during the search process to select the best solutions. "Survival of the fittest" (i.e. to keep updating the recent best solution obtained so far) is the most common method. Moreover, certain elitism is generally utilized to confirm that the best solutions are not lost and should be passed on to the next generations.

IV. Evolutionary Computation (EC)

EC is a part of computational intelligence that is dependent on the ideas and theories of biological evolution. Generally, EC comprises of Evolutionary Algorithms (EAs), SI, and other methods. EC methods perform well in approximating solutions in various kinds of problems, due to their capability of not creating any supposition about the basic fitness landscape. For that reason, these methods have shown better performance in different areas that include industrial applications, cutting- edge technology, and academic research [5]. Figure 1 shows the hierarchical distribution of nature-inspired algorithms.

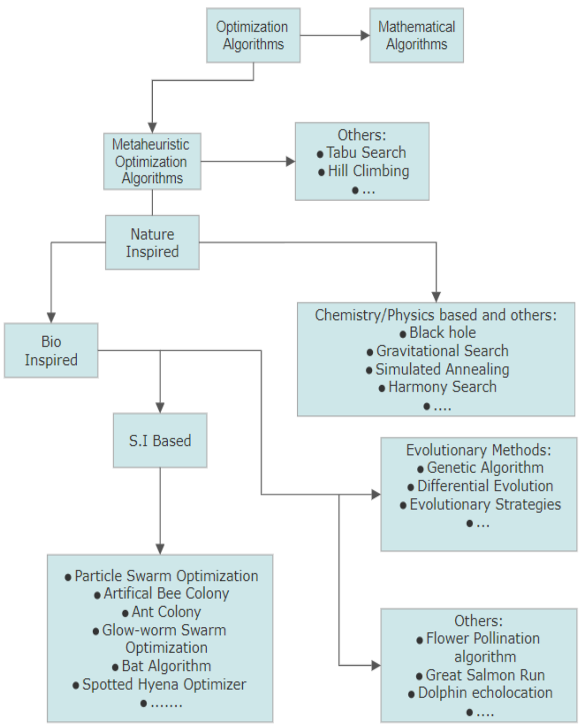

Figure 1 Hierarchy of nature inspired algorithms.

A. *Evolutionary Algorithms (EA)*

EAs are the subcategory of EC that are population-based metaheuristic optimization techniques. EA algorithms utilize a few procedures depends on biological evolution such as mutation, recombination, selection, and reproduction. Every solution in the EA algorithm for the given optimization problem is denoted by a single agent in the whole population. The fitness of each agent is evaluated by a function. Through genetic operators and selection procedure evolution of the population is performed. These operators are cross over, reproduction and mutation. EAs are classified into four main types. These are as under:

1) *Genetic Algorithms(GA):*

In the early 1970's Holland presented a novel algorithm called a Genetic Algorithm (GA) [6]. The algorithm is founded on Darwin's theory of survival of the fittest. Through utilizing crossover and mutation genetic operators along with the Darwin principle of natural selection, the population having a related fitness value is iteratively determined. GA is famous because it can solve complex optimization problems without using the initial values. Despite having pros it has a few cons as well, for instance, a slow rate of convergence and even non-convergence. The probabilities of the crossover and mutation are the significant parameters that control the GA's performance. Larger values of the crossover probability resulted in a faster rate of the production of the new individuals. Extremely large values can destroy the genetic model as well as the individuals' structure with high fitness that leads to the slow searching process. In contrast, small values of the crossover probability help in the production of new individuals.

2) *Genetic Programming (GP):*

GP is an evolutionary technique that expands the use of genetic algorithms to permit the exploration of the space of computer programs [7]. Similar to the other evolutionary algorithms, it works by defining fitness criteria and then using this measure to develop the population of the agents by imitating the fundamental concepts of Darwinian evolution. By using an iterative approach it breeds the solutions to problems that involve the probabilistic selection of the fittest solutions and their difference utilizing a set of genetic operators, generally mutation and crossover. The key variance between GA and GP is that the population is represented as an array in GA whereas, each agent is a computer program in GP. GP has been effectively used on different real-world problems without telling the computers how to solve them.

3) *Evolutionary Strategies (ES):*

ES is similar to the evolutionary methods and applied in the continuous search domain for black-box optimization problems. Based on the biological evolution, their unique creation is reliant on the usage of recombination, mutation, and selection in populations of candidate solutions. An algorithmic perspective shows that ES is the optimization technique that stochastically samples new candidate solutions, usually from a multivariate normal probability distribution [8].

4) *Evolutionary Programming (EP):*

EP is analogous to GP, however, the program structure is fixed. In EP a population of chromosomes is utilized to develop finite-state machines (FSMs) – referred to as a program [9]. Till now, the sequences of symbols that are detected are provided to every FSM. Every agent (individual) is then assessed by its capability of guessing future symbols. EP uses fitness values to choose agents (individuals) similar to other EAs and then uses evolutionary operators to explore new solutions. EP is dissimilar to GA in the sense that it uses two evolutionary operators, which are selection operators and changes by the use of mutation. Original EP does not use the recombination operators [10].

B. SI Based Techniques:

Several methods are constructed on the performance of different natural swarms that are different kinds of birds, ants, bees, fireflies, fishes, spiders, wolf, and many others. The common characteristics of all these methods are the same as they all are population-based and are interactive methods. On the other hand, their searching criteria are different. A few of the most famous SI algorithms are discussed and summarized in Table I.

1) Ant Colony Optimization

In 1992, during his Ph.D. studies, Dorigo proposed a new algorithm based on the foraging behavior of ants which is now known as the Ant Colony Optimization (ACO) algorithm [11]. ACO comprises of four key parts (ant daemon action, ants, decentralized control, and pheromone) that give support to the whole system. Since this method is inspired by the ant system, ants are the virtual agents that are used to imitate the exploitation and exploration of the search area. Ants while moving over the paths drop a chemical substance known as a pheromone in the real world. Because of the evaporation, the intensity of this material varies over time. ACO uses the same phenomena, where ants spread this substance while moving in the search area and the amounts of this chemical show the strength of the trail. The criteria for selecting the direction based on the path by the ants are depending on the higher trail intensity. This path intensity is regarded as the system's global memory [12]. Daemon's actions are used to collect global information. The single ant does not be able to perform this action; therefore, it uses this information to

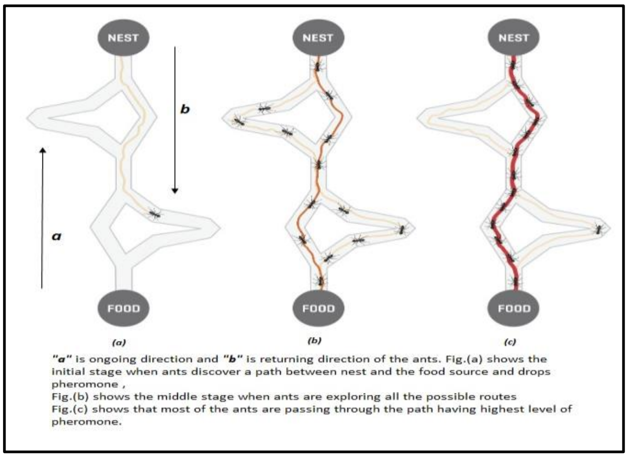

Figure 2 Path selection criteria of ants in ACO

decide if more pheromone is added so that convergence of the algorithm will be increased. Decentralized Control System (DCS) is used to make the ACO robust and flexible within a dynamic environment. The significance of using a DCS is that in case of any ant disappear and system failure it makes the ACO more flexible. All these actions give a supportive and mutual collaboration which helps to identify the shortest routes [13]. The process of choosing the shortest path is highlighted in Figs 2(a-c) [14], which shows the early stage, the middle phase, and the finishing result of the algorithm.

Fig 2(a) shows the early scenario at the beginning of the ACO when an ant moves back and forth from its nest and the source. Fig 2(b) shows that over the iterations when the ants explore several probable routes between nest and source. The best route chosen by the ants due to the higher intensity of the pheromone is shown in fig 2(c). To find out the probability from the present position to the updated position (3) is used.

$$p_{(i,j)}^{k}(t) = \left\{ \begin{array}{ll} \frac{([\tau_{ij}(t)]^{\alpha}.[\eta_{ij}]^{\beta}}{(\sum_{k \epsilon j_k}[\tau_{ij}(t)]^{\alpha}.[\eta_{ij}]^{\beta})} & \\ 0, & Otherwise \end{array} \right\} \tag{3}$$

Where J_k is the nodes that the ant is permitted to move back and forth from node *i*. $p_{(i,\,j)}$ is the probability of going from node *i* to node *j*. At time *t*, τ_{ij} *(t)* denotes the quantity of unevaporated pheromone between node *i* and node *j*, η_{ij} gives to the visibility between node *i* and node *j*. β and α are used to manage the impact of τ_{ij} *(t)* and η_{ij}, whether β" is having a larger value the ants searching behavior is dependent on its knowledge or visibility. If "α" has, a larger value than the ants searching is dependent on the pheromone quantity. To preclude the ants from traveling to the same nodes, repeatedly, they have a taboo list. Since, pheromones are the key factor in ACO, which help the ants to choose the path of having a higher intensity, the relation for depositing the pheromone can be expressed as:

$$\Delta\tau_{ij}{}^{k}(t) = \left\{ \begin{array}{ll} & \frac{Q}{L_k}(t) \\ 0, & Otherwise \end{array} \right. \tag{4}$$

Where *k* denotes any specific ant, *L* is the length of the route (i.e. the cost of the ant travel), *Q* is a constant, and *t* represents the iterations. At iteration *t*, the value of this factor highlights the pheromone rate that the ant moves between node *i* and *j*. For all the routes that are not chosen the pheromone, the deposition value is zero [15]. The pheromone evaporation rate is one of the other major factors in ACO. Which is used to find out the exploitation and exploration behavior of the ants? Greater values of this factor lead to exploration whereas the lower values cause exploitation. If this factor has a very low value than the ants are failed to get the optimal path, on the contrary, very high value causes the ants to get lost [16]. The evaporating factor is expressed as:

$$\tau_{ij}(t+1) = (1-p).\tau_{(ij)(t)} + \sum_{k=1}^{m}[\Delta\tau_{ij}{}^{k}(t)] \tag{5}$$

where *p* is the pheromone evaporation rate and *m* is the number of ants in the system.

ACO has many advantages as compared to other EC methods some of them are highlighted as under [17]:

- It helps to find the optimal solution quickly because of positive feedback.
- Distributed computation helps to avoid premature convergence.
- Collective interaction of a population of agents.

On the other hand, ACO has several disadvantages as well these are as under:

- ACO has a slower convergence in comparison with other heuristic methods.
- The absence of the centralized processor prevents the ants to move towards good solutions.
- The time of convergence is ambiguous.
- For the problems having a larger search area, ACO shows poor performance.

After the introduction of the standard ACO, it became an area of interest among researchers and scientists. Many versions of the ACO have been presented so far to expand the efficiency of the standard method. The first modification suggested by Dorigo et.al [18] entailed modification of three important characteristics of the ACO (local search procedures, pheromone, and state transition rule) they named it as Ant Colony System (ACS). In this system, for updating the pheromone a global update strategy is used so that the ants focus on the searching areas having the best solution. This amendment intents to better the convergence of the algorithm. The state transition rule involves the second modification, which differs from ACO. The stated probability *"q_0"* In ACS, whereas has to choose (behavior used by the ant) where *"q_0"* is commonly set to 0.9 and compare to a value of q ($0 \leq q \leq 1$). In case of a lower value of q than this range, the exploitation is used and vice versa. Local search procedures are performed through local optimization heuristic-based edge exchange methods, for instance, 2-opt, 3-opt, or Lin-Kernighan is used for. The method is used on each solution produced by an ant to achieve its local minima. This new improved ACO is then applied to the TSP problem for validating its performance.

The other most prominent version of ACO is Max-Min Ant System. In 2000, Hoos et.al proposed this variant of ACO [19]. They presented three modifications in ACO; first, they proposed an interval [τ_{min}, τ_{max}] to bounds the pheromone trail values. Secondly, the pheromone trails values are set to the maximum to facilitate the exploration. In the last variation, only one ant is permitted to add pheromone that helps to exploit the best solutions. Two techniques are used to add the pheromone it is by either a global- best approach or an iteration-best approach. In the global best method, the ants with the best solution in the same iteration can add the pheromone without considering the other ants. In the iteration-best method, for every iteration, the ant with the best solution only adds the pheromone.

A number of the optimization problems have been solved by the ACO to show its proficiency in the field of Telecommunication [20-22], Robotics [23-25], Railway Engineering [26, 27], Solving Travel Salesman Problem (TSP) [28-30], Image processing[31, 32], Finance [33, 34], Biology [35, 36], etc.

2) Artificial Bee Colony

Artificial Bee Colony (ABC) was introduced by Dervis in 2005 and is the most current SI algorithm [37]. The efficiency of the ABC was analyzed in 2007 when compared with other SI techniques [38]. A similar study was also conducted in 2009 [39] by using different benchmark functions and it is found that the ABC method outperformed other methods. ABC is stirred by the conduct of honeybees for finding the food sources, called nectar, and by sharing the food source information with other bees. This method is easy to implement similar to PSO and DE [40]. This method consists of the agents (bees) which are classified into three categories: a) the scout bee b) the employed bee c) the onlooker bee. These bees have several duties allocated to execute the algorithm. The duties of the employed bees are to discover the food source and to memorize the food source location information. The number of employed bees is like the number of food sources as one food source is looked after by each employed bee. The employed bees then pass on the information of the food source to the onlooker bees in the hive. The food source is then selected to collect the nectar. Finally, the scout bee is responsible for looking for other food sources and the new nectar.

The steps for the ABC algorithms are as under:

Initialization (Stage I): The controlling parameters are adjusted and scout bees initialized the vectors of the population of the food source. Every vector contains n variables that are optimized, to minimize the fitness function. For the initialization stage, the equation used is defined as:

$$x_i = l_i + rand \times (u_i - l_i) \tag{6}$$

Where *rand* is the random number from (0-1), u_i and l_i are the upper and lower bound of x_i.

Employed bees (Stage II): In this part of the algorithm, there is an extensive search is conducted around the neighborhood for the new food source so more nectar is gathered. After finding the food source its fitness is calculated. To generate a new food position from the previous in the memory following equation is used.

$$v_i = x_i + \emptyset_i(x_i - x_j) \tag{7}$$

where $\emptyset_i$ is a random number between the bounds [-a, a] and x_j is a randomly selected food source. After producing a new its fitness is calculated. a greedy selection is applied between x_j and v_j. For the smaller difference between $(x_i - x_j)$ exploitation occur and if it's large then the exploration takes place. To calculate the fitness value following relationship is used:

$$fit_i(x_i) = \begin{cases} \frac{1}{1+f_i(x_i)} & if\ f_i(x_i) \geq 0 \\ 1 + abs\big(f_i(x_i)\big) & if\ f_i(x_i) < 0 \end{cases} \tag{8}$$

where f_i is the objective function value.

Onlooker Bees(Stage III): Based on the information given by the employed bees and probability calculated by using the fitness value the onlooker bees select their food sources. p_i can be calculated as :

$$p_i = \frac{fit_i(x_i)}{\sum_{i=1}^{SN} fit_i(x_i)} \tag{9}$$

Scout Bees (Stage IV): The scout bees are unemployed bees that randomly select their food sources. If the fitness values of the employed bees are not getting better over a fixed number of iterations known as abandonment criterion or limit, they turned in to the scout bees and their food sources have been deserted.

Stage V: The best position and its fitness value are memorized.

Stopping Criteria Check (Stage VI): If the stopping criteria are achieved, the program stops, if not then it goes back to stage II and redo till the stopping condition is obtained.

There are many pros of ABC these include simple implementation, robust and adaptable. As it needs two controlling parameters only it is considered as the highly flexible algorithm, due to its flexibility as compared to the other SI methods it is used for solving many real-world optimization problems [41]. A few drawbacks of the ABC are; slow when used in serial processing because a lot of computation is required for fitness function assessment [42].

Despite the fact, that ABC is a new algorithm many versions of the standard algorithm have already been published and available in the literature. The most notable is the Modified ABC proposed by his creator in which they have introduced two new controlling factors perturbation frequency and magnitude to solve benchmark functions [43], the other variant is proposed by Liu et.al in which they introduced new search strategy and elective probability *P*. The new mechanism helps to omit scout bee stage and probabilistic selection scheme. The method is compared with two other ABC based techniques by using 28 benchmark functions [44]. A number of the problems have been solved by ABC in different areas these include Communication [45-47], TSP [48, 49], Power Engineering [50, 51], Health care [52-54], Management [55, 56], Image processing [57, 58], and many other applications.

3) *Cuckoo Search Algorithm*

Yang et.al have developed a novel algorithm in 2009 called the Cuckoo Search Algorithm (CSA)[59]. The algorithm is relying on the cuckoo species brood parasite behavior along with the levy flight characteristics of fruit flies and birds. Three simple rules are followed for the implementation of the CSA algorithm.

1. In every iteration, only one egg is allowed to be laid and the nest is selected randomly.
2. Only the good nests and eggs are allowed to take into the next stage.
3. The host bird explored the nests with a probability $p_a \in [0, 1]$ in which the egg is present and the host nests are in a fixed amount. Depending on the value the host bird either build a new nest, throw the egg or simply move from the nest.

To make it simple, the third rule is estimated by the fraction p_a of the n nests and are replaced by new nests. The complexity of the algorithm can be increased by adding more eggs in the nest. Different steps involved based on the three key factors in CSA are as follows:

A levy flight is performed to generate a new position for the cuckoo indexed m:

$$x_m(t+1) = x_m(t) + \alpha \oplus Levy\ (\lambda) \tag{10}$$

where α is the step size, α= 1 in most cases. The product $\oplus$ is an entry wise multiplication analogous to the approach used in PSO but its more effective due to the levy flight for exploring the search area as the step size is bigger. The Levy flight necessarily gives a random walk while the random step length is taken from a Levy distribution

$$Levy \sim \mu = t^{-\lambda}, (1 < \lambda \leq 3) \tag{11}$$

(11) has an infinite mean and variance. To fulfill the requirement of the step length distribution, it is essential to achieved steps of a cuckoo from a random walk process. The new nests at the new locations can be made by discarding the worst net fraction, p_a. Based on the difference or the similarity of the host eggs the mixing of the solution is done by using random permutation. The step size, α, is initialized with a bigger value and it reduces linearly over the iterations. The reason of linearly decreasing the step size is allowing the population to converge towards the optimal solution in the last stage. The modification done by Deb et.al [60] in (10) is defined as:

$$\mathrm{x_m(t+1) = x_m(t) + \alpha \oplus Levy\ (\lambda) \sim 0.01} \frac{\mu}{|v|^{\frac{1}{\lambda}}} (x_n(t) - x_m(t)) \tag{12}$$

where u and v are taken from a normal distribution that is

$$\mu \sim N(0, \sigma_\mu^2), v \sim N(0, \sigma_\mu^2), \tag{13}$$

where,

$$\sigma_\mu = \left(\frac{(\gamma(1+\lambda) \sin(\frac{\pi\lambda}{2}))}{\gamma(\frac{(1+\lambda)}{2})\lambda 2^{\frac{(\lambda-1)}{2}}} \right)^{\frac{1}{\lambda}} \quad , \quad \sigma_v = 1 \tag{14}$$

γ is the standard gamma function. If term $(x_n(t) - x_m(t))$ has a smaller difference, then exploitation occurs otherwise for the large differences it will facilitate the exploration.

With the multimodal functions, CSA gives better performance because it needs only a few parameters to control than the other SI techniques [61]. Some of the variants of the CSA are: In 2011, Hassan et.al [62] presented a Modified Cuckoo Search (MCS) algorithm to improve the convergence of the algorithm. The amendment for this improved version consists of the exchange of information between the top solutions (eggs). To validate its performance and compare it with other algorithms, a different benchmark is used. Quantum Inspired Cuckoo Search Algorithm (QICSA) is another improved version of CSA presented in 2012 by Abdesslem et.al [63]. The improved variant uses the concepts of quantum computing and merged it with CSA. The primary purpose is to increase the stability and convergence of the method.

CSA is also used in many application these includes Power Engineering [64, 65], Telecommunication [66, 67], Robotics [68], TSP problem [69], Image processing [70, 71], embedded systems [72], and etc.

4) *Glow-worm Swarm Optimization*

Ghose et.al in 2009 proposed a novel algorithm known as Glowworm Swarm Optimization (GSO), which shares some properties of ABC and PSO for solving multimodal functions [73, 74]. The algorithm is based on the agents (glow-worms) that carry with them a minescence quantity called luciferin. The fitness of their current position is calculated by the given objective function, which they transformed into the luciferin value and broadcast it to the neighboring worms. Three steps in which the GSO works are luciferin level, neighborhood range update, and update glow-worm movement. The glow-worm is initialized randomly and over the iteration, the above three stages are repeated until the stopping condition is met. The fitness of the current position of the glow-worm is determined for updating the luciferin level by the following expression:

$$l_a(t+1) = (1-p).\, l_a(t-1) + \gamma J(x_a(t+1)) \qquad (15)$$

where $\gamma$is the luciferin enhancement constant, l_a is the luciferin level of the glow-worm "*a"* at time *t*, *p* is luciferin decay constant, and $J(x_a(t))$ is the value of the fitness function of glow-worm "*a*" position.

By using a probabilistic strategy during the movement phase, every glow-worm moves towards its neighbor having a higher luciferin value. The probability of its movement towards its neighboring glow-worm is calculated as:

$$p_{ab}(t) = \frac{l_b(t) - l_a(t)}{\sum_{k \in N_i(t)} l_k(t) - l_a(t)} \qquad (16)$$

where $b \in N_a(t), N_a(t) = \{b : d_{ab}(t) < r_d^a(t);\, l_a(t) < l_b(t)\}$ is the set of neighbors of glow-worm "*a*" at time *t*.

The position of the glow-worm in the searching area can be calculated as :

$$x_a(t+1) = x_a(t) + s\left(\frac{x_b(t) - x_a(t)}{\|x_b(t) - x_a(t)\|}\right) \qquad (17)$$

where "*s"* is the step size, and ||.|| is Euclidean norm operator. For the smaller difference value of the term $(x_b(t) - x_a(t))$ resulted in exploitation, however, larger values will facilitate the exploration behavior. In the next phase, every glow-worm tries to find out its neighbors. Each glow-worm decides to choose its neighbor depending upon the condition of having the shorter distance between as compared to the neighborhood range $r_m(t)$, the other criteria which take into account are that glow-worm *a* is brighter as compared to the glow-worm *b*. But, to choose among many neighbors, then the neighbor is chosen by using the probability equation.

$$p_{ab}(t) = \frac{l_b(t) - l_a(t)}{\sum_{k \in N_a(t)} (l_k(t) - l_a(t))} \qquad (18)$$

Finally, to restrict the range of communication in a group of glow-worms let's assume that if the initial range of every glow-worm is ($r_d^a = 0 = r_0$), the neighborhood range $r_m(t)$ is defined as:

$$r_d^a(t+1) = \min\{r_s, \max\{0, r_d^a(t) + \beta(n_t - |N_a(t)|)\}\} \qquad (19)$$

The values of these parameters are set as *ρ=0.4 ,γ=0.6, s=0.03,β=0.08,* $r_0=r_s$ and $n_t=5$. where r_s is a sensor range.

Some of the suggestions to modify the GSO algorithm, in general, are as under:

1) The range of the neighborhood can be an increase so that more glow-worms can be included. After the fitness evaluation of each glow-worm, all the glow-worms move towards the glow-worm (having the best solution). In this way, the proficiency of the algorithm increases in the exploitation phase, as more glow-worms are within range of the best solution.

2) To decrease the computational cost of the GSO and increase the rate of convergence, there will be a small number of glow-worms within the neighborhood range.

Like other methods, GSO has many modified versions as well that are proposed to ameliorate its performance. For instance, Bin et al. [75] proposed two approaches to the movement stage of the GSO. The first modification is the greedy acceptance criteria in which every glow-worm updates its position one dimension by one dimension. The other amendment is introducing new movement formulas that are inspired by the PSO and ABC algorithms. This modification helps to enhance the accuracy and convergence of the GSO method. A modified version of GSO is proposed in [76], which introduced some modifications to adjust the step size, the local decision, and the selection approach. The standard GSO while solving the multi-peak benchmark functions, the convergence rate is slow, and the accuracy is not high to solve the drawback Peng et al. presented a modified version. They introduced a fluorescent factor that adaptively fine-tunes the step length of the algorithm [77].

Applications related to different fields such as Image processing [78-80], Communication [81, 82], Robotics [83, 84], and Power Engineering [85-87] used the GSO algorithm.

5) *Particle Swarm Optimization Algorithm*

In 1995, Kennedy and Eberhart that are inspired by the social behavior of the birds present the PSO algorithm. Similarly, to the flock of birds, the method comprises the number of agents to form a swarm. Each agent in the search area is looking for an optimal solution. The description of all the standard and the modified PSO algorithm are presented as under:

A. *Standard Particle Swarm Optimization Algorithm (SPSO):*

In the beginning, a swarm of agents is created with random positions and velocities. The evaluation of every agent's fitness is made by a given benchmark function. After each iteration, the position for the next function assessment, and the velocity of the particles is calculated by (1) and (2). As a result, if the position found out is better as compared to the last best position is stored in the memory. v_{max} is defined to control the unnecessary movement of the agents outside the search space. If the velocity goes above v_{max} it is set to zero. Each particle moves in the search area for finding the best solution. The position of each particle is defined as

$$x_i(t+1) = x_i(t) + v_i \qquad (20)$$

The knowledge of every particle depends upon its own experience and the surrounding particle's information. These elements have equivalent importance and might be changed based upon particles decision so the velocity equation will be

$$v_i(t+1) = v_i + R_1\left(P_i^p - x_i\right) + R_2\left(P_g - x_i\right) \qquad (21)$$

where

$P^p = [P_1^p, P_2^p, P_3^p, P_D^p]$

$P_g = [Pg_1, Pg_2, Pg_3, Pg_D]$

i=1, 2, 3… D.

P_g is the global best position, the v_i = {v_1, v_2... v_n} is the velocity of the particles, R is the random number [0-1], D is the dimension of the search space $D \in \{1, 2, 3\ldots D$, P_i is the local best, and x_i is the current position. Each particle is assessed by a given fitness function. The motivation of the PSO is to reduce the cost values of the particles iteratively for the given fitness function. The particles progress from iteration t to $t + 1$ by iterating the process.

B. SPSO with Constriction factor and Inertial Weight:

Shi *et al*. have first introduced the inertial weight "w" and constriction factor "χ" [88]. By introducing these two parameters the (2) will be changed to

$$v_i = \chi.\{w.v_i + C_1R_1(P_i^p - x_i) + C_2R_2(P_g - x_i)\} \quad (22)$$

Where χ is the constriction factor, w is the inertial weight, and C_1, C_2 are two acceleration constants numbers.

The first term in (22) represents the Inertia component it is also called the momentum of habit. It supports the particle to move in the same way in which it has been traveling. The second term stated as the cognitive part. This part is the distance that a particle is from the best solution found by itself. It shows the propensity of particles' to come back to environments where they experienced their best performance. The third term denoted as the social part. It shows the distance that a particle is from the best position found by its neighborhood. It characterizes the inclination of particles to follow the success of other agents.

The new parameter i.e. inertial weight used to find out the impact of the previous velocity on the current update. Higher values of the *'w''* assist in global search while lesser values facilitate the local search. The inertial weight "w" is decreased linearly from the current iteration to the later iteration. Factors *i.e.* w_{max} and w_{min} are used to control inertial weight. The relationship is used as follows:

$$w = w_{max} - \left(\frac{w_{max} - w_{min}}{T}\right) * t \quad (23)$$

where T is the total number of iterations and t is the current iteration. The constriction factor has also been presented in [89]. (24) calculates the constriction factor as

$$\chi = \frac{2}{\Phi - 2 + \sqrt{\Phi^2 - 4*\Phi}} \quad (24)$$

where $\boldsymbol{\Phi}$=4.1, constriction factor is used to adjust the inertial weight by the following relation.

$$w = \chi * \left(0.0005 + w * \left(\frac{T-(t-30)}{T}\right)\right) \quad (25)$$

The agent velocity is restricted by the highest value of v_{max} in (22), v_{max} is used to find out what areas are needed to be explored between the current and the target position. If the value of v_{max} is very high than the particles move unsteadily and will go distant to the good solution; conversely, if the value is small it restricts the mobility of the particle and they do not move towards the best solution.

To enhance the efficiency of the PSO algorithm in general the following steps must be considered:

- As the population is a key parameter, the larger population resulted in the accurate and swift convergence.
- Maintaining a trade-off between exploration and exploitation. The higher exploration resulted in exploring the new searching areas, whereas, exploitation in the final phase helps to confine the search.
- Having a swarm of particles within the swarm (sub swarm) is another common approach. This approach is effective to solve the multi-objective problems by allocating tasks to each sub-swarm [90].
- Modifying the velocity equation of the PSO that is dynamic velocity adjustment. This technique moves the particles in various directions resulted in fast convergence.

There are many advantages and a few disadvantages of the PSO algorithm these include simple implementation, efficient global searching, few parameters settings, and design variables that can be modified. PSO has a propensity to trapped in local minima resulted in a premature convergence and weak exploitation in the final stage. Over the years, PSO has been used in many areas these include in the field of Communication [91-93], Robotics [94-96], Image processing [97-99], Electrical [100-102], Management [103, 104], and many others.

6) Bat Algorithm:

In 2010 Xin proposed a novel algorithm known as the Bat algorithm [105]. The algorithm is based on the echolocation behavior of microbats. Microbats release a sound wave a kind of sonar and listen to it when reflected from the nearby objects. They use this approach to prevent hurdles, find out prey and locate their roosting crevices in the dark. Based on the type of micro-bats each produces a different type of pulse and can correspond to their correlated with their hunting scheme. Few of them produce constant frequency waves for echolocation, on the other hand, the majority of them generate low-frequency pulse [106]. The approach is based on the three main principles.

a) They all use the echolocation approach to observe the distance and remarkably, they find out the difference between prey/food and the obstacles.

b) Every bat has a frequency range [f_{min}, f_{max}] and moves with a velocity v_i at position x_i randomly. They vary their loudness A_0 and emission rate $r \in [0, 1]$ to find out prey based on the closeness of their target.

$$x_i^{t+1} = x_i^t + v_i^t \quad (26)$$

$$f_i = f_{min} + (f_{max} - f_{min})\,\epsilon \quad (27)$$

$$v_i^{t+1} = v_i^t + (x_i^t - x^*)\,f_i \quad (28)$$

c) The loudness A_0 changes from the maximum value of A_0 to the minimum value of A_0.

$$A_i^{t+1} = \propto A_i^t \quad (29)$$

$$r_i^t = r_i^0[1 - e^{-\beta t}] \quad (30)$$

Where x is the position, v is the velocity, ε is a random number drawn from a uniform distribution, and α β are constants. Many areas in engineering use this algorithm such as in Communication [107, 108], Power [109, 110], Robotics [111, 112], etc.

7) Other SI based Algorithms:

Many other algorithms are proposed during the last few decades. Table I summarized a list of the remaining algorithms. The proficiency of the SI methods is based on the principle that they mimic the best properties of nature, mainly the selection of the fittest in biological systems that have evolved by nature over a millennium.

Table I: List of the Swarm Intelligence Algorithms

Algorithm	Year of Publication	Inspiration	Authors
Boids [113]	1987	Inspired by the behavior of flocks of birds. Instead of, simulating the whole flock, the algorithm only specifies the behavior of a single bird.	Craig W. Reynolds
MBO: Marriage in Honey Bees Optimization[114]	2001	The unified model of the marriage in honeybees inspires the method.	H.A. Abbass
Bacterial Foraging[115]	2002	The algorithm is inspired by the foraging behavior of *Escherichia* coli bacteria.	K.M. Passino
Bacteria chemotaxis (BC) algorithm [116]	2002	Based on the biological model of the Bacteria chemotaxis.	Muller et al.
Fish Swarm Optimization Algorithm [117]	2002	It depends on the behavior of fish such as making groups.	Li et al.
Shuffled frog-leaping algorithm [118]	2003	The algorithm is based on the natural memetic in which a set of the virtual population of frogs interact with each other and grouped into various memeplexes.	Eusuff et al.
BeeHive [119]	2004	It depends on the communicative and evaluative approaches and processes of honeybees.	Horst et al.
Virtual Bees [120]	2005	Depends upon the communication model of the bees, they interact whenever they find the targeted food source.	Xin-She Yang
Bee colony optimization[121]	2005	It depends on the communicating behaviors of the real bees to solve a ride-matching problem.	Dusan et al.
Bacterial Colony Chemotaxis (BCC) algorithm[122]	2005	It is based on the BC algorithm; it used the single bacterium's reaction to chemoattractants and the communication among the bacteria.	Wu et al.
Bees Swarm Optimization [123]	2005	It depends on the intelligent behavior of real bees for solving a hard Johnson benchmark.	Safa et al.
Honey-Bees Mating Optimization (HBMO) Algorithm [124]	2006	Inspired by the honeybees mating process.	Haddad et al.
Cat Swarm Optimization (CSO) [125]	2006	Inspired by the behaviors of cats, such as seeking and tracing.	Pan et al.
Fish School Behaviour [126]	2008	The algorithm depends on the feeding, swimming, and breeding behavior of the fish school for high dimensional search space problems.	Filho et al.
Roach Infestation Optimization [127]	2008	Inspired by the social characteristics of cockroaches.	Spain et al.
Fast Bacterial Swarming Algorithm (FBSA) [128]	2008	The algorithm is based on the swarming behavior of birds and foraging behavior of *Escherichia* coli bacteria.	Hua et al.
Bumblebees [129]	2009	Inspired by the mutual behavior of social insects.	Padro et al.
Group Search Optimizer [130]	2009	The algorithm is based on animal searching behavior	He et al.
Firefly Algorithm [131]	2009	This algorithm is based on the bioluminescence process of fireflies.	Xin-She Yang
Bumble Bees Mating Optimization Algorithm [132]	2010	The algorithm is dependent on the mating behavior of the bumblebees.	Yannis et al.
Cockroach Swarm Optimization [133]	2010	This algorithm is based on the social behavior of cockroaches.	Hui et al.
Hunting Search [134]	2010	This algorithm is based on the group hunting skills of animals like dolphins, wolves, and lions.	Mahjoob et al.
Krill Herd [135]	2012	The algorithm is based on the herding behavior of krill individuals.	Amir et al.
Wolf search algorithm [136]	2012	This algorithm mimics how wolves can search for food and stay alive by circumventing their adversaries.	Rui et al.
Bacterial Colony Optimization [137]	2012	The algorithm depends upon the entire life cycle of the E. coli bacteria that include communication, chemotaxis, reproduction, elimination, and migration.	Ben et al.
Lion's Algorithm [138]	2012	Based on the social behavior of the lion that helps it to keep strong.	B.R.Rajakumar
Blind, naked mole-rats (BNMR) algorithm [139]	2012	Inspired by the Social behavior of the blind naked mole-rats colony.	Mohammad Taherdangkoo

Fruit Fly Optimization Algorithm [140]	2012	The algorithm is based on the behavior of fruit flies.	Wen-TsaoPan
Social Spider Optimization (SSO) [141]	2013	The algorithm is based on the cooperative behavior of social spiders.	Erik et al.
Cuttlefish Algorithm [142]	2014	Imitates the process of color-changing behavior of the cuttlefish	Adel et al.
Grey Wolf Optimizer [143]	2014	The algorithm mimics the leadership hierarchy and hunting strategy of grey wolves.	Ali et al.
Spider Monkey Optimization algorithm [144]	2014	Based on the foraging behavior of spider monkeys.	Bansal et al.
Animal migration optimization [145]	2014	The algorithm is based on the migration behavior of animals.	Li et al.
Monarch butterfly optimization[146]	2015	Inspired by the migration of monarch butterflies.	Gai et al.
Moth-flame optimization algorithm [147]	2015	The algorithm is inspired by the navigation approach of moths known as transverse orientation.	Seyed Ali Mirjalili
Elephant Herding Optimization[148]	2015	Inspired by the herding behavior of the elephant group.	Gai et al.
Ant Lion Optimizer	2015	This algorithm is based on the hunting skills of ant lions.	Seyed Ali Mirjalili
Crow search algorithm [149]	2016	The algorithm works on the concept of how the crows stock their extra food in hiding places and use it when required.	Alireza Askarzadeh
Dolphin swarm algorithm [150]	2016	Inspired by the dolphins' behavior of echolocation, information exchanges, cooperation, and division of labor.	Tian et al.
Dynamic Virtual Bats Algorithm [151]	2016	This algorithm is based on the bat's capability during hunting to alter the frequency and wavelength of the sound waves.	Topal et al.
Dragonfly algorithm [152]	2016	This algorithm is based on the dynamic and static swarming behaviors of dragonflies	Seyed Ali Mirjalili
The Swarm Dolphin Algorithm (SDA) [153]	2016	This algorithm work on the three main characteristics of dolphins (a) Search (b) Detects (c) Capture.	Yi et al.
Wolf-pack algorithm [154]	2016	The Algorithm is inspired by the social behaviors of the wolf pack in besieging, calling, and scouting.	Wang et al.
Whale Optimization Algorithm [155]	2016	The algorithm imitates the social behavior of humpback whales and based on its bubble-net hunting strategy.	Ali et al.
Spotted Hyena Optimizer [156]	2017	This algorithm is based on the spotted hyena's behavior. The primary idea is the social relationship between spotted hyenas and their collective behavior.	Gaurav et al.
Grasshopper Optimization Algorithm [157]	2017	The algorithm is inspired by the grasshopper behavior.	Saremi et al.
Salp Swarm Algorithm [158]	2017	Inspired by the swarming behavior of salps in oceans while foraging and navigating.	Ali et al.
Donkey and smuggler optimization algorithm [159]	2019	The algorithm is based on the searching behavior of donkeys.	Ahmed et al.
Fitness Dependent Optimizer [160]	2019	The algorithm is based on the bees' reproductive process and their collective decision-making behavior.	Jaza et al.

V. TESTING OF SI METHODS ON STANDARD BENCHMARK FUNCTIONS:

A. Benchmark Functions:

Several optimization algorithms claim their proficiency than the other methods in the literature. Therefore, to examine the efficiency of any algorithm benchmark test functions are used. In this paper, to examine the proficiency of the SI based methods a set of 23 standard benchmark functions are used. The testing is done on a selected SI based algorithms that have been used widely for decades on different optimization problems. These functions are tabulated in Table II. These functions are divided into two categories and they are as under:

a) Unimodal: This is the asymmetric model with a single minimum $f_1 - f_6$ and f_{19} belong to this type.

b) Multimodal with a few and several numbers of minima: These functions from f_7- f_{23} is in the multimodal type having a few and several local minima; f_7, f_8 f_{17}, f_{18}, $f_{21} - f_{23}$ are the low dimensions functions having only a few local minima.

Table II: Standard Benchmark Test Functions

Test Function	***Domain Range***	***Optimal point***
$f_1(x) = \sum_{i=1}^{d} x_i^2$	$-100 \le x_i \le 100$	$f_1(0) = 0$
$f_2(x) = \sum_{i=1}^{d} [100\,(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$-2.048 \le x_i \le 2.048$	$f_2(1) = 0$
$f_3(x) = \sum_{i=1}^{100} (\lvert x_i + 0.5 \rvert)^2$	$-10 \le x_i \le 10$	$f_3(0) = 0$
$f_4(x) = \sum_{i=1}^{10} i x_i^4 + random[0,1]$	$-2.56 \le x_i \le 2.56$	$f_4(0) = 0$
$f_5(x) = max_i\{\lvert x_i \rvert, 1 \le i \le 30$	$-100 \le x_i \le 100$	$f_5(0) = 0$
$f_6(x) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	$-10 \le x_i \le 10$	$f_6(0) = 0$
$f_7(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}]\right]^{-1}$	$-65.536 \le x_i \le 65.536$	$f_7([-32, -32]) \approx 1$
$f_8(x) = \sum_{i=1}^{9} \left[a_i - \sum_{j=1}^{25} \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}]\right]^2$	$-5 \le x_i \le 5$	$f_8(0.1928, 0.1928, 0.1231, 0.1358) \approx 0.0003075$
$f_9(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3} x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	$-5 \le x_1, x_2 \le 5$	$f_9 = ([0.08983, -0.7126]) = f_{11} = ([-0.08983, 0.7126]) \approx -1.0316$
$f_{10}(x) = -\sum_{i=1}^{d} c_i\, exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right]$	$0 \le x_i \le 1$	$f_{10} = (0.114, 0.556, 0.852) \approx -3.8628$
$f_{11}(x) = -\sum_{i=1}^{d} c_i\, exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$	$0 \le x_i \le 1$	$f_{11} = ([0.201, 0.15, 0.477, 0.275, 0.311, 0.627]) \approx -3.32$
$f_{12}(x) = 0.1 \left\{ \begin{matrix} sin^2(\pi 3 x_1) \\ + \sum_{i=1}^{29} (x_i - 1)^2 . [1 + sin^2(3\pi x_{i+1})] \\ + (x_{30} - 1)^2\, [1 + sin^2(2\pi x_{30})] \end{matrix} \right\} + \sum_{i=1}^{30} u(x_i, 5, 100, 4)$	$-50 \le x_i \le 50$	$f_{12}(1) = 0$
$f_{13}(x) = \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i) + 10$	$-50 \le x_i \le 50$	$f_{13}(0) = 0$
$f_{14}(x) = \frac{1}{4000} \sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$-600 \le x_i \le 600$	$f_{14}(0) = 0$
$f_{15}(x) = -20exp\left(-0.2\sqrt{\frac{1}{d} \sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d} \sum_{i=1}^{d} \cos(2\pi x_i)\right) + 20 + e$	$-32 \le x_i \le 32$	$f_{15}(0) = 0$
$f_{16}(x) = -\sum_{i=1}^{10} \left(x_i \sin(\sqrt{\lvert x_i \rvert})\right)$	$-500 \le x_i \le 500$	$f_{16} = ([420.9687 \ldots\ldots 420.9687]) = 10 \times 418.9829 = 4189.829$
$f_{17}(x) = (x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right) cos x_1 + 10$	$-5 \le x_i \le 5$	$f_{17\text{-}min} = 0.398$
$f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	$-2 \le x_i \le 2$	$f_{18\text{-}min} = 3.0$
$f_{19}(x) = \sum_{i=1}^{d} (\sum_{j-1}^{i} x_j)^2$	$-100 \le x_i \le 100$	$f_{19}(0) = 0$
$f_{20}(x) = \frac{\pi}{d} \left\{ \begin{matrix} 10\, sin(\pi y_1) \\ + \sum_{i=1}^{d-1} (y_i - 1)^2 . [1 + 10\, sin^2(\pi y_{i+1})] \\ + (y_{29} - 1)^2 \end{matrix} \right\} + \sum_{i=1}^{30} u(x_i, 10, 100, 4)$	$-50 \le x_i \le 50$	$f_{20}(1) = 0$
$f_{21}(x) = -\sum_{i=1}^{5} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	$0 \le X \le 10$	$f_{21\text{-}min} = -10.1532$
$f_{22}(x) = -\sum_{i=1}^{7} [(X - a_i)(X - a_i)^T + c_i]^{-1}$	$0 \le X \le 10$	$f_{22\text{-}min} = -10.4028$
$\boldsymbol{f_{23}(x) = -\sum_{i=1}^{10} [(X - a_i)(X - a_i)^T + c_i]^{-1}}$	$0 \le X \le 10$	$f_{23\text{-}min} = -10.5363$

B. Results and Discussion:

The results presented in this section are based on the performance of the different SI based methods that are applied to the 23-benchmark functions. The proficiency of each SI approach is tested on the standard versions of these methods and no modifications are applied. The performance is evaluated in terms of the standard deviation and the mean value. The benchmark functions are divided into two types the first is unimodal and the other is the multimodal functions with some or many local minima. Table III shows the results of the unimodal functions. The results show that the WOA (Whale Optimization Algorithm) gives better performance as compared to the standard versions of the other SI algorithms. WOA also reaches the global or near-global optimum faster as compared to the other optimization methods. PSO algorithm gives the second-best proficiency for a few unimodal functions. Similarly, for the Type II benchmark functions, WOA outperformed the remaining algorithms. WOA is a new metaheuristic algorithm proposed by Ali et.al that is inspired by the social behavior of humpback whales and based on its bubble-net hunting strategy.

TABLE III. The mean and standard deviation evaluation between SI algorithms for Type I Benchmark Functions for *D*=30

Functions		**PSO**	**ACO**	**ABC**	**GSO**	**CSA**	**WOA**
$f_1(x)$	Mean	0.000136	$1.7596e^{-4}$	$1.1820e^{-5}$	$1.1844e^{-06}$	$4.4138e^{-4}$	$1.41e^{-30}$
	Std. Dev	0.000202	$1.8603e^{-3}$	$8.3508e^{-3}$	$8.0723e^{-04}$	$5.5047e^{-4}$	$4.91e^{-30}$
$f_2(x)$	Mean	96.71832	$6.6160e^{1}$	$7.3760e^{+01}$	$1.1701e^{+02}$	$6.4181e^{+01}$	27.86558
	Std. Dev	60.11559	$3.7940e^{+01}$	$2.8049e^{+01}$	$2.6130e^{+01}$	$5.5250e^{+00}$	0.763626
$f_3(x)$	Mean	0.000102	3.38563	3.7395	3.3792	3.5441	3.116266
	Std. Dev	$8.28e^{-05}$	0.9363	0.7830	0.59301	0.68829	0.532429
$f_4(x)$	Mean	0.122854	$5.035e^{-1}$	$8.43\ e^{-2}$	$2.939e^{-01}$	$7.692e^{-02}$	0.001425
	Std. Dev	0.044957	$1.068e^{-1}$	$9.019\ e^{-2}$	$4.3811e^{-01}$	$6.0392e^{-02}$	0.001149
$f_5(x)$	Mean	1.086481	1.9737	0.7013	$3.90932e^{-1}$	$5\ .36e^{-1}$	0.072581
	Std. Dev	0.317039	0.7602	0.60031	$8.1938e^{-1}$	$9\ .630e^{-1}$	0.39747
$f_6(x)$	Mean	0.042144	$1.038e^{-3}$	$8.0986e^{-6}$	$3.0032e^{-8}$	$6.1718e^{-09}$	$1.06e^{-21}$
	Std. Dev	0.045421	$1.78375e^{-3}$	$5.3819e^{-6}$	$4.92938e^{-8}$	$9.8728e^{-09}$	$2.39e^{-21}$
$f_{19}(x)$	Mean	70.12562	$4.762e^{1}$	$5.3439e^{1}$	$1.9664e^{-2}$	$4.2038e^{-3}$	$5.39e^{-07}$
	Std. Dev	22.11924	$5.1531e^{1}$	$3.08892e^{1}$	$4.8427e^{-2}$	$7.2998e^{-3}$	$2.93e^{-06}$

TABLE IV. The mean and standard deviation evaluation between SI algorithms for Type II Benchmark Functions for *D*=30

Functions		**PSO**	**ACO**	**ABC**	**GSO**	**CSA**	**WOA**
$f_7(x)$	Mean	3.627168	2.4324	2.52729	2.27382	2.80071	2.111973
	Std. Dev	2.560828	2.39313	2.469293	2.42819	2.50288	2.498594
$f_8(x)$	Mean	0.000577	$1.8201e^{-3}$	$7.9875e^{-3}$	$9.1103e^{-03}$	$4.7793e^{-03}$	0.000572
	Std. Dev	0.000222	$6.7301e^{-3}$	0.000324	$5.8018e^{-03}$	$4.5842e^{-03}$	0.000324
$f_9(x)$	Mean	−1.03163	-1.03163	−1.03163	−1.03163	-1.031614	−1.03163
	Std. Dev	$3.9802e^{-7}$	$9.40823e^{-7}$	$5.337e^{-07}$	$8.27939e^{-7}$	$6.3347e^{-07}$	$4.2e^{-07}$

$f_{10}(x)$	Mean	−3.86278	-3.87073	−3.88939	-3.86641	-3.8628	−3.85616
	Std. Dev	$2.58e^{-15}$	0.006392	0.005492	0.00497	0.006827	0.002706
$f_{11}(x)$	Mean	−3.26634	-3.2902	−3.10773	-2.9863	-3.2792	−2.98105
	Std. Dev	0.060516	$4.668e^{-2}$	0.47991	0.53814	0.42775	0.376653
$f_{12}(x)$	Mean	0.006675	3.0142	2.8085	1.14322	3.3901	1.889015
	Std. Dev	0.008907	2.69025	1.59925	2.831	2.1682	0.266088
$f_{13}(x)$	Mean	46.70423	20.792	$2.8310e^{1}$	$4.8429e^{1}$	21.6331	0.000289
	Std. Dev	11.62938	3.0742	$1.330e^{1}$	$2.4031e^{1}$	10.7601	0.001586
$f_{14}(x)$	Mean	0.009215	$1.1711e^{+00}$	$3.0996e^{+01}$	$9.3869e^{+01}$	$9.2549e^{+00}$	0.0000
	Std. Dev	0.007724	$2.9271e^{-02}$	$2.2269e^{+00}$	$3.0447e^{+00}$	$3.3997e^{-01}$	0.0000
$f_{15}(x)$	Mean	0.276015	$1.5884e^{+01}$	$2.0681e^{+01}$	$1.9896e^{+01}$	$1.2795e^{+01}$	7.4043
	Std. Dev	0.50901	$1.2211e^{+00}$	$3.8721e^{-02}$	$5.3227e^{-01}$	$8.4147e^{-01}$	9.897572
$f_{16}(x)$	Mean	−4841.29	-5658.37	−5490.76	-5197.0	-5509.7	−5080.76
	Std. Dev	1152.814	7203.56	242.778	8920.93	763.32	695.7968
$f_{17}(x)$	Mean	0.397887	$3.9789e^{-01}$	$3.9789e^{-01}$	$3.7481e^{-01}$	$3.9789e^{-01}$	0.397914
	Std. Dev	0.000	$1.781e^{-01}$	$2.925e^{-01}$	$8.6588e^{-01}$	$3.77e^{-01}$	$2.7e^{-05}$
$f_{18}(x)$	Mean	3.00	3.0	3.0	3.0	3.0	3
	Std. Dev	$1.33e^{-15}$	$5.541e^{-10}$	$3.143e^{-11}$	$8.2517e^{-9}$	$6.0247e^{-9}$	$4.22e^{-15}$
$f_{20}(x)$	Mean	0.006917	$5.921e^{-1}$	0.35738	$8.92113e^{-1}$	$2.5739e^{-1}$	0.339676
	Std. Dev	0.026301	$1.5730e^{-1}$	0.37949	$6.1135e^{-2}$	0.835834	0.214864
$f_{21}(x)$	Mean	−6.8651	−7.1892	−7.2940	−8.0942	-7.3440	−7.04918
	Std. Dev	3.019644	3.519303	3.70638	$6.57e^{-02}$	3.7929	3.629551
$f_{22}(x)$	Mean	−8.45653	−8.6903	−8.2947	−8.5783	-7.53978	−8.18178
	Std. Dev	3.087094	3.02792	3.5013	3.4935	3.79391	3.829202
$f_{23}(x)$	Mean	−9.95291	−9.6938	−9.89299	−9.47492	-9.44929	−9.34238
	Std. Dev	1.782786	1.893683	2.50027	2.68282	2.472820	2.414737

VI. TEAM 22 AN ELECTROMAGNETIC DESIGN PROBLEM:

A. *Description of the Problem:*

The SMES (Superconducting Magnetic Energy Storage System) is a design problem that is formulated to store energy in magnetic fields that were generated by current densities in their superconducting coil system. It is famous with the name of "TEAM 22 problem" as it was presented by the team in a workshop where its number is 22. TEAM 22 is an optimization version of the SMES and it is used widely in magnetostatics as a benchmark problem. The system comprises two coils carrying current in the opposite direction. The outer shielding and the inner main solenoid are used to reduce the effect of the stray field. The configuration of the system is shown in figure.3.

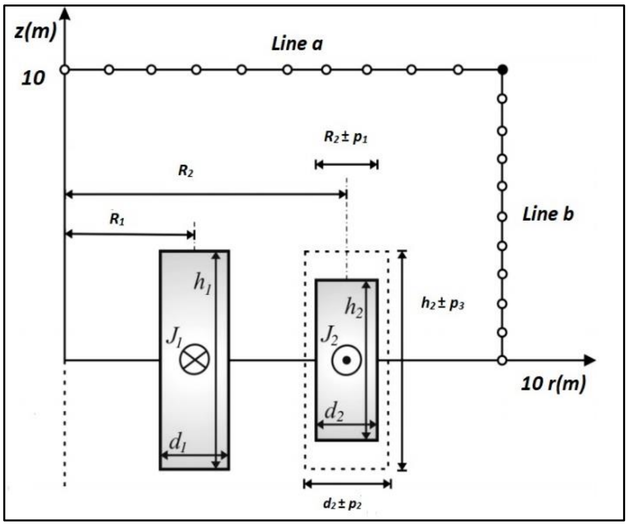

Figure 3 Configuration of Team 22 Problem

So, the primary purpose of the problem is to store the required amount of energy keeping the negligible stray field. To meet the above objective following conditions must be considered.

- The energy stored should be 180 MJ.
- The produced magnetic field inside the solenoids must not violate a certain physical condition that confirms superconductivity.
- The average stray field along "line a" and "line b" at 10 meters should be minimal.

The two objectives are mapped into a single objective function by:

$$min\ F = \frac{B_{stray}^2}{B_{norm}^2} + \frac{|E - E_{ref}|}{E_{ref}} \qquad (31)$$

where E_{ref} = 180 MJ, B_{norm} = 200 μT and B^2_{stray} is defined as:

$$B_{stray}^2 = \frac{\sum_{i=1}^{22} |B_{stray,i}|^2}{22} \qquad (32)$$

Some of the design constraints are that the solenoids should not overlap with each other.

$$R_1 + \frac{d_1}{2} < R_2 - \frac{d_2}{2} \qquad (34)$$

The superconducting material should not violate (35) condition that links together the value of the current density and the maximum value of magnetic flux density.

$$|J| = (-6.4|B| + 54) A/mm^2 \qquad (35)$$

B. *Results and discussion:*

The performance of the SI algorithm (Standard PSO) is tested on an electromagnetic design problem. Moreover, some other techniques are also compared with the SI method that is available in the literature. A modified version of the PSO algorithm that is developed by the authors previously is also used to solve the TEAM 22 problem. The design parameters for the problem are defined in table V.

Table V shows the parameters of the TEAM 22 Problem.

	R_1 *(m)*	R_2 *(m)*	$h_1/2$ *(m)*	$h_2/2$ *(m)*	d_1 *(m)*	d_2 *(m)*	J_1 A/mm^2	J_2 A/mm^2
Fixed	2	-	0.8	-	0.27	-	22.5	-22.5
Minimum	-	2.6	-	0.204	-	0.1	-	-
Maximum	-	3.4	-	1.1	-	0.4	-	-

Table VI shows the comparison between the standard SI methods, some state of the art algorithms applied to this problem previously and advanced PSO presented in [161] by the authors.

Table VI Comparison of the SI methods, state-of-the-art algorithms, and APSO for TEAM 22 Problem.

Methods	R_2 *(m)*	d_2 *(m)*	B^2_{stray} *(nT)*	$h_2/2$ *(m)*	Fitness value
APSO [161]	2.683	0.1792	0.06291	0.3083	$7.972e^{-5}$
Standard PSO	2.950	0.3853	6.345	0.529	0.5379
Standard ABC	3.429	0.3727	5.3915	0.8683	0.7738
HAL Method [162]	2.78	0.11	4.55	1.479	$5.16e^{-4}$
MQPSO [163]	3.139	0.2871	-	0.316	0.0716
MOCSA [164]	4.0355	0.2470	-	0.4753	$3.12e^{-3}$
MPSO[165]	2.9927	0.2976	-	0.2139	0.0929
POS[166]	3.080	0.394	0.7913	0.478	0.0881
Bayes Opt [167]	3.093	0.369	0.7736	0.500	0.0917
EQBPSO [168]	1.800	0.2851	1.79	1.800	1.1730
Team22 Standard Problem [169]	1.80	0.195	0.0724	1.513	0.0018

The result analysis validates the proficiency of the APSO algorithm that is presented by the authors in [161] as compared to the other state-of-the-art techniques available in the literature.

VII. CONCLUSIONS

In this paper, a summary of the famous SI methods is presented. These SI methods have been used to solve different problems related to diverse fields. Moreover, an electromagnetic inverse problem is solved by using the APSO algorithm that has been proposed by the authors previously, some SI methods, and it is also compared with the other state-of-the-art techniques available in the literature. The results obtained show that the APSO has better performance than the other methods.

REFERENCES

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz, *From Natural to Artificial Swarm Intelligence*. Oxford University Press, Inc., 1999.

[2] A. Chakraborty and A. K. Kar, "Swarm Intelligence: A Review of Algorithms," in *Nature-Inspired Computing and Optimization: Theory and Applications*, S. Patnaik, X.-S. Yang, and K. Nakamatsu Eds. Cham: Springer International Publishing, 2017, pp. 475-494.

[3] X. Li and M. Clerc, "Swarm Intelligence," in *Handbook of Metaheuristics*, M. Gendreau and J.-Y. Potvin Eds. Cham: Springer International Publishing, 2019, pp. 353-384.

[4] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*. 2010.

[5] M. Mitchell and C. E. Taylor, "Evolutionary Computation: An Overview," *Annual Review of Ecology and Systematics*, vol. 30, no. 1, pp. 593-616, 1999, DOI: 10.1146/annurev.ecolsys.30.1.593.

[6] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. MIT Press, 1975.

[7] J. R. Koza and R. Poli, "Genetic Programming," in *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, E. K. Burke and G. Kendall Eds. Boston, MA: Springer US, 2005, pp. 127-164.

[8] T. Bäck and F. Hoffmeister, "Basic aspects of evolution strategies," *Statistics and Computing*, vol. 4, no. 2, pp. 51-63, 1994/06/01 1994, DOI: 10.1007/BF00175353.

[9] D. B. Fogel, "An Overview of Evolutionary Programming," in *Evolutionary Algorithms*, L. D. Davis, K. De Jong, M. D. Vose, and L. D. Whitley Eds. New York, NY: Springer New York, 1999, pp. 89-109.

[10] A. E. Eiben and J. E. Smith, "Evolutionary Programming," in *Introduction to Evolutionary Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 89-99.

[11] M. Dorigo, "Optimization, Learning and Natural Algorithms," *Ph.D. Thesis, Politecnico di Milano, Italy*, 1992 1992. [Online]. Available: https://ci.nii.ac.jp/naid/10016599043/en/.

[12] Y. Fan, G. Wang, X. Lu, and G. Wang, "Distributed forecasting and ant colony optimization for the bike-sharing rebalancing problem with unserved demands," *PLOS ONE*, vol. 14, no. 12, p. e0226204, 2020, doi: 10.1371/journal.pone.0226204.

[13] M. Dorigo and T. Stützle, "Ant Colony Optimization: Overview and Recent Advances," in *Handbook of Metaheuristics*, M. Gendreau and J.-Y. Potvin Eds. Boston, MA: Springer US, 2010, pp. 227-263.

[14] D. HANSFORD, "MOB MENTALITY," no. 123.

[15] Q. Yang *et al.*, "Adaptive Multimodal Continuous Ant Colony Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 191-205, 2017, doi: 10.1109/TEVC.2016.2591064.

[16] D. Zhang, X. You, S. Liu, and K. Yang, "Multi-Colony Ant Colony Optimization Based on Generalized Jaccard Similarity Recommendation Strategy," *IEEE Access*, vol. 7, pp. 157303-157317, 2019, doi: 10.1109/ACCESS.2019.2949860.

[17] J. Shang *et al.*, "A Review of Ant Colony Optimization Based Methods for Detecting Epistatic Interactions," *IEEE Access*, vol. 7, pp. 13497-13509, 2019, doi: 10.1109/ACCESS.2019.2894676.

[18] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, 1997, doi: 10.1109/4235.585892.

[19] T. Stützle and H. H. Hoos, "MAX–MIN Ant System," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889-914, 2000/06/01/ 2000, doi: https://doi.org/10.1016/S0167-739X(00)00043-1.

[20] R. Jian, Y. Chen, and T. Chen, "Multi-Parameters Unified-Optimization for Millimeter-Wave Microstrip Antenna Based on ICACO," *IEEE Access*, vol. 7, pp. 53012-53017, 2019, doi: 10.1109/ACCESS.2019.2912461.

[21] X. Wang, H. Gu, Y. Liu, and H. Zhang, "A Two-Stage RPSO-ACS Based Protocol: A New Method for Sensor Network Clustering and Routing in Mobile Computing," *IEEE Access*, vol. 7, pp. 113141-113150, 2019, doi: 10.1109/ACCESS.2019.2933150.

[22] H. Zhang, X. Wang, P. Memarmoshrefi, and D. Hogrefe, "A Survey of Ant Colony Optimization Based Routing Protocols for Mobile Ad Hoc Networks," *IEEE Access*, vol. 5, pp. 24139-24161, 2017, doi: 10.1109/ACCESS.2017.2762472.

[23] H. Wang, Z. A. Wang, L. Yu, X. Wang, and C. Liu, "Ant Colony Optimization with Improved Potential Field Heuristic for Robot Path Planning," in *2018 37th Chinese Control Conference (CCC)*, 25-27 July 2018 2018, pp. 5317-5321, doi: 10.23919/ChiCC.2018.8483844.

[24] Y. Huang, Y. Gu, and Z. Zheng, "Research on the Path Planning of Hair-Insertion Robot Arm Based on Ant Colony Optimization," in *2018 37th Chinese Control Conference (CCC)*, 25-27 July 2018 2018, pp. 5191-5195, doi: 10.23919/ChiCC.2018.8483149.

[25] R. Singh and L. B. Prasad, "Optimal Trajectory Tracking of Robotic Manipulator using Ant Colony Optimization," in *2018 5th IEEE Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, 2-4 Nov. 2018 2018, pp. 1-6, doi: 10.1109/UPCON.2018.8597087.

[26] W. Zhu, P. Hou, L. Chang, and X. Xu, "Disjunctive Belief Rule Base Optimization by Ant Colony Optimization for Railway Transportation Safety Assessment," in *2019 Chinese Control And Decision Conference (CCDC)*, 3-5 June 2019 2019, pp. 6120-6124, doi: 10.1109/CCDC.2019.8833179.

[27] J. Eaton, S. Yang, and M. Gongora, "Ant Colony Optimization for Simulated Dynamic Multi-Objective Railway Junction Rescheduling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 11, pp. 2980-2992, 2017, doi: 10.1109/TITS.2017.2665042.

[28] M. Mavrovouniotis, S. Yang, M. Van, C. Li, and M. Polycarpou, "Ant Colony Optimization Algorithms for Dynamic Optimization: A Case Study of the Dynamic Travelling Salesperson Problem [Research Frontier]," *IEEE Computational Intelligence Magazine*, vol. 15, no. 1, pp. 52-63, 2020, doi: 10.1109/MCI.2019.2954644.

[29] C. Ratanavilisagul, "Modified Ant Colony Optimization with pheromone mutation for travelling salesman problem," in *2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 27-30 June 2017 2017, pp. 411-414, doi: 10.1109/ECTICon.2017.8096261.

[30] M. Mavrovouniotis, F. M. Müller, and S. Yang, "Ant Colony Optimization With Local Search for Dynamic Traveling Salesman Problems," *IEEE Transactions on Cybernetics*, vol. 47, no. 7, pp. 1743-1756, 2017, doi: 10.1109/TCYB.2016.2556742.

[31] R. Contreras, M. A. Pinninghoff, and J. Ortega, "Using Ant Colony Optimization for Edge Detection in Gray Scale Images," Berlin, Heidelberg, 2013: Springer Berlin Heidelberg, in Natural and Artificial Models in Computation and Biology, pp. 323-331.

[32] S. Kaur and P. Kaur, "An Edge detection technique with image segmentation using Ant Colony Optimization: A review," in *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, 19-19 Nov. 2016 2016, pp. 1-5, doi: 10.1109/GET.2016.7916741.

[33] U. J, N. Metawa, K. Shankar, and S. K. Lakshmanaprabu, "Financial crisis prediction model using ant colony optimization," *International Journal of Information Management*, vol. 50, pp. 538-556, 2020/02/01/ 2020, doi: https://doi.org/10.1016/j.ijinfomgt.2018.12.001.

[34] Y. Marinakis, M. Marinaki, M. Doumpos, and C. Zopounidis, "Ant colony and particle swarm optimization for financial

classification problems," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10604-10611, 2009/09/01/ 2009, doi: https://doi.org/10.1016/j.eswa.2009.02.055.
[35] R. Kleinkauf, M. Mann, and R. Backofen, "antaRNA: ant colony-based RNA sequence design," *Bioinformatics*, vol. 31, no. 19, pp. 3114-3121, 2015, doi: 10.1093/bioinformatics/btv319.
[36] D. Do Duc, H. Q. Dinh, T. H. Dang, K. Laukens, and X. H. Hoang, "AcoSeeD: An Ant Colony Optimization for Finding Optimal Spaced Seeds in Biological Sequence Search," Berlin, Heidelberg, 2012: Springer Berlin Heidelberg, in Swarm Intelligence, pp. 204-211.
[37] D. Karaboga, "An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical Report - TR06," *Technical Report, Erciyes University*, 01/01 2005.
[38] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459-471, 2007/11/01 2007, doi: 10.1007/s10898-007-9149-x.
[39] D. Karaboga and B. Akay, "A comparative study of Artificial Bee Colony algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108-132, 2009/08/01/ 2009, doi: https://doi.org/10.1016/j.amc.2009.03.090.
[40] Y. Gao, "An Improved Hybrid Group Intelligent Algorithm Based on Artificial Bee Colony and Particle Swarm Optimization," in *2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*, 10-11 Aug. 2018 2018, pp. 160-163, doi: 10.1109/ICVRIS.2018.00046.
[41] B. Wang and L. Wang, "A Novel Artificial Bee Colony Algorithm for Numerical Function Optimization," in *2012 Fourth International Conference on Computational and Information Sciences*, 17-19 Aug. 2012 2012, pp. 172-175, doi: 10.1109/ICCIS.2012.32.
[42] F. Chengli, F. Qiang, L. Guangzheng, and X. Qinghua, "Hybrid artificial bee colony algorithm with variable neighborhood search and memory mechanism," *Journal of Systems Engineering and Electronics*, vol. 29, no. 2, pp. 405-414, 2018, doi: 10.21629/JSEE.2018.02.20.
[43] B. Akay and D. Karaboga, "A modified Artificial Bee Colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120-142, 2012/06/01/ 2012, doi: https://doi.org/10.1016/j.ins.2010.07.015.
[44] W.-f. Gao and S.-y. Liu, "A modified artificial bee colony algorithm," *Computers & Operations Research*, vol. 39, no. 3, pp. 687-697, 2012/03/01/ 2012, doi: https://doi.org/10.1016/j.cor.2011.06.007.
[45] L. Wang, X. Zhang, and X. Zhang, "Antenna Array Design by Artificial Bee Colony Algorithm With Similarity Induced Search Method," *IEEE Transactions on Magnetics*, vol. 55, no. 6, pp. 1-4, 2019, doi: 10.1109/TMAG.2019.2896921.
[46] H. Liang and H. Jiang, "The Modified Artificial Bee Colony-Based SLM Scheme for PAPR Reduction in OFDM Systems," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, 11-13 Feb. 2019 2019, pp. 504-508, doi: 10.1109/ICAIIC.2019.8669020.
[47] A. Salman, I. M. Qureshi, S. Saleem, and S. Saeed, "Optimization of Resource Allocation for Heterogeneous Services in OFDM Based Cognitive Radio Networks Using Artificial Bee Colony," in *2019 International Symposium on Recent Advances in Electrical Engineering (RAEE)*, 28-29 Aug. 2019 2019, vol. 4, pp. 1-5, doi: 10.1109/RAEE.2019.8886951.
[48] A. Rekaby, A. A. Youssif, and A. S. Eldin, "Introducing Adaptive Artificial Bee Colony algorithm and using it in solving traveling salesman problem," in *2013 Science and Information Conference*, 7-9 Oct. 2013 2013, pp. 502-506.
[49] Y. Wang, "Improving Artificial Bee Colony and Particle Swarm Optimization to Solve TSP Problem," in *2018 International Conference on Virtual Reality and Intelligent Systems (ICVRIS)*, 10-11 Aug. 2018 2018, pp. 179-182, doi: 10.1109/ICVRIS.2018.00051.
[50] D. Kumar, A. Mishra, and K. Chatterjee, "Power and frequency control of a wind energy power system using artificial bee colony algorithm," in *2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM)*, 23-24 March 2017 2017, pp. 561-565, doi: 10.1109/ICONSTEM.2017.8261385.
[51] Ç. M and A. Kaygusuz, "Optimum Fuel Cost in Load Flow Analysis of Smart Grid by Using Artificial Bee Colony Algorithm," in *2019 International Artificial Intelligence and Data Processing Symposium (IDAP)*, 21-22 Sept. 2019 2019, pp. 1-5, doi: 10.1109/IDAP.2019.8875893.
[52] Z. Salehahmadi and A. Manafi, "How can bee colony algorithm serve medicine?," (in eng), *World J Plast Surg*, vol. 3, no. 2, pp. 87-92, 2014. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25489530 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4236990/.
[53] G. S. Gopika, J. Shanthini, and S. Karthik, "Hybrid Approach for the Brain Tumors Detection & Segmentation Using Artificial Bee Colony Optimization with FCM," in *2018 International Conference on Soft-computing and Network Security (ICSNS)*, 14-16 Feb. 2018 2018, pp. 1-5, doi: 10.1109/ICSNS.2018.8573648.
[54] T. Keerthika, "A Hybrid Fish – Bee Optimization Algorithm for Heart Disease Prediction using Multiple Kernel SVM Classifier," 08/31 2019.
[55] Q. Khalid and M. Arshad Bangash, "An Artificial Bee Colony Algorithm Based on a Multi-Objective Framework for Supplier Integration," *Applied Sciences*, vol. 9, 02/11 2019, doi: 10.3390/app9030588.
[56] D. Xiaoyi, "An Efficient Hybrid Artificial Bee Colony Algorithm for Customer Segmentation in Mobile E-commerce," *Journal of Electronic Commerce in Organizations (JECO)*, vol. 11, no. 2, pp. 53-63, 2013, doi: 10.4018/jeco.2013040105.
[57] E. Cuevas, F. Sención-Echauri, D. Zaldivar, and M. Pérez, "Image Segmentation Using Artificial Bee Colony Optimization," in *Handbook of Optimization: From Classical to Modern Approach*, I. Zelinka, V. Snášel, and A. Abraham Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 965-990.
[58] A. Yimit, Y. Hagihara, T. Miyoshi, and Y. Hagihara, *Automatic image enhancement by artificial bee colony algorithm* (2012 International Conference on Graphic and Image Processing). SPIE, 2013.
[59] X. Yang and D. Suash, "Cuckoo Search via Lévy flights," in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, 9-11 Dec. 2009 2009, pp. 210-214, doi: 10.1109/NABIC.2009.5393690.
[60] X.-S. Yang and S. Deb, "Multiobjective cuckoo search for design optimization," *Computers & Operations Research*, vol. 40, no. 6, pp. 1616-1624, 2013/06/01/ 2013, doi: https://doi.org/10.1016/j.cor.2011.09.026.
[61] M. Mareli and B. Twala, "An adaptive Cuckoo search algorithm for optimisation," *Applied Computing and Informatics*, vol. 14, no. 2, pp. 107-115, 2018/07/01/ 2018, doi: https://doi.org/10.1016/j.aci.2017.09.001.
[62] S. Walton, O. Hassan, K. Morgan, and M. R. Brown, "Modified cuckoo search: A new gradient-free optimisation algorithm," *Chaos, Solitons & Fractals*, vol. 44, no. 9, pp. 710-718, 2011/09/01/ 2011, doi: https://doi.org/10.1016/j.chaos.2011.06.004.
[63] A. Layeb and S. R. Boussalia, "A Novel Quantum Inspired Cuckoo Search Algorithm for Bin Packing Problem," *International Journal of Information Technology and Computer Science*, vol. 4, pp. 58-67, 05/05 2012, doi: 10.5815/ijitcs.2012.05.08.
[64] W. Han, X. S. Lu, M. Zhou, X. Shen, J. Wang, and J. Xu, "An Evaluation and Optimization Methodology for Efficient Power Plant Programs," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 2, pp. 707-716, 2020, doi: 10.1109/TSMC.2017.2714198.
[65] D. A. Nugraha, K. L. Lian, and Suwarno, "A Novel MPPT Method Based on Cuckoo Search Algorithm and Golden Section Search Algorithm for Partially Shaded PV System," *Canadian Journal of Electrical and Computer Engineering*, vol. 42, no. 3, pp. 173-182, 2019, doi: 10.1109/CJECE.2019.2914723.
[66] G. Sun, Y. Liu, Z. Chen, S. Liang, A. Wang, and Y. Zhang, "Radiation Beam Pattern Synthesis of Concentric Circular Antenna Arrays Using Hybrid Approach Based on Cuckoo Search," *IEEE Transactions on Antennas and Propagation*, vol. 66, no. 9, pp. 4563-4576, 2018, doi: 10.1109/TAP.2018.2846771.

[67] G. Sun, Y. Liu, J. Li, Y. Zhang, and A. Wang, "Sidelobe reduction of the large-scale antenna array for 5G beamforming via hierarchical cuckoo search," *Electronics Letters,* vol. 53, no. 16, pp. 1158-1160, 2017, doi: 10.1049/el.2016.4768.

[68] P. Savsani, R. L. Jhala, and V. J. Savsani, "Comparative Study of Different Metaheuristics for the Trajectory Planning of a Robotic Arm," *IEEE Systems Journal,* vol. 10, no. 2, pp. 697-708, 2016, doi: 10.1109/JSYST.2014.2342292.

[69] S. Laha, "A quantum-inspired cuckoo search algorithm for the travelling salesman problem," in *2015 International Conference on Computing, Communication, and Security (ICCCS)*, 4-5 Dec. 2015 2015, pp. 1-6, doi: 10.1109/CCCS.2015.7374201.

[70] N. A. Jebril and Q. Abu Al-Haija, "Cuckoo Optimization Algorithm (COA) for Image Processing," in *Nature-Inspired Optimization Techniques for Image Processing Applications*, J. Hemanth and V. E. Balas Eds. Cham: Springer International Publishing, 2019, pp. 189-213.

[71] A. S. Ashour, S. Samanta, N. Dey, N. Kausar, W. B. Abdessalemkaraa, and A. E. Hassanien, "Computed Tomography Image Enhancement Using Cuckoo Search: A Log Transform Based Approach," *Journal of Signal and Information Processing,* vol. Vol.06No.03, p. 14, 2015, Art no. 59273, doi: 10.4236/jsip.2015.63023.

[72] H. H. Issa and S. M. E. Ahmed, "FPGA Implementation of Floating Point-Based Cuckoo Search Algorithm," *IEEE Access,* vol. 7, pp. 134434-134447, 2019, doi: 10.1109/ACCESS.2019.2942205.

[73] K. N. Krishnanand and D. Ghose, "Glowworm swarm optimisation: a new method for optimising multi-modal functions," *Int. J. Comput. Intell. Stud.*, vol. 1, no. 1, pp. 93–119, 2009, doi: 10.1504/ijcistudies.2009.025340.

[74] K. N. Krishnanand and D. Ghose, "Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions," *Swarm Intelligence,* vol. 3, no. 2, pp. 87-124, 2009/06/01 2009, doi: 10.1007/s11721-008-0021-5.

[75] B. Wu, C. Qian, W. Ni, and S. Fan, "The improvement of glowworm swarm optimization for continuous optimization problems," *Expert Systems with Applications,* vol. 39, no. 7, pp. 6335-6342, 2012/06/01/ 2012, doi: https://doi.org/10.1016/j.eswa.2011.12.017.

[76] S. A. Ludwig, "Improved glowworm swarm optimization algorithm applied to multi-level thresholding," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 24-29 July 2016 2016, pp. 1533-1540, doi: 10.1109/CEC.2016.7743971.

[77] P. Qiong, Y. Liao, P. Hao, X. He, and C. Hui, "A Self-Adaptive Step Glowworm Swarm Optimization Approach," *International Journal of Computational Intelligence and Applications,* vol. 18, no. 01, p. 1950004, 2019, doi: 10.1142/s1469026819500044.

[78] X. Zheng, Z. Gui, and Y. Wang, "Support vector machine model based on glowworm swarm optimization in the application of vibrant fault diagnosis for the hydro-turbine generating unit," in *2017 IEEE 3rd Information Technology and Mechatronics Engineering Conference (ITOEC)*, 3-5 Oct. 2017 2017, pp. 238-141, doi: 10.1109/ITOEC.2017.8122427.

[79] J. Senthilnath, S. N. Omkar, V. Mani, N. Tejovanth, P. G. Diwakar, and A. S. B, "Multi-spectral satellite image classification using Glowworm Swarm Optimization," in *2011 IEEE International Geoscience and Remote Sensing Symposium*, 24-29 July 2011 2011, pp. 47-50, doi: 10.1109/IGARSS.2011.6048894.

[80] Y.-Q. Zhou, Z. Ouyang, J. Liu, and G. Sang, "A Novel K-means Image Clustering Algorithm Based on Glowworm Swarm Optimization," *Electrical Review,* vol. 88, 01/01 2012.

[81] T. Zeng, Y. Hua, X. Zhao, and T. Liu, "Research on glowworm swarm optimization localization algorithm based on wireless sensor network," in *2016 IEEE International Frequency Control Symposium (IFCS)*, 9-12 May 2016 2016, pp. 1-5, doi: 10.1109/FCS.2016.7546730.

[82] H. Jiang and X. Tang, "Polarimetric MIMO radar target detection based on glowworm swarm optimization algorithm," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4-9 May 2014 2014, pp. 805-809, doi: 10.1109/ICASSP.2014.6853708.

[83] Y. Zhang, X. Ma, and Y. Miao, "Localization of multiple odor sources using modified glowworm swarm optimization with collective robots," in *Proceedings of the 30th Chinese Control Conference*, 22-24 July 2011 2011, pp. 1899-1904.

[84] K. N. Krishnanand and D. Ghose, "A Glowworm Swarm Optimization Based Multi-robot System for Signal Source Localization," in *Design and Control of Intelligent Robotic Systems*, D. Liu, L. Wang, and K. C. Tan Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 49-68.

[85] N. N. Quang, E. R. Sanseverino, M. L. D. Silvestre, A. Madonia, C. Li, and J. M. Guerrero, "Optimal power flow based on glow worm-swarm optimization for three-phase islanded microgrids," in *2014 AEIT Annual Conference - From Research to Industry: The Need for a More Effective Technology Transfer (AEIT)*, 18-19 Sept. 2014 2014, pp. 1-6, doi: 10.1109/AEIT.2014.7002028.

[86] S. Surender Reddy and C. Srinivasa Rathnam, "Optimal Power Flow using Glowworm Swarm Optimization," *International Journal of Electrical Power & Energy Systems,* vol. 80, pp. 128-139, 2016/09/01/ 2016, doi: https://doi.org/10.1016/j.ijepes.2016.01.036.

[87] X. Wang, K. Yang, and X. Zhou, "Two-stage glowworm swarm optimisation for economical operation of hydropower station," *IET Renewable Power Generation,* vol. 12, no. 9, pp. 992-1003, 2018, doi: 10.1049/iet-rpg.2017.0466.

[88] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No.00TH8512)*, 16-19 July 2000 2000, vol. 1, pp. 84-88 vol.1, doi: 10.1109/CEC.2000.870279.

[89] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 4-9 May 1998 1998, pp. 69-73, doi: 10.1109/ICEC.1998.699146.

[90] S. Sedarous, S. M. El-Gokhy, and E. Sallam, "Multi-swarm multi-objective optimization based on a hybrid strategy," *Alexandria Engineering Journal,* vol. 57, no. 3, pp. 1619-1629, 2018/09/01/ 2018, doi: https://doi.org/10.1016/j.aej.2017.06.017.

[91] L. Lizzi, F. Viani, R. Azaro, and A. Massa, "Optimization of a Spline-Shaped UWB Antenna by PSO," *IEEE Antennas and Wireless Propagation Letters,* vol. 6, pp. 182-185, 2007, doi: 10.1109/LAWP.2007.894157.

[92] Y. Li, W. Shao, L. You, and B. Wang, "An Improved PSO Algorithm and Its Application to UWB Antenna Design," *IEEE Antennas and Wireless Propagation Letters,* vol. 12, pp. 1236-1239, 2013, doi: 10.1109/LAWP.2013.2283375.

[93] Z. Wang, T. Zhang, L. Kong, and G. Cui, "Prediction-based PSO algorithm for MIMO radar antenna deployment in a dynamic environment," *The Journal of Engineering,* vol. 2019, no. 20, pp. 6646-6650, 2019, doi: 10.1049/joe.2019.0188.

[94] E. Masehian and D. Sedighizadeh, "An Improved Particle Swarm Optimization Method for Motion Planning of Multiple Robots," in *Distributed Autonomous Robotic Systems: The 10th International Symposium*, A. Martinoli *et al.* Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 175-188.

[95] N. A. A. Aziz and Z. Ibrahim, "Asynchronous Particle Swarm Optimization for Swarm Robotics," *Procedia Engineering,* vol. 41, pp. 951-957, 2012/01/01/ 2012, doi: https://doi.org/10.1016/j.proeng.2012.07.268.

[96] A. Ayari and S. Bouamama, "A new multiple robot path planning algorithm: dynamic distributed particle swarm optimization," *Robotics and Biomimetics,* vol. 4, no. 1, p. 8, 2017/11/02 2017, doi: 10.1186/s40638-017-0062-6.

[97] K. Venkatalakshmi and S. M. Shalinie, "A customized Particle Swarm Optimization algorithm for image enhancement," in *2010 INTERNATIONAL CONFERENCE ON COMMUNICATION CONTROL AND COMPUTING TECHNOLOGIES*, 7-9 Oct. 2010 2010, pp. 603-607, doi: 10.1109/ICCCCT.2010.5670768.

[98] T. R. Farshi, J. H. Drake, and E. Özcan, "A multimodal particle swarm optimization-based approach for image segmentation," *Expert Systems with Applications,* vol. 149, p. 113233, 2020/07/01/ 2020, doi: https://doi.org/10.1016/j.eswa.2020.113233.

[99] F. Mohsen, M. M. Hadhoud, and K. Amin, "A new image segmentation method based on particle swarm optimization,"

International Arab Journal of Information Technology, vol. 9, 09/01 2012.
[100] A. Esmin and G. Lambert-Torres, "Application of particle swarm optimization to an optimal power system," *International Journal of Innovative Computing, Information, and Control*, vol. 8, 03/01 2012.
[101] T. K. Das and G. K. Venayagamoorthy, "Optimal design of power system stabilizers using a small population-based PSO," in *2006 IEEE Power Engineering Society General Meeting*, 18-22 June 2006 2006, p. 7 pp., doi: 10.1109/PES.2006.1709322.
[102] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *IEEE Transactions on Power Systems*, vol. 15, no. 4, pp. 1232-1239, 2000, doi: 10.1109/59.898095.
[103] P. Pandey and S. Soni, "Enhance Clustering Approach using PSO-A* for E-Commerce," *International Journal of Computer Applications*, vol. 182, pp. 57-60, 01/17 2019, doi: 10.5120/ijca2019918405.
[104] W. Yang, Q. Xie, and M. Li, "Inventory Control Method of Reverse Logistics for Shipping Electronic Commerce Based on Improved Multi-objective Particle Swarm Optimization Algorithm," *Journal of Coastal Research*, vol. 83, no. SI, pp. 786-790, 2018, doi: 10.2112/si83-128.1.
[105] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 65-74.
[106] Y. Wang *et al.*, "A Novel Bat Algorithm with Multiple Strategies Coupling for Numerical Optimization," *Mathematics*, vol. 7, no. 2, p. 135, 2019. [Online]. Available: https://www.mdpi.com/2227-7390/7/2/135.
[107] S. Swayamsiddha, Prateek, S. S. Singh, S. Parija, and D. K. Pratihar, "Reporting cell planning-based cellular mobility management using a Binary Artificial Bat algorithm," *Heliyon*, vol. 5, no. 3, p. e01276, 2019/03/01/ 2019, doi: https://doi.org/10.1016/j.heliyon.2019.e01276.
[108] C. K. Ng, C. H. Wu, W. H. Ip, and K. L. Yung, "A Smart Bat Algorithm for Wireless Sensor Network Deployment in 3-D Environment," *IEEE Communications Letters*, vol. 22, no. 10, pp. 2120-2123, 2018, doi: 10.1109/LCOMM.2018.2861766.
[109] B. R. Adarsh, T. Raghunathan, T. Jayabarathi, and X.-S. Yang, "Economic dispatch using chaotic bat algorithm," *Energy*, vol. 96, pp. 666-675, 2016/02/01/ 2016, doi: https://doi.org/10.1016/j.energy.2015.12.096.
[110] S. Biswal, A. K. Barisal, A. Behera, and T. Prakash, "Optimal power dispatch using BAT algorithm," in *2013 International Conference on Energy Efficient Technologies for Sustainability*, 10-12 April 2013 2013, pp. 1018-1023, doi: 10.1109/ICEETS.2013.6533526.
[111] M. Rahmani, A. Ghanbari, and M. M. Ettefagh, "Robust adaptive control of a bio-inspired robot manipulator using bat algorithm," *Expert Systems with Applications*, vol. 56, pp. 164-176, 2016/09/01/ 2016, doi: https://doi.org/10.1016/j.eswa.2016.03.006.
[112] M. Rahmani, A. Ghanbari, and M. M. Ettefagh, "A novel adaptive neural network integral sliding-mode control of a biped robot using bat algorithm," *Journal of Vibration and Control*, vol. 24, no. 10, pp. 2045-2060, 2018, doi: 10.1177/1077546316676734.
[113] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 25–34, 1987, doi: 10.1145/37402.37406.
[114] H. A. Abbass, "MBO: marriage in honey bees optimization-a Haplometrosis polygynous swarming approach," in *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546)*, 27-30 May 2001 2001, vol. 1, pp. 207-214 vol. 1, doi: 10.1109/CEC.2001.934391.
[115] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52-67, 2002, doi: 10.1109/MCS.2002.1004010.
[116] S. D. Muller, J. Marchetto, S. Airaghi, and P. Kournoutsakos, "Optimization based on bacterial chemotaxis," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 16-29, 2002, doi: 10.1109/4235.985689.
[117] X. Li, Z. Shao, and J. I. Qian, "An optimizing method based on autonomous animate: Fish swarm algorithm," *System Engineering Theory and Practice*, vol. 22, pp. 32-38, 11/01 2002.
[118] M. M. Eusuff and K. E. Lansey, "Optimization of Water Distribution Network Design Using the Shuffled Frog Leaping Algorithm," *Journal of Water Resources Planning and Management*, vol. 129, no. 3, pp. 210-225, 2003, doi: doi:10.1061/(ASCE)0733-9496(2003)129:3(210).
[119] H. F. Wedde, M. Farooq, and Y. Zhang, "BeeHive: An Efficient Fault-Tolerant Routing Algorithm Inspired by Honey Bee Behavior," Berlin, Heidelberg, 2004: Springer Berlin Heidelberg, in Ant Colony Optimization and Swarm Intelligence, pp. 83-94.
[120] X.-S. Yang, "Engineering Optimizations via Nature-Inspired Virtual Bee Algorithms," Berlin, Heidelberg, 2005: Springer Berlin Heidelberg, in Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, pp. 317-323.
[121] D. Teodorović and M. Dell'Orco, "Bee colony optimization - A cooperative learning approach to complex transportation problems," *Advanced OR and AI Methods in Transportation*, pp. 51-60, 01/01 2005.
[122] W. H. LI Wei-wu, ZOU Zhi-jun, QIAN Ji-xin, "Function optimization method based on bacterial colony chemotaxis," *Journal of Circuits and Systems*, vol. 10, no. 01, pp. 58-63, 2005.
[123] H. Drias, S. Sadeg, and S. Yahi, "Cooperative Bees Swarm for Solving the Maximum Weighted Satisfiability Problem," Berlin, Heidelberg, 2005: Springer Berlin Heidelberg, in Computational Intelligence and Bioinspired Systems, pp. 318-325.
[124] O. B. Haddad, A. Afshar, and M. A. Mariño, "Honey-Bees Mating Optimization (HBMO) Algorithm: A New Heuristic Approach for Water Resources Optimization," *Water Resources Management*, vol. 20, no. 5, pp. 661-680, 2006/10/01 2006, doi: 10.1007/s11269-005-9001-3.
[125] S.-C. Chu, P.-w. Tsai, and J.-S. Pan, "Cat Swarm Optimization," Berlin, Heidelberg, 2006: Springer Berlin Heidelberg, in PRICAI 2006: Trends in Artificial Intelligence, pp. 854-858.
[126] C. Bastos-Filho, F. Lima Neto, A. Lins, A. Nascimento, and M. Lima, *A novel search algorithm based on fish school behavior*. 2008, pp. 2646-2651.
[127] T. C. Havens, C. J. Spain, N. G. Salmon, and J. M. Keller, "Roach Infestation Optimization," in *2008 IEEE Swarm Intelligence Symposium*, 21-23 Sept. 2008 2008, pp. 1-7, doi: 10.1109/SIS.2008.4668317.
[128] C. Ying, M. Hua, L. Huilian, J. Zhen, and Q. H. Wu, "A Fast Bacterial Swarming Algorithm for high-dimensional function optimization," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, 1-6 June 2008 2008, pp. 3135-3140, doi: 10.1109/CEC.2008.4631222.
[129] F. Padró and J. Navarro, "Bumblebees: a multiagent combinatorial optimization algorithm inspired by social insect behaviour," 01/01 2009, doi: 10.1145/1543834.1543949.
[130] S. He, Q. H. Wu, and J. R. Saunders, "Group Search Optimizer: An Optimization Algorithm Inspired by Animal Searching Behavior," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 973-990, 2009, doi: 10.1109/TEVC.2009.2011992.
[131] X.-S. Yang, "Firefly Algorithms for Multimodal Optimization," Berlin, Heidelberg, 2009: Springer Berlin Heidelberg, in Stochastic Algorithms: Foundations and Applications, pp. 169-178.
[132] Y. Marinakis, M. Marinaki, and N. Matsatsinis, "A Bumble Bees Mating Optimization Algorithm for Global Unconstrained Optimization Problems," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 305-318.
[133] C. ZhaoHui and T. HaiYan, "Cockroach Swarm Optimization," *Proceedings of the 2nd International Conference on Computer*

Engineering and Technology (ICCET '10), vol. 6, 01/01 2010, doi: 10.1109/ICCET.2010.5485993.
[134] R. Oftadeh, M. J. Mahjoob, and M. Shariatpanahi, "A novel meta-heuristic optimization algorithm inspired by group hunting of animals: Hunting search," *Computers & Mathematics with Applications*, vol. 60, no. 7, pp. 2087-2098, 2010/10/01/ 2010, doi: https://doi.org/10.1016/j.camwa.2010.07.049.
[135] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831-4845, 2012/12/01/ 2012, doi: https://doi.org/10.1016/j.cnsns.2012.05.010.
[136] R. Tang, S. Fong, X. Yang, and S. Deb, "Wolf search algorithm with ephemeral memory," in *Seventh International Conference on Digital Information Management (ICDIM 2012)*, 22-24 Aug. 2012 2012, pp. 165-172, doi: 10.1109/ICDIM.2012.6360147.
[137] B. N. a. H. Wang, "Bacterial Colony Optimization," *Discrete Dynamics in Nature and Society* vol. 2012, pp. 1-29, 2012.
[138] B. R. Rajakumar, "The Lion's Algorithm: A New Nature-Inspired Search Algorithm," *Procedia Technology*, vol. 6, pp. 126-135, 2012/01/01/ 2012, doi: https://doi.org/10.1016/j.protcy.2012.10.016.
[139] M. Taherdangkoo, "A novel meta-heuristic algorithm for numerical function optimization: Blind, naked mole-rats (BNMR) algorithm," *Scientific research and essays*, 11/27 2012.
[140] W.-T. Pan, "A new Fruit Fly Optimization Algorithm: Taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69-74, 2012/02/01/ 2012, doi: https://doi.org/10.1016/j.knosys.2011.07.001.
[141] E. Cuevas, M. Cienfuegos, D. Zaldívar, and M. Pérez-Cisneros, "A swarm optimization algorithm inspired in the behavior of the social-spider," *Expert Systems with Applications*, vol. 40, no. 16, pp. 6374-6384, 2013/11/15/ 2013, doi: https://doi.org/10.1016/j.eswa.2013.05.041.
[142] A. Eesa, A. Mohsin Abdulazeez, and Z. Orman, *A New Tool for Global Optimization Problems- Cuttlefish Algorithm*. 2014.
[143] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46-61, 2014/03/01/ 2014, doi: https://doi.org/10.1016/j.advengsoft.2013.12.007.
[144] J. C. Bansal, H. Sharma, S. S. Jadon, and M. Clerc, "Spider Monkey Optimization algorithm for numerical optimization," *Memetic Computing*, vol. 6, no. 1, pp. 31-47, 2014/03/01 2014, doi: 10.1007/s12293-013-0128-0.
[145] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: an optimization algorithm inspired by animal migration behavior," *Neural Computing and Applications*, vol. 24, no. 7, pp. 1867-1877, 2014/06/01 2014, doi: 10.1007/s00521-013-1433-8.
[146] G.-G. Wang, S. Deb, and Z. Cui, "Monarch butterfly optimization," *Neural Computing and Applications*, vol. 31, no. 7, pp. 1995-2014, 2019/07/01 2019, doi: 10.1007/s00521-015-1923-y.
[147] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowledge-Based Systems*, vol. 89, pp. 228-249, 2015/11/01/ 2015, doi: https://doi.org/10.1016/j.knosys.2015.07.006.
[148] G.-G. Wang, S. Deb, and L. Coelho, *Elephant Herding Optimization*. 2015.
[149] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm," *Computers & Structures*, vol. 169, pp. 1-12, 2016/06/01/ 2016, doi: https://doi.org/10.1016/j.compstruc.2016.03.001.
[150] T.-q. Wu, M. Yao, and J.-h. Yang, "Dolphin swarm algorithm," *Frontiers of Information Technology & Electronic Engineering*, vol. 17, no. 8, pp. 717-729, 2016/08/01 2016, doi: 10.1631/FITEE.1500287.
[151] A. O. Topal and O. Altun, "A novel meta-heuristic algorithm: Dynamic Virtual Bats Algorithm," *Information Sciences*, vol. 354, pp. 222-235, 2016/08/01/ 2016, doi: https://doi.org/10.1016/j.ins.2016.03.025.
[152] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Computing and Applications*, vol. 27, no. 4, pp. 1053-1073, 2016/05/01 2016, doi: 10.1007/s00521-015-1920-1.
[153] Y. Chen and B. Peng, "Multi-objective optimization on the multi-layer configuration of cathode electrode for polymer electrolyte fuel cells via computational-intelligence-aided design and engineering framework," *Applied Soft Computing*, vol. 43, pp. 357-371, 2016/06/01/ 2016, doi: https://doi.org/10.1016/j.asoc.2016.02.045.
[154] Y. Chen, Z. Wang, E. Yang, and Y. Li, "Pareto-optimality solution recommendation using a multi-objective artificial wolf-pack algorithm," in *2016 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA)*, 15-17 Dec. 2016 2016, pp. 116-121, doi: 10.1109/SKIMA.2016.7916207.
[155] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software*, vol. 95, pp. 51-67, 2016/05/01/ 2016, doi: https://doi.org/10.1016/j.advengsoft.2016.01.008.
[156] G. Dhiman and V. Kumar, "Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications," *Advances in Engineering Software*, vol. 114, pp. 48-70, 2017/12/01/ 2017, doi: https://doi.org/10.1016/j.advengsoft.2017.05.014.
[157] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper Optimisation Algorithm: Theory and application," *Advances in Engineering Software*, vol. 105, pp. 30-47, 2017/03/01/ 2017, doi: https://doi.org/10.1016/j.advengsoft.2017.01.004.
[158] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems," *Advances in Engineering Software*, vol. 114, pp. 163-191, 2017/12/01/ 2017, doi: https://doi.org/10.1016/j.advengsoft.2017.07.002.
[159] A. S. Shamsaldin, T. A. Rashid, R. A. Al-Rashid Agha, N. K. Al-Salihi, and M. Mohammadi, "Donkey and smuggler optimization algorithm: A collaborative working approach to pathfinding," *Journal of Computational Design and Engineering*, vol. 6, no. 4, pp. 562-583, 2019/10/01/ 2019, doi: https://doi.org/10.1016/j.jcde.2019.04.004.
[160] J. M. Abdullah and T. Ahmed, "Fitness Dependent Optimizer: Inspired by the Bee Swarming Reproductive Process," *IEEE Access*, vol. 7, pp. 43473-43486, 2019, doi: 10.1109/ACCESS.2019.2907012.
[161] T. A. Khan, S. H. Ling, and A. S. Mohan, "Advanced Particle Swarm Optimization Algorithm with Improved Velocity Update Strategy," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 7-10 Oct. 2018 2018, pp. 3944-3949, doi: 10.1109/SMC.2018.00669.
[162] S. Coco, A. Laudani, F. Riganti Fulginei, and A. Salvini, "TEAM problem 22 approached by a hybrid artificial life method," *COMPEL - The international journal for computation and mathematics in electrical and electronic engineering*, vol. 31, no. 3, pp. 816-826, 2012, doi: 10.1108/03321641211209726.
[163] O. U. Rehman, S. U. Rehman, S. Tu, S. Khan, M. Waqas, and S. Yang, "A Quantum Particle Swarm Optimization Method With Fitness Selection Methodology for Electromagnetic Inverse Problems," *IEEE Access*, vol. 6, pp. 63155-63163, 2018, doi: 10.1109/ACCESS.2018.2873670.
[164] F. G. Guimaraes, F. Campelo, R. R. Saldanha, H. Igarashi, R. H. C. Takahashi, and J. A. Ramirez, "A multiobjective proposal for the TEAM benchmark problem 22," *IEEE Transactions on Magnetics*, vol. 42, no. 4, pp. 1471-1474, 2006, doi: 10.1109/TMAG.2006.871570.
[165] S. U. Khan, S. Yang, L. Wang, and L. Liu, "A Modified Particle Swarm Optimization Algorithm for Global Optimizations of Inverse Problems," *IEEE Transactions on Magnetics*, vol. 52, no. 3, pp. 1-4, 2016, doi: 10.1109/TMAG.2015.2487678.
[166] P. Alotto *et al.*, "SMES Optimization Benchmark Extended: Introducing Pareto Optimal Solutions Into TEAM22," *IEEE Transactions on Magnetics*, vol. 44, no. 6, pp. 1066-1069, 2008, doi: 10.1109/TMAG.2007.916091.
[167] P. Karban, P. Kropík, V. Kotlan, and I. Doležel, "Bayes approach to solving T.E.A.M. benchmark problems 22 and 25 and its comparison with other optimization techniques," *Applied Mathematics and Computation*, vol. 319, pp. 681-692,

2018/02/15/ 2018, doi: https://doi.org/10.1016/j.amc.2017.07.043.

[168] L. Coelho and P. Alotto, "Global Optimization of Electromagnetic Devices Using an Exponential Quantum-Behaved Particle Swarm Optimizer," *Magnetics, IEEE Transactions on*, vol. 44, pp. 1074-1077, 07/01 2008, doi: 10.1109/TMAG.2007.916032.

[169] U. B. P.G. Alotto, F. Freschi, M. Jaindl, A. K¨ostinger, and W. R. Ch. Magele, M. Repetto. "TEAM Workshop Problem 22: SMES Optimization Benchmark [Online]." (accessed.