FAGA: Hybridization of Fractional Order ABC and GA for Optimization

Lavanya Gunasekaran and Srinivasan Subramaniam
Department of Computer Science and Engineering, Anna University Regional Centre, Madurai

Abstract: In order to solve problems of optimization, Swarm Intelligence (SI) algorithms are extensively becoming more popular. Many swarm intelligence based optimization techniques are present but most face problems like convergence problem and local minimization problem. In this paper, a hybrid optimization algorithm is proposed using fractional order Artificial Bee Colony (ABC) and Genetic Algorithm (GA) for optimization to solve the existing problems. The proposed algorithm has four phases such as, employee bee, onlooker bee, mutation and scout bee. In employee bee phase, neighbour solution is generated based on ABC algorithm. Then, in onlooker bee, the probability is used to select a solution and new solution is generated based on fractional calculus-dependent neighbor solution. The mutation operation of genetic algorithm is used in the mutation module and then the scout bee phase is carried out. The proposed algorithm is implemented in MATLAB. For experimentation, the unimodal benchmark functions such as: De jong’s, axis parallel hyper-ellipsoid, rotated hyper-ellipsoid and multi-modal functions such as: Griewank and rastrigin are utilized to analyse the performance of the algorithm. Then, the comparison of the algorithm is also, carried out with the existing ABC, GA and hybrid algorithm. From the results, we can see that the proposed technique has obtained better results by acquiring better minimization and convergence rate.

Keywords: Optimization, ABC, GA, fractional order, mutation, test functions.

Received April 2, 2013; accepted December 24, 2013

1. Introduction

In recent years, Swarm Intelligence (SI) becomes more and more attractive for the researchers. It is one of the branches in evolutionary computing. The algorithms in SI are often applied to solve problems of optimization [30]. It can be defined as the measure, which introduces the collective behaviour of social insect colonies, other animal societies or the relationship description of unsophisticated agents interacting with their environment, to design algorithms or distributed problem-solving devices. By collecting the characteristics and the behaviours of creatures, several algorithms of the optimization issues related to SI are proposed one after another. In addition, several applications of optimization algorithms based on computational intelligence or SI are also presented continuously [8, 17, 21].

Over the years many different techniques for solving optimization problems were developed [10, 15, 25]. Besides many traditional methods, heuristic methods become very prominent. Special place among heuristic methods belongs to the techniques based on the social behaviour of certain animals and insects [22, 23, 29]. These methods are known as swarm intelligence algorithms. Swarms inherently use forms of decentralized control and self-organization to achieve their goals [17, 19]. SI is the collective behaviour of decentralized, self-organized natural or artificial systems. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment [13]. The agents follow very simple rules and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of “intelligent” global behaviour, unknown to the individual agents [7, 18].

SI studies the collective behaviour of systems composed of many individuals interacting locally with each other and with their environment. The recent research has focused on the meta-heuristics approaches such as Ant Colony Optimization (ACO) [9], Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC) [2]. It has also been shown that these algorithms can provide better solutions in comparison to classical algorithms. By mimicking nature inspired swarming behaviour in computing methodologies, techniques emerge for hard optimization problems become robust, scalable and easily distributed [4]. Some of the significant swarm intelligence techniques are as follows: PSO, ACO, ABC and Consultant Guided Search (CGS) and so many techniques. Yet, the complete swarm exhibits intelligent behaviour, providing efficient solutions for complex problems such as predator evasion and shortest path finding [24, 27].

Among various algorithms, GA is a traditional optimization algorithm based on natural selection and the mechanisms of population genetics [12, 14]. This technique is useful for finding the optimal or near optimal solutions for combinatorial optimization problems that traditional methods fail to solve...
efficiently. GA differs from traditional search and optimization methods in four significant points: GA search parallel from a population of points. Therefore, it has the ability to avoid being trapped in local optimal solution like traditional methods, which search from a single point, GA use probabilistic selection rules, not deterministic ones, GA work on the Chromosome, which is encoded version of potential solution’s parameters, rather the parameters themselves and GA use fitness score, which is obtained from objective functions, without other derivative or auxiliary information [26].

Recently, ABC algorithm is introduced for optimization problems. In the ABC algorithm, bee’s duty is to looking for flowers and tries to find the best place of flowers. This process is distributed in an asynchronous process means in this process the bees exchange their information with each other about the food sources after dancing on the searched sources of food [28]. Some important advantage of this algorithm is follows: The structure of the algorithm is favourable for parallel processing, thus saving time and high flexibility, which allows adjustments and the introduction of specific knowledge of the problem by observing nature [11].

A hybrid optimization algorithm using fractional order ABC and GA is proposed in this paper. The proposed optimization technique solves the existing problems faced by the exiting optimization problem such as convergence problem and local minimization problem. The proposed algorithm has four phases such as, employee bee, onlooker bee, mutation and scout bee. In employee bee phase, neighbour solution is generated based on ABC algorithm. Then, in onlooker bee, the probability is used to select a solution and new solution is generated based on fractional calculus-dependent neighbour solution. The mutation operation of GA is used in the mutation module and then the scout bee phase is carried out.

The rest of the paper is organized as follows: Section 2 gives the literature review. Section 3 gives the problem definition and section 4 describes the proposed technique. Section 5 gives results and discussions. Conclusions are summed up in Section 6.

2. Literature Review

Despite a plenty of works available in the literature, a handful of significant research works are reviewed here. Vesterstrom and Thomsen [31] have evaluated the performance of Differential Evolution (DE), PSO and Evolutionary Algorithms (EAs) regarding their general applicability as numerical optimization techniques. The comparison was performed on a suite of 34 widely used benchmark problems. Brest et al. [3] have described an efficient technique for adapting control parameter settings associated with DE. The DE algorithm had been used in many practical cases and has demonstrated good convergence properties. It has only a few control parameters, which were kept fixed throughout the entire evolutionary process. However, it is not an easy task to properly set control parameters in DE.

Karaboga and Basturk [16] have compared the performance of the ABC with that of GA, PSO and PS-EA which were also swarm intelligence and population based algorithms as the ABC algorithm. In order to demonstrated the performance of the ABC algorithm, PSO, PS-EA, GA and ABC algorithms were tested on five high dimensional numerical benchmark functions that have multimodality. From the simulation results it was concluded that the presented algorithm has the ability to get out of a local minimum and was efficiently used for multivariable, multimodal function optimization. Anghinolfi and Paolucci [1] have presented a Discrete Particle Swarm Optimization (DPSO) approach to face the NP-hard single machine total weighted tardiness scheduling problem in presence of sequence-dependent setup times. Differently from previous approaches the proposed DPSO uses a discrete model both for particle position and velocity and a coherent sequence metric. They tested the presented DPSO mainly over a benchmark originally developed by Cicirello in 2003 and available online. The results obtained showed the competitiveness of our DPSO, which was able to outperformed the best known results for the benchmark.

Deng et al. [6] have presented a two-stage hybrid swarm intelligence optimization algorithm called GA-PSO-ACO algorithm that combined the evolution ideas of the GA, PSO and ACO based on the compensation for solving the traveling salesman problem. In the presented hybrid algorithm, the whole process was divided into two stages. In the first stage, they make use of the randomness, rapidity and wholeness of the GA and PSO to obtain a series of sub-optimal solutions (rough searching) to adjust the initial allocation of pheromone in the ACO. In the second stage, they make use of these advantages of the parallel, positive feedback and high accuracy of solution to implement solving of whole problem (detailed searching). Tsai et al. [30] have developed a hybrid optimization algorithm based on Cat Swarm Optimization (CSO) and ABC. CSO was an optimization algorithm designed to solve numerical optimization problems, and ABC was an optimization algorithm generated by simulating the behavior of bees finding foods. By hybridizing those two algorithms, the hybrid algorithm called hybrid Particle Cat Swarm Optimization Artificial Bee Colony (PCSOABC) was presented. Five benchmark functions were used to evaluate the accuracy, convergence, the speed and the stabilization of the hybrid PCSOABC.

In order to integrate BA global search ability with the local search advantages of PSO, Cheng and Lien [5] have presented a optimization hybrid swarm algorithm the Particle Bee Algorithm (PBA) which imitated the intelligent swarming behaviour of honeybees and birds. This study compared the performance of PBA with that of GA, DE, BA and
PSO for multi-dimensional benchmark numerical problems. Besides, this study compared the performance of PBA with that of BA and PSO for practical construction engineering of Construction Site Layout (CSL) problem. The results showed that the performance of PBA was comparable to those of the mentioned algorithms in the benchmark functions and was efficiently employed to solve a hypothetical CSL problem with high dimensionality.

3. Problem Definition and Solution

Literature presents various optimization algorithms in which ABC and GA have more advantages. The ABC algorithm makes use the advantages such as L Global optimization strategy, local optimization strategy (in employee bee), random selection strategy (probability in onlooker) and feedback strategy (scout limit). On the other hand, GA is a simple algorithm and the problem can be easily solved without requiring mathematical knowledge. But, even both the algorithm seems good in optimization, they also faces some challenges in their process. In the traditional GA, solution can have tendency to converge towards local minimum since the “better” solution is only in comparison with other solutions. On the other hand, ABC algorithm does not consider the secondary information about the problem so that convergence rate of optimization problem is very slow. So, the hybridization of both the algorithm leads to an even better solution since the advantages are boosted and disadvantages are compensated with other one. The combination of both the algorithm can easily solve the local minimum problem and also speeds up the convergence rate as the new solution generation is combined with GA. The secondary information problem of ABC algorithm is solved by incorporating the fractional calculus, which is one of the effective mathematical calculations based on differential operator. This leads to having better solutions and better optimization.

4. Proposed Hybridization of Fractional Order ABC and GA for Optimization

In this paper, a hybrid optimization algorithm is proposed by blending fractional order ABC and GA for optimization. The proposed algorithm has four phases namely, employee bee phase, onlooker bee phase, mutation phase and scout bee phase. In employee bee phase, neighbour solution is generated based on ABC algorithm. Then, in onlooker bee, the probability measure is used to select a solution and new solution is generated based on fractional calculus-dependent neighbour solution. Mutation operation of GA is used in the mutation phase and subsequently scout bee phase is carried out.

4.1. Employee Bee Phase

The colony of artificial bees contains three groups of bees: employed bees, onlookers and scouts. A bee waiting on the dance area for making decision to choose a food source is called an onlooker and a bee going to the food source visited by itself previously is named an employed bee. A bee carrying out random search is called a scout. First half of the colony consists of employed artificial bees and the second half constitutes the onlookers. For every food source, there is only one employed bee. In other words, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source is exhausted by the employed and onlooker bees becomes a scout. In each cycle, the search consists of four steps: Sending the employed bees onto the food sources and then measuring their nectar amounts; selecting of the food sources by the onlookers after sharing the information of employed bees and determining the nectar amount of the foods; mutation operation; determining the scout bees and then sending them onto possible food sources. Here, the position of a food source represents a possible solution of the optimization problem and the nectar amount of a food source corresponds to the fitness of the associated solution. The number of the employed bees or the onlooker bees is equal to the number of solutions in the population. In employee bee phase, neighbour solution is generated based on ABC algorithm.

At the initialization stage, a set of food source positions are randomly selected by the employed bees and their nectar amounts are determined. Then, these bees come into the hive and share the nectar information of the sources with the onlooker bees waiting on the dance area within the hive. Initially, ABC generates a randomly distributed initial population represented by $\text{pop}_{\text{ini}}$ having $N_p$ solutions, where each solution is the food source position and $N_p$ is the population size. Each solution is represented by $\mathbf{g}_j$, where $1 \leq j \leq N_p$ is a M-dimensional vector, where M is the number of optimization parameters taken into consideration. After initialization, the population of the positions is subjected to repeated cycles of the search processes of the employed bees, onlooker bees, mutation and scout bees.

4.2. Onlooker Bee Phase

In this phase, selection of the food sources by the onlookers after receiving the information of employed bees and generation of new solution based on fractional calculus is carried out. The onlooker bee prefers a food source area depending on the nectar information distributed by the employed bees on the dance area. As the nectar amount of a food source increases, the probability with which that food source is chosen by an onlooker increases, too. Hence, the dance of employed bees carrying higher nectar recruits the onlookers for the food source areas with higher nectar amount.

An onlooker bee chooses a food source depending on the probability value associated with that food source ($P$) given by the expression:
\[ P_j = \frac{F_j}{\sum_{k=1}^{N_p} F_k} \]

Where \( F_j \) is the fitness value of the solution \( j \) evaluated by its employed bee, which is proportional to the nectar amount of the food source in the position \( j \) and \( N_p \) is the number of food sources which is equal to the number of employed bees. After arriving at the selected area, onlooker chooses a new food source in the neighbourhood of the one in the memory depending on visual information. Visual information is based on the comparison of food source positions. When the nectar of a food source is abandoned by the bees, a new food source is randomly determined by a scout bee and replaced with the abandoned one. The position update is made with the use of fractional calculus.

An artificial onlooker bee probabilistically produces a modification on the position (solution) in her memory for finding a new food source and tests the nectar amount (fitness value) of the new source (new solution). In case of real bees, the production of new food sources is based on a comparison process of food sources in a region depending on the information gathered, visually, by the bee. In our case, the production of a new food source position is also based on a comparison process of food source positions. However, in the model, the artificial bees do not use any information in comparison. They randomly select a food source position and produce a modification on the one existing in their memory. Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount and then, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution).

Providing that its nectar is higher than that of the one existing in their memory. Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount and then, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution). Providing that its nectar is higher than that of the one existing in their memory. Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount and then, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution). Providing that its nectar is higher than that of the one existing in their memory. Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount and then, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution). Providing that its nectar is higher than that of the one existing in their memory. Provided that the nectar amount of the new source is higher than that of the previous one the bee memorizes the new position and forgets the old one. Otherwise she keeps the position of the previous one. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount and then, she produces a modification on the position (solution) in her memory and checks the nectar amount of the candidate source (solution).

Let the old position be represented by \( z_{j,k} \) and the new position be represented by \( y_{j,k} \), which is defined by the Equation 1:

\[ z_{j,k} = y_{j,k} + j_k (y_{j,k} - y_{j,k}) \quad i \neq j \]

Where, \( i \in \{1, 2, ..., N_p\} \) and \( k \in \{1, 2, ..., M_i\} \). \( \psi_{j,k} \) is a random number in the range \(-1, 1\). Which controls the production of a neighbour food source position around \( y_{j,k} \) and the modification represents the comparison of the neighbour food positions visually by the bee. The position update equation shows that as the difference between the parameters of the \( y_{j,k} \) and \( y_{j,k} \) decreases, the perturbation on the position \( y_{j,k} \) also decreases, too.

Thus, as the search approaches to the optimum solution in the search space, the step length is adaptively reduced.

Fractional Calculus (FC) is a natural extension of the classical mathematics and extents the possibility of taking real number powers or even complex number powers of the differentiation operator or the integration operator. In our case, fractional calculus is made use of in the position updation step.

Rearranging the position update step, we have Equation 2 as:

\[ z_{j,k} - y_{j,k} = \phi_{j,k} (y_{j,k} - y_{j,k}) \quad (2) \]

As \( z_{j,k} \) is the position update from \( y_{j,k} \) in the previous step, representing in the time domain, we can write \( y_{j,k} \) as \( z_i \) when \( z_{j,k} \) is taken as \( z_{i-1} \). Hence, we have Equation 3:

\[ z_{i+1} - z_i = \phi_{j,k} (y_{j,k} - y_{j,k}) \quad (3) \]

The left side \( z_{i+1} \), is the discrete version of the derivative of order \( \alpha=1 \). Hence, we have Equation 4:

\[ D^\alpha [z_{i+1}] = \phi_{j,k} (y_{j,k} - y_{j,k}) \quad (4) \]

Here, by discrete time approximation (taking first four terms), we have Equation 5:

\[ z_{i+1} = z_i + \frac{\alpha-1}{2} z_i - \frac{\alpha}{6} z_i + \frac{\alpha-1}{2} - \frac{\alpha}{24} z_i + \frac{\alpha}{24} \phi_{j,k} (y_{j,k} - y_{j,k}) \quad (5) \]

Rearranging, we have the updated position Equation as 6:

\[ s_{i+1} = \alpha z_i - \left( 1 - \alpha \right) z_i + \frac{\alpha-1}{2} z_i - \frac{\alpha}{6} z_i + \frac{\alpha-1}{2} - \frac{\alpha}{24} z_i + \frac{\alpha}{24} \phi_{j,k} (y_{j,k} - y_{j,k}) \quad (6) \]

4.3. Mutation Module

In this module, mutation operation is carried out to have better solution. The mutation operator is one of the operators used in GA. GAs are adaptive heuristic search algorithm premised on the evolutionary ideas of natural selection and genetic which has been widely studied, experimented and applied in many fields in engineering worlds. GA belong to the larger class of EA, which generate solutions to optimization problems using techniques inspired by natural evolution, such as: Inheritance, mutation, selection and crossover.

The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated; multiple individuals are stochastically selected from the current population based on their fitness and modified using recombination and mutation to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

In mutation, individuals are perturbed probabilistically to bring a change in the individuals.
Using mutation operator, there is a probability that some new features might appear due to change in the chromosome. Mutation is a genetic operator that alters one or more gene values in a chromosome from its initial state. This can result in entirely new gene values being added to the gene pool. With these new gene values, the genetic algorithm may be able to arrive at better solution than was previously possible. Mutation is an important part of the genetic search as help helps to prevent the population from stagnating at any local optima. Mutation occurs during evolution according to a user-definable mutation probability. This probability should usually be set fairly low and if it is set to high, the search will turn into a primitive random search.

Many mutation methods like flip bit mutation, boundary mutation, non-uniform mutation, uniform mutation and Gaussian mutation are been commonly used. In our proposed technique, we make use of Gaussian mutation to mutate the individuals. Mutation is performed on the basis of pre determined mutating probability. Gaussian mutation consists in adding a random value from a Gaussian distribution to each element of an individual’s vector to create a new offspring.

4.4. Scout Bee Phase
The employed bee whose food source is exhausted by the employed and onlooker bees becomes a scout and it carries out random search. The food source whose nectar is abandoned by the bees is replaced with a new food source by the scouts. This is simulated by randomly producing a position and replacing it with the abandoned one. Here, if a position cannot be improved further through a predetermined number of cycles called limit then that food source is assumed to be abandoned.

The control parameters used in the algorithm consists of the number of the food sources which is equal to the number of employed or onlooker bees, the value of limit, mutation operation and the maximum cycle number. Here, the onlookers and employed bees carry out the exploitation process in the search space, the scouts control the exploration process and mutation betters the solution.

5. Results and Discussions
The results obtained for the proposed hybridization of fractional order ABC and GA (FAGA) is discussed in this section. Section 5.1 gives the experimental set up and details on the test functions used for evaluation and section 5.2 gives the comparative analysis.

5.1. Experimental Setup and Test Functions
The proposed technique is implemented using MATLAB on a system having the configuration of 6 GB RAM and 2.8 GHz Intel i-7 processor. In order to compare the performance of the proposed FAGA, we used six classical benchmark functions described in [20]:

1. De Jong’s Function: So, called first function of De Jong’s is one of the simplest test benchmark. Function is continuous, convex and unimodal. It has the following general definition 7:

   \[ f(x) = \sum_{i=1}^{n} x_i^2 \]  

   Test area is usually restricted to hypercube \(-50 \leq x_i \leq 50, \ i = 1, 2, ..., n\). Global minimum \(f(x)=0\) is obtainable for \(x_i=0, \ i = 1, 2, ..., n\).

2. Axis Parallel Hyper-Ellipsoid Function: The axis parallel hyper-ellipsoid is similar to function of De Jong. It is also known as the weighted sphere model. Function is continuous, convex and unimodal. It has the following general definition 8:

   \[ f(x) = \sum_{i=1}^{n} (i.x_i^2) \]  

   Test area is usually restricted to hypercube \(-50 \leq x_i \leq 50, \ i = 1, 2, ..., n\). Global minimum \(f(x)=0\) is obtainable for \(x_i=0, \ i = 1, 2, ..., n\).

3. Rotated Hyper-Ellipsoid Function: An extension of the axis parallel hyper-ellipsoid is schwefel’s function. With respect to the coordinate axes, this function produces rotated hyper-ellipsoids. It is continuous, convex and unimodal. Function has the following general definition 9:

   \[ f(x) = \sum_{i=1}^{n} \sum_{j=1}^{i} x_j^2 \]  

   Test area is usually restricted to hypercube \(-50 \leq x_i \leq 50, \ i = 1, 2, ..., n\). Global minimum \(f(x)=0\) is obtainable for \(x_i=0, \ i = 1, 2, ..., n\).

4. Rastrigin’s Function: Is based on the function of De Jong with the addition of cosine modulation in order to produce frequent local minima. Thus, the test function is highly multimodal. However, the location of the minima is regularly distributed. Function has the following definition 10:

   \[ f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10 \cos(2 \pi x_i)] \]  

   Test area is usually restricted to hypercube \(-50 \leq x_i \leq 50, \ i = 1, 2, ..., n\). Global minimum \(f(x)=0\) is obtainable for \(x_i=0, \ i = 1, 2, ..., n\).

5. Griewank’s Function: Is similar to the function of Rastrigin. It has many widespread local minima regularly distributed. Function has the following definition 11:

   \[ f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \]  

   Test area is usually restricted to hypercube \(-50 \leq x_i \leq 50, \ i = 1, 2, ..., n\). Global minimum \(f(x)=0\) is obtainable for \(x_i=0, \ i = 1, 2, ..., n\). The function interpretation changes with the scale; the general overview suggests convex function, medium-scale view suggests existence of local extremum and finally zoom on the details indicates complex structure of numerous local extremum.
6. **Ackley’s Function**: Is a widely used multimodal test function. It has the following definition:

\[ f(x) = -a \exp \left( -b \sum_{i=1}^{n} \frac{x_i^2}{2} \right) - \exp \left( \sum_{i=1}^{n} \cos(x_i) \right) + a + \exp(l) \]  \hspace{2cm} (12)

It is recommended to set \(a=20\), \(b=0.2\), \(c=2\pi\). Test area is usually restricted to hypercube \(-50 \leq x_i \leq 50\), \(i=1, 2, ..., n\). Global minimum \(f(x)=0\) is obtainable for \(x_i=0\), \(i=1, 2, ..., n\).

### 5.2. Comparative Analysis

Our proposed technique is compared with other optimization techniques with the use of test functions in this section. The comparison is made to ABC, GA and ABC-GA. The test functions are evaluated for two test conditions where in one case, dimension is set as 30 and in the other case dimension is taken as 60. Our aim is to minimize the objective function which is directly proportional to minimizing the fitness functions when test functions are taken for evaluation. The test function used are axis parallel hyper-ellipsoid function, dejong’s function, rotated hyper-ellipsoid function, griewangk’s function, rastrigin’s function and ackley’s function.

a) Case 1 with dimension 30:

- **Figure 1.** Axis parallel hyper-ellipsoid function.
- **Figure 2.** De Jong’s function.
- **Figure 3.** Rotated hyper-ellipsoid function.

b) Case 2 with dimension 60:

- **Figure 4.** Griewangk’s function.
- **Figure 5.** Rastrigin’s function.
- **Figure 6.** Ackley’s function.
- **Figure 7.** Axis parallel hyper-ellipsoid function.
- **Figure 8.** De Jong’s function.
- **Figure 9.** Rotated hyper-ellipsoid function.
From Figures 1 to 12, we can infer that the fitness obtained with respect to the iteration count is less for the proposed FAGA algorithm for six standard test functions.

- Our aim is to minimize the objective function which is directly proportional to minimizing the fitness functions when test functions are taken for evaluation.
- The functions include Axis parallel hyper-ellipsoid function, deJong’s function, rotated hyper-ellipsoid function, griewangk’s function, rastrigin’s function and ackley’s function.
- Analysis is carried for two dimensions ($M=30$ and $M=60$) as two cases to check the robustness of the proposed technique.
- Comparative analysis is made by comparing our technique (FAGA) to other standard optimization techniques (GA, ABC and ABC-GA).
- Observations are made up to iteration number 50.
- Figures 1 and 7 give fitness graph for Axis parallel hyper-ellipsoid function with $M=30$ and $M=60$ respectively.
- Figures 2 and 8 give fitness graph for DeJong’s function with $M=30$ and $M=60$ respectively.
- Figures 3 and 9 give fitness graph for Rotated hyper-ellipsoid function with $M=30$ and $M=60$ respectively.
- Figures 4 and 10 give fitness graph for Griewangk’s function with $M=30$ and $M=60$ respectively.
- Figures 5 and 11 give fitness graph for Rastrigin’s function with $M=30$ and $M=60$ respectively.
- Figures 6 and 12 give fitness graph for Ackley’s function with $M=30$ and $M=60$ respectively.
- We can see from all graphs, that our proposed technique have achieved better minimization when compared to others techniques such as ABC, GA and ABC-GA.
- Result shows better performance for our proposed optimization technique when compared to others techniques such as ABC, GA and ABC-GA.
- The result also shows the better convergence when compared with others techniques such as ABC, GA and ABC-GA.
- The technique performed well for all test functions and dimensions taken for the evaluation so as to prove the effectiveness and robustness of the proposed technique.
- From the Figure 1, the fitness value obtained for the Axis parallel hyper-ellipsoid function by the proposed FAGA algorithm is $2.45E^{-48}$ which is less than the existing GA methods that obtained $60.0104075486298$.
- From the Figure 2, the fitness value obtained for the DeJong’s function by the proposed FAGA algorithm is $1.58E^{-49}$ which is less than the existing GA methods that obtained $2.881424036$.
- From the Figure 3, the fitness value obtained for the Rotated hyper-ellipsoid function by the proposed FAGA algorithm is $2.17E^{-48}$ which is less than the existing GA methods that obtained $50.19178412$.
- From the Figure 4, the fitness value obtained for the Griewangk’s function by the proposed FAGA algorithm is $0$ which is less than the existing GA methods that obtained $0.154898886$.
- From the Figure 5, the fitness value obtained for the Rastrigin’s function by the proposed FAGA algorithm is $0$ which is less than the existing GA methods that obtained $126.8224121$.
- From the Figure 6, the fitness value obtained for the Griewangk’s function by the proposed FAGA algorithm is $8.88E^{-16}$ which is less than the existing GA methods that obtained $0.096378571$.
- From the Figure 7, the fitness value obtained for the Axis parallel hyper-ellipsoid function by the proposed FAGA algorithm is $8.43E^{-48}$ which is less than the existing GA methods that obtained $299.1169328$.
- From the Figure 8, the fitness value obtained for the Axis parallel hyper-ellipsoid function by the proposed FAGA algorithm is $3.26E^{-49}$ which is less than the existing GA methods that obtained $10.36081944$.
- From the Figure 9, the fitness value obtained for the Rotated hyper-ellipsoid function by the proposed
FAGA algorithm is 8.73E-48 which is less than the existing GA methods that obtained 285.2345166.

- From the Figure 10, the fitness value obtained for the Griewank’s function by the proposed FAGA algorithm is 0 which is less than the existing GA methods that obtained 0.256749065.
- From the Figure 11, the fitness value obtained for the Rastrigin’s function by the proposed FAGA algorithm is 0 which is less than the existing GA methods that obtained 410.9625062.
- From the Figure 12, the fitness value obtained for the Griewank’s function by the proposed FAGA algorithm is 8.38E-16 which is less than the existing GA methods that obtained 0.216643961.

6. Conclusions
In this paper, a hybrid optimization algorithm is proposed using fractional order ABC and GA for optimization of different objective functions. The proposed algorithm has four phases such as: Employee bee, onlooker bee, mutation and scout bee. The proposed algorithm is implemented in MATLAB. For experimentation, the unimodal benchmark functions such as: De Jong’s, axis parallel hyper-ellipsoid, rotated hyper-ellipsoid and multi-modal functions such as: Griewank and rastrigin are utilized to analyze the performance of the algorithm. Then, the comparison of the algorithm is also carried out with the existing ABC, genetic algorithm and hybrid algorithm. From the results, we can see that the proposed technique has obtained better results by acquiring better minimization and convergence rate.

References


---

**Gunasekaran Lavanya** received her Bachelor’s degree in Computer science and Engineering from MS University, Tirunelveli. Then she received her Master’s degree in computer science and information technology from M.S University, Tirunelveli. She is currently an associate professor in Department of Computer Science and Engineering at OCOE, Theni. Her area of interests includes optimization, swarm intelligence, image processing. She is currently pursuing her doctoral research at Anna University in the area of Swarm Intelligence.

**Subramaniam Srinivasan** received the PhD degree in computer science and Engineering from the Sathyabama University, Chennai, and M.Tech. in Computer Science and Engineering from the Indian Institute of Technology, Chennai and MBA in System and Finance from Sathyabama Engineering College, University of Madras, Chennai and the MS. in Mathematics from the Gobi Arts College, Gobichettipalayam, Bharathiar University, Coimbatore. He is presently working as Professor and Head, Department of Computer Science and Engineering Anna University, Regional office, Madurai. He has 22 year of experience in Teaching and Research. He has also held various positions and responsibilities in Technical Institutions. He is acting as expert member at various Universities in various capacities. He has published more 80 research papers in journals, conferences and workshops. His research interest includes text mining, data mining and data warehouse and Software Engineering. He is a member in the Professional bodies: ISTE and CSI.