SN Computer Science (2020) 1:59
https://doi.org/10.1007/s42979-020-0062-4

ORIGINAL RESEARCH

Fuzzy Galactic Swarm Optimization with Dynamic Adjustment of Parameters Based on Fuzzy Logic

Emer Bernal[1] · Oscar Castillo[1] · José Soria[1] · Fevrier Valdez[1]

Received: 27 March 2018 / Accepted: 12 November 2018 / Published online: 21 January 2020

Abstract
In this work, a fuzzy method for dynamic adjustment of parameters in galactic swarm optimization is presented. Galactic swarm optimization is based on the movement of stars and galaxies in the universe, as well as their attractive influence allowing the use of multiple cycles of exploration and exploitation to solve complex optimization problems. It has been observed in the literature that the utilization of fuzzy systems for dynamic adjustment of parameters in metaheuristic algorithms produces good results when compared to using fixed parameters in the algorithms. In this work, the adjustment of the c_3 and c_4 parameters is made through the use of fuzzy systems because these parameters have a significant role in the operation of galactic swarm optimization. We tested the fuzzy approach with a set of benchmark mathematical functions and with the fuzzy controller of the water tank problem to measure the performance. Finally, a comparison of the results is presented among the proposed method and other metaheuristics.

Keywords Fuzzy method · Galactic swarm optimization · Mathematical functions · Adjustment of parameters · Fuzzy controller

Introduction

Optimization techniques that can produce solid performance in high-dimensional and multimodal functions have been of great interest to researchers. It is known that the classic search methods usually get stuck in local minima and do not behave well when faced with a high number of dimensions, such as particle swarm optimization (PSO), ant colony optimization (ACO), genetic algorithm (GA), among others [1–4].

Swarm intelligence originated from the study of colonies, swarms or flocks of social organisms and the study of the behavior of individuals in this type of groupings impelled the design of very efficient metaheuristic optimization algorithms [5, 6]. For example, the study of the flocks of birds helped the design of the particle swarm optimization algorithm and the study of the colonies of different groupings of organism gave origin to the bee colony algorithm and the ant colony optimization, among other existing optimization algorithms that base their performance on the behavior of groupings of organisms [7–11].

Fuzzy logic and fuzzy sets were initially proposed by Zadeh [12]. Based on these concepts, fuzzy systems are built with if–then rules constructed through knowledge and heuristics that are based on human knowledge [13].

In fuzzy sets, each element can belong to a set with a degree of certainty, and fuzzy logic allows us to reason with inaccurate or uncertain facts to infer from them new facts with a degree of certainty associated with each particular event. This allows us to model the facts as they occur in real life [14].

Galactic swarm optimization proposed by Muthiah-Nakarajan and Noel has shown to behave well when facing multimodal problems as well with a high number of dimensions, since it presents multiple cycles of exploration and exploitation, and this increases the chances of obtaining better solutions and not getting stuck in local minima [1, 2].

Galactic swarm optimization is based on the movement of stars and galaxies in the universe as well as their attractive influence, and the initial population is divided into subpopulations where each solution is attracted towards the best

Fuzzy Galactic Swarm Optimization with Dynamic Adjustment of Parameters Based on Fuzzy Logic was initially published in Metaheuristics that was closed without publishing an Issue. Consequently, the article has been re-published in SN Computer Science.

✉ Oscar Castillo
ocastillo@tectijuana.mx

[1] Tijuana Institute of Technology, Tijuana, BC, Mexico

solutions within each subpopulation thus achieving a better exploration of the possible solutions giving rise to the possibility of solving more complex optimization problems [2].

Galactic swarm optimization (GSO) is described in its original form to appreciate which are the necessary equations to perform its operation and determine the parameters that we can adapt to achieve a better behavior of the galactic swarm optimization. The study of galactic swarm optimization performance is made to show the efficacy of the GSO algorithm to solve optimization problems, taking the original GSO as a basis for the modification of the galactic swarm optimization with dynamic adjustment of parameters. Normally to use the adaptation of parameters during execution of metaheuristic algorithms helps to obtain better results than when using fixed parameters [15–17].

The main contribution of this work is the use of fuzzy logic and fuzzy systems to help the galactic swarm optimization to automatically make the adjustment of the parameters without the need for the user to manually move the values of the parameters of the algorithm. In addition, it was used for the optimization of the fuzzy controller of the case study of the water tank control, expecting to obtain a better controller than the one used by traditional control methods.

In this approach, two different fuzzy systems were used to perform the dynamic adjustment of the c_3 and c_4 parameters of galactic swarm optimization. The first fuzzy system has one input variable and two output variables. The second fuzzy system has two input variables and two output variables. For both fuzzy systems the output variables are the c_3 and c_4 parameters, this with the idea of obtaining better results than those obtained when using fixed parameters. In the case of the study of the water tank controller, the fuzzy controller optimization is performed, dynamically adjusting the parameters of the membership functions to obtain different and varied fuzzy controllers that can improve the results obtained with the original controller.

The paper is organized as follows: in "Basic Concepts", a brief description of the concepts of metaheuristic algorithms, fuzzy logic and fuzzy sets is presented. In "Galactic Swarm Optimization", the galactic swarm optimization is described where we can observe the steps and necessary equations to carry out its operation. In "Fuzzy Approach in Galactic Swarm Optimization", we can find the proposed fuzzy approach to perform the parameter adaptation in galactic swarm optimization. In "Mathematical Functions and the Benchmark Water Tank Problem for Testing the Galactic Swarm Optimization", we can find the mathematical functions and the benchmark water tank problem used to test the proposed method. "Experiments and Comparison of Results" shows the results obtained from the execution of the galactic swarm optimization and the fuzzy approach to fuzzy galactic swarm optimization (FGSO), in addition to a statistical test to be able to confirm that significant improvements were obtained and finally "Conclusions" describes the conclusions.

Basic Concepts

Metaheuristic Algorithms

Metaheuristic algorithms are usually classified into three classes: evolutionary, based on physical laws and swarm intelligence. Evolutionary algorithms are inspired by the concept of evolution as it occurs in nature [18, 19], and one of the most popular evolutionary algorithms is the genetic algorithm [20] that emulates the concept of evolution according to the theory of Darwin, which has been used in a large number of applications in the areas of engineering, and industry, among others.

Physical-based algorithms try to imitate physical laws such as the gravitational force and the electromagnetic force, and some of these algorithms are the gravitational search algorithm, big bang–big crunch among others. In these algorithms, the agents communicate and move in the space search according to the rules of physics [21, 22].

Swarm intelligence algorithms imitate the social behavior of herds, swarms, and groups of animals in nature, in these algorithms the search agents move and communicate using the social behavior of some type of animal species [20, 23]. The most popular swarm intelligence algorithm is the PSO [24] algorithm that is inspired by the behavior of a flock of birds.

Fuzzy Logic and Fuzzy Sets

Fuzzy logic is a branch of artificial intelligence that allows a system to analyze information from the real world on a scale of values between the false and the true. It supports vague concepts and allows the construction of heuristics capable of interpreting information difficult to define [15, 25].

Fuzzy logic has the virtue of better adapting to the real world and can even understand and function with every day or vague expressions such as "it is very hot", "is very high" among some others. The key is that their adaptation to natural language is based on quality quantifiers to make our inferences. For every fuzzy set, there is a membership function associated with each of its elements that indicates to what extent the element forms part of that fuzzy set [26].

A fuzzy set is a generalization of the classical sets, the main difference lies in that in the theory of classical sets an element may or may not belong to a set and in the theory of fuzzy sets an element may belong to more than one set with different membership degrees. One of the qualities of a fuzzy set is the handling of ambiguous information [25, 26].

The application of fuzzy logic is to imitate human reasoning in computer programming, where traditional computing can only manipulate strictly dual values, as true or false, yes or no, etc. In fuzzy logic, mathematical models are used to represent subjective notions, such as "high, low" or "cold, hot" for specific values that can be manipulated by computers. This paradigm has a special variable value with respect to time, since control systems or of other types may need feedback in a specific space of time since they may need previous data to perform an average evaluation of the situation in an earlier period of time [12].

Fuzzy logic is based on heuristic rules of if (antecedent) then (consequent) form to represent knowledge that is imprecise and inaccurate in nature. This is achieved using linguistic variables that have the ability to express and work with observations and measures of uncertainty in addition to describing uncertain conditions. Linguistic variables facilitate the extraction and storage of knowledge in a simple way. In addition to providing a gradual transition of states [13].

Galactic Swarm Optimization

Galactic swarm optimization, proposed by Muthiah-Nakarajan and Noel, has shown to behave well when facing multimodal problems and with a high number of dimensions, since it presents multiple cycles of exploration and exploitation, which increases the chances of obtaining better solutions and not getting stuck in local minima [1].

The way stars are attracted into a galaxy and a galaxy within a group of galaxies is emulated in galactic swarm optimization according to two levels of grouping. First, the population is divided into subpopulations, in level 1 all the subpopulations are attracted towards the best solution according to the particle swarm optimization (PSO) algorithm. In level 2 each subpopulation will be represented by the best solution found in each subpopulation treating the best solutions as a super swarm, and they will move according to the PSO algorithm. In this way, all individuals will be attracted towards the global best solution [1, 2].

PSO is a swarm intelligence metaheuristic that is based on the behavior of birds in nature, where each particle or individual has a position and a velocity with which it moves through the search space [27, 28]. In the real world the particles have an amount of inertia that maintains them in the direction in which they moved in the same way they have an acceleration or change of velocity that depends mainly on two characteristics [29, 30]:

1. Each particle is attracted to the best local position it has found also known as local best.
2. Each particle is attracted towards the best global position found in a set of particles or neighbourhood known as the global best.

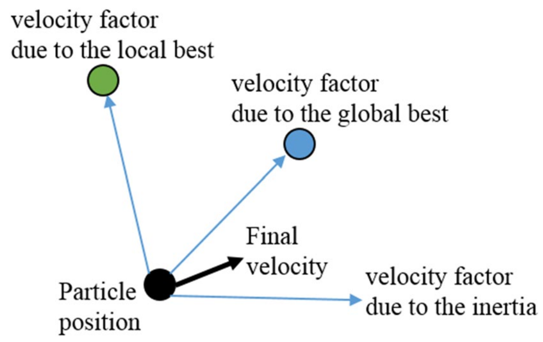
velocity factor due to the local best
velocity factor due to the global best
Final velocity
Particle position
velocity factor due to the inertia

Fig. 1 Approximation of the particle position

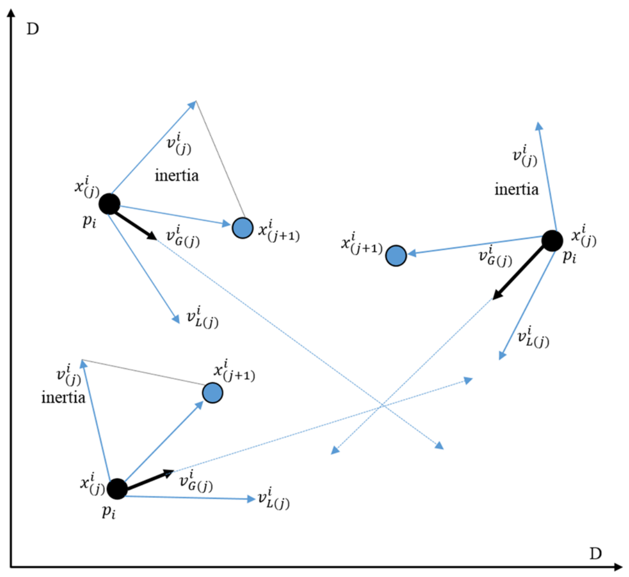

Fig. 2 Movement of the particles in PSO

In Fig. 1 we can find the factors that influence the movement of the particles.

The force with which the particles are moving in each of these directions can be adjusted by the cognitive component c_1 and the social component c_2 so that when the particles move away from the best positions the attraction force is greater [31, 32].

In Fig. 2 we observe how a set of randomly initialized particles perform their movement in which different factors influence how to update their position and velocity. This figure illustrates how the particles will move in particular space for a given problem.

In galactic swarm optimization, a set X represents a swarm that is formed of elements $X_j^{(i)}$ that consist of M partitions denominated subswarms X_i all of size N. All the elements of the swarm are initialized randomly in the search space $[X_{\max}, X_{\min}]^D$, where D represents the number of dimensions of the search space [2].

Level 1

The intention of having several swarms exploring at the same time is to lead to a better exploration of possible solutions, since each subswarm independently explores in the search space. The GSO algorithm begins the iterations by computing the position and velocity of the particles using the equations presented below [1, 2]:

$$v_j^{(i)} \leftarrow W_1 v^{(i)} + c_1 r_1 \left(p_j^{(i)} - x_j^{(i)} \right) + c_2 r_2 \left(g^{(i)} - x_j^{(i)} \right) \tag{1}$$

$$x_j^{(i)} \leftarrow x_j^{(i)} + v_j^{(i)}. \tag{2}$$

Where the current velocity is $v^{(i)}$, $p_j^{(i)}$ is the personal best of particle $x_j^{(i)}$, $g^{(i)}$ is the global best solution of subswarm X_i, c_1 and c_2 are constants that indicate the direction towards the best local and global solutions, weight of inertia is W_1, r_1 and r_2 are calculated by the following expressions [1]:

$$W_1 = 1 - \frac{K}{I_1 + 1}, \tag{3}$$

$$r_1 = \cup(-1,\ 1). \tag{4}$$

The current iteration is K ranging from 0 to I_1, and r_1 is used to obtain random numbers chosen from the range of −1 to 1.

Level 2

In this level of grouping, the global best solutions are of great importance since they participate in the formation of the superclusters, we have a superswarm Y that is formed of the best solutions found in each subpopulation or subswarm X_i.

$$Y^{(i)} = g^{(i)}. \tag{5}$$

Similar to level 1, in this level the position and velocity of the particles are also computed taking as a basis the PSO algorithm [33], the equations show some slight modifications unlike those used in level 1, as shown below [2]:

$$v^{(i)} \leftarrow W_2 v^{(i)} + c_3 r_3 \left(p^{(i)} - y^{(i)} \right) + c_4 r_4 \left(g - y^{(i)} \right), \tag{6}$$

$$y^{(i)} \leftarrow y^{(i)} + v^{(i)}. \tag{7}$$

In this level $p^{(i)}$ represents the personal best, g is the global best solution, c_3 and c_4 are the acceleration constants. W_2, r_3 and r_4 are calculated with equations similar to those shown in level 1.

In the superswarm, we use the best solutions of each of the sub swarms and thus exploit the already calculated information. The individuals in the superswarm are more dispersed in comparison with the individuals of the subswarms, reason why an independent exploration is not realized since it focuses on exploiting the solutions found by the subswarms. In galactic swarm optimization, feedback is avoided to help retain the diversity in the solutions and the global search capability.

In the GSO algorithm the movement of the subswarms in level 1 consists mainly in a phase of exploration and in level 2 it is a phase of exploitation, in this way the algorithm alternates between the exploration and the exploitation [1, 2]. The operation of the galactic swarm optimization can be summarized in the pseudocode presented below.

Pseudocode GSO

```
1. Start GSO
2. The population is divided into M subpopulations.
3. Initialization of the population randomly.
4. Begin Level 1:
     PSO begins for each of the M subpopulations.
     For i=1 To M
       For k=1 To $I_1$
         For j=1 To N
     Calculate the position and velocity of the particles according to
     expressions 1 and 2.
         If $f(x_j^{(i)}) < f(p_j^{(i)})$   Then $f(p_j^{(i)}) = f(x_j^{(i)})$
         If $f(p_j^{(i)}) < f(g_{(i)})$   Then $f(g_{(i)}) = f(p_j^{(i)})$
         If $f(g_{(i)}) < f(g)$       Then $g = g_{(i)}$
         Endfor j
       Endfor k
     Endfor i
5. Begin Level 2:
     Initialization of the swarm $Y^{(i)} \in Y: i = 1, 2, \ldots, M$.   $Y^{(i)} = g^{(i)}$
     For k=0 Hasta $I_2$
       For i=1 Hasta M
      Calculate the position and velocity of the particles according to
      expressions 6 and 7.
          If $f(Y^{(i)}) < f(p^{(i)})$  Then $f(p^{(i)}) = f(Y^{(i)})$
          If $f(p^{(i)}) < f(g)$     Then $g = p^{(i)}$
       Endfor i
     Endfor k
6. Return the best position g and its fitness value f(g) .
7. End
```

Fuzzy Approach in Galactic Swarm Optimization

Fuzzy logic and fuzzy sets were initially proposed by Zadeh [8]. In this case, fuzzy systems are constructed based on if–then rules representing the knowledge and heuristics that are based on human knowledge [12, 17].

In fuzzy sets, each element can be part of a set with a degree of certainty, and fuzzy logic allows us to reason with inaccurate or uncertain facts to infer from them new facts with a degree of certainty associated with each particular event. This allows us to model the facts as they occur in real life [13, 14].

According to the literature [7, 29] the recommended values for the c_3 cognitive component and c_4 social component are in the range of 0.5–2.5. On the other hand, it is suggested that the dynamic adjustment of the parameters during the execution can produce better results. It has been observed that measures such as the iteration and diversity of the swarm should be taken into consideration for the execution of the algorithm, since it has been demonstrated in other studies [15, 29] that the use of these measures helps to control the parameters of the metaheuristic algorithms during the execution.

The optimal values of the parameters in a fuzzy system help the algorithms to find better solutions, and the objective of our proposal is to dynamically find these values with an adaptation of the parameters utilizing fuzzy logic as a means to achieve this and in this way get better performance of galactic swarm optimization (GSO). The general idea of the proposed approach is illustrated in Fig. 3.

After performing different tests a decision was made about using the c_3 and c_4 parameters from Eqs. (6) and (7) presented in "Basic Concepts", since they are of great importance for finding the position and velocity of the particles in the second level of the galactic swarm optimization, and therefore, they become the fuzzy parameters that are dynamically adapted. In GSO the c_3 parameter is the cognitive component that measures the performance of the particle with respect to its previous positions; its objective is that the particles are attracted towards the best positions, in the same way that the individuals return to situations or places where they were previously better. The social component is c_4 that measures the performance of the particle in relation to a group of particles, the importance of the social component c_3 lies in that each particle is grouped into the best position found in a neighborhood or in its search space [8, 33].

The main difference of using the dynamic adjustment of parameters with respect to using fixed parameters is that the selected parameters used as fuzzy parameters are modified as the iterations are being performed in galactic swarm optimization, and this way in each iteration we are dynamically changing the values of the parameters with the idea of finding better solutions.

In this work, everything previously mentioned has been taken into account for the design of fuzzy systems that dynamically adapt the c_3 and c_4 parameters during the execution of galactic swarm optimization. In this case, these parameters are fuzzy values that are defined as shown in expressions (8) and (9) [17]:

$$C_3 = \frac{\sum_{i=1}^{r_{c_3}} \mu_i^{C_3}\left(C_{3i}\right)}{\sum_{i=1}^{r_{c_3}} \mu_i^{C_3}}, \tag{8}$$

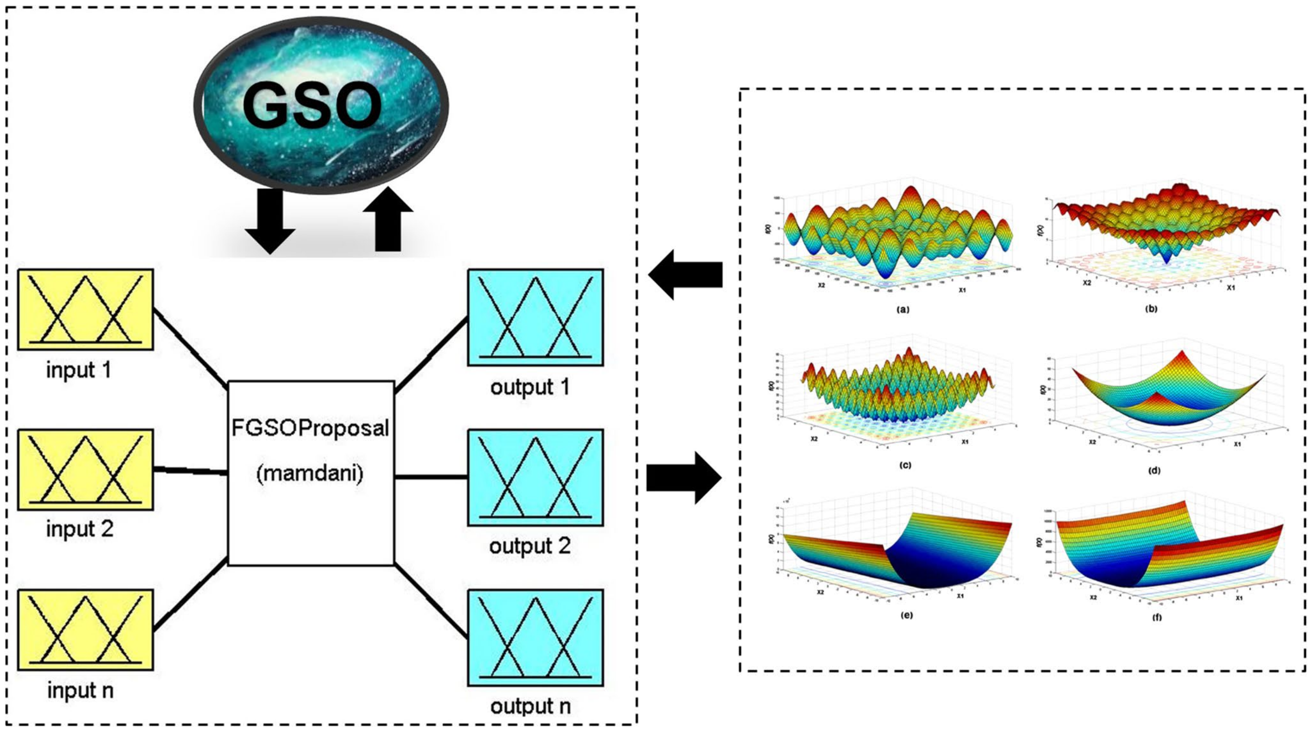

Fig. 3 Proposed FGSO

where r_{c_3} is the total number of rules for the fuzzy system, the output result for rule i is represented by C_{3i} and $\mu_i^{C_3}$ is the membership function associated to rule i

$$C_4 = \frac{\sum_{i=1}^{r_{c_4}} \mu_i^{C_4}\left(C_{4i}\right)}{\sum_{i=1}^{r_{c_4}} \mu_i^{C_4}}, \tag{9}$$

where r_{c_4} is the total number of rules for the fuzzy system, the output result for rule i is represented by C_{4i} and $\mu_i^{C_4}$ is the membership function associated to rule i.

The design of the FGSO1 fuzzy system was based on our previous work [2] since it was our first proposal of the dynamic adjustment of parameters using fuzzy logic in galactic swarm optimization. Given that a good behavior was observed in relation to mathematical functions, the decision was made to propose a variant of the FGSO1 fuzzy system, therefore, the FGSO2 fuzzy system is proposed in which diversity has been added as an input variable expecting to obtain significant improvements.

The fuzzy system design is illustrated in Figs. 4 and 5, the fuzzy systems are of Mamdani type, where in the first fuzzy system the iteration is used as an input variable and as outputs we have the c_3 and c_4 parameters. In the second fuzzy system, we used the iteration and the diversity as input variables and c_3 and c_4 as output variables.

The input variables are divided into three triangular membership functions labeled as "low", "medium" and "high" in a range from 0 to 1 for both fuzzy systems as can be noticed in Figs. 6 and 7. A triangular membership function is represented by three parameters a, b, and c as shown below [8]:

$$\text{triangle}\,(x;\, a,\, b,\, c) = \begin{cases} 0, & x \le a \\ \frac{x-a}{b-a} & a \le x \le b \\ \frac{c-x}{c-b} & b \le x \le c \\ 0, & c \le x \end{cases}. \tag{10}$$

In this case, the parameters a, b and c with $a < b$ and $b < c$ are the coordinates of the three corners of the triangular membership function.

For the first fuzzy system, the output variables are granulated into three triangular functions of the same type as "low", "medium" and "high" with a range of 0–3 as shown in Figs. 8 and 9, and for the second fuzzy system the output variables are granulated into five triangular membership functions "low", "medium low", "medium", "medium high", and "high" in a range from 0 to 3 as shown in Figs. 10 and 11, respectively.

The iteration is normalized to obtain a percentage of the current iteration for the total number of iterations, the diversity represents the mean of the Euclidean distance between all the particles and the best particle found, in other words,

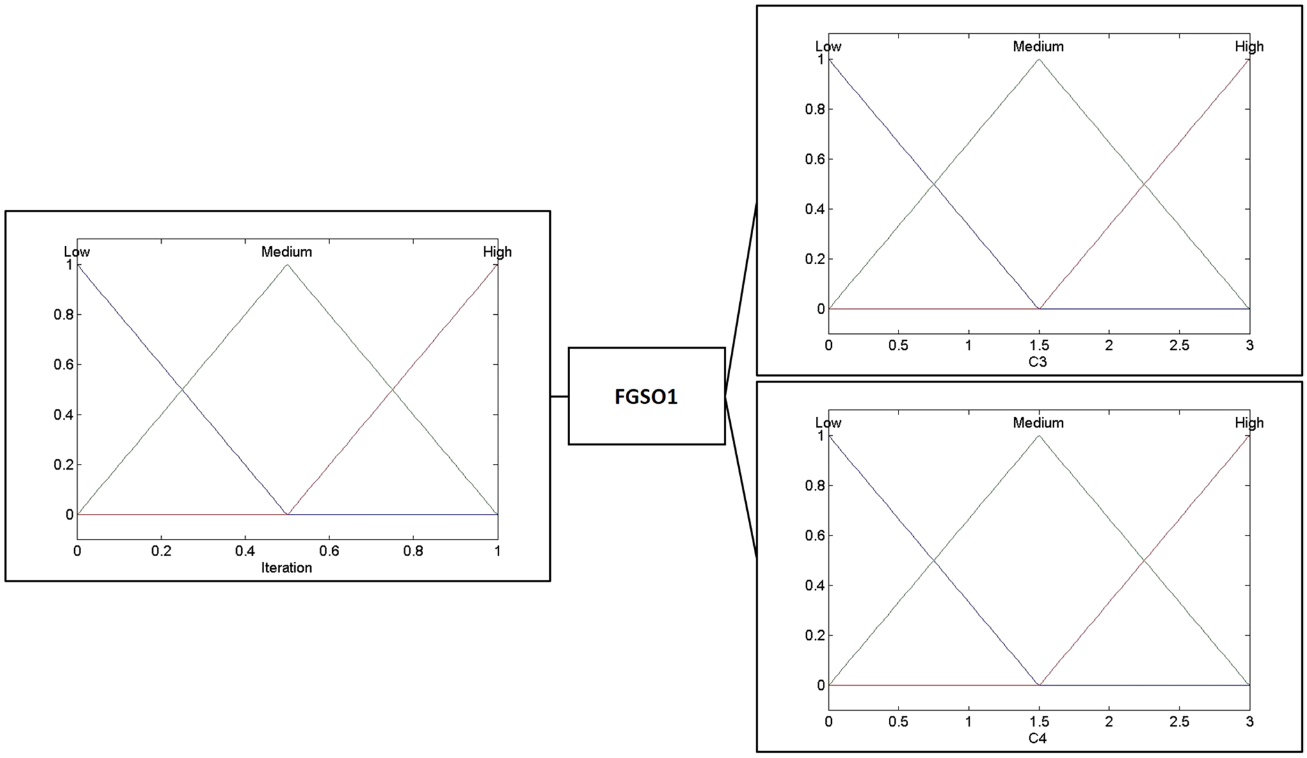

Fig. 4 Fuzzy system FGSO1

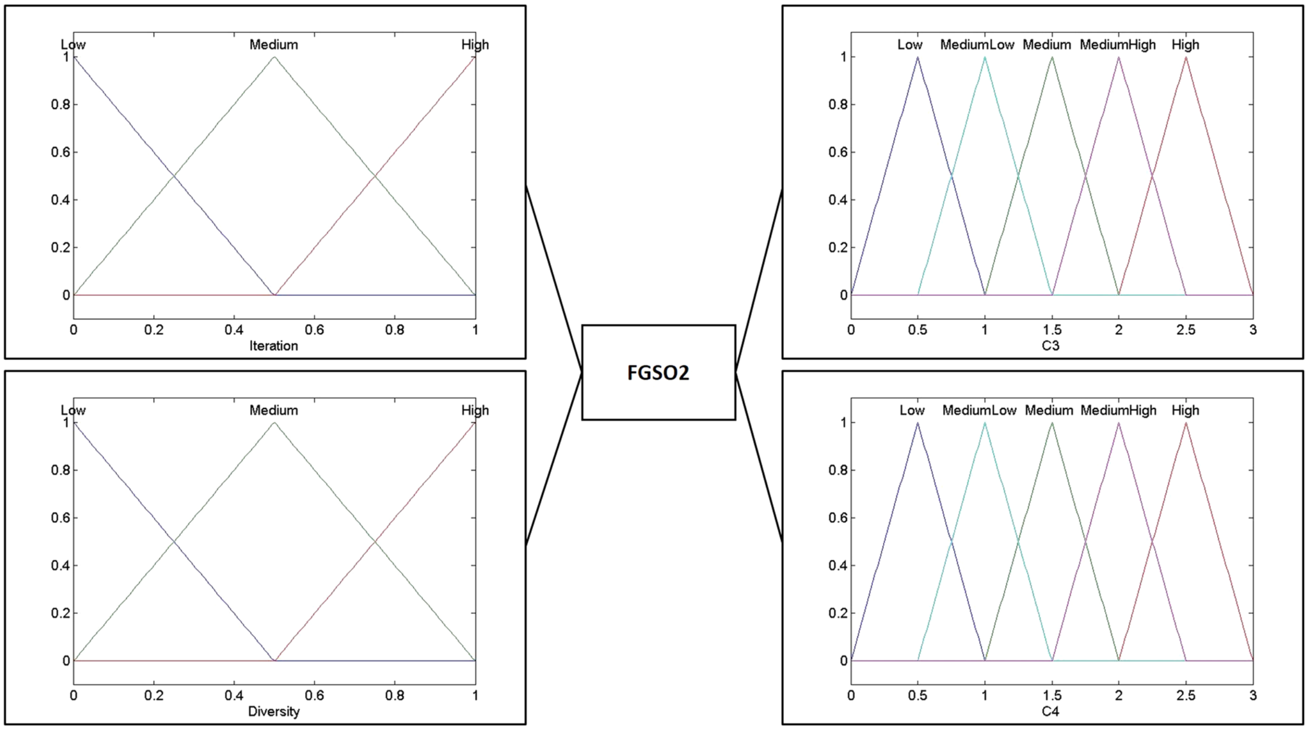

Fig. 5 Fuzzy system FGSO2

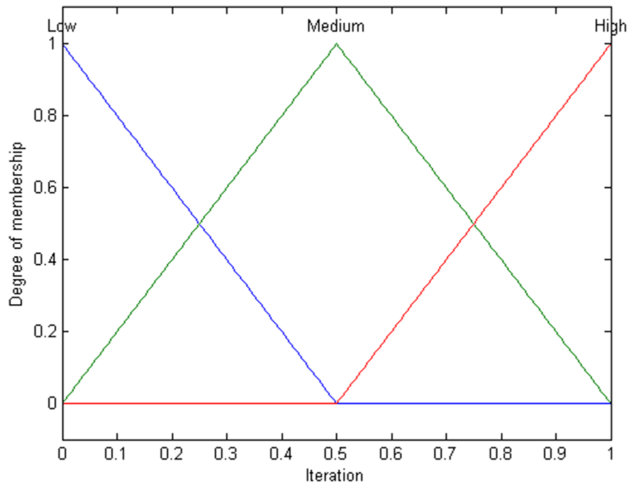

Fig. 6 Input iteration

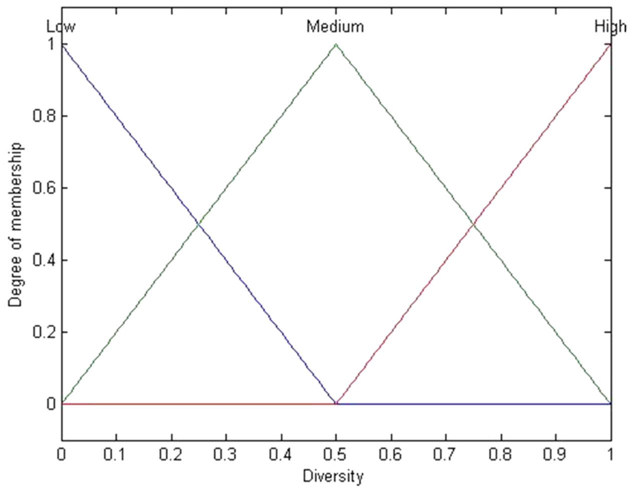

Fig. 7 Input diversity

it measures the degree of dispersion of the particles, and the expressions used to obtain the iteration and the diversity are presented below [15, 30]:

$$\text{Iteration} = \frac{\text{Current Iteration}}{\text{Total of Iterations}}, \tag{11}$$

$$\text{Diversity}\ \ (S(t)) = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\sum_{j=1}^{D}\left(x_{ij}(t)\right)^2 - \left(\bar{x}_j(x)\right)^2}, \tag{12}$$

where the population of the GSO is represented by S, the size of the population is n, D is the number of dimensions in which the population moves, x_{ij} is the solution i in the dimension j and $\bar{x}_j$ is the best solution found in dimension j [30].

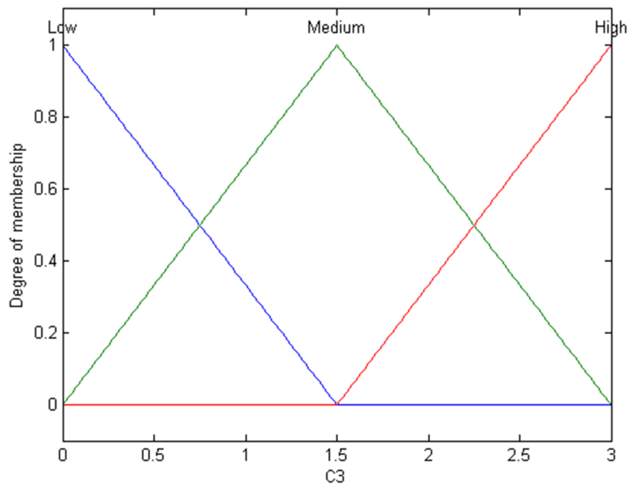

Fig. 8 Output c_3 of FGSO1

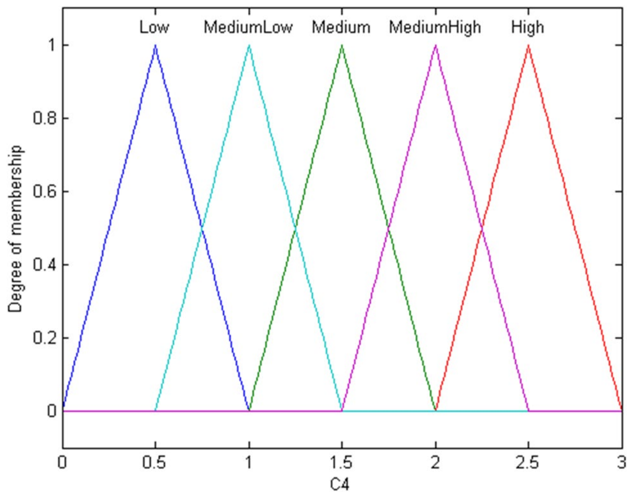

Fig. 11 Output c_4 of FGSO2

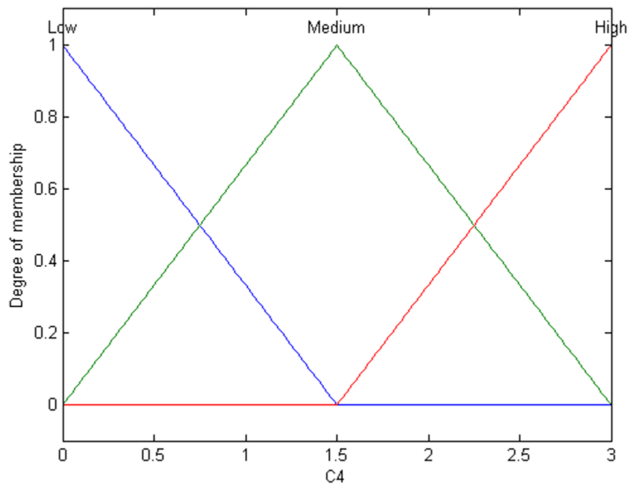

Fig. 9 Output c_4 of FGSO1

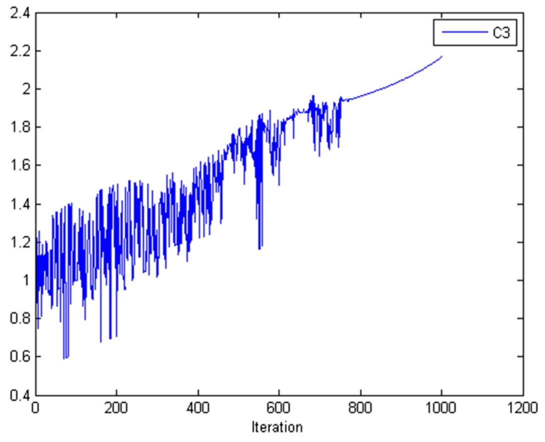

Fig. 12 Behavior of the c_3 parameter

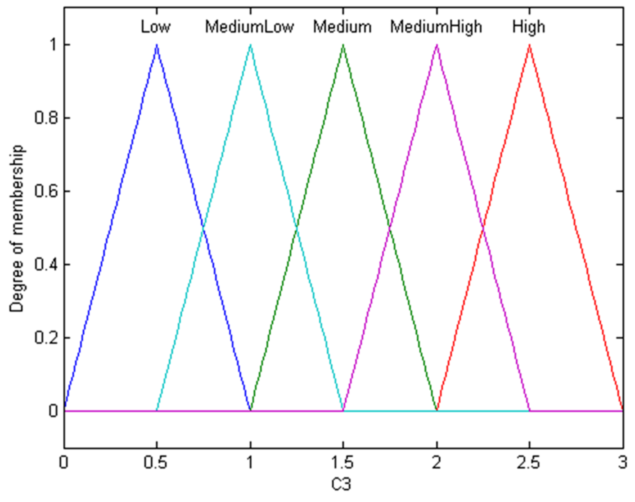

Fig. 10 Output c_3 of FGSO2

In Figs. 12 and 13 we can find an example of the behavior of the c_3 and c_4 parameters as the iterations are executed, in addition, in Fig. 14 we illustrate the diversity that exists between the individuals of galactic swarm optimization and how it varies as the iterations pass.

The rules of the FGSO1 fuzzy system were designed with the idea that c_3 is increasing and c_4 is decreasing in this way aiming at achieving that when the algorithm is in the initial iterations it can explore and when it is in the last iterations it can exploit [2, 13].

Rules of the FGSO1 fuzzy system:

1. If (Iteration is Low) then (c_3 is Low) and (c_4 is High)
2. If (Iteration is Medium) then (c_3 is Medium) and (c_4 is Medium)
3. If (Iteration is High) then (c_3 is High) and (c_4 is Low)

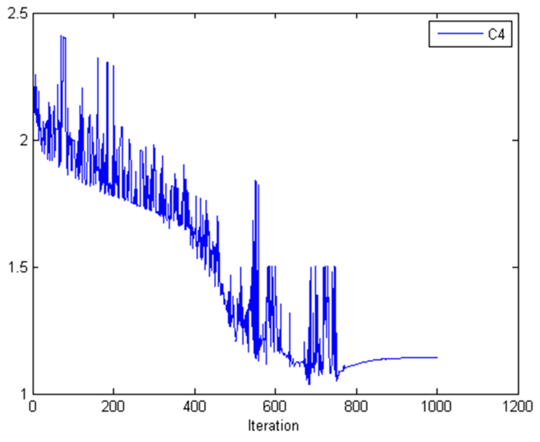

Fig. 13 Behavior of the c_4 parameter

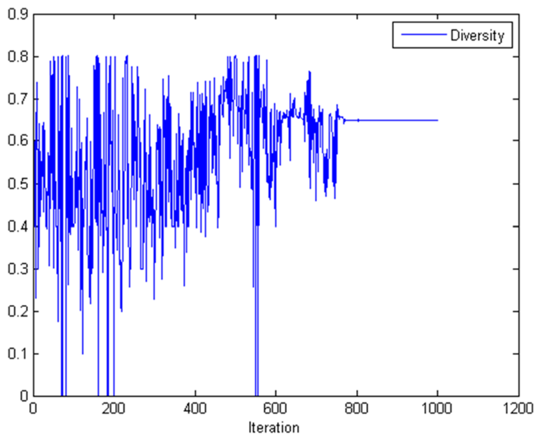

Fig. 14 Behavior of the diversity parameter

Taking into account the diversity as input variable used in the second fuzzy system along with the iteration input variable, the FGSO2 fuzzy system rules are designed in such a way that when the diversity is low, the particles are too close together then we need to explore and when we have a high diversity we need to exploit. The idea of the rule design is very similar to those used in the FGSO1 fuzzy system, since when we are in the initial iterations we need to explore and when we are in the final stage of the execution of the algorithm then we need to exploit [29].

Rules of the FGSO2 fuzzy system:

1. If (Iteration is Low) and (Diversity is Low) then (c_3 is Low) (c_4 is High)
2. If (Iteration is Low) and (Diversity is Medium) then (c_3 is MediumLow) (c_4 is MediumHigh)
3. If (Iteration is Low) and (Diversity is High) then (c_3 is Medium) (c_4 is High)
4. If (Iteration is Medium) and (Diversity is Low) then (c_3 is MediumLow) (c_4 is MediumHigh)
5. If (Iteration is Medium) and (Diversity is Medium) then (c_3 is Medium) (c_4 is Medium)
6. If (Iteration is Medium) and (Diversity is High) then (c_3 is MediumHigh) (c_4 is MediumLow)
7. If (Iteration is High) and (Diversity is Low) then (c_3 is MediumHigh) (c_4 is MediumLow)
8. If (Iteration is High) and (Diversity is Medium) then (c_3 is MediumHigh) (c_4 is Medium)
9. If (Iteration is High) and (Diversity is High) then (c_3 is High) (c_4 is Low)

Mathematical Functions and the Benchmark Water Tank Problem for Testing the Galactic Swarm Optimization

Water Tank Controller Problem

Most optimization problems in the real world require a lot of computational cost for the evaluation of their possible solutions, and given the limitations of computational or time resources, specialized optimization algorithms are usually required in industrial applications. In the last years, several methods of computational optimization have been proposed to deal with problems where too much computational cost is required and with these methods good results have been obtained [2, 15].

To test the proposed fuzzy approach of the fuzzy galactic swarm optimization with parameter adaptation we use a set of 20 mathematical functions found in [19, 20], where their mathematical representation is presented as well as their search space and their global minimum. We also tested the proposed method with the problem of the water controller, which is described in more detail below. The main idea of our proposed method is to search for the minimum of the benchmark mathematical functions, as well as the best fuzzy controller for the plant of the water tank [34, 35]. In Table 1 we can find the mathematical functions used to test the proposed approach and the original GSO algorithm.

In Table 1 we can find the set of mathematical functions used to measure the performance of the GSO algorithm and the proposed fuzzy GSO approach, in addition to the range of the search space and its global minimum.

Water Tank Controller Problem

The fuzzy controller for the benchmark water tank problem refers to controlling the water level in a tank, therefore, it is necessary to know the water level in the tank and with this be able to establish the opening of the valve. To evaluate the opening of the valve in an accurate way, we rely on

Table 1 Benchmark mathematical functions

Function
$f_1(x) = \sum_{1=1}^{n} x_i^2$
Search space $x_j \in [-5.12,\ 5.12]$ and $f(x^*) = 0$
$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
Search space $x_j \in [-10,\ 10]$ and $f(x^*) = 0$
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j \right)^2$
Search space $x_j \in [-100,\ 100]$ and $f(x^*) = 0$
$f_4(x) = \max_i \left\{ \lvert x_i \rvert,\ 1 \le i \le n \right\}$
Search space $x_j \in [-100,\ 100]$ and $f(x^*) = 0$
$f_5(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2 \right]$
Search space $x_j \in [-30,\ 30]$ and $f(x^*) = 0$
$f_6(x) = \sum_{i=1}^{n} \left(\left[x_i + 0.5 \right] \right)^2$
Search space $x_j \in [-100,\ 100]$ and $f(x^*) = 0$
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}\ [0,\ 1]$
Search space $x_j \in [-1.28,\ 1.28]$ and $f(x^*) = 0$
$f_8(x) = \sum_{i-1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert} \right)$
Search space $x_j \in [-500,\ 500]$ and $f(x^*) = -418.9829$
$f_9(x) = \sum_{i=1}^{n} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10 \right]$
Search space $x_j \in [-5.12,\ 5.12]$ and $f(x^*) = 0$
$f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos\left(2\pi x_i\right) \right) + 20 + e$
Search space $x_j \in [-32,\ 32]$ and $f(x^*) = 0$
$f_{11}(x) = \frac{1}{400}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$
Search space $x_j \in [-600,\ 600]$ and $f(x^*) = 0$
$f_{12}(x) = \frac{\pi}{n}\left\{ 10\sin\left(\pi y_1\right) + \sum_{i=1}^{n-1} \left(y_i - 1\right)^2 \left[1 + 10\sin^2\left(\pi y_{i+1}\right)\right] + \left(y_n - 1\right)^2 \right\} + \sum_{i=1}^{n} u\left(x_i, 10, 100, 4\right)$
$y_i = 1 + \frac{x_i + 1}{4}$
Search space $x_j \in [-50,\ 50]$ and $f(x^*) = 0$
$f_{13}(x) = 0.1\left\{ \sin^2\left(3\pi x_1\right) + \sum_{i=1}^{n} \left(x_i - 1\right)^2 \left[1 + \sin^2\left(3\pi x_i + 1\right)\right] + \left(x_n - 1\right)^2 \left[1 + \sin^2\left(2\pi x_n\right)\right] \right\} + \sum_{i=1}^{n} u\left(x_i, 5, 100, 4\right)$
Search space $x_j \in [-50,\ 50]$ and $f(x^*) = 0$
$\text{Rosenbrock}\ (x) = \sum_{i=1}^{n-1} \left[100\left(x_i + x_i^2\right)^2 + \left(x_{i-1}\right)^2 \right]$
Search space $x_j \in [-5,\ 10]$ and $f(x^*) = 0$
$\text{SumSquares}\ (x) = \sum_{i=1}^{n} i x_i^2$
Search space $x_j \in [-10,\ 10]$ and $f(x^*) = 0$
$\text{Zakharov}\ (x) = \sum_{i=1}^{n} x_i^2 + \left(\sum_{i=1}^{n} 0.5 i x_i \right)^2 + \left(\sum_{i=1}^{n} 0.5 i x_i \right)^4$
Search space $x_j \in [-5,\ 10]$ and $f(x^*) = 0$
$\text{Shubert}\ (x) = \left(\sum_{i=1}^{5} i\cos\left((i+1)x_1 + i\right) \right)\left(\sum_{i=1}^{5} i\cos\left((i+1)x_2 + i\right) \right)$
Search space $x_j \in [-10,\ 10]$ and $f(x^*) = -186.7309$

Table 1 (continued)

Function
Baele $(x) = \left(1.5 - x_1 + x_1x_2\right)^2 + \left(2.25 - x_1 + x_1x_2^2\right)^2 + \left(2.625 - x_1 + x_1x_2^3\right)^2$ Search space $x_j \in [-4.5, 4.5]$ and $f(x^*) = 0$
Booth $(x) = \left(x_1 + 2x_2 - 7\right)^2 + \left(2x_1 + x_2 - 5\right)^2$ Search space $x_j \in [-10,\ 10]$ and $f(x^*) = 0$
Dixon - Price $(x) = \left(x_1 - 1\right)^2 + \sum_{i=2}^{n} i\left(2x_i^2 - x_{i-1}\right)^2$ Search space $x_j \in [-10,\ 10]$ and $f(x^*) = 0$

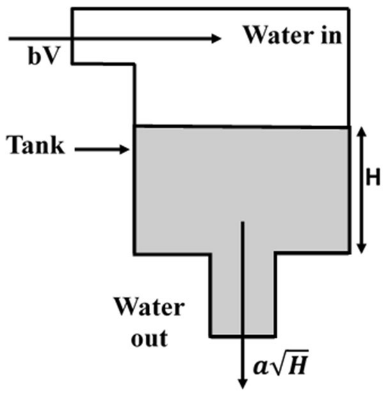

Fig. 15 Water tank model

fuzzy logic, which we implement as a fuzzy controller for the water level and how quickly it is introduced to maintain the water level in the tank.

The filling of the water tank is represented by a differential equation for the water level of the tank, H, as shown below, and Fig. 15 shows a representation of the water tank model [34]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{Vol} = A\frac{\mathrm{d}H}{\mathrm{d}t} = bV - a\sqrt{H}, \tag{13}$$

where the variable H is the level of the water in the tank, Vol is the volume of water contained in the tank and V is the voltage that is applied to the pump.

Parameter A is the cross-sectional area of the tank, b is a constant associated with the flow of water to the tank and a is a constant associated to the outflow of the tank [35].

Fuzzy System for the Fuzzy Controller of the Water Tank Problem

The fuzzy system of the water tank controller is of the Mamdani type and composed of two input variables and one output variable [35]: the first input variable is *Level*, which is composed of three Gaussian membership functions labeled *High, Okay* and *Low.* The second input variable is *Rate*, which has three Gaussian membership functions labeled *Negative, None,* and *Positive.*

The output is *Valve* (Tank Fill Valve), which is composed of five triangular membership functions labeled as *Close_fast, Close_slow, No_change, Open_slow*, and *Open_fast*.

In Fig. 16 we can find the fuzzy system of the fuzzy controller for the water tank, where the input and output variables are displayed graphically.

The rules of the fuzzy system of the water tank help maintain control of the opening of the valve and thus the water level in the tank.

Rules for the fuzzy system of the water tank controller:

1. If (level is okay) then (valve is no_change)
2. If (level is low) then (valve is open_fast)
3. If (level is high) then (valve is close_fast)
4. If (level is okay) and (rate is positive) then (valve is close_slow)
5. If (level is okay) and (rate is negative) then (valve is open_slow)

Experiments and Comparison of Results

To test and compare the proposed method with the original GSO algorithm, we used 20 benchmark mathematical functions and the fuzzy controller for the water tank controller presented in "Fuzzy Approach in Galactic Swarm Optimization". In this work, each metaheuristic algorithm is tested with 30 independently performed tests to obtain greater confidence on the obtained results in the simulations.

The c_3 and c_4 parameters are fixed for the results obtained with the original GSO algorithm and in our proposal c_3 and c_4 are dynamically adjusted using fuzzy systems to perform the adaptation of the parameters.

The results presented in Tables 3, 4 and 5 are separated by the number of dimensions ranging from 10, 30 and 50 dimensions. The parameters used during the simulations are presented below in Table 2, so that a fair comparison for the tests is performed with the same parameters.

The nomenclature used in the tables is summarized below:

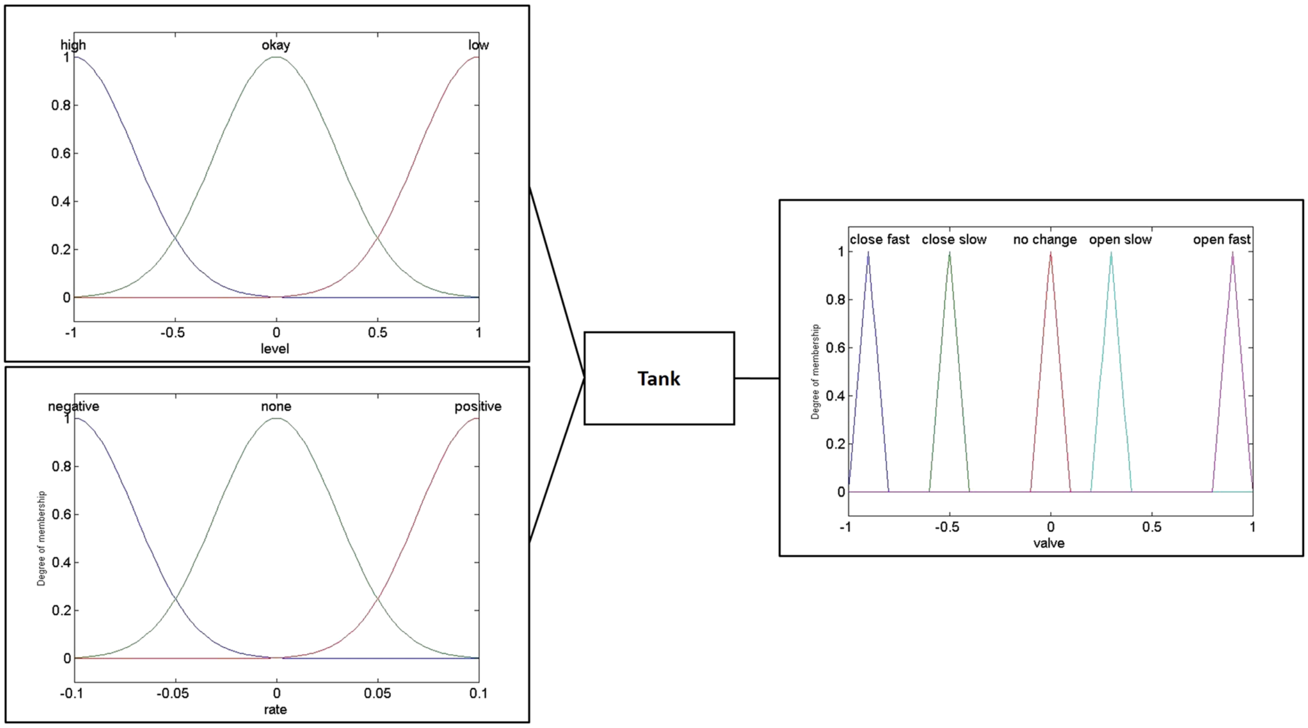

Fig. 16 Fuzzy system for the tank

Table 2 Parameters for the simulations

Parameter	Value	Value	Value
Dimensions	10	30	50
Population	5	5	5
Subpopulation	10	20	20
Iteration 1	100	150	250
Iteration 2	1000	1500	1500
No epochs	5	5	9
c_1 and c_2	2	2	2
c_3 and c_4	2	2	2

GSO: is the original galactic swarm optimization.

FGSO1: is the first proposed fuzzy system with iteration as the input variable and c_3 and c_4 as the output variables.

FGSO2: is the second proposed fuzzy system with iteration and diversity as the input variables and c_3 and c_4 as the output variables.

In Table 3 we can find the averages for each of the 20 mathematical functions used to measure the performance of GSO and FGSO after 30 runs for 10 dimensions.

In Table 3 we can find the averages for each of the 20 mathematical functions used to measure the performance of GSO and FGSO after 30 runs for 10 dimensions. Where it is observed that the original galactic swarm optimization has better performance than the proposed FGSO1 approach in functions f5, f6, f7 and f8, and the FGSO1 approach manages to improve the performance in Rosenbrock, Baele and Dixon-price functions. The FGSO2 approach shows improvements in Rosenbrock, Baele, Dixon-price and f8 functions, but lowers the performance in functions f5, f6, f7, f12 and f13, and in the remaining functions the proposed approach manages to reach the global minimum for each of the functions.

In Table 4 we can find the averages for each of the 20 mathematical functions used to measure the performance of GSO and FGSO after 30 runs for 30 dimensions. Where it is observed that the original galactic swarm optimization has better performance than the proposed FGSO1 approach in functions f5, f6, f7 and f8, and the FGSO1 approach manages to improve the performance in Rosenbrock, Baele and Dixon-price functions. The FGSO2 approach shows improvements in Rosenbrock, Baele, Dixon-price and f8 functions, but lowers the performance in functions f5, f6, f7, f12 and f13, and in the remaining functions the proposed approach manages to reach the global minimum for each of the functions.

In Table 5 we can find the averages for each of the 20 mathematical functions used to measure the performance of GSO and FGSO after 30 runs for 50 dimensions. In this case, it is observed that the original galactic swarm optimization has better performance than the proposed FGSO1 approach in functions f7 and f8, and the FGSO1 approach manages to

Table 3 Experimental results with GSO and FGSO for ten dimensions

Function	GSO		FGSO1		FGSO2	
	Average	Std	Average	Std	Average	Std
f1	0	0	0	0	0	0
f2	0	0	0	0	0	0
f3	0	0	0	0	0	0
f4	0	0	0	0	0	0
f5	0	0	5.710E−06	2.419E−05	1.937E−03	8.830E−03
f6	0	0	4.810E−06	9.248E−06	6.196E−05	1.128E−04
f7	0	0	1.350	0.133	1.348	0.151
f8	−4067.029	92.873	−4102.303	128.559	−4040.478	118.381
f9	0	0	0	0	0	0
f10	0	0	0	0	0	0
f11	0	0	0	0	0	0
f12	0	0	0	0	1.052E−04	2.032E−04
f13	0	0	0	0	8.140E−05	1.348E−04
Rosenbrock	6.670E−07	1.037E−06	7.510E−12	2.705E−11	3.150E−08	1.034E−07
Sumsquare	0	0	0	0	0	0
Zakharov	0	0	0	0	0	0
Shubert	−186.731	0	−186.731	0	−186.731	0
Baele	3.000E−21	9.201E−21	1.910E−24	7.060E−24	2.057E−23	6.924E−23
Booth	0	0	0	0	0	0
Dixon-Price	7.725E−02	3.195E−02	0	0	0	0

Table 4 Experimental results with GSO and FGSO for 30 dimensions

Function	GSO		FGSO1		FGSO2	
	Average	Std	Average	Std	Average	Std
f1	0	0	0	0	0	0
f2	0	0	0	0	0	0
f3	0	0	0	0	0	0
f4	0	0	0	0	0	0
f5	0	0	3.930E−07	2.052E−06	7.687E−03	2.110E−02
f6	0	0	6.450E−07	1.385E−06	5.334E−05	1.044E−04
f7	0	0	7.911	0.305	7.962	0.337
f8	−11,366.043	456.361	−11,580.548	560.526	−10,959.443	477.883
f9	0	0	0	0	0	0
f10	0	0	0	0	0	0
f11	0	0	0	0	0	0
f12	0	0	0	0	0.128	0.122
f13	0	0	0	0	1.877E−02	4.163E−02
Rosenbrock	4.910E−07	1.420E−06	3.210E−08	1.748E−07	0	0
Sumsquare	0	0	0	0	0	0
Zakharov	0	0	0	0	0	0
Shubert	−186.731	0	−186.731	0	−186.731	0
Baele	0	0	0	0	0	0
Booth	0	0	0	0	0	0
Dixon-Price	0.667	1.841E−08	0	0	0	0

improve the performance in Rosenbrock, Baele and Dixon-price functions. The FGSO2 approach shows improvements in Rosenbrock, Baele, Dixon-price and f8 functions, but lowers the performance in functions f7, f12 and f13, and in the remaining functions the proposed approach manages to reach the global minimum for each of the functions.

Table 5 Experimental results with GSO and FGSO for 50 dimensions

Function	GSO		FGSO1		FGSO2	
	Average	Std	Average	Std	Average	Std
f1	0	0	0	0	0	0
f2	0	0	0	0	0	0
f3	0	0	0	0	0	0
f4	0	0	0	0	0	0
f5	0	0	0	0	0	0
f6	0	0	0	0	0	0
f7	0	0	15.561	0.359	15.419	0.457
f8	− 18,626.611	806.593	− 18,964.119	895.925	− 17,992.640	768.958
f9	0	0	0	0	0	0
f10	0	0	0	0	0	0
f11	0	0	0	0	0	0
f12	0	0	0	0	0.437	0.276
f13	0	0	0	0	3.545E−02	0.105
Rosenbrock	4.400E−09	2.232E−08	3.690E−12	1.968E−11	0	0
Sumsquare	0	0	0	0	0	0
Zakharov	0	0	0	0	0	0
Shubert	− 186.731	0	− 186.731	0	− 186.731	0
Baele	0	0	0	0	0	0
Booth	0	0	0	0	0	0
Dixon-Price	0.667	2.314E−08	0	0	0	0

To clearly observe the behavior of our proposal and be able to test it, a toolbox was designed to run functions f1–f13 presented in "Mathematical Functions and the Benchmark Water Tank Problem for Testing the Galactic Swarm Optimization", and below the link to access the FGSO toolbox is provided https://www.dropbox.com/s/z3epfnx857wrtro/FGSO.rar?dl=0.

The results obtained after performing the parameter optimization of the membership functions of the fuzzy controller for the water tank were performed by dynamically adapting the c_3 and c_4 parameters of the GSO algorithm, as well as with the original GSO algorithm.

In this case, the expression to calculate the fitness value of each particle is replaced by the expression to calculate the mean squared error (MSE) and thus in this way measure the efficiency of the fuzzy controller obtained by galactic swarm optimization, and the expression of the mean squared error (MSE) is the following [35, 36]:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(X_i - Y_i\right)^2, \quad (14)$$

where MSE is the sum of squared errors, that is, the difference between the estimator and what is being estimated, X_i represents the reference value and Y_i represents the value produced by the system.

Table 6 Configuration parameters for GSO and FGSO applied to the water tank problem

Parameter	Value
Population (N)	10
Subpopulation (M)	5
Iteration 1 (I_1)	50
Iteration 2 (I_2)	100
No. of epochs	5
c_1 and c_2	2
c_3 and c_4	2

Table 7 Experimental results for GSO and FGSO applied to the water tank problem

	GSO		FGSO	
	MSE	RMSE	MSE	RMSE
Best	7.603E−02	0.276	1.348E−02	0.116
Worst	7.875E−02	0.281	7.893E−02	0.281
Average	7.801E−02	0.279	4.515E−02	0.202
STD	5.758E−04	1.034E−03	2.722E−02	6.614E−02

The parameters used for the tests performed with the fuzzy controller of the water tank problem are the following (Table 6).

In Table 7 we can find the best, worst, average and standard deviation of the fuzzy controller of the water tank used to measure the performance of the GSO algorithm and the proposed FGSO after 30 runs.

For this case study, the proposed approach manages to obtain improvements with respect to the original GSO algorithm. In Fig. 17, we can find the best simulation of the fuzzy controller obtained by the proposed fuzzy galactic swarm optimization (FGSO), where the red line represents the reference data, and the blue line represents the data obtained by the fuzzy controller developed to follow the reference.

After analyzing the results of the fuzzy controller of the water tank problem, we decided to perform a statistical test between the fuzzy galactic swarm optimization (FGSO) and the galactic swarm optimization to obtain more evidence of the improvements obtained with the proposed method. The statistical test used for the comparison is the z test [3, 34] with the following characteristics:

- μ_1 = Mean of fuzzy galactic swarm optimization (FGSO).
- μ_2 = Mean of galactic swarm optimization (GSO).
- $H_0 = \mu_1 \geq \mu_2$.
- $H_a = \mu_1 < \mu_2$ (claim).
- Claim: the average of the FGSO is lower than the average of the GSO (claim).
- Confidence level = 95%.
- $\propto = 0.05$.

Table 8 Results of z test for FGSO and GSO for the water tank problem

FGSO		GSO		Value of Z
Average	Std	Average	Std	
4.515E−02	2.722E−02	7.801E−02	5.758E−04	**− 6.612**

Table 8 shows the averages, standard deviations and the value of the statistical z test for fuzzy galactic swarm optimization (FGSO) and galactic swarm optimization (GSO) applied to the fuzzy controller of the water tank problem.

The statistical z test for the fuzzy controller of the water tank problem was performed with a confidence level of 95%, alpha 0.05, where Ha states that the average of the FGSO is lower than the average of the GSO. Ho states that the average of the FGSO is greater than or equal to the average of the GSO, with a critical value of − 1.645. The resulting value confirms that there is sufficient evidence to reject Ho, therefore, Ha is accepted, indicating that the average of the fuzzy galactic swarm optimization is lower than the average of the galactic swarm optimization.

Statistical Comparison of Fuzzy Galactic Swarm Optimization and the Modified Grey Wolf Optimizer

The fuzzy galactic swarm optimization (FGSO) and modified grey wolf optimizer are compared with 7 benchmark functions with 30, 64 and 128 dimensions, the results

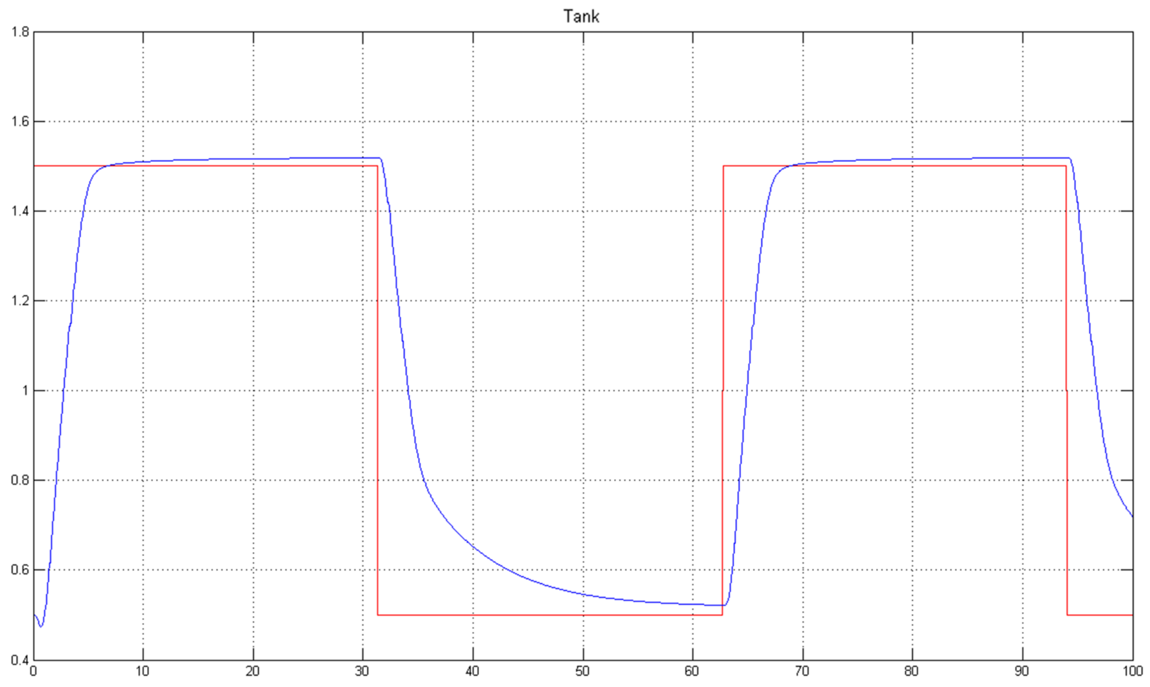

Fig. 17 Best simulation of fuzzy controller for water tank problem

obtained by the fuzzy galactic swarm optimization and modified grey wolf optimizer are presented in different tables according to the number of dimensions.

With the intention that the statistical comparison was as fair as possible, we aimed at having a similar number of evaluations between fuzzy galactic swarm optimization (FGSO) and the modified GWO. To obtain the number of evaluations of FGSO, first, the number of evaluations of the first level is obtained by multiplying the number of subpopulations, the size of the population and the number of iterations of the first level in the following way ($M \times N \times I_1$). In the second level, the number of subpopulations is multiplied by the number of iterations of the second level ($M \times I_2$), then the number of evaluations of the first level and those of the second level are added, and finally, it is multiplied by the number of predefined epochs [1].

The parameters of fuzzy galactic swarm optimization and the modified grey wolf optimizer are given in Table 9.

A statistical test was performed between the fuzzy galactic swarm optimization and modified grey wolf optimizer. The statistical test that was used is the *z* test [3] with the following characteristics:

- μ_1 = Mean of the fuzzy galactic swarm optimization (FGSO)
- μ_2 = Mean of the Modified Grey Wolf Optimizer.
- $H_0 = \mu_1 \geq \mu_2$.
- $H_a = \mu_1 < \mu_2$ (claim).
- Claim: the average of the FGSO is lower than the average of the modified grey wolf optimizer (claim).
- Confidence level = 95%.
- $\propto = 0.05$.
- Critical value $Z_0 = -1.645$.

Table 9 Parameters of FGSO and the modified GWO

FGSO		Modified GWO [3]	
Parameter	Value	Parameter	Value
Population (N)	10	Wolves	30
Subpopulation (M)	5	a[0.5–2.5]	Dynamic
Iteration 1 (I_1)	50	Iteration	500
Iteration 2 (I_2)	500	No. of evaluations	15,000
No. of epoch	3		
No. of evaluations	15,000		

Table 10 shows the averages, standard deviations and the values of the statistical *z* test for the Sphere, Rosenbrock, Quartic, Schwefel, Rastrigin, Ackley and Griewank functions for 30 dimensions.

In Table 11 we can find the averages, standard deviations and the values of the statistical *z* test for the Sphere, Rosenbrock, Quartic, Schwefel, Rastrigin, Ackley and Griewank functions for 64 dimensions.

In Table 12 we can find the averages, standard deviations and the values of the statistical *z* test for the Sphere, Rosenbrock, Quartic, Schwefel, Rastrigin, Ackley and Griewank functions for 128 dimensions.

A statistical *z* test is applied, with a confidence level of 95%, an alpha value of 0.05, where Ha states that the average of the FGSO is lower than the average modified grey wolf optimizer and Ho states that the average of FGSO is greater than or equal to the average modified grey wolf optimizer, with a critical value of − 1.645. The results support that for the Sphere, Rosenbrock, Quartic, Schwefel, Rastrigin, Ackley and Griewank functions there is sufficient evidence to reject Ho, therefore, Ha is accepted, stating that the average of FGSO is lower than the average of the modified grey wolf optimizer according to the results shown in Table 10 for experiments with 30 dimensions.

In applying the statistical *z* test, with a confidence level of 95%, an alpha value of 0.05, where Ha states that the average of the FGSO is lower than the average modified grey wolf optimizer and Ho states that the average of FGSO is greater than or equal to the average modified grey wolf optimizer, with a critical value of − 1.645. The results support that for the Sphere, Rosenbrock, Quartic, Schwefel, Rastrigin, Ackley and Griewank functions there is sufficient evidence to

Table 10 Results of the *z* test for FGSO and modified GWO with 30 dimensions

30 Dimensions					
Function	FGSO1		Modified GWO [3]		Value of *Z*
	Average	Std	Average	Std	
Sphere	0	0	5.610E−43	1.020E−42	− 3.012
Rosenbrock	2.142E−03	7.112E−03	27.274	0.772	− 193.555
Quartic	0	0	1.500E−03	8.470E−04	− 9.700
Schwefel	− 7705.048	682.176	− 4258.280	440.440	23.250
Rastrigin	0	0	0.384	1.462	− 1.438
Ackley	0	0	1.500E−14	2.380E−15	− 34.520
Griewank	0	0	3.400E−03	8.000E−03	− 2.328

Table 11 Results of the *z* test for FGSO and modified GWO with 64 dimensions

64 Dimensions					
Function	FGSO1		Modified GWO [3]		Value of *Z*
	Average	STD	Average	STD	
Sphere	0	0	1.540E−26	1.820E−26	− 4.635
Rosenbrock	4.272E−02	7.941E−02	61.736	0.745	− 450.949
Quartic	0	0	3.000E−03	1.420E−03	− 11.572
Schwefel	− 11,322.755	844.075	− 6424.990	882.538	21.967
Rastrigin	0	0	3.322	6.532	− 2.785
Ackley	0	0	8.760E−14	1.280E−14	− 37.485
Griewank	0	0	2.900E−03	7.200E−03	− 2.206

Table 12 Results of the *z* test for FGSO and modified GWO with 128 dimensions

128 Dimensions					
Function	FGSO1		Modified GWO [3]		Value of *Z*
	Average	STD	Average	STD	
Sphere	0	0	1.720E−17	1.830E−17	− 5.148
Rosenbrock	0.253	0.303	125.884	0.663	− 943.980
Quartic	0	0	6.800E−03	3.100E−03	− 12.015
Schwefel	− 16,650.884	1425.982	− 9000.040	1179.718	22.643
Rastrigin	0	0	8.337	10.175	− 4.487
Ackley	0	0	3.650E−10	1.770E−10	− 11.295
Griewank	0	0	5.930E−03	1.180E−02	− 2.753

reject Ho, therefore, Ha is accepted, stating that the average of FGSO is lower than the average of the modified grey wolf optimizer according to the results shown in Table 11 for experiments with 64 dimensions.

Once the statistical *z* test was performed, with a confidence level of 95%, an alpha value of 0.05, where Ha states that the average of the FGSO is lower than the average of the modified grey wolf optimizer and Ho states that the average of FGSO is greater than or equal to the average modified grey wolf optimizer, with a critical value of -1.645. The results support that for the Sphere, Rosenbrock, Quartic, Rastrigin, Ackley and Griewank functions there is sufficient evidence to reject Ho, therefore, Ha is accepted, stating that the average of FGSO is lower than the average of the modified grey wolf optimizer according to the results shown in Table 12 for experiments with 128 dimensions.

Conclusions

The galactic swarm optimization has been shown to behave well in the face of multimodal problems and with a high number of dimensions, since it presents several cycles of exploration and exploitation, which increases the chances of obtaining better solutions and not getting stuck in local minima.

This work proposes a modification to the galactic swarm optimization, which consists of dynamically adjusting the parameters using fuzzy logic. In this case, the adjustment of the parameters was tested with two different fuzzy systems, which were validated with a set of 20 mathematical functions. Another case was also considered with a fuzzy controller and was also tested to observe how our proposed method behaves for parameter optimization of the membership functions of the fuzzy controller of the water tank. We statistically compared the proposed approach with the modified grey wolf optimizer where fuzzy logic was also used to adjust the parameters, and this comparison was made to measure the performance of our proposal versus other metaheuristics that exists in the literature.

We can conclude that the adjustment of the parameters using fuzzy logic in galactic swarm optimization applied to mathematical functions is a good option since competitive results were obtained. It was observed that in some mathematical functions where the original algorithm does not perform well, the proposed approach obtains significant improvements. It is also observed that as the number of dimensions increases the galactic swarm optimization continues to present competitive results, in the case of the study of the fuzzy controller of the water tank, it has been possible to gradually improve the results using our proposed method to optimize the fuzzy controller. All supported by the results shown in the tables in "Experiments and Comparison of Results".

Galactic swarm optimization is a metaheuristic of recent creation, therefore, it is proposed as future work of the following tasks to study more thoroughly its operation and thus to obtain a better performance of galactic swarm optimization.

- Consider type-2 fuzzy systems for the adaptation of the parameters in GSO.
- Optimize the rules of the fuzzy systems.
- Apply GSO with fuzzy logic for control problems (elevators, washing machines, etc.).
- Apply GSO with fuzzy logic for image processing.
- Apply GSO with fuzzy logic for medical diagnosis.
- Apply GSO with fuzzy logic for systems automation.
- Adapt the GSO algorithm for use with other metaheuristics.

References

1. Muthiah-Nakarajan V, Noel MM. Galactic Swarm Optimization: a new global optimization metaheuristic inspired by galactic motion. Appl. Soft Comput. 2016;38:771–87.
2. Bernal E, Castillo O, Soria J. A fuzzy logic approach for dynamic adaptation of parameters in galactic swarm optimization. IEEE Symp Ser Comput Intell (SSCI). 2016;2016:1–7.
3. Rodriguez L, Castillo O, Soria J. Grey wolf optimizer with dynamic adaptation of parameters using fuzzy logic. IEEE Congress Evol. Comput. (CEC). 2016;2016:3116–23.
4. Soto C, Valdez F, Castillo O. A Review of Dynamic Parameter Adaptation Methods for the Firefly Algorithm, Studies in Computational Intelligence, vol. 667. Cham: Springer; 2017. p. 285–95.
5. Kuo HC, Lin CH. UNAM, Centro de Ciencias Aplicadas y Desarrollo Tecnológico. J. Appl. Res. Technol. 2013;11(4):408–36.
6. Lagunes, ML, Castillo O, Soria J. Methodology for the optimization of a fuzzy controller using a bio-inspired algorithm BT—fuzzy logic in intelligent system design, pp. 131–137 (2018)
7. Engelbrecht AP. Computational Intelligence. Pretoria: Wiley; 2007.
8. Jang JSR, Sun CT, Jang ME. Neuro-fuzzy and soft computing; a computational approach to learning and machine intelligence. Upper Saddle River: Prentice Hall; 1997.
9. Mitchell M. An Introduction to Genetic Algorithms. Cambridge: MIT Press; 1998.
10. Vatin N, Murgul V. Using the Big Bang-Big Crunch Algorithm for Rational Design of an Energy-Plus Building-NC-ND license. (http://creativecommons.org/licenses/by-nc-nd/4.0/). Peer-review under responsibility of the ScienceDirect Using the Big Bang-Big Crunch Algorithm for Rational Design of an Energy-Plus Building," vol. 117, pp. 916–923 (2015)
11. Sedighizadeh M, Bakhtiary R. Optimal multi-objective reconfiguration and capacitor placement of distribution systems with the Hybrid Big Bang-Big Crunch algorithm in the fuzzy framework. Ain Shams Eng. J. 2016;7(1):113–29.
12. Mahmoodabadi MJ, Jahanshahi H. Multi-objective optimized fuzzy-PID controllers for fourth order nonlinear systems. Eng. Sci. Technol. Int. J. 2016;19(2):1084–98.
13. Bernal E, Castillo O, Soria J. Fuzzy logic for dynamic adaptation in the imperialist competitive algorithm. In: Annual Conference of the North American Fuzzy Information Processing Society—NAFIPS, no. 2, pp. 0–5 (2017)
14. Khehra BS, Pharwaha APS, Kaushal M. Fuzzy 2-partition entropy threshold selection based on Big Bang-Big Crunch Optimization algorithm. Egypt. Inform. J. 2015;16(1):133–50.
15. Bernal E, Castillo O, Soria J, Valdez F, Bernal E, Castillo O, Soria J, Valdez F. Imperialist competitive algorithm with dynamic parameter adaptation using fuzzy logic applied to the optimization of mathematical functions. Algorithms. 2017;10(1):18.
16. Bernal E, Castillo O, Soria J. Imperialist Competitive Algorithm with Dynamic Parameter Adaptation Applied to the Optimization of Mathematical Functions, Nature Inspired Design of Hybrid Intelligent Systems. Cham: Springer; 2017. p. 329–41.
17. Peraza C, Valdez F, Garcia M, Melin P, Castillo O, Peraza C, Valdez F, Garcia M, Melin P, Castillo O. A new fuzzy harmony search algorithm using fuzzy logic for dynamic parameter adaptation. Algorithms. 2016;9(4):69.
18. Ochoa P, Castillo O, Soria J. Differential Evolution with Dynamic Adaptation of Parameters for the Optimization of Fuzzy Controllers, Recent Advances on Hybrid Approaches for Designing Intelligent Systems. Cham: Springer; 2014. p. 275–88.
19. Haupt RL, Haupt SE. Practical Genetic Algorithms. Second Edition with CD-ROM. 2nd ed. New York: Wiley; 2004.
20. Mirjalili S, Mirjalili SM, Lewis A. Grey Wolf Optimizer. Adv. Eng. Softw. 2014;69:46–61.
21. Rashedi E, Nezamabadi-pour H, Saryazdi S. GSA: a gravitational search algorithm. Inf. Sci. (NY). 2009;179(13):2232–48.
22. Erol OK, Eksin I. A new optimization method: Big Bang-Big Crunch. Adv. Eng. Softw. 2006;37(2):106–11.
23. Dorigo M, Blum C. Ant colony optimization theory: a survey. Theor. Comput. Sci. 2005;344(2–3):243–78.
24. Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of ICNN'95—International Conference on Neural Networks, vol. 4, pp. 1942–1948
25. Zadeh LA. Fuzzy logic = computing with words. IEEE Trans. Fuzzy Syst. 1996;4(2):103–11.
26. Chen G, Tat Pham T, Pham TT. Introduction to Fuzzy Sets, Fuzzy Logic, and Fuzzy Control Systems. 1st ed. Boca Raton: CRC Press; 2000.
27. Anantathanavit, M., Munlin, M.-A.: Radius particle swarm optimization. In: 2013 International Computer Science and Engineering Conference (ICSEC), pp. 126–130 (2013)
28. Dai J, Han H, Hu Q, Liu M. Discrete particle swarm optimization approach for cost sensitive attribute reduction. Knowl. Based Syst. 2016;102:116–26.
29. Melin P, Olivas F, Castillo O, Valdez F, Soria J, Valdez M. Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic. Expert Syst. Appl. 2013;40(8):3196–206.
30. Olivas F, Valdez F, Castillo O. A fuzzy system for dynamic parameter adaptation in gravitational search algorithm. In: 2016 IEEE 8th International Conference on Intelligent Systems (IS), pp. 146–151 (2016)
31. Valdez F, Melin P, Castillo O. An improved evolutionary method with fuzzy logic for combining Particle Swarm Optimization and Genetic Algorithms. Appl. Soft Comput. 2011;11(2):2625–32.
32. Zhang J, Tang Q, Chen Y, Lin S. A hybrid particle swarm optimization with small population size to solve the optimal

short-term hydro-thermal unit commitment problem. Energy. 2016;109:765–80.
33. Luo Y, Che X. Chaos immune particle swarm optimization algorithm with hybrid discrete variables and its application to mechanical optimization. In: 2009 Third International Symposium on Intelligent Information Technology Application Workshops, pp. 190–193 (2009)
34. Peraza C, Valdez F, Melin P, Peraza C, Valdez F, Melin P. Optimization of intelligent controllers using a type-1 and interval type-2 fuzzy harmony search algorithm. Algorithms. 2017;10(3):82.
35. Caraveo C, Valdez F, Castillo O. Optimization of fuzzy controller design using a new bee colony algorithm with fuzzy dynamic parameter adaptation. Appl. Soft Comput. 2016;43:131–42.
36. Amador-Angulo L, Mendoza O, Castro J, Rodríguez-Díaz A, Melin P, Castillo O, Amador-Angulo L, Mendoza O, Castro JR, Rodríguez-Díaz A, Melin P, Castillo O. Fuzzy sets in dynamic adaptation of parameters of a bee colony optimization for controlling the trajectory of an autonomous mobile robot. Sensors. 2016;16(9):1458.