International Journal of Computational Intelligence Systems
Vol. **13(1)**, 2020, *pp.* 893–903
DOI: https://doi.org/10.2991/ijcis.d.200615.001; ISSN: 1875-6891; eISSN: 1875-6883
https://www.atlantis-press.com/journals/ijcis/

Research Article

# An Improved Parameter Control Based on a Fuzzy System for Gravitational Search Algorithm

Yu Xianrui[2], Yu Xiaobing[1,2,*], Li Chenliang[2], Chen Hong[2]

[1]*Key Laboratory of Meteorological Disaster (KLME), Ministry of Education and Collaborative Innovation Center on Forecast and Evaluation of Meteorological Disasters (CIC-FEMD), Nanjing University of Information Science and Technology, Nanjing 210044, China*
[2]*School of Management Science and Engineering Nanjing University of Information Science and Technology, Nanjing 210044, China*

**ARTICLE INFO**

*Article History*
Received 07 Oct 2019
Accepted 12 May 2020

*Keywords*
Gravitational search algorithm
Fuzzy system
Fuzzy rules
Optimization

**ABSTRACT**

Recently, a kind of heuristic optimization algorithm named gravitational search algorithm (GSA) has been rapidly developed. In GSA, there are two main parameters that control the search process, namely, the number of applied agents (*Kbest*) and the gravity constant (*G*). To balance exploration and exploitation, a fuzzy system containing twelve fuzzy rules is proposed to intelligently control the parameter setting of the GSA. The proposed method can enhance the convergence ability and yield better optimization results. The performance of fuzzy GSA (FGSA) is examined by fifteen benchmark functions. Extensive experimental results are tested and compared with those of the original GSA, CGSA, CLPSO, NFGSA, PSGSA and EKRGSA.



## 1. INTRODUCTION

As optimization problems have become increasingly complicated, the traditional methods cannot effectively solve them in a high-dimensional search space. Hence, heuristic search algorithms have come into being. Heuristic optimization methods are developed based on natural or physical processes. Algorithms inspired by the behavior of natural phenomena have received widespread attention over the past few decades [1–4].

The superior performance of gravitational search algorithm (GSA) has attracted increasing attention. In recent years, many modified versions of GSA have been proposed. The most famous are the real GSA [5], introduced for real-valued variables; the binary GSA [6], which has variables with a value of either 0 or 1; the discrete GSA [7], possessing variables with discrete values; and the mixed GSA [8], containing both continuous and binary variables.

To control exploitation and exploration efficiently, some new GSA operators have been designed, including disruption [9–11], chaotic [12,13], mutation [14], crossover [15], and so on.

In the original GSA, the parameters mainly consist of the following aspects: mass of agent ($M$), number of agents ($N$), gravitational constant ($G$), distance between agents in the search space ($R$), power between distances ($P$) and number of applied agents (*Kbest*). The number of agents is generally set at the beginning of the algorithm and fixed during operation [16]. $M$ has three attributes [5]: active gravitational mass, which determines the intensity of the gravitational field produced by an agent; passive gravitational mass, which determines the strength of an agent's interaction with the gravitational field and inertial mass, which represents the intensity of resistance to motion change when a force acts on an agent. The value of $N$ is set according to the requirements of the specific algorithm used. The distance between agents is calculated as a function of the Euclidian distance, and the power between distances is set to one in the GSA and all its variants.

The values of the masses in the original GSA, including the passive mass, active mass and inertia mass, are set to the same value. Javidi suggested that mass calculation should be improved by defining an opportune function named sigma scaling and the Boltzmann functions [17]. These functions try to balance exploration and exploitation to prevent the algorithm from falling into local optimum. Khajooei and Rashedi proposed a function related to the concept of antigravity, which utilizes both positive and negative masses. The algorithm aims to enhance the ability to explore the search space [18].

In most GSA versions, both $G$ and *Kbest* follow the same rules and decrease as the number of iterations increases [19]. In the original GSA, *Kbest* is obtained by the decreasing function in Eq. (1) [5]:

$$Kbest = 0.01 \times N \times \left( final\_per + \left(1 - \frac{t}{T}\right) \times \left(100 - final\_per\right) \right), \tag{1}$$

where $N$ is the number of agents, *final_per* is a control parameter allowing only one agent to apply a force to others, which denotes the percentage of particles that exert gravitational force on all particles in the end, $t$ is the number of iterations and $T$ is the maximum

* *Corresponding author. Email: yuxb111@163.com*

number of iterations. S. He et al. proposed EKRGSA and modified the function of *Kbest*, which decreases exponentially with the number of iterations from $N$ to 1 in Eq. (2) [20]:

$$Kbest = N \times \left( \frac{final\_per}{100} \right)^{\frac{t}{T}}, \tag{2}$$

where $N$ is the number of agents, *final_per* is the percentage of particles that exert gravitational force on all particles in the end, $t$ is the number of iterations and $T$ is the maximum number of iterations.

$G$ is controlled by an exponential function in Eq. (3) [5]:

$$G(t) = G_0 \times e^{-\alpha \frac{t}{T}}, \tag{3}$$

where $G_0$ denotes the initial value, $\alpha$ denotes the gravitational coefficient and $T$ is the maximum number of iterations.

The NFGSA [21] proposed a neural network combined with a fuzzy system to adjust the parameter $\alpha$ and redefined the definition of the parameter *Kbest*. The PSGSA [21] redefined the calculation of the parameter $\alpha$ combined with the plane output surface based on NFGSA.

In the binary GSA, $G$ is considered as a linear decreasing function in Eq. (4). Rashedi et al. examined the effects of different values of $G$ and *Kbest* [6]. The results of their experiments have shown that the optimal value of $G$ is different for different functions and that $G$ is a parameter determined by the problem. Furthermore, the value of *Kbest* was experimentally proven to change with time.

$$G(t) = G_0 \times \left(1 - \frac{t}{T}\right), \tag{4}$$

where $G_0$ denotes the initial value, $T$ denotes the maximum number of iterations and $t$ is the iteration counter.

$$ED = \frac{R_{ave} - R_{min}}{R_{max} - R_{min}} \tag{5}$$

$$CM = \frac{f_{ave}(t) - f_{ave}(t-1)}{f_{ave}(t)} \tag{6}$$

Fatemeh and Esmat use a fuzzy rule for controlling GSA setting [22]. In their opinion, the distance between agents ($R$) can be rewritten into a new parameter *ED*, calculated according to Eq. (5), which detects the ability of algorithms in terms of diversity. In Eq. (5), $R_{ave}$, $R_{max}$ and $R_{min}$ are the average, maximum and minimum Euclidian distances between agents, respectively. Then, a new parameter *CM* (see Eq. (6)) is introduced to measure the progress of the algorithm, where $f_{ave}(t)$ is the average value of the agent finesses at iteration $t$. The combination of *ED* and *CM* can be expressed by a fuzzy logic controller to prevent the algorithm from falling into local optimum and converging too early. During the operation of the algorithm, $G$ is increased by these fuzzy rules when the optimization progress of the algorithm is stagnant. At the later stage of the algorithm, $G$ is decreased so that the exploitation capabilities of the algorithm can be improved. This is consistent with the rules for strengthening the exploration ability in the early stage and increasing the exploitation capability in the later stage.

In another study [23], the authors deemed that the value of *Kbest* should be increased to escape trapping local optima when the best fitness is not improved after multiple iterations. For the parameter $G$, the authors suggested that a simple exponential decreasing function for all iterations is unreasonable when solving complicated problems. Therefore, eight fuzzy rules are proposed to update $G$ and *Kbest*.

According to the two parameter control methods based on fuzzy theory mentioned above, the former uses two functions to control two input parameters, which represent the population diversity and population progress respectively. However, although these fuzzy rules of the former are used to control the parameter $G$, they cannot be used to control the parameter *Kbest*, which is also significant in determining the motion of agents. Conversely, the latter uses fuzzy rules to control $G$ and *Kbest*, but they cannot use appropriate functions such as Eqs. (5–6) to control the input parameters. The goal of a fuzzy system is to produce output parameters through the values of the corresponding input parameters, which should be calculated in reasonable ways. These descriptive languages without specific control functions for calculating the input parameters are obviously insufficient. Based on this observation, a fuzzy system including new control functions is introduced to control $G$ and *Kbest*. The aim is to balance exploration and exploitation, which can improve the performance of GSA.

1. It is important to control exploitation and exploration in GSA. A new fuzzy system is applied to intelligently control the parameters of GSA, which includes twelve fuzzy rules. They are adopted to increase the convergence rate and prevent the algorithm from falling into local optimum.
2. New control functions are designed to calculate the value of the input variables so that the agent diversity and optimization progress of GSA can be well monitored.
3. Compared with the original GSA, CGSA, CLPSO, NFGSA, PSGSA and EKRGSA, the proposed method has achieved better results for fifteen well-known standard functions. Additional experiments have indicated that these fuzzy rules are effective.

This paper consists of five sections. Introduction is firstly suggested as Section 1. After that, GSA is reviewed in Section 2. In Section 3 fuzzy GSA is described in detail. Experimental results are presented in Section 4. At last, conclusions are included in Section 5.

## 2. GRAVITATIONAL SEARCH ALGORITHM

In GSA, the objective function of an optimization problem is based on many variables. Each variable has an upper and a lower bound, as shown in Eq. (7), represented by $xl^d$ and $xu^d$, respectively. A search field is set up using these boundaries.

$$xl^d \le x^d \le xu^d, d = 1, 2, \dots, m \tag{7}$$

Let $X_i (i = 1, 2, \dots, k)$ be the agents, and the position of the $i$th agent can be presented in Eq. (8):

$$X_i = \left(x_i^1, \dots, x_i^m\right), \ i = 1, 2, \dots, k, \tag{8}$$

where each value in $X_i$ represents the position of the $i$th agent in the search space. At the iteration $t$, the total forces applied

to agent $i$ from a set of heavier agents should be defined as Eq. (9):

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i} rand_j F_{ij}^d = \sum_{j \in Kbest, j \neq i} rand_j G(t) \frac{M_{aj}(t) M_{pi}(t)}{R_{ij}(t) + \varepsilon} (x_j^d(t) - x_i^d(t)), \tag{9}$$

where $M_{aj}(t)$ is the active mass related to agent $j$, $M_{pi}(t)$ is the passive mass related to agent $i$, $G(t)$ is the gravitational constant at iteration $t$, $\varepsilon$ is a small value, $R_{ij}(t)$ is the Euclidian distance between the two agents $i$ and $j$ and $rand_j$ is a random variable in the interval [0, 1]. *Kbest* is a function of time, which is initialized to $k_0$ at the beginning and decreases over time.

Then, the law of motion is used to calculate the acceleration of the agents, as shown in Eq. (10):

$$a_i^d(t) = \frac{F_i^d(t)}{M_{Ii}(t)}, \tag{10}$$

where $a_i^d(t)$ is the acceleration of the $i$th agent in the $d$th dimension at iteration $t$. $M_{Ii}(t)$ is the inertia mass related to the $i$th agent at iteration $t$. Afterward, the next velocity and its next position are calculated using Eqs. (11–12):

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{11}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), \tag{12}$$

where $rand_i$ is a uniform random variable in the interval $[0, 1]$, $v_i^d(t)$ is the velocity of the $i$th agent in the $d$th dimension at iteration $t$ and $x_i^d(t)$ is the position of the $i$th agent in the $d$th dimension at iteration $t$.

The gravitational constant $G$ is a function of time $t$ (see Eq. (13)), which takes an initial value $G_0$. It is reduced with the iterative time $t$ to control the search accuracy.

$$G(t) = G(G_0, t) \tag{13}$$

The masses of the agents are calculated by the fitness evaluation. Supposing that the gravitational mass is numerically equal to the inertia mass, the mass $M_i(t)$ is computed by Eqs. (14–17):

$$M_i = M_{Ii}, i = 1, 2, \ldots, m$$

$$m_i(t) = \frac{fit_i(t) - f_{max}(t)}{f_{min}(t) - f_{max}(t)} \tag{14}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{k} m_j(t)} \tag{15}$$

$$f_{min}(t) = \min fit_j(t) \, j \in \{1, \ldots, m\} \tag{16}$$

$$f_{max}(t) = \max fit_j(t) \, j \in \{1, \ldots, m\}, \tag{17}$$

where $M_{Ii}$ denotes the inertia mass of the $i$th agent, $fit_i(t)$ is the fitness value of the $i$th agent at iteration $t$ and $f_{min}(t)$ and $f_{max}(t)$ represent the minimum and maximum fitness values at iteration $t$, respectively (for a minimization problem). The better the function value is, the larger the value of the mass will be. A heavier mass represents a more efficient agent, which has a greater attractive force and runs more slowly.

To control important parameters, the original GSA uses a mathematical model, which includes linearity, an exponential character or both. As the iteration proceeds, the exploration becomes weaker and the exploitation becomes stronger. However, when faced with complex engineering problems, such as big data analysis, the problem becomes nonlinear and complex. It is not enough to control the algorithm process through a mathematical model [23]. An effective solution is to establish a balance between exploration and exploitation through parameter control.

## 3. FUZZY GSA

Currently, many scholars combine heuristics methods with fuzzy logic to improve algorithms [22,24–26]. They integrate algorithm processes with the linguistic description to create a fuzzy system to intelligently control parameters. In GSA, the larger the agent is, the slower the convergence rate is and the better the diversity is; the smaller the agent is, the easier it is for the agent to fall into the local optimum.

In the original GSA, there are two important parameters, i.e., *Kbest* and $G$, which are used to control the search process. Among them, $G$ can be dominated by the gravitational coefficient $\alpha$.

### 3.1. *Kbest* and *G*

*Kbest* denotes the number of agents that effectively apply force on other agents. The larger *Kbest* is, the more attractive it is between agents. By changing *Kbest*, the performance of GSA can be guaranteed by controlling the balance between exploration and exploitation. In the original GSA, all agents initially apply a force to each other. As the iteration evolves, *Kbest* is gradually reduced by a linear function. Finally, only one maximum fitness value exists, and only the heaviest agent applies a force to the other agents, which is shown in Eq. (9). The main idea of the *Kbest* formula is that when the iteration time reaches the later stage, only agents with a large weight can exert force on other agents. When *Kbest* increases, there is more attraction, more movement and a lower convergence rate. On the other hand, when *Kbest* decreases, the attraction and movement decrease, which may result in premature convergence and easily falling into the local optimum.

The parameter $G$ is directly related to the acceleration of the agents. Its size has a great influence on the value of the acceleration, and the acceleration affects the position of agents after the gravitational force is applied. In the original GSA, $G$ is gradually reduced by a monotonic exponential function. $G$ is determined by the gravitational coefficient $\alpha$. When $\alpha$ is larger, then $G$ is smaller, the acceleration of the agent is smaller, the positional change is less, the exploration ability is weaker, and it is easy to fall into the local optimum. On the other hand, when $\alpha$ is smaller, then $G$ is larger, the acceleration of the agent is greater, the positional change is more obvious, the exploration capability is stronger, and the convergence rate is slower.

## 3.2. Agent Diversity and Optimization Progress

Agent diversity and optimization progress are the most effective tools for monitoring the performance of the algorithm [22]. The parameter *SD* is introduced to measure the agent diversity, which is calculated by Eq. (18):

$$SD = \frac{f_{ave}(t) - f_{min}(t)}{f_{max}(t) - f_{min}(t)}, \tag{18}$$

where $f_{ave}(t)$ represents the average fitness value obtained from the $t$th iteration, $f_{min}(t)$ and $f_{max}(t)$ represent the minimum and maximum fitness values in the $t$th iteration, respectively. When *SD* is small, the fitness values are similar, indicating the following situations:

1. The positions of the agents in the search space are not far apart. In this case, the objective function has fallen into the local optimum at similar locations in the search space.
2. The positions of the agents in the search space are far apart. In this case, the objective function has at least 2 local optima at different locations in the search space.

Both conditions indicate that the objective function is trapped in the local optimum regardless of whether the locations of the agents are similar or different, which implies that the diversity of the agents is poor, and vice versa.

In addition to *SD*, the parameter *ZM* is considered for measuring the optimization progress. *ZM* is calculated as Eq. (19) in minimization problems:

$$ZM = \frac{f_{best}(t-1) - f_{best}(t)}{f_{best\_ave}(t) - f_{best}(t)}, \tag{19}$$

where $t$ is the number of iterations, $f_{best}(t)$ is the optimal fitness value in the $t$th iteration, and $f_{best\_ave}(t)$ is the average optimal fitness value of the previous $t$ iterations. When *ZM* is small, the current iterative optimal value is not very different from that of the previous iteration, which indicates that the optimization progress is not good, and vice versa.

The smaller *SD* is, the worse the diversity of the agents is. The larger *SD* is, the better the diversity of the agents is. The smaller *ZM* is, the worse the optimization process is. The larger *ZM* is, the better the algorithm optimization process is.

## 3.3. Membership Functions

A fuzzy inference system is composed of three input variables and two output variables. *SD*, *ZM* and $t$ are the input variables. $t$ is the current iteration. $\alpha$ and *Kbest* are the output variables. Fuzzy theory uses linguistic descriptions instead of mathematical language. In this paper, five membership functions are proposed to describe the input and output variables. The role of the membership functions is to label $t$, $\alpha$, and *Kbest* as low, medium and high. *SD* and *ZM* are converted to low and high labels. The membership functions are depicted in Figure 1. In most cases, the membership functions are devised by experts. The range of *SD* is the interval [0, 0.6], the range of *ZM* is the interval [0, 1], the range of $t$ is the interval [0, 1000], the range of $\alpha$ is the interval [28, 30] and the range of *Kbest* is the interval [2, 50].

## 3.4. Fuzzy Rules

Twelve fuzzy rules are extracted from the previous subsections with three input and two output variables, as shown in Table 1. These rules are proposed to intelligently control the search process of GSA, prevent the algorithm from falling into a local optimum and avoid premature convergence. Each of the fuzzy rules is set up to enhance the performance of the algorithm.

The first rule in Table 1 indicates that in the early stage of the iterations, the optimization progress and the diversity are good. At this time, when $\alpha$ increases, $G$ decreases; then, the gravitation between agents decreases, and the motion becomes slower. When *Kbest* decreases, the number of agents that exert a force is reduced, and the movement decreases. Hence, $\alpha$ takes a large value and *Kbest* takes a median value, which can enhance the local search ability and convergence ability of the algorithm.

The second rule indicates that in the early stage of the iterations, the optimization progress is good and the diversity is bad. This may reveal that the algorithm is prematurely trapped; thus, $\alpha$ and *Kbest* should increase. Therefore, $\alpha$ gives a medium value, and *Kbest* gives a high value. By the rule, the acceleration increases, and the movement is faster. As a result, the convergence rate is lower.

The third rule is that at early iterations, the optimization progress and the diversity are poor. This shows that the algorithm is trapped in local optimum and suffers from premature convergence. At this moment, by decreasing $\alpha$ and increasing *Kbest*, the acceleration and velocity are increased. Then, agents can escape from the local optimum.

The fourth rule is that at early iterations, the optimization progress is poor and the diversity is good. This scenario shows that the algorithm converges slowly. It is necessary to increase $\alpha$ and decrease *Kbest*; hence, $\alpha$ takes a high value and *Kbest* takes a medium value to increase the convergence speed.

The seventh rule indicates that at medium iterations, the optimization progress and the diversity are poor. Similar to the third rule, the algorithm falls into local optimum, and premature convergence occurs. $\alpha$ gives a low value, and *Kbest* is taken as a high value in the medium term. Then, the acceleration increases, the position changes more quickly and escape from the local optimum occurs as soon as possible.

Regarding the ninth rule, when the algorithm is at a later iteration, the optimization progress and the diversity are good. In this case, the exploitation capability should be enhanced. Therefore, $\alpha$ gives a high value, and *Kbest* takes a low value, which speeds up the convergence.

The difference between the twelfth rule and the ninth rule is that the optimization progress is low. Because the diversity is good, it is necessary to strengthen the convergence ability, which requires a high value of $\alpha$ and a low value of *Kbest*.

In this paper, inputs are combined logically using the AND operator to obtain a crisp decision front, and the center-of-gravity method is utilized for defuzzification.

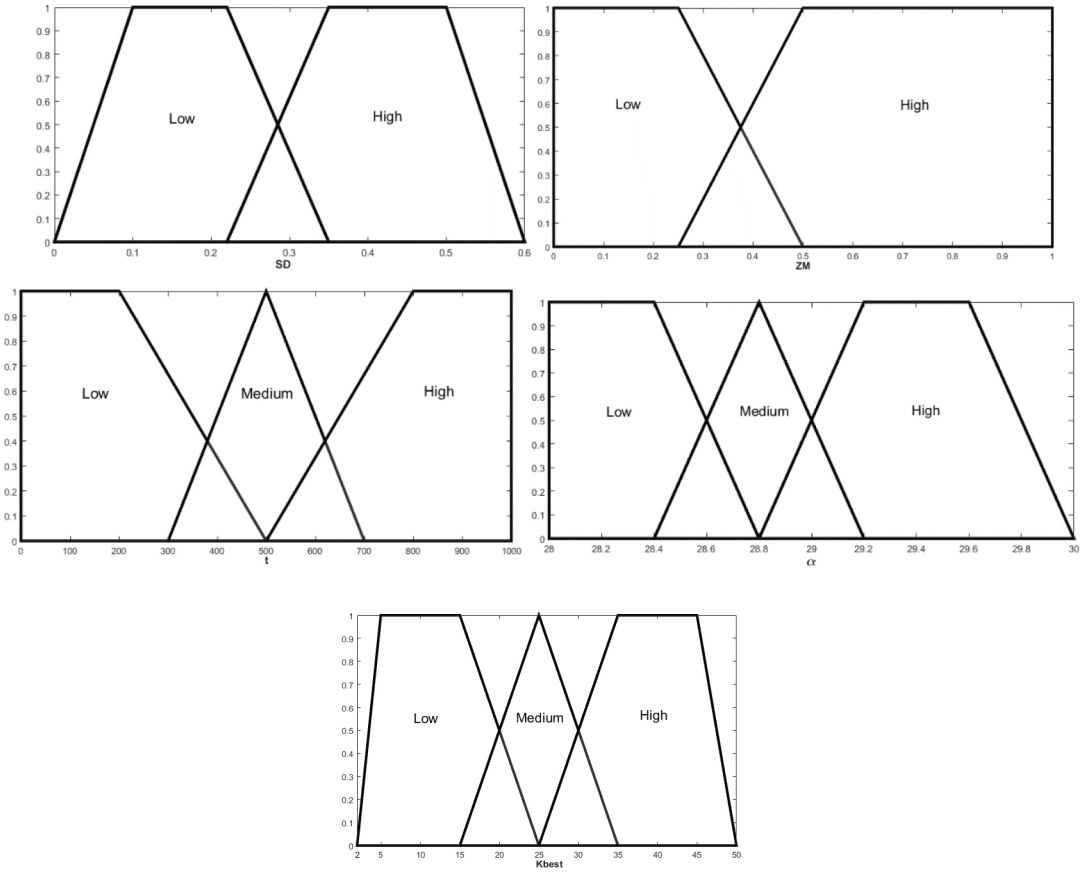


**Figure 1** | Membership functions.

**Table 1** | Fuzzy rules for controlling the parameter of $\alpha$ and *Kbest*.

| Rule | | *SD* | *T* | *ZM* | | $\alpha$ | *Kbest* |
|---|---|---|---|---|---|---|---|
| 1 | | high | low | high | | high | medium |
| 2 | | low | low | high | | medium | high |
| 3 | | low | low | low | | low | medium |
| 4 | | high | low | low | | high | medium |
| 5 | | high | medium | high | | high | medium |
| 6 | If | low | medium | high | Then | medium | high |
| 7 | | low | medium | low | | low | high |
| 8 | | high | medium | low | | medium | medium |
| 9 | | high | high | high | | high | low |
| 10 | | low | high | high | | medium | medium |
| 11 | | low | high | low | | low | medium |
| 12 | | high | high | low | | high | low |

## 3.5. Pseudocode of FGSA

Pseudocode of FGSA
**Input**: *N*: the number of agents;
*T*: maximum number of iterations;
*Rpower*: power of the Euclidian distance between two agents
*F_index*: the index of the test function
*Min_flag*: type of problem (*min_flag* = 1, minimization or *min_flag* = 0, maximization)
(1) *Rnorm* = 2 (norm in the Euclidian distance);
(2) Generate the initial position of the *i*th agent by randomly forming from search space;
(3) **for** *iteration* = 1: *T* (the maximum iteration)
(4) Checks the search boundaries for agents;
(5) Evaluate the objective function values of agents;
(6) **If** *min_flag* = 1
(7) Select the minimum value in the calculation result;
(8) **else**
(9) Select the maximum value in the calculation result;
(10) **end If**
(11) Calculate the values of masses of each agent (*M*);
(12) Update gravitational coefficient ($\alpha$) and the number of effective agents (*Kbest*) by twelve fuzzy rules;
(13) Calculate gravitational constant (*G*);
(14) Calculate acceleration and velocity;
(15) Update agents' position;
(16) **end for**
**Output**: *Fbest*: the best result of the objective function values;
*Lbest*: the best solution (the location of *Fbest* in search space);
*BestChart*: the best function values over iterations.

## 4. EXPERIMENTS AND DISCUSSION

The performance of the proposed algorithm can be tested by fifteen standard benchmark functions [27], including some unimodal and some multimodal criteria, which are presented in Table 2. The proposed algorithm is tested for minimization and compared with the original GSA, CGSA [13] and CLPSO [28]. The CGSA (chaotic GSA) embeds chaotic maps into the gravitational constant (*G*) and proposes an adaptive normalization method to

**Table 2** | Test functions.

| Test Functions | Bounds |
|---|---|
| $f_1(X) = \sum_{i=1}^{n} x_i^2$ | $[-100, 100]^n$ |
| $f_2(X) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$ | $[-10, 10]^n$ |
| $f_3(X) = \sum_{i=1}^{n} \left( \sum_{j=1}^{i} x_j \right)^2$ | $[-100, 100]^n$ |
| $f_4(X) = \max_i \{ \lvert x_i \rvert, 1 \le i \le n \}$ | $[-100, 100]^n$ |
| $f_5(X) = \sum_{i=1}^{n-1} \left[ 100 \left( x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right]$ | $[-30, 30]^n$ |
| $f_6(X) = \sum_{i=1}^{n} \left( [x_i + 0.5] \right)^2$ | $[-100, 100]^n$ |
| $f_7(X) = \sum_{i=1}^{n} i x_i^4 + random[0, 1]$ | $[-1.28, 1.28]^n$ |
| $f_8(X) = \sum_{i=1}^{n} -x_i \sin\left( \sqrt{\lvert x_i \rvert} \right)$ | $[-500, 500]^n$ |
| $f_9(X) = \sum_{i=1}^{n} \left[ x_i^2 - 10\cos(2\pi x_i) + 10 \right]$ | $[-5.12, 5.12]^n$ |
| $f_{10}(X) = -20\exp\left( -0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}} \right) - \exp\left( \frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e$ | $[-32, 32]^n$ |
| $f_{11}(X) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left( \frac{x_i}{\sqrt{i}} \right) + 1$ | $[-600, 600]^n$ |
| $f_{12}(X) = \frac{\pi}{n} \left\{ 10\sin(\pi y_1) + \sum_{i=1}^{n} (y_i - 1)^2 \left[ 1 + 10\sin^2(\pi y_i + 1) \right] + (y_n - 1)^2 \right\} y_i = 1 + \frac{x_i + 1}{4}$ $+ \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ | $[-50, 50]^n$ |
| $f_{13}(X) = 0.1 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2 \left[ 1 + \sin^2(3\pi x_i + 1) \right] + (x_n - 1)^2 \left[ 1 + \sin^2(2\pi x_n) \right] \right\}$ $+ \sum_{i=1}^{n} u(x_i, 5, 100, 4)$ | $[-50, 50]^n$ |
| $f_{14}(X) = \left( \frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} \left( x_i - a_{ij} \right)^6} \right)^{-1}$ | $[-65.53, 65.53]^2$ |
| $f_{15}(X) = \sum_{i=1}^{11} \left[ a_i - \frac{x_1 \left( b_i^2 + b_i x_2 \right)}{b_i^2 + b_i x_3 + x_4} \right]^2$ | $[-5, 5]^4$ |

make the transition from the exploration stage to the exploitation stage smoother, which are all used to balance exploration and exploitation. The CLPSO (comprehensive learning particle swarm optimizer) introduces a novel learning strategy and combines the particles' historical best information to update the particle velocity in PSO. In all algorithms, the number of agents is 50 ($N$ = 50). The dimensions of the test function are set to 30, the maximum number of iterations is 1000 and $G_0$ is set to 100. From $f_1$ to $f_{13}$, the minimum value is 0, except for $f_8$, which has a minimum value of $-418.9823 \times n$. The optimum locations for the functions of Table 2 can be found in $[0]^n$, except for $f_5$, $f_{12}$, and $f_{13}$ in $[1]^n$ and $f_8$. in $[420.96]^n$. For $f_{14}$ and $f_{15}$, the minimum values are 1 and 0.0003, respectively. The optimum locations of $f_{14}$ and $f_{15}$ are $(-32, 32)$ and $(0.1928, 0.1908, 0.1231, 0.1358)$, respectively.

The results are averaged over 25 runs. In all tables, the mean represents the average of the 25 best fitness values, the median represents the median of the 25 best fitness values, and the variance represents the variance of the 25 best fitness values, among which the best results have been highlighted in bold.

The results are shown in Table 3 and Figure 2. Table 3 illustrates that the FGSA provides a better mean, median and variance than those of the original GSA in $f_1, f_2, f_3, f_4, f_5, f_8, f_{10}, f_{12}, f_{13}$ and $f_{14}$. The results of the mean, median and variance in $f_6$ indicate that the FGSA is the same as the original GSA. At the same time, the results of the original GSA are worse than those of the FGSA in $f_7, f_9, f_{11}$ and $f_{15}$.

For CGSA and CLPSO, the results of FGSA are better in most functions. In particular, the mean, median and variance of $f_1$, $f_2$, $f_4$, $f_{10}, f_{12}$, and $f_{13}$ are significantly better than those of the CGSA and CLPSO. In $f_3, f_9, f_{11}, f_{14}$ and $f_{15}$, FGSA is less effective than CGSA and CLPSO; FGSA is worse than CGSA and better than CLPSO in $f_5$. In the result of $f_6$, FGSA is the same as CGSA and better than CLPSO. For $f_7$ and $f_8$, FGSA is better than CGSA and worse than CLPSO. Further analysis of the results by a statistical method is shown in later paragraphs.

The progress of GSA, CGSA and CLPSO for $f_1, f_2, f_4, f_{10}, f_{12}$ and $f_{13}$ are plotted in Figure 2. It can be observed that the convergence

**Table 3** The best results of 15 benchmark function with GSA, CGSA and CLPSO.

| Function | | FGSA | GSA | CGSA | CLPSO |
|---|---|---|---|---|---|
| $f_1$ | Mean | **6.74E−24** | 2.069E−17 | 1.620E−05 | 5.502E−02 |
| | Median | **6.32E−24** | 1.978E−17 | 1.430E−5 | 4.516E−01 |
| | Variance | **4.2E−48** | 2.847E−35 | 2.088E−11 | 7.247E−03 |
| $f_2$ | Mean | **3.55E−12** | 2.272E−08 | 2.014E−02 | 4.612E−02 |
| | Median | **3.56E−12** | 2.244E−08 | 1.996E−02 | 4.382E−02 |
| | Variance | **2.66E−25** | 1.687E−17 | 7.659E−06 | 8.210E−05 |
| $f_3$ | Mean | 1.05E+02 | 2.546E+02 | **1.534E−03** | 8.585E+01 |
| | Median | 9.007E+01 | 2.286E+02 | **1.298E−02** | 8.726E+01 |
| | Variance | 2.271E+03 | 6.670E+03 | **6.764E−07** | 1.639E+02 |
| $f_4$ | Mean | **1.26E−12** | 3.188E−09 | 4.267E−03 | 2.509E+01 |
| | Median | **1.19E−12** | 3.213E−09 | 4.429E−03 | 2.519E+01 |
| | Variance | **4.49E−26** | 3.655E−19 | 4.558E−07 | 3.369E+00 |
| $f_5$ | Mean | 2.66E+01 | 2.857E+01 | **2.508E+01** | 4.021E+02 |
| | Median | 2.65E+01 | 2.610E+01 | **2.499E+01** | 3.997E+02 |
| | Variance | 1.028E−01 | 1.576E+02 | **9.226E−02** | 7.377E+03 |
| $f_6$ | Mean | **0** | **0** | **0** | 4.721E−02 |
| | Median | **0** | **0** | **0** | 4.497E−02 |
| | Variance | **0** | **0** | **0** | 1.380E−04 |
| $f_7$ | Mean | 4.57E−02 | **1.798E-02** | 4.919E−02 | 3.123E−02 |
| | Median | 4.412E−02 | **1.808E-02** | 4.762E−02 | 3.106E−02 |
| | Variance | 1.82E−03 | **3.836E−05** | 7.182E−05 | 5.638E−05 |
| $f_8$ | Mean | −2.928E+03 | −2.840E+03 | −2.789E+03 | **−1.254E+04** |
| | Median | −2.877E+03 | −2.865E+03 | −2.722E+03 | **−1.255E+04** |
| | Variance | 2.130E+05 | 3.766E+05 | 9.139E+04 | **1.130E+03** |
| $f_9$ | Mean | 2.69E+01 | 1.496E+01 | 2.080E+01 | **1.001E+01** |
| | Median | 2.587E+03 | 1.492E+01 | 2.139E+01 | **1.046E+01** |
| | Variance | 2.799E+03 | 1.068E+01 | 1.054E+01 | **2.805E+00** |
| $f_{10}$ | Mean | **2.08E−12** | 3.732E−9 | 3.038E−03 | 1.581E−01 |
| | Median | **2.15E−12** | 3.671E−09 | 3.059E−03 | 1.485E−01 |
| | Variance | **1.39E−25** | 1.986E−19 | 3.014E−07 | 1.049E−03 |
| $f_{11}$ | Mean | 2.06E+01 | 3.777 | **0.022** | 0.152 |
| | Median | 2.017E+01 | 3.433 | **0.013** | 0.145 |
| | Variance | 2.29E+01 | 2.071 | **0.001** | 0.002 |
| $f_{12}$ | Mean | **4.32E−26** | 1.431E−19 | 1.126E−07 | 1.818E+01 |
| | Median | **4.32E−26** | 1.484E−19 | 1.103E−07 | 1.828E+01 |
| | Variance | **1.39E−52** | 1.804E−39 | 1.141E−15 | 9.706E−01 |
| $f_{13}$ | Mean | **6.53E−25** | 1.900E−18 | 1.940E−06 | 5.245E−02 |
| | Median | **5.8E−25** | 1.747E−18 | 1.794E−06 | 4.956E−02 |
| | Variance | **7.05E−50** | 2.006E−37 | 6.331E−13 | 5.622E−04 |
| $f_{14}$ | Mean | 3.506 | 3.656 | 2.530 | **0.998** |
| | Median | 2.209 | 2.982 | 2.317 | **0.998** |
| | Variance | 6.916 | 6.926 | 1.020 | **0** |
| $f_{15}$ | Mean | 2.11E−03 | 2.007E−03 | 1.889E−03 | **5.807E−04** |
| | Median | 2.102E−03 | 1.991E−03 | 1.219E−03 | **5.857E−04** |
| | Variance | 2.44E−07 | 4.99263E−07 | 2.59053E−06 | **5.45262E−09** |

GSA, gravitational search algorithm; FGSA, fuzzy GSA.

rate of FGSA is faster than that of the other algorithms and that the mean of the iterative full period is the smallest. This fully demonstrates that FGSA tends to look for the global optimum faster than the other algorithms. Additionally, FGSA shows better convergence ability than that of the others. Therefore, FGSA has excellent performance and a high convergence rate.

The non-parametric statistical method Wilcoxon test was used to make a compete comparison. Table 4 lists the results of the Wilcoxon test on every test function between FGSA and the other algorithms. Rows "+," "=" and "−" denote the number of test functions of the other algorithms that FGSA performs better than, almost the same as, and worse than, respectively. Row "Score" indicates the difference between the number of "1"s and "−1"s. It is used to make an overall comparison between two algorithms.

For example, FGSA outperforms GSA on 8 functions ($f_1, f_2, f_3, f_4, f_5, f_{10}, f_{12}, f_{13}$), performs the same as GSA on 4 functions ($f_6, f_8, f_{14}, f_{15}$) and performs worse on 3 functions ($f_7, f_9, f_{11}$). Therefore, the score is $8 - 3 = 5$, which indicates that FGSA is better than GSA.

## 4.1. Comparison with the Latest GSA Variants

To further validate the performance of the proposed algorithm, the three latest GSA variants, i.e., NFGSA [21], PSGSA [21] and EKRGSA [20], are selected for comparison.

In all algorithms, the test functions are the same as in Table 2. The number of agents is 50 ($N = 50$). The dimension of the test function

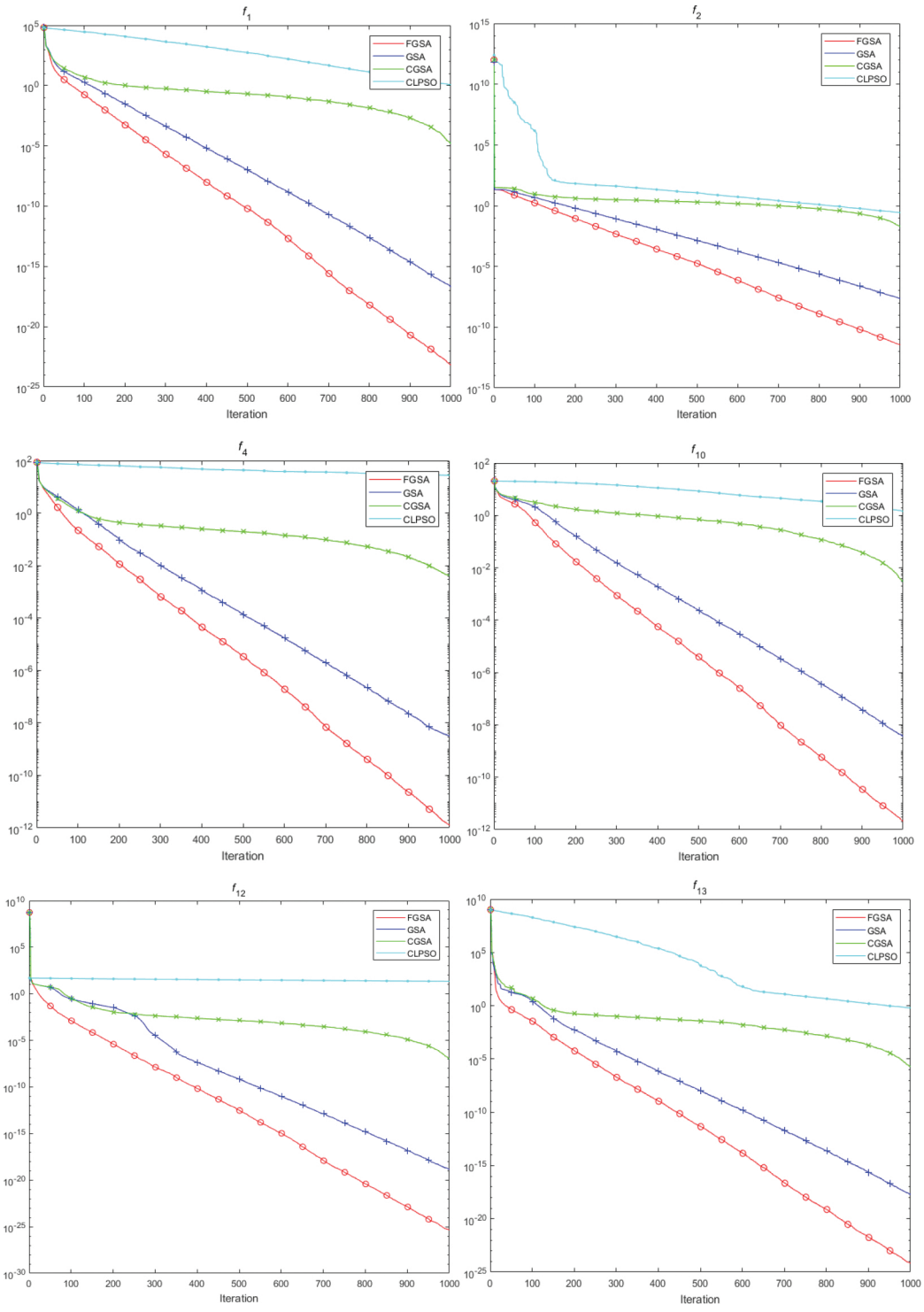


**Figure 2** | Comparison of performance of $f_1, f_2, f_4, f_{10}, f_{12}$ and $f_{13}$.

is set to 30, the maximum number of iterations is 1000 and $G_0$ is set to 100 in FGSA, NFGSA and PSGSA and to 1000 in the EKRGSA. The results are averaged over 25 runs. Note that the results of NFGSA, PSGSA and EKRGSA are taken from their respective reference papers. Because there are no experimental results on the function $f_6$ for the EKRGSA, the comparison with EKRGSA on $f_6$ is omitted.

Table 5 illustrates that the mean of FGSA is the best for $f_1$. The results of the mean rank second compared to that of NFGSA, PSGSA and EKRGSA on $f_3$, $f_8$, $f_{14}$ and $f_{15}$. For $f_5$, FGSA provides better result than does PSGSA but worse than that of NFGSA and EKRGSA. In general, the results of FGSA are better on most functions.

**Table 4** The Wilcoxon test of the best results for 15 benchmark function.

| Function | FGSA | GSA | CGSA | CLPSO |
|---|---|---|---|---|
| | | Wilcoxon | Wilcoxon | Wilcoxon |
| $f_1$ | | + | + | + |
| $f_2$ | | + | + | + |
| $f_3$ | | + | − | = |
| $f_4$ | | + | + | + |
| $f_5$ | | + | − | = |
| $f_6$ | | = | = | + |
| $f_7$ | | − | = | − |
| $f_8$ | | = | = | − |
| $f_9$ | | − | − | − |
| $f_{10}$ | | + | + | + |
| $f_{11}$ | | − | − | − |
| $f_{12}$ | | + | + | + |
| $f_{13}$ | | + | + | + |
| $f_{14}$ | | = | = | − |
| $f_{15}$ | | = | − | − |
| + | | 8 | 6 | 7 |
| = | | 4 | 4 | 2 |
| − | | 3 | 5 | 6 |
| Score | | 5 | 1 | 1 |

GSA, gravitational search algorithm; FGSA, fuzzy GSA.

**Table 5** The best results of 15 benchmark function with NFGSA, PSGSA and EKRGSA.

| Function | | FGSA | NFGSA | PSGSA | EKRGSA |
|---|---|---|---|---|---|
| $f_1$ | Mean | **6.74E−24** | 7.51E−23 | 1.04E−02 | 3.68E−19 |
| $f_2$ | Mean | **3.55E−12** | 7.69E−09 | 3.77E−02 | 3.74E−09 |
| $f_3$ | Mean | 1.05E+02 | **9.13E +01** | 2.72E+02 | 2.46E+03 |
| $f_4$ | Mean | **1.26E−12** | 3.42E−08 | 1.67E−02 | 2.16E−08 |
| $f_5$ | Mean | 2.66E+01 | 2.50E+01 | 5.53E+01 | **2.457E+01** |
| $f_6$ | Mean | **0** | **0=** | 3.33E−02 | – |
| $f_7$ | Mean | **4.57E−02** | 5.06E−02 | 1.05E−01 | 6E−02 |
| $f_8$ | Mean | −2.928E+03 | −2.68E+03 | **−3.39E+03** | – |
| $f_9$ | Mean | 2.69E+01 | 2.33E+01 | **2.13E+01** | – |
| $f_{10}$ | Mean | **2.08E−12** | 5.16E−12 | 1.48E−02 | 3.61E−10 |
| $f_{11}$ | Mean | 2.06E+01 | **1.50E+00** | 1.01E+01 | – |
| $f_{12}$ | Mean | **4.32E−26** | 2.07E−02 | 1.17E−02 | 1.91E−21 |
| $f_{13}$ | Mean | **6.53E−25** | 1.10E−03 | 1.30E−01 | 3.13E−20 |
| $f_{14}$ | Mean | 3.506 | 4.32 | 5.93 | **1.02** |
| $f_{15}$ | Mean | 2.11E−03 | 2.78E−03 | 4.52E−03 | **9.05E−04** |

## 4.2. Discussion

To set the parameters of GSA more reasonably, a fuzzy system with twelve fuzzy rules is introduced. However, the setting of the three input parameters needs to confirm the rationality. Thus, an experiment is designed to test the performance of three different combinations of input parameters: $SD + t$, $ZM + t$, and $SD + ZM + t$. Three groups are tested independently. The contrast is tested for minimization, and the results are averaged over 25 runs. The experimental results are shown in Table 6.

The average of the 25 best fitness values is chosen as the standard for testing the pros and cons. The combination of *SD*, *ZM* and *t* provides smaller results than those of the others on $f_1$,$f_2$,$f_3$,$f_4$,$f_5$,$f_8$, $f_{10}$,$f_{12}$,$f_{13}$,$f_{14}$ and $f_{15}$. The results of the three methods for the input parameters are the same as for $f_6$. The results of the combination of *SD*, *ZM* and *t* are worse than the others for $f_7$ and $f_{11}$. For $f_9$, the combination of *SD*, *ZM* and *t* is more efficient than the combination of *SD* and *t* and less efficient than the combination of *ZM* and *t*. In general, the combination of *SD*, *ZM* and *t* is better than the others on most functions, which represents its excellent performance and effectiveness.

**Table 6** Mean of three ways of input parameters.

| Function | $SD + t$ | $ZM + t$ | $SD + ZM + t$ |
|---|---|---|---|
| $f_1$ | 2.91E−21 | 1.43E−23 | **6.74E−24** |
| $f_2$ | 6.99E−11 | 1.60E−11 | **3.55E−12** |
| $f_3$ | 1.83E+02 | 1.93E+02 | **1.05E+02** |
| $f_4$ | 2.73E−11 | 1.67E−12 | **1.26E−12** |
| $f_5$ | 2.68E+01 | 2.88E+01 | **2.66E+01** |
| $f_6$ | 0 | 0 | 0 |
| $f_7$ | 3.98E−02 | **1.96E−02** | 4.57E−02 |
| $f_8$ | −2.84E+03 | −2.73E+03 | **−2.93E+03** |
| $f_9$ | 2.95E+01 | **1.91E+01** | 2.69E+01 |
| $f_{10}$ | 3.80E−11 | 2.77E−12 | **2.08E−12** |
| $f_{11}$ | 1.60E+01 | **1.15E+01** | 2.06E+01 |
| $f_{12}$ | 1.69E−23 | 8.48E−26 | **4.32E−26** |
| $f_{13}$ | 2.27E−22 | 1.34E−24 | **6.53E−25** |
| $f_{14}$ | 5.17E+00 | 3.91E+00 | **3.82E+00** |
| $f_{15}$ | 3.27E−03 | 3.27E−03 | **2.11E−03** |

The inference mechanism behind these fuzzy rules in Table 1 is herein explained. The process of controlling parameters by the corresponding fuzzy rules for $f_3$ is selected to describe the inference mechanism. The results are represented in Table 7, which includes the early, median and late iterations separately. For the parameters of Table 7, the inputs are *t* (representing the current iteration), *SD* (measuring the agent diversity) and *ZM* (weighing the optimization progress), and the outputs are *Kbest*, $\alpha$ and the corresponding rule of the *t*th iteration. For example, when *t* is 199, *SD* is 0.47 and *ZM* is 0.746, it is revealed that at early iterations, *SD* and *ZM* are described as high, which is verified by the member functions in Figure 1. The agent diversity measured by *SD* and the optimization progress weighed by *ZM* are good, which corresponds to rule 1 in Table 1. As a result, the value of *Kbest* is 25, which is consistent with taking a median value, and the value of $\alpha$ is 29.49, meeting the requirements of taking a large value. The other iteration stages all have corresponding rules, which are shown in Table 7. Through the above inference mechanism, the algorithm can escape from the local optimum and enhance the rate of convergence.

Due to the addition of a fuzzy system, the computational cost may be slightly worse. Nowadays, the performance of computer increasingly becomes improved. The gap of response time between FGSA and GSA is within seconds. However, we admit that our proposed algorithm has increased some computational cost. This is the limitation of the paper.

## 5. CONCLUSIONS

The idea of intelligently controlling the parameters of GSA is introduced, and a fuzzy system is designed for this purpose. In this article, an improved idea of controlling the important parameters of GSA is proposed, which shows that a fuzzy inference system containing 12 fuzzy rules can be used to control the parameters *Kbest* and *G*.

The new strategy gives GSA a higher convergence rate and the ability to escape the local minimum. A balance between exploration

**Table 7** | Control parameters by corresponding fuzzy rules for $f_3$.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Inputs | $t$ | 199 | 200 | 201 | 499 | 500 | 501 | 799 | 800 | 801 |
| | $SD$ | 0.470 | 0.580 | 0.294 | 0.394 | 0.228 | 0.433 | 0.458 | 0.430 | 0.452 |
| | $ZM$ | 0.746 | 0.903 | 0.276 | 0.656 | 0.367 | 0.603 | 0.405 | 0.177 | 0.377 |
| Outputs | $Kbest$ | 25 | 25 | 25 | 31 | 40 | 25 | 11 | 11 | 11 |
| | $\alpha$ | 29.49 | 29.47 | 28.36 | 28.76 | 28.31 | 29.28 | 29.50 | 29.49 | 29.50 |
| Corresponding rule | | 1 | 1 | 3 | 6 | 7 | 5 | 12 | 12 | 12 |

and exploitation is also realized. The performance of FGSA is examined on fifteen well-known standard functions. The experimental results are compared with those of the original GSA, CGSA, CLPSO, NFGSA, PSGSA and EKRGSA. The results indicate that FGSA achieves superior performances on most functions.

## CONFLICT OF INTEREST

The authors declare no conflict of interest.

## AUTHORS' CONTRIBUTIONS

Xianrui Yu conceived the research, drafted the manuscript and performed the experiments; Xiaobing Yu edited, and revised the manuscript. Chenliang Li and Hong Chen designed the experiments and analyzed the data.

## ACKNOWLEDGMENTS

This research was funded by the National Natural Science Foundation of China (No.71974100, No.71503134), Natural Science Foundation of Jiangsu Province (No. BK20191402), Major Project of Philosophy and Social Science Research in Colleges and Universities of Jiangsu province (2019SJZDA039), Key Project of National Social and Scientific Fund Program (16ZDA047), Qing Lan project (R2019Q05), Postgraduate research and innovation plan project of Jiangsu Province (SJKY19_0983) and National Undergraduate Training Program for Innovation and Entrepreneurship (201910300016Z), the Joint Open Project of KLME & CIC-FEMD, NUIST (KLME202004).

## REFERENCES

[1] D.E. Goldberg, D.M. Goldberg, D.E. Goldberg, D. Goldberg, E.D. Goldberg, E. Goldberg, *et al.* Genetic Algorithm is Search Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[2] B.K. Pedersen, L. Hoffman-Goetz, Exercise and the immune system: regulation, integration, and adaptation, Physiol. Rev. 80 (2000), 1055–1081.

[3] R.A. Formato, Central force optimization: a new nature inspired computational framework for multidimensional search and optimization, in: N. Krasnogor, G. Nicosia, M. Pavone, D. Pelta (Eds.), Nature Inspired Cooperative Strategies for Optimization (NICSO 2007), Studies in Computational Intelligence, Springer, Berlin, Heidelberg, Germany, 2008.

[4] J. Kennedy, Particle swarm optimization, in Proceeding of 1995 IEEE International Conference on Neural Networks, Perth, Australia, 2011, pp. 1942–1948.

[5] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, GSA: a gravitational search algorithm, Inf. Sci. 179 (2009), 2232–2248.

[6] E. Rashedi, H. Nezamabadi-pour, S. Saryazdi, BGSA: binary gravitational search algorithm, Natural Comput. 9 (2009), 727–745.

[7] H.C. Shamsudin, A. Irawan, Z. Ibrahim, A.F.Z. Abidin, S. Wahyudi, M.A.A. Rahim, K. Khalil, A fast discrete gravitational search algorithm, in Proceedings of International Conference on Computational Intelligence, Kuantan, Malaysia, 2012, pp. 24–28.

[8] S. Sarafrazi, H. Nezamabadi-Pour, Facing the classification of binary problems with a GSA-SVM hybrid system, Math. Comput. Modell. 57 (2013), 270–278.

[9] S. Sarafrazi, H. Nezamabadi-pour, S. Saryazdi, Disruption: a new operator in gravitational search algorithm, Sci. Iran. 18 (2011), 539–548.

[10] N. Gouthamkumar, V. Sharma, R. Naresh, Disruption based gravitational search algorithm for short term hydrothermal scheduling, Expert Syst. Appl. 42 (2015), 7000–7011.

[11] V. Sharma, R. Naresh, G. Nadakuditi, Application of non-dominated sorting gravitational search algorithm with disruption operator for stochastic multiobjective short term hydrothermal scheduling, IET Gener. Trans. Distrib. 10 (2016), 862–872.

[12] X. Han, X. Chang, A chaotic digital secure communication based on a modified gravitational search algorithm filter, Inf. Sci. 208 (2012), 14–27.

[13] S. Mirjalili, A.H. Gandomi, Chaotic gravitational constants for the gravitational search algorithm, Appl. Soft Comput. 53 (2017), 407–419.

[14] N. Hadi, N. Mahdi, S. Patrick, A Multi-Objective Gravitational Search Algorithm Based on Non-Dominated Sorting, Int. J. of Swarm Intel. Res. 3 (2012), pp. 32–49.

[15] M. Khatibinia, S. Khosravi, A hybrid approach based on an improved gravitational search algorithm and orthogonal crossover for optimal shape design of concrete gravity dams, Appl. Soft Comput. J. 16 (2014), 223–233.

[16] I. Fister, D. Strnad, X.-S. Yang, I. Fister, Adaptation and hybridization in nature-inspired algorithms, in: I. Fister, I. Fister Jr (Eds.), Adaptation and Hybridization in Computational Intelligence, Springer, Cham, Switzerland, 2015, pp. 3–50.

[17] S.E. Mood, E. Rashedi, M.M. Javidi, New functions for mass calculation in gravitational search algorithm, J. Comput. Secur. 2 (2015), 233–246.

[18] F. Khajooei, E. Rashedi, A new version of gravitational search algorithm with negative mass, in Proceedings of Conference on Swarm Intelligence & Evolutionary Computation, Bam, Iran, 2016.

[19] E. Rashedi, E. Rashedi, H. Nezamabadi-pour, A comprehensive survey on gravitational search algorithm, Swarm Evol. Comput. 41 (2018), 141–158.

[20] S. He, L. Zhu, L. Wang, L. Yu, C. Yao, A modified gravitational search algorithm for function optimization, IEEE Access. 7 (2019), 5984–5993.
[21] D. Pelusi, R. Mascella, L. Tallini, J. Nayak, B. Naik, A. Abraham, Neural network and fuzzy system for the tuning of gravitational search algorithm parameters, Expert Syst. Appl. 102 (2018), 234–244.
[22] F.S. Saeidi-Khabisi, E. Rashedi, Fuzzy gravitational search algorithm, in 2012 2nd International eConference on Computer and Knowledge Engineering (ICCKE), Mashhad, Iran, 2012.
[23] H. Askari, S.-H. Zahiri, Decision function estimation using intelligent gravitational search algorithm, Int. J. Mach. Learn. Cybern. 3 (2011), 163–172.
[24] M.A. Kacimi, O. Guenounou, L. Brikh, F. Yahiaoui, N. Hadid, New mixed-coding PSO algorithm for a self-adaptive and automatic learning of Mamdani fuzzy rules, Eng. Appl. Artif. Intell. 89 (2020), 103417.
[25] A. Ragmani, A. Elomri, N. Abghour, K. Moussaid, M. Rida, An improved hybrid fuzzy-ant colony algorithm applied to load balancing in cloud computing environment, Procedia Comput. Sci. 151 (2019), 519–526.
[26] J. Vijaya Kumar, D.M. Vinod Kumar, K. Edukondalu, Strategic bidding using fuzzy adaptive gravitational search algorithm in a pool based electricity market, Appl. Soft Comput. 13 (2013), 2445–2455.
[27] X. Yao, L. Yong, Evolutionary programming made faster, IEEE Trans. Evol. Comput. 3 (1999), 82–102.
[28] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evol. Comput. 10 (2006), 281–295.