Received November 15, 2018, accepted December 2, 2018, date of publication January 1, 2019, date of current version January 16, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2889854

A Modified Gravitational Search Algorithm for Function Optimization

SHOUSHUAI HE, LEI ZHU, LEI WANG, LU YU, AND CHANGHUA YAO
College of Communication Engineering, Army Engineering University of PLA, Nanjing 210007, China

Corresponding authors: Lei Zhu (zhulei_paper@126.com), Lei Wang (iponly@126.com), and Changhua Yao (ych2347@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702543 and Grant 71501186 and in part by the 333 High-Level Talent Training Project of Jiangsu Province of China under Grant BRA 2016542.

ABSTRACT Gravitational search algorithm (GSA) is a population-based heuristic algorithm, which is inspired by Newton's laws of gravity and motion. Although GSA provides a good performance in solving optimization problems, it has a disadvantage of premature convergence. In this paper, the concept of repulsive force is introduced and the definition of exponential *Kbest* is used in a new version of GSA, which is called repulsive GSA with exponential *Kbest* (EKRGSA). In this algorithm, heavy particles repulse or attract all particles according to distance, and all particles search the solution space under the combined action of repulsive force and gravitational force. In this way, the exploration ability of the algorithm is improved and a proper balance between exploration and exploitation is established. Moreover, the exponential *Kbest* significantly decreases the computational time. The proposed algorithm is tested on a set of benchmark functions and compared with other algorithms. The experimental results confirm the high efficiency of EKRGSA.

INDEX TERMS Gravitational search algorithm, repulsive force, exponential *Kbest*, function optimization.

I. INTRODUCTION

Optimization means finding the global optimal solution of problem. Over the past few decades, the complexity of optimization problems has increased dramatically. Therefore, effective optimization methods have become more and more valuable. There are two kinds of methods for optimization, classical algorithms and heuristic algorithms [1]. Classical algorithms based on derivative use the gradient information of objective function to find the optimal solution. So this kind of methods are not suitable for non-differentiable functions. Besides, the computational complexity of classical algorithms increases exponentially as the dimension of problems increases. Thus, it is not practical to use them to solve high-dimensional problems. Since 1990s, a series of heuristic algorithms have been proposed, which can solve complex high-dimensional problems with incomplete information [2]. This characteristic makes them different from the classical mathematical algorithms.

Heuristic algorithms are stochastic algorithms, which seek for good solutions at reasonable computational costs without guaranteeing optimality. They are inspired by natural, biological and physical phenomena, and most of them are population-based algorithms [3]. Population-based algorithms search the solution space with a set of candidate solutions that are generated randomly. In each iteration, the population is updated by using some specific operations. These operations are almost simple, but their collective effect tends to improve the quality of new population. The iteration is repeated until the stop criterion is met.

Some of the most famous heuristic algorithms are Simulated Annealing (SA) which is designed based on the thermodynamic effects [4], Particle Swarm Optimization (PSO) which mimics the behavior of birds [5], Ant Colony Optimization (ACO) which simulates the foraging process of ants [6], and Genetic Algorithm (GA) which is inspired by Darwin's theory of evolution [7]. These algorithms are well suited to solve complex computational problems and show good performance in many fields such as function optimization [8], data clustering [9], pattern recognition [10], data mining [11], image processing [12], computer vision [13] and neural network training [14]. However, most of the existing algorithms have the disadvantage of premature convergence. And this issue remains to be solved.

Gravitational Search Algorithm is one of the new heuristic algorithms, which was proposed by Rashedi *et al.* [15] in 2009. It is inspired by Newton's laws of gravity and motion. In this algorithm, the performance of each particle is represented by its mass. All particles attract each other by gravitational force, and the force causes all particles to move towards the heaviest particle whose position represents the optimal solution. The salient characteristics of this algorithm are simplicity and efficiency, which make GSA useful for solving complex problems in technology and engineering [16]–[20]. However, like other heuristic algorithms, the major disadvantage of GSA is premature convergence.

The key to solving the issue of premature convergence in heuristic algorithms is to establish a proper balance between exploration and exploitation. Exploration is the ability to explore the solution space and find new solutions. Exploitation is the ability to find the optimal solution among good solutions. As iterations go on, exploration fades out and exploitation fades in. The agents in population-based heuristic algorithms do exploration and exploitation by three steps: self-adaptation, cooperation and competition. In self-adaptation stage, each agent improves its own performance. In cooperation stage, agents cooperate with each other by transmitting information. And in competition stage, agents compete for survival. The three steps inspired by natural process usually have stochastic forms. They can be implemented in different ways and guide the algorithm to find the global optimal solution.

In recent years, some modified versions of GSA have been proposed in order to avoid trapping in the local optimum and satisfy the computational requirements. In [21], a novel gravitational search algorithm with negative mass is proposed, which simulates the phenomenon of antigravity and modifies the definition of mass. In [22], a new attractive-repulsive gravitational search algorithm is proposed, in which the uniform circular motion and centripetal force are introduced. A chaotic optimization mechanism making *Kbest*, one of the parameters in the algorithm, decrease chaotically is applied in [23]. And a fuzzy logic controller is utilized in [24] to control the changes of some parameters and improve the convergence rate of the algorithm.

Since the attraction of gravitational force causes particles to approach each other merely, the original algorithm loses the ability to explore the solution space after premature convergence. Hence, in this paper, a new version of GSA based on repulsive force is proposed to enhance the exploration ability and establish a proper balance between exploration and exploitation. Simultaneously, the exponential *Kbest* further balances the exploration and exploitation, and significantly improves the computational efficiency of the algorithm.

This paper is organized as follows. In the next section, a brief review of GSA is provided. In Section III, the proposed EKRGSA and its characteristics are described in detail. In Section IV, the performance of EKRGSA is evaluated by a set of benchmark functions, and the experimental results are compared with the original algorithm and another modified algorithm. This paper is concluded in Section V.

II. GRAVITATIONAL SEARCH ALGORITHM

In this section, the original Gravitational Search Algorithm is introduced, which is a new population-based heuristic algorithm inspired by Newton's laws of gravity and motion. More precisely, particles in the algorithm obey the following laws:

- *Law of gravity:* Each particle attracts other particles. The gravitational force between two particles is directly proportional to the product of their masses and inversely proportional to the distance between them.
- *Law of motion:* The acceleration of each particle is equal to the resultant force acting on it divided by its inertial mass.

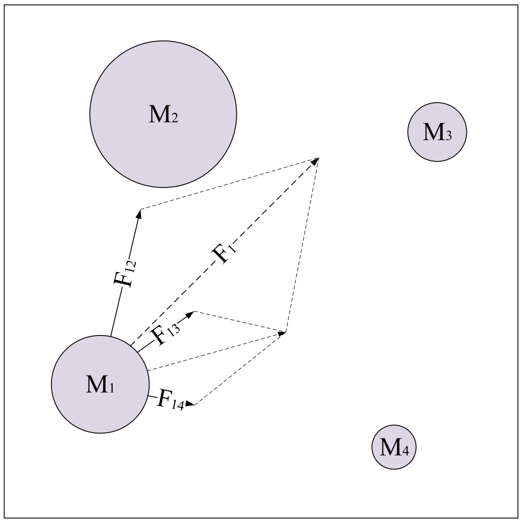


FIGURE 1. Principle of GSA.

The principle of GSA is depicted in Fig. 1. Particles in this algorithm simulate the planets in space. They move in the multidimensional solution space under the effect of gravitational force to find the optimal solution. Each particle has four attributes:

- *Position:* The position of each particle in the solution space corresponds to a candidate solution of problem.
- *Active gravitational mass:* This mass determines the intensity of gravitational field produced by a particle. And a particle with larger active gravitational mass produces a stronger gravitational field than other particles.
- *Passive gravitational mass:* This mass determines the intensity of gravitational force acting on a particle in a known gravitational field. Within the same gravitational field, a particle with larger passive gravitational mass experiences stronger gravitational force than others.
- *Inertial mass:* This mass determines the intensity of resistance to motion change when force acts on a particle. And a particle with large inertial mass changes its motion slowly, while a particle with small inertial mass changes rapidly.

In GSA, the active gravitational, passive gravitational and inertial masses of each particle are equal and determined by a fitness function.

This algorithm is advanced by adjusting the mass of each particle, which represents its performance. Heavier particles correspond to more effective solutions than others. All particles attract each other according to the law of gravity and move according to the law of motion. As iterations go on, the population will be attracted by the heaviest particle whose position represents the optimal solution in the solution space.

In a D-dimensional solution space that contains N particles, the position of particle i is defined as follows:

$$X_i(t)=(x_i^1(t),\ldots,x_i^d(t),\ldots,x_i^D(t)),\quad i=1,2,\ldots,N, \tag{1}$$

where $x_i^d(t)$ denotes the position of particle i in dimension d.

G is named as the gravitational constant but decreased with iterations to control the search accuracy:

$$G = G_0 \times e^{-\alpha \frac{t}{max\_it}}, \tag{2}$$

where G_0 is the initial value, t is the current iteration, max_it is the maximum number of iterations, and α is a shrink constant, which controls the decay rate of the exponential function.

Kbest is the number of particles with large masses. Only the *Kbest* particles exert gravitational force on all particles. It is a function of iteration and decreased linearly from N to 1:

$$Kbest = N \times \frac{final\_per + \left(1-\frac{t}{max\_it}\right)\times(100-final\_per)}{100}, \tag{3}$$

where *final_per* denotes the percent of particles that exert gravitational force on all particles in the end. The decreasing process of the linear *Kbest* is shown in Fig. 2, where N is 50, *final_per* is 2, and *max_it* is 1000.

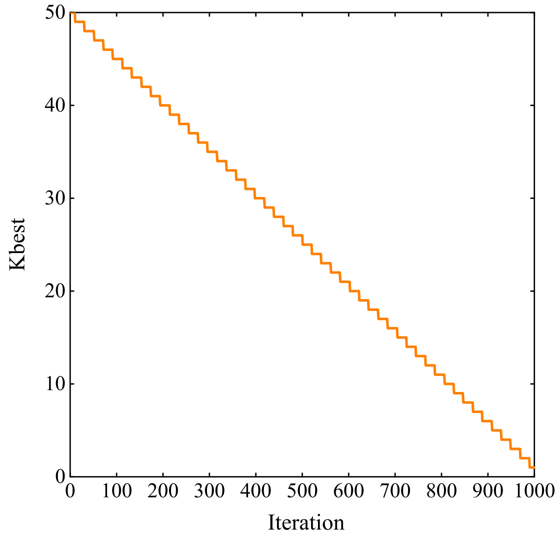


FIGURE 2. Decreasing process of linear *Kbest*.

The parameters G and *Kbest* play an important role in balancing the exploration and exploitation. In order to improve the performance of the algorithm, the values of G and *Kbest* are large at the beginning and decreased with iterations.

The mass of particle i is updated by the following equations:

$$M_i(t) = M_{ai}(t) = M_{pi}(t) = M_{ii}(t), \tag{4}$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)}, \tag{5}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \tag{6}$$

where $M_{ai}(t)$, $M_{pi}(t)$ and $M_{ii}(t)$ denote the active gravitational, passive gravitational and inertial masses of particle i, respectively. And $fit_i(t)$ denotes the fitness value of particle i, which is evaluated by the fitness function. In this algorithm, $best(t)$ and $worst(t)$ denote the best and worst fitness value among N particles, respectively. For minimization problems, they are defined as follows:

$$best(t) = \min_{j\in\{1,\ldots,N\}} fit_j(t), \tag{7}$$

$$worst(t) = \max_{j\in\{1,\ldots,N\}} fit_j(t). \tag{8}$$

And for maximization problems, they are changed to the following forms:

$$best(t) = \max_{j\in\{1,\ldots,N\}} fit_j(t), \tag{9}$$

$$worst(t) = \min_{j\in\{1,\ldots,N\}} fit_j(t). \tag{10}$$

The gravitational force acting on particle i from particle j in dimension d is calculated by the following equation:

$$F_{ij}^d(t) = G \times \frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon} \times (x_j^d(t) - x_i^d(t)), \tag{11}$$

where ε is a small constant, and $R_{ij}(t)$ is the Euclidean distance between particle i and particle j, which is defined as follows:

$$R_{ij}(t) = \left\| X_i(t), X_j(t) \right\|_2. \tag{12}$$

The resultant force acting on particle i in dimension d is a random weighted sum of gravitational force exerted from the *Kbest* particles:

$$F_i^d(t) = \sum_{j\in Kbest, j\neq i} rand_j \times F_{ij}^d(t), \tag{13}$$

where $rand_j$ is a uniformly distributed random number in interval [0, 1], which provides a stochastic characteristic for the algorithm.

According to the law of motion, the acceleration of particle i in dimension d is calculated by the following equation:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}. \tag{14}$$

The next velocity of particle i in dimension d is equal to a fraction of its current velocity plus its acceleration:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t), \tag{15}$$

where $rand_i$ is a uniformly distributed random number in interval [0, 1], which enhances the stochastic characteristic of the search.

Furthermore, the next position of particle i in dimension d is updated as follows:

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1). \tag{16}$$

The process of GSA are listed as Algorithm 1.

Algorithm 1 Gravitational Search Algorithm (GSA)

```
Input:
    $N$: population size
    $max\_it$: maximum number of iterations
Output:
    $S$: optimal solution
 1: Identify the solution space
 2: Initialize the population
 3: while the stop criterion is not satisfied do
 4:     Update the gravitational constant and linear $Kbest$
 5:     Evaluate the fitness value of each particle by the fitness
        function
 6:     Update the mass of each particle
 7:     Calculate the gravitational force between particles in
        different dimensions
 8:     Calculate the resultant force and acceleration of each
        particle in different dimensions
 9:     Update the velocity and position of each particle in
        different dimensions
10: end while
11: return $S$
```

III. REPULSIVE GRAVITATIONAL SEARCH ALGORITHM WITH EXPONENTIAL KBEST

Finding the approximate global optimal solution within the reasonable computational time is the primary task of heuristic algorithms. One way to achieve this task is to provide a proper balance between exploration and exploitation. In this section, a new version of GSA inspired by the physical phenomenon of repulsive force is proposed to enhance the exploration ability and computational efficiency of the algorithm.

A. REPULSIVE FORCE

The main characteristic of GSA is that the force is always attractive, while in the proposed algorithm, the definition of force is modified. In addition to the gravitational force, the concept of repulsive force is introduced. Similar to the interaction between charges, under the same conditions, the repulsive force between heterogeneous charges and the attractive force between homogeneous charges are equal in size but opposite in direction. Therefore, the repulsive force defined in this paper is equal to the gravitational force in size but exactly opposite to it in direction.

In the original algorithm, the *Kbest* particles with large masses exert gravitational force on all particles. It means that heavy particles merely attract all particles. But in the proposed algorithm, the *Kbest* particles exert both repulsive force and gravitational force on all particles according to distance. In other words, heavy particles affect near and far particles in two different ways. Supposing that each particle has a repulsive radius (R_r), which is changed with iterations. The force field generated by a particle provides repulsive force when the distance to another particle is more than R_r, and provides gravitational force when the distance is less than R_r. The repulsive radius is defined by the following equation:

$$R_r = A \times \log_{max\_it} t, \tag{17}$$

where t is the current iteration, max_it is the maximum number of iterations, and A is a positive variable, which is changed with the ranges of solution space to enhance the adaptability of the repulsive radius.

The three situations mentioned below correspond to Fig. 3. In early iterations, the value of R_r is small, and most of particles are repulsed by heavy particles because their distances to the heavy particles are more than R_r. This strategy improves the exploration ability of the algorithm at the beginning. As iterations go on, the value of R_r is increased. In middle iterations, some particles are repulsed by heavy particles to explore the solution space, while the others are attracted by heavy particles to find the optimal solution. And in late iterations, the value of R_r is large, most of particles are attracted by heavy particles because their distances to the heavy particles are less than R_r. This strategy maintains the exploitation ability of the algorithm. Therefore, the proposed algorithm enhances the exploration ability, maintains the exploitation ability, and establishes a proper balance between them.

In Fig. 3(b), it is assumed that particle i, whose repulsive radius is R_r, is a heavy particle, and there are other particles j and k. If the distance between particle i and particle j is less than R_r, then particle i will attract particle j. The gravitational force acting on particle j from particle i in dimension d when $\left|x_i^d(t) - x_j^d(t)\right| < R_r$ is the same as the original gravitational force:

$$Fg_{ji}^d(t) = G \times \frac{M_{pj}(t) \times M_{ai}(t)}{R_{ji}(t) + \varepsilon} \times (x_i^d(t) - x_j^d(t)), \tag{18}$$

If the distance between particle i and particle k is more than R_r, then particle i will repulse particle k. As mentioned above, the repulsive force acting on particle k from particle i in dimension d when $\left|x_i^d(t) - x_k^d(t)\right| \geq R_r$ is defined as follows:

$$Fr_{ki}^d(t) = -G \times \frac{M_{pk}(t) \times M_{ai}(t)}{R_{ki}(t) + \varepsilon} \times (x_i^d(t) - x_k^d(t)). \tag{19}$$

The gravitational force and repulsive force acting on particle i in dimension d are calculated by the following equations, respectively:

$$Fg_i^d(t) = \sum_{j \in Kbest, j \neq i} rand_j \times Fg_{ij}^d(t), \tag{20}$$

$$Fr_i^d(t) = \sum_{k \in Kbest, k \neq i} rand_k \times Fr_{ik}^d(t), \tag{21}$$

where $rand_j$ and $rand_k$ are uniformly distributed random numbers in interval [0, 1].

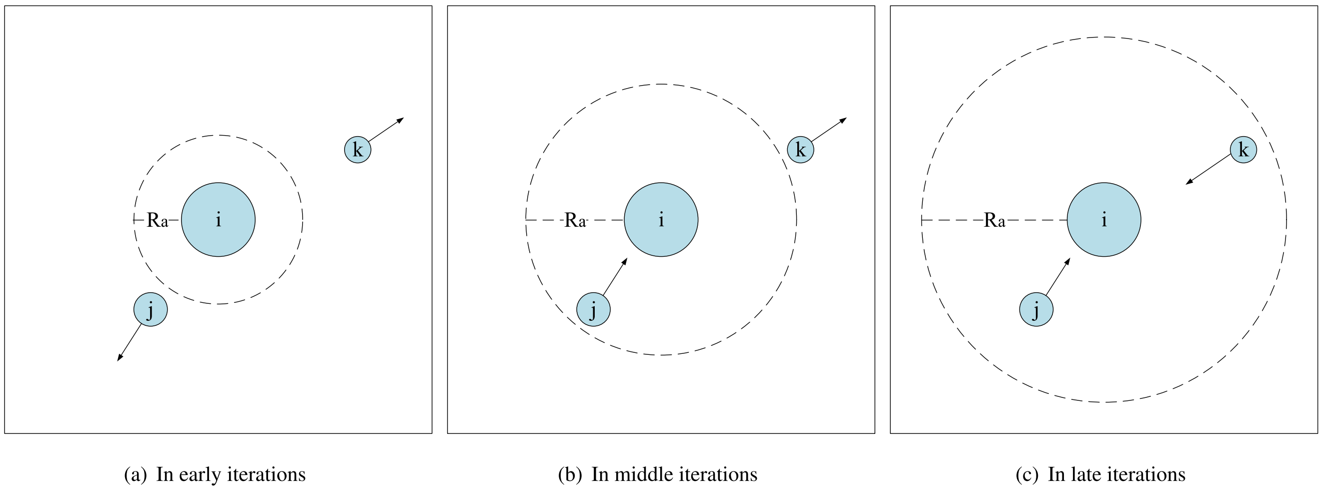


FIGURE 3. Sizes of repulsive radius and directions of force exerted from heavy particle *i* in early, middle and late iterations. (a) In early iterations. (b) In middle iterations. (c) In late iterations.

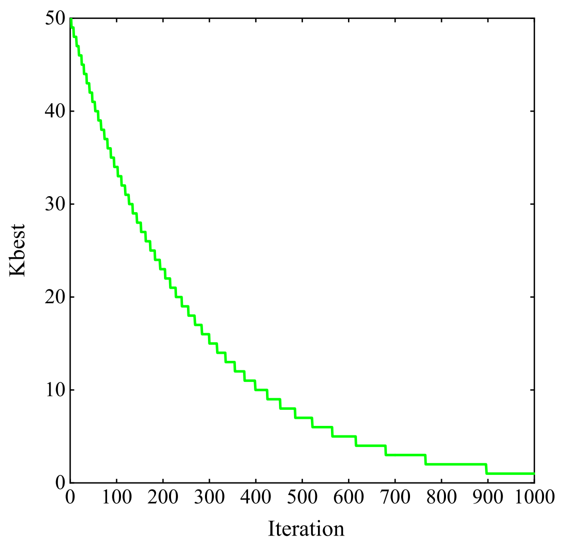


FIGURE 4. Decreasing process of exponential *Kbest*.

If $Fr_i^d(t) > Fg_i^d(t)$, the repulsive force will be dominant and particle i will be repulsed. On the contrary, if $Fr_i^d(t) < Fg_i^d(t)$, the gravitational force will be dominant and particle i will be attracted. Hence, the resultant force acting on particle i in dimension d is equal to the sum of repulsive force and gravitational force, which is a modified version of the original resultant force:

$$F_i^d(t) = Fr_i^d(t) + Fg_i^d(t). \tag{22}$$

Algorithm 2 Repulsive Gravitational Search Algorithm With Exponential *Kbest* (EKRGSA)

```
Input:
    $N$: population size
    $max\_it$: maximum number of iterations
Output:
    $S$: optimal solution
1:  Identify the solution space
2:  Initialize the population
3:  while the stop criterion is not satisfied do
4:      Update the gravitational constant, exponential $Kbest$ and repulsive radius
5:      Evaluate the fitness value of each particle by the fitness function
6:      Update the mass of each particle
7:      if the distance between particles $\geq R_r$ then
8:          Calculate the repulsive force between particles in different dimensions
9:      else
10:         Calculate the gravitational force between particles in different dimensions
11:     end if
12:     Calculate the resultant force and acceleration of each particle in different dimensions
13:     Update the velocity and position of each particle in different dimensions
14: end while
15: return $S$
```

B. EXPONENTIAL KBEST

In order to improve the computational efficiency of the algorithm and enhance the balance between exploration and exploitation, the definition of *Kbest* is modified in this paper.

In the original algorithm, *Kbest* is the number of particles that exert force on all particles. It is a function of iteration and decreased linearly from N to 1. But in the proposed algorithm, the function is modified, and *Kbest* is decreased exponentially with iterations from N to 1 as follows:

$$Kbest = N \times \left(\frac{final\_per}{100}\right)^{\frac{t}{max\_it}}, \tag{23}$$

where N is the population size, *final_per* is the percent of particles that exert force on all particles in the end, t is the

TABLE 1. Benchmark functions.

	Function	Range	Minimum value
Unimodal	$F_1(x) = \sum_{i=1}^{D} {x_i}^2$	$[-100, 100]^D$	0
	$F_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$	0
	$F_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^D$	0
	$F_4(x) = \max_i \{ \lvert x_i \rvert, 1 \le i \le D \}$	$[-100, 100]^D$	0
	$F_5(x) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^D$	0
	$F_6(x) = \sum_{i=1}^{D} i x_i^4 + random[0, 1)$	$[-1.28, 1.28]^D$	0
Multimodal	$F_7(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{D} \cos(2\pi x_i) \right) + 20 + e$	$[-32, 32]^D$	0
	$F_8(x) = \frac{\pi}{n} \left\{ 10 \sin(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 \left[1 + 10\sin^2(\pi y_{i+1}) \right] + (y_n - 1)^2 \right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	$[-50, 50]^D$	0
	$F_9(x) = 0.1 \left\{ \sin^2(3\pi x_1) + \sum_{i=1}^{D} (x_i - 1)^2 \left[1 + \sin^2(3\pi x_i + 1) \right] + (x_n - 1)^2 \left[1 + \sin^2(2\pi x_n) \right] \right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$	0
	$F_{10}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1 (b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4} \right]^2$	$[-5, 5]^4$	0.00030
	$F_{11}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \right)^{-1}$	$[-65.53, 65.53]^2$	1
	$F_{12}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	-1.0316
	$F_{13}(x) = \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2) \right] \times \left[30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2) \right]$	$[-5, 5]^2$	3
	$F_{14}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2 \right)$	$[0, 1]^3$	-3.86
	$F_{15}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij} (x_j - p_{ij})^2 \right)$	$[0, 1]^6$	-3.32
	$F_{16}(x) = -\sum_{i=1}^{5} \left[(X - a_i)(X - a_i)^T + c_i \right]^{-1}$	$[0, 10]^4$	-10.1532
	$F_{17}(x) = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i \right]^{-1}$	$[0, 10]^4$	-10.4029
	$F_{18}(x) = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i \right]^{-1}$	$[0, 10]^4$	-10.5364

current iteration, and *max_it* is the maximum number of iterations. The decreasing process of the exponential *Kbest* is shown in Fig. 4, where N is 50, *final_per* is 2 and *max_it* is 1000.

Neglecting the inconsequential particles with small masses can effectively reduce the redundant computations then improve the computational efficiency. Simultaneously, such modification makes the algorithm do comprehensive exploration in early iterations and precise exploitation in late iterations [25]. So the exploration and exploitation abilities of the algorithm get further balanced.

In addition, other equations in the proposed algorithm such as mass and acceleration calculations, velocity and position updates are the same as the original algorithm.

In summary, the detailed steps of EKRGSA are listed as Algorithm 2.

The time complexity of the proposed algorithm is analyzed as follows:

- Population initialization requires $O(N \times D)$, where D denotes the dimension of solution space.
- Gravitational constant, exponential *Kbest* and repulsive radius updates require $O(max\_it)$ each.

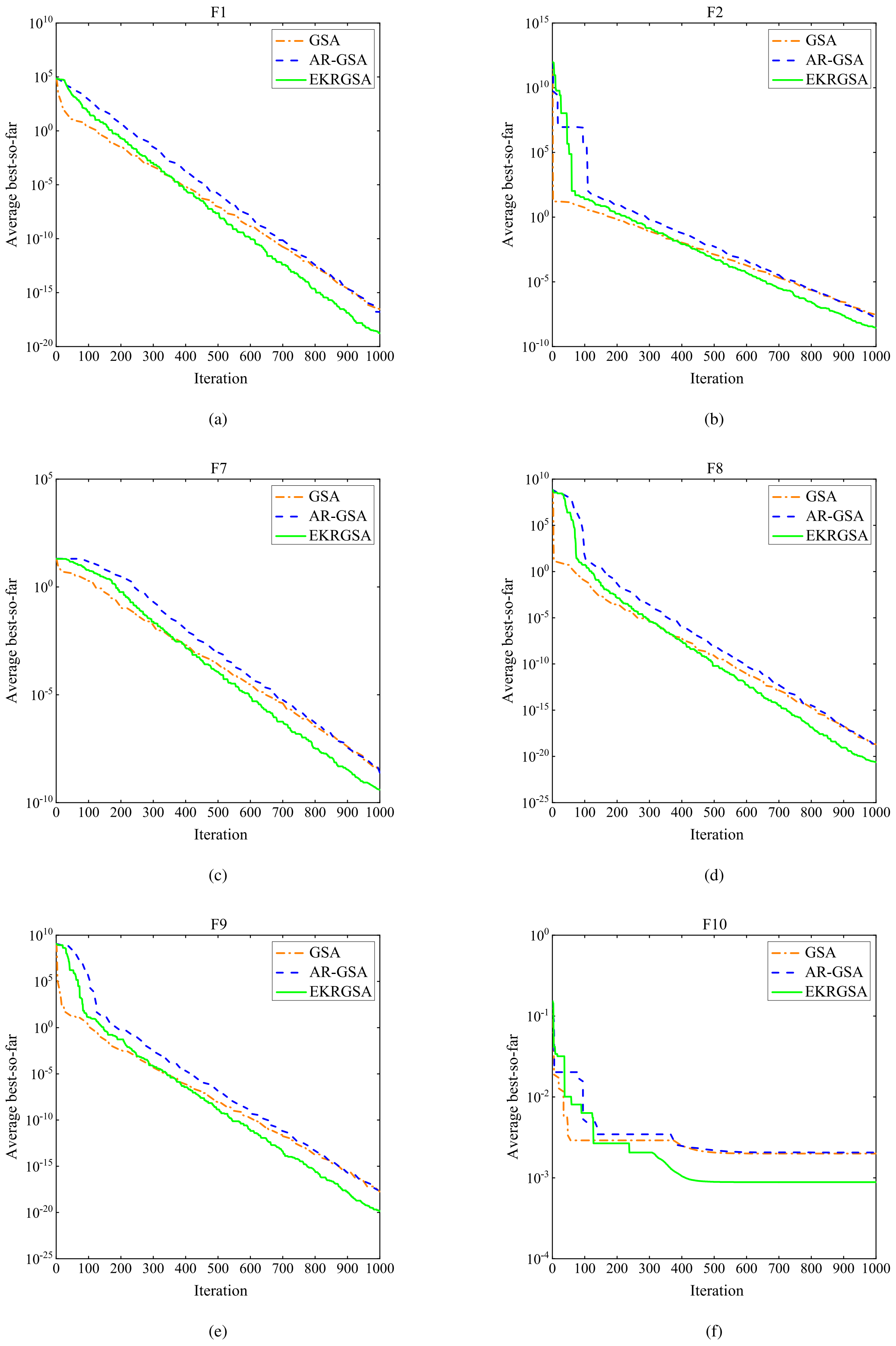


FIGURE 5. **Performance comparisons. (a) Performance comparison for F_1. (b) Performance comparison for F_2. (c) Performance comparison for F_7. (d) Performance comparison for F_8. (e) Performance comparison for F_9. (f) Performance comparison for $F_1$0.**

TABLE 2. Parameter settings.

Parameter	GSA	AR-GSA	EKRGSA
N	50	50	50
max_it	1000	1000	1000
G_0	100	4000	1000
α	20	25	25
$Kbest$	linear	$0.6 \times N$	exponential

- Fitness value evaluation requires $O(N \times D \times max\_it)$.
- Mass calculation requires $O(N \times max\_it)$.
- Repulsive force and gravitational force calculations approximately require $O(N^2 \times D \times \max\_it)$, when *Kbest* is decreased exponentially with iterations and *max_it* is large enough.
- Resultant force and acceleration calculations, velocity and position updates require $O(N \times D \times max\_it)$ each.

Considering the complexities of the above steps, the total time complexity of EKRGSA is $O(N^2 \times D \times \max\_it)$, which is equal to the original algorithm.

The space requirement of the proposed algorithm is related to the population size and dimension of solution space. Thus, the total space complexity of EKRGSA is $O(N \times D)$, which is also equal to the original algorithm.

In EKRGSA, each particle is affected by the *Kbest* particles according to distance, where *Kbest* is decreased exponentially with iterations. All particles search the solution space under the combined action of repulsive force and gravitational force. In this way, the search ability and efficiency of the algorithm get a significant boost.

IV. EXPERIMENTAL RESULTS

Satisfactory solution and reasonable computational costs are two major criteria for heuristic algorithms to solve practical problems. In this section, in order to investigate the performance of EKRGSA, it is compared with the original algorithm and another modified algorithm: Attractive-Repulsive Gravitational Search Algorithm (AR-GSA), in which the uniform circular motion in physics is introduced, and particles exert centripetal force on each other based on the hypothetical absorption radius to improve the exploration ability [22].

The existing and proposed algorithms are tested on a set of benchmark functions including some unimodal and multimodal functions as presented in Table 1, followed by the dimension and range of each function, where D is 30. All benchmark functions are minimization problems, and the minimum values are presented at the end of Table 1. Some detailed descriptions of the functions are given in [15]. The convergence rate is more important for the unimodal functions, while the minimization results are more important for the multimodal functions because they have many local minimums which make algorithms more difficult to find the global minimum.

TABLE 3. Minimization results.

Function	Best-so-far	GSA	AR-GSA	EKRGSA
$F_1(x)$	Average	2.03×10^{-17}	1.83×10^{-17}	$\mathbf{3.68\times10^{-19}}$
	Median	1.92×10^{-17}	1.82×10^{-17}	$\mathbf{2.13\times10^{-19}}$
$F_2(x)$	Average	2.32×10^{-8}	1.85×10^{-8}	$\mathbf{3.74\times10^{-9}}$
	Median	2.28×10^{-8}	1.83×10^{-8}	$\mathbf{2.71\times10^{-9}}$
$F_3(x)$	Average	$\mathbf{2.73\times10^{2}}$	8.80×10^{2}	2.46×10^{3}
	Median	$\mathbf{2.64\times10^{2}}$	8.22×10^{2}	2.34×10^{3}
$F_4(x)$	Average	3.64×10^{-9}	$\mathbf{2.01\times10^{-9}}$	2.16×10^{-8}
	Median	3.44×10^{-9}	$\mathbf{1.98\times10^{-9}}$	1.38×10^{-8}
$F_5(x)$	Average	34.84	26.08	**24.57**
	Median	26.14	26.06	**24.56**
$F_6(x)$	Average	**0.02**	0.03	0.06
	Median	**0.02**	0.03	0.06
$F_7(x)$	Average	3.54×10^{-9}	3.14×10^{-9}	$\mathbf{3.61\times10^{-10}}$
	Median	3.35×10^{-9}	3.15×10^{-9}	$\mathbf{3.01\times10^{-10}}$
$F_8(x)$	Average	2.08×10^{-19}	1.91×10^{-19}	$\mathbf{1.91\times10^{-21}}$
	Median	1.49×10^{-19}	1.33×10^{-19}	$\mathbf{1.18\times10^{-21}}$
$F_9(x)$	Average	2.46×10^{-18}	2.66×10^{-18}	$\mathbf{3.13\times10^{-20}}$
	Median	2.07×10^{-18}	2.00×10^{-18}	$\mathbf{2.30\times10^{-20}}$
$F_{10}(x)$	Average	2.15×10^{-3}	1.98×10^{-3}	$\mathbf{9.05\times10^{-4}}$
	Median	2.01×10^{-3}	2.02×10^{-3}	$\mathbf{8.60\times10^{-4}}$
$F_{11}(x)$	Average	4.47	1.28	**1.02**
	Median	3.13	1.08	**1.00**
$F_{12}(x)$	Average	**-1.0316**	**-1.0316**	**-1.0316**
	Median	**-1.0316**	**-1.0316**	**-1.0316**
$F_{13}(x)$	Average	**3**	**3**	**3**
	Median	**3**	**3**	**3**
$F_{14}(x)$	Average	**-3.86**	**-3.86**	**-3.86**
	Median	**-3.86**	**-3.86**	**-3.86**
$F_{15}(x)$	Average	**-3.32**	**-3.32**	**-3.32**
	Median	**-3.32**	**-3.32**	**-3.32**
$F_{16}(x)$	Average	-7.1037	-7.4958	**-8.0777**
	Median	**-10.1532**	**-10.1532**	**-10.1532**
$F_{17}(x)$	Average	**-10.4029**	**-10.4029**	**-10.4029**
	Median	**-10.4029**	**-10.4029**	**-10.4029**
$F_{18}(x)$	Average	**-10.5364**	**-10.5364**	**-10.5364**
	Median	**-10.5364**	**-10.5364**	**-10.5364**

TABLE 4. Computational time (in seconds).

Function	GSA	AR-GSA	EKRGSA
$F_1(x)$	7.65	10.50	**4.98**
$F_2(x)$	7.57	10.52	**5.08**
$F_3(x)$	9.93	12.94	**7.47**
$F_4(x)$	7.49	10.47	**4.99**
$F_5(x)$	7.88	10.87	**5.21**
$F_6(x)$	7.77	10.72	**5.43**
$F_7(x)$	7.56	10.68	**5.00**
$F_8(x)$	8.69	11.68	**6.17**
$F_9(x)$	8.63	11.81	**6.23**
$F_{10}(x)$	4.81	6.72	**3.44**
$F_{11}(x)$	7.75	9.62	**6.19**
$F_{12}(x)$	4.41	6.21	**3.11**
$F_{13}(x)$	4.45	6.16	**3.09**
$F_{14}(x)$	4.92	6.72	**3.57**
$F_{15}(x)$	5.21	7.15	**3.76**
$F_{16}(x)$	5.04	6.92	**3.63**
$F_{17}(x)$	5.15	7.09	**3.78**
$F_{18}(x)$	5.33	7.28	**3.97**

In all tests, the population size is 50, and the maximum number of iterations is 1000. In GSA, *Kbest* is initialized to N and decreased linearly with iterations to 1. In AR-GSA, *Kbest* is considered as a constant and set to 0.6 times the population size. And in EKRGSA, A is set to the size of range of each function, G_0 is set to 1000 and α is set to 25, which are determined by voluminous experiments and comparisons. Besides, *Kbest* is initialized to N and decreased exponentially with iterations to 1. The detailed parameter settings of three algorithms are shown in Table 2.

The average and median best-so-far results in the last iteration listed in Table 3 and the computational time listed in Table 4 are the average of 30 independent runs. The minimum value and the shortest computational time of each benchmark function are presented in the bold form.

A. MINIMIZATION RESULTS

The repulsive force prevents the proposed algorithm from trapping in the local optimum. It is clear from Table 3 that EKRGSA obtains better or equal results compared with other algorithms in most of the benckmark functions except for F_3, F_4 and F_6, in which GSA and AR-GSA provide slightly smaller results, respectively.

B. COMPUTATIONAL TIME

The exponential *Kbest* neglects the force exerted from the inconsequential particles with small masses. Thus, it can improve the computational efficiency. Table 4 shows that EKRGSA has shorter computational time than other algorithms in all cases.

C. CONVERGENCE RATE

Since EKRGSA adequately explores the solution space, it converges slowly in early iterations. But under the combined action of repulsive force and exponential *Kbest*, EKRGSA can provide better results and faster convergence rate than other algorithms in late iterations. The convergence trends of three algorithms for part of representative unimodal and multimodal benchmark functions are presented in Fig. 5. This figure confirms the high performance of EKRGSA.

V. CONCLUSION

In recent years, the demands for heuristic algorithms to solve complex optimization problems have been increased dramatically. As one of the new heuristic algorithms, GSA is constructed based on Newton's laws of gravity and motion. In this paper, a new version of GSA is proposed, which is called EKRGSA. In this algorithm, the concept of repulsive force, similar to the interaction between heterogeneous charges in physics, is introduced. Particles search the solution space under the combined action of repulsive force and gravitational force. In this way, the exploration ability of the algorithm is improved and a proper balance between exploration and exploitation is established. Simultaneously, the exponential *Kbest* can not only further balance the exploration and exploitation, but also significantly improve the computational efficiency. In order to evaluate the proposed algorithm, a set of benchmark functions are used in the experiment. The experimental results show that, in most cases, EKRGSA can obtain better solutions and provide higher performance. Thus, it is confirmed that EKRGSA is suitable for solving function optimization problems.

REFERENCES

[1] D. K. de Sa Vieira and M. H. S. Mendes, "A comparison of algorithms for solving multicomponent optimization problems," *IEEE Latin Amer. Trans.*, vol. 15, no. 8, pp. 1474–1479, Jul. 2017, doi: 10.1109/TLA.2017.7994795.

[2] J. Gu, S. J. Bae, S. F. Hasan, and M. Y. Chung, "Heuristic algorithm for proportional fair scheduling in D2D-cellular systems," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 769–780, Jan. 2016, doi: 10.1109/TWC.2015.2477998.

[3] R. E. Zich, A. Niccolai, M. Ruello, F. Grimaccia, and M. Mussetta, "Novel population-based algorithms for reflectarray optimization," in *Proc. IEEE ICEAA*, Palm Beach, Netherlands Antilles, Aug. 2014, pp. 818–821.

[4] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983. May 1983, doi: 10.1126/science.220.4598.671.

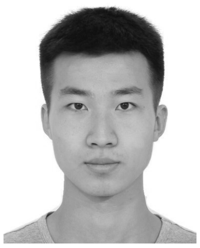
[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE ICNN*, Perth, WA, Australia, Nov./Dec. 1995, pp. 1942–1948.

[6] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996, doi: 10.1109/3477.484436.

[7] K. S. Tang, K. F. Man, S. Kwong, and Q. He, "Genetic algorithms and their applications," *IEEE Signal Process. Mag.*, vol. 13, no. 6, pp. 22–37, Nov. 1996, doi: 10.1109/79.543973.

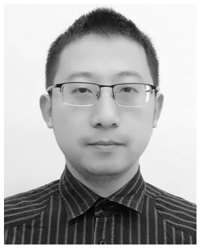
[8] S. K. Sarangi, R. Panda, S. Priyadarshini, and A. Sarangi, "A new modified firefly algorithm for function optimization," in *Proc. IEEE ICEEOT*, Chennai, India, Mar. 2016, pp. 2944–2949.

[9] C. A. Dhote, A. D. Thakare, and S. M. Chaudhari, "Data clustering using particle swarm optimization and bee algorithm," in *Proc. IEEE ICCCNT*, Tiruchengode, India, Jul. 2013, pp. 1–5.
[10] V.-E. Neagoe and E.-C. Neghina, "Feature selection with Ant Colony Optimization and its applications for pattern recognition in space imagery," in *Proc. IEEE COMM*, Bucharest, Romania, Jun. 2016, pp. 101–104.
[11] M. Grami, R. Gheibi, and F. Rahimi, "A novel association rule mining using genetic algorithm," in *Proc. IEEE IKT*, Hamedan, Iran, Sep. 2016, pp. 200–204.
[12] R. Wang, "Research on image processing based on improved particle swarm optimization," in *Proc. IEEE ICMTMA*, Changsha, China, Feb. 2018, pp. 538–540.
[13] G. Lee, R. Mallipeddi, G.-J. Jang, and M. Lee, "A genetic algorithm-based moving object detection for real-time traffic surveillance," *IEEE Signal Process. Lett.*, vol. 22, no. 10, pp. 1619–1622, Oct. 2015, doi: 10.1109/LSP.2015.2417592.
[14] I. O. Aksu and R. Coban, "Training the multifeedback-layer neural network using the Particle Swarm Optimization algorithm," in *Proc. IEEE ICECCO*, Ankara, Turkey, Nov. 2013, pp. 172–175.
[15] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *J. Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009, doi: 10.1016/j.ins.2009.03.004.
[16] L. Dos Santos Coelho, V. C. Mariani, N. Tutkun, and P. Alotto, "Magnetizer design based on a quasi-oppositional gravitational search algorithm," *IEEE Trans. Magn.*, vol. 50, no. 2, pp. 705–708, Feb. 2014, doi: 10.1109/TMAG.2013.2285277.
[17] H. Nikbakht and H. Mirvaziri, "A new algorithm for data clustering based on gravitational search algorithm and genetic operators," in *Proc. IEEE AISP*, Mashhad, Iran, Mar. 2015, pp. 222–227.
[18] A. Sheshasaayee and D. Sridevi, "Fuzzy C-means algorithm with gravitational search algorithm in spatial data mining," in *Proc. IEEE ICICT*, Coimbatore, India, Aug. 2016, pp. 1–5.
[19] J. M. Tripathi and R. Krishan, "Optimal coordination of overcurrent relays using gravitational search algorithm with DG penetration," in *Proc. IEEE PIICON*, Delhi, India, Dec. 2014, pp. 1–6.
[20] S. Sheikhpour, M. Sabouri, and S.-H. Zahiri, "A hybrid Gravitational search algorithm—Genetic algorithm for neural network training," in *Proc. IEEE ICEE*, Mashhad, Iran, May 2013, pp. 1–5.
[21] F. Khajooei and E. Rashedi, "A new version of Gravitational Search Algorithm with negative mass," in *Proc. IEEE CSIEC*, Bam, Iran, Mar. 2016, pp. 1–5.
[22] H. Zandevakili, E. Rashedi, and A. Mahani, "Gravitational search algorithm with both attractive and repulsive forces," *Soft Comput.*, vol. 2, pp. 1–43, Aug. 2017, doi: 10.1007/s00500-017-2785-2.
[23] H. Mittal, R. Pal, A. Kulhari, and M. Saraswat, "Chaotic Kbest gravitational search algorithm (CKGSA)," in *Proc. IEEE IC3*, Noida, India, Aug. 2016, pp. 1–6.
[24] F. S. Saeidi-Khabisi and E. Rashedi, "Fuzzy gravitational search algorithm," in *Proc. IEEE ICCKE*, Mashhad, Iran, Oct. 2012, pp. 156–160.
[25] H. Mittal and M. Saraswat, "An optimum multi-level image thresholding segmentation using non-local means 2D histogram and exponential Kbest gravitational search algorithm," *Eng. Appl. Artif. Intell.*, vol. 71, pp. 226–235, May 2018, doi: 10.1016/j.engappai.2018.03.001.

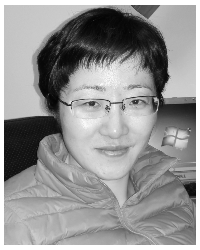


SHOUSHUAI HE received the B.S. degree in information security from the Harbin Institute of Technology, in 2017. He is currently a Graduate Student with the Army Engineering University of PLA. His research interests include artificial intelligence and network security.

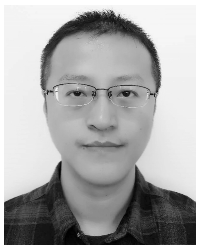
LEI ZHU received the Ph.D. degree from the College of Communications Engineering, PLA University of Science and Technology, China, in 2002. He is currently a Professor of system engineering. He has authored more than 35 scientific articles. His research interests include network planning and system simulation.



LEI WANG received the Ph.D. degree in military operational research from the PLA University of Science and Technology, in 2014, China. He holds a Postdoctoral position in communication and information system and an Engineer of optimization and system engineering with the College of Communications Engineering, Army Engineering University of PLA, China. He has authored more than 25 scientific articles. His research interests include knowledge engineering, data mining, artificial intelligence, network planning, and military operational research.



LU YU received the Ph.D. degree from Southeast University, in 2007. She is currently with the Institute of Communications Engineering, Army Engineering University of PLA. She has authored more than 25 scientific articles. Her current research interests include machine learning, image understanding, and pattern recognition.



CHANGHUA YAO received the B.S. degree in automation from Zhejiang University, in 2005, and the Ph.D. degree in communications and information systems from the PLA University of Science and Technology, in 2016. He has been holding a Postdoctoral position at the Army Engineering University of PLA, since 2017. He has authored more than 25 scientific articles. His research interests include wireless networks, network security, opportunistic spectrum access, distributed optimization, data analysis, artificial intelligence, and machine learning.