RGSA: A New Improved Gravitational Search Algorithm

Ruopeng Wang[1], Fang Su[1], Tongan Hao[1] and Jilong Li[2]
[1]Beijing University of Posts and Telecommunications, Beijing, China
[2]Academy of Broadcasting Science, Beijing, China

***Abstract*—In this paper, an improved gravitational search algorithm, revolving gravitational search algorithm (RGSA) is proposed. RGSA optimizes the selection of kbest in GSA by importing revolving operator, lets agents not included in kbest have opportunity to influence other agents' movement. RGSA could further exploit potential districts and strengthen the ability of exploration than GSA. Experiments confirm the high efficiency of RGSA.**

***Keywords*—gravitational search algorithm; exploration; exploitation; revolving operator**

I. Introduction

Gravitational search algorithm (GSA) is an optimization algorithm proposed by Rashedi et al. [1]. GSA is a powerful heuristic algorithm and is widely used in various real optimization problems, such as neural network [2], optimal capacitor placement [3], clustering and classification problem [4], feature selection [5].

As a swarm optimization algorithm, GSA has great ability of exploration and exploitation. Exploration represents the ability to search other districts which may bring better solutions while exploitation is the ability to search carefully around a good solution [6]. Both of the two abilities are important to the performance of GSA.

Since the proposal of GSA, GSA has attracted much scholarly attention. Up to now, there has been many researches to improve GSA's performance. The main achievements are as follows: Sarafrazi et al. [6] proposed disruption operator to improve the performance of GSA by emphasizing in exploration at start and switch to exploitation as time goes by. Kun Qian et al. [7] introduced hybrid GSA incorporates the local search technique (LST) into GSA, makes full use of the exploration ability of GSA and the exploitation ability of LST. Mirjalili et al. [8] use chaotic maps to change the gravity constant of GSA to improve the ability of exploitation and exploration. Mirjalili et al. [9] archive and use best mass to accelerate the exploitation phase and ameliorate the slow exploitation weakness.

In this paper, we propose a new improved GSA, named RGSA. RGSA modifies the $kbest$ selection of standard GSA. At the base of $kbest$, we introduce revolving operator and define group M, which make the agents not included in $kbest$ have opportunity to exert force to other agents, by this way, GSA's ability of exploration and exploitation is enhanced. Experiments demonstrate that RGSA can improve the performance of GSA.

The rest of this paper is organized as follows: the section "Analyzation of $kbest$ Selection Problem" takes completely analyze of $kbest$ selection problem in GSA. The section "Revolving Gravitational Search Algorithm" explains how to manipulate revolving operator and group M in RGSA to make better performance. A comparative study is presented in section "Experimental results" to test our algorithm. Finally, section 5 gives a conclusion of this paper.

II. Analyzation Of Kbest Selection Problem

The main process of standard GSA is described as follows: GSA models a group of agents scatter in the search space, each agent possesses specific mass and position. Agents' positions are solution for target function and the mass $M_i(t)$ can be calculated as follows [1]:

$$\mathrm{fit}_i(t) = f(x_i(t)), i = 1,2,\dots,N \tag{1}$$

$$\mathrm{best}(t) = \min_{i\in\{1,2,\dots,N\}} \mathrm{fit}_i(t) \tag{2}$$

$$\mathrm{worst}(t) = \max_{i\in\{1,2,\dots,N\}} \mathrm{fit}_i(t) \tag{3}$$

$$m_i(t) = \frac{\mathrm{fit}_i(t) - \mathrm{worst}(t)}{\mathrm{best}(t) - \mathrm{worst}(t)} \tag{4}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{5}$$

Where $\mathrm{fit}_i(t)$ is the function value of agent i at time t.

GSA simulates the gravity force between agents to make the movement of agents. At time t, the force $F_{ij}^d(t)$ and acceleration $a_i^d(t)$ of agent i can be calculated as follows [1]:

$$F_{ij}^d(t) = G(t)\,\frac{M_i(t)\times M_j(t)}{R_{ij}(t)+\epsilon}\left(x_j^d(t) - x_i^d(t)\right) \tag{6}$$

$$F_i^d(t) = \sum_{j\in kbest, j\neq i} \mathrm{rand}_j \times F_{ij}^d(t) \tag{7}$$

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \tag{8}$$

For each iteration, GSA updates the velocity $v_i^d(t)$ and position $x_i^d(t)$ of each object as follows [1]:

$$v_i^d(t+1) = \mathrm{rand}_i \times v_i^d(t) + a_i^d(t), t = 0,1,..,T \tag{9}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), t = 0,1,..,T \quad (10)$$

GSA defines $kbest$ to influence the movement of all other agents. $kbest$ is the set of first K agents with the best fitness value and biggest mass [1]. Only agents from $kbest$ are capable to exert force to all other agents. Therefore, $kbest$ plays an important role in guiding the search direction for whole group.

There is a problem in the *kbest* selection of standard GSA. If the agent, which is nearby the global optimal district, is not included in $kbest$, it may blemish the search process, makes it can't focus search space on global minima in the late stage. Fig. 1 shows a situation of $kbest$ selection.

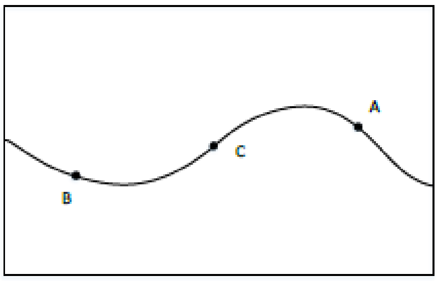


FIGURE I. A SITUATION OF $kbest$ SELECTION

As $K = 2$, agent B, C will be included in $kbest$ and agent A will lose the ability to attract other agents. In this iteration, agent A will be attracted by B, C and moves away from the global minimal district. Then the whole group may converge to local minima in the end.

We take experiment to testify the influence of $kbest$ selection to the performance of GSA.

For example, we apply GSA to optimize Rastrigin function [1] (whose optimum value is 0). Population size is set as 100 and iteration number is 1000. The initial number of agents in $kbest$, K_0, is set as $N, 05N, 0.2N$ to compare with each other. Other parameters are same as [6]. The result of experiment is shown in Fig. 2.

As the convergence curve shows in Fig. 2, the algorithm converges to local minima too early and can't optimize to global minima or nearby solution. As $K_0 = 0.5N$ and $K_0 = 0.2N$, the problem described above is more likely to appear, so the optimize results are worse than $K_0 = N$, which testifies our analyzation.

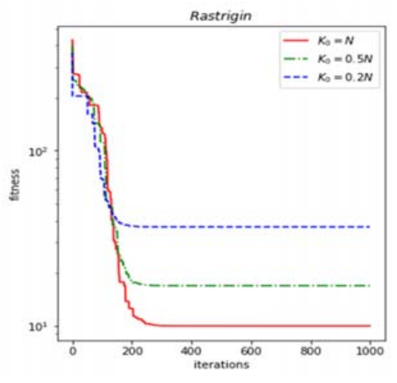


FIGURE II. CONVERGENCE CHARACTERISTICS FOR RASTRIGIN WITH DIFFERENT K_0

Therefore, GSA selects first K agents with the best fitness value included in $kbest$ is not reasonable, that may lead the whole group be trapped in local minima.

III. REVOLVING GRAVITATIONAL SEARCH ALGORITHM

To mitigate this problem, we introduce revolving operator and define group M at the base of GSA. Revolving operator is inspired by satellite phenomenon. In our life, although the earth is much lighter than the sun, the earth still has the moon as its satellite, and the force applied by the earth significantly influences the moon's movement to make it moves around the earth steadily.

Based on this idea, we propose RGSA with revolving operator. Revolving operator builds group M as supplement of $kbest$. As an agent is not included in $kbest$, it still has opportunity to be included in group M, to attract part of other agents move toward it. Which could make potential districts be further exploited and improve the performance of GSA.

Definition of group M. Producing the group M is vital to the effectiveness of revolving operator. For revolving operator, the population of group M needs be controlled.

If there are too many agents in group M, then will be plenty of agents be captured and the influence of $kbest$ will be damaged. So the diversity of whole group can't be guaranteed, and the exploration ability will be depressed. Therefore, group M is defined and built as follows:

Group M is defined as the set of first L agents with the best fitness value whose fitness value is within the range of $(F_k, C]$.

$$F_k(t) = \max_{i \in kbest}\left(fit_i(t)\right) \quad (11)$$

$$C = \begin{cases} F_k(t) \times 10 \text{ if } F_k(t) > 0 \\ F_k(t) \times 0.1 \text{ else} \end{cases} \quad (12)$$

$$L = \lfloor (N - K) \times 0.1 \rfloor \quad (13)$$

Where L is the population of group M, C is the threshold of group M, $F_k(t)$ is the worst fitness value in $kbest$. K is the number of objects in $kbest$, N is the population of whole group.

RGSA needs build group M for every iteration. In each iteration, RGSA sorts agents whose fitness value is smaller than C and larger than F_k in ascending order, and select L smallest agents included in group M or all of them when the population is less than L.

With the supplement of group M, revolving operator modifies the rules of motion in GSA as follows:

1. For the agents included in group M, they are not capable to attract all of other agents but are capable to capture k nearest agents exclusively. Objects could be captured must not be included in group M and $kbest$.

2. If object i is captured by object j, then object i and object j construct a capture pair and the force applied to i is calculated as follows:

$$F_i^d(t) = rand_j \times F_{ij}^d(t) \quad (14)$$

Where $rand_j$ is a uniform random number in the interval [0, 1].

3. Object i will get rid of forces from $kbest$ and it is only influenced by the force from object j, with the value of $v_i^d(t)$, object i is capable to search around object j further more.

After the movement, we get the fitness value of i at $t+1$, $fit_i(t+1)$ and compare it with $fit_j(t)$.

If $fit_i(t+1) \le fit_j(t)$, that means object i searchs to a potential district may bring better solution in the later iterations. The algorithm does not need to remedy for this capture.

However, if $fit_i(t+1) > fit_j(t)$ is satisfied, we need to mutate the position of object i as follows to remedy the misdirection.

$$x_i^d(t+1) = \left(1 - \frac{t}{T}\right) \times x_i^d(t) + \left(\frac{t}{T}\right) \times x_{best}^d(t) + c \times rand_1 \times \left(rand_2 x_{best}^d(t) - x_i^d(t)\right) \quad (15)$$

Where $x_{best}^d(t)$ is the global best agent's position at dth dimension. $rand_1$ is a uniform random number in the interval [-0.5, 0.5], $rand_2$ is a uniform random number in the interval [0, 1]. c is set as 2 [10]. In the early stage of iteration, $x_i^d(t)$ is more important than $x_{best}^d(t)$, and the importance of $x_{best}^d(t)$ promotes as time goes by. That is because the algorithm needs more diversity in the early stage to explore while needs to converge in small search space in the end to exploit.

By importing the revolving operator, one agent may be captured by objects in group M or be attracted by $kbest$. Therefore, the algorithm could search the potential districts further more. In addition, the capture behavior could add stochastic feature to the movement of objects with worse fitness value and improve exploration ability of GSA.

The pseudo code of RGSA is given in Fig. 3.

```
RGSA
1   Search space identification, t = 0
2   Randomized initialization value of $X_i(0), i = 1,2,\ldots,N$
3   Fitness evaluation of agents
4   Updating global optimization result
5   Calculating $M_i(t), i = 1,2,\ldots,N$ based on fitness evaluation of all agents
6   Calculating $C$ and $L$ for group M
7   Selecting agents included in group M for this iteration
8   Traversing agents from group M in ascending order of fitness value to determine capture pairs
9   Calculating $F_i(t), i = 1,2,\ldots,N$ applied by $kbest$ or agent form group M
10  Calculating $a_i(t)$ and $v_i(t+1), i = 1,2,\ldots,N$ based on $v_i(t)$ and $F_i(t)$
11  Updating $X_i(t+1), i = 1,2,\ldots,N$ based on $v_i(t+1)$ and $X_i(t)$
12  Checking fitness value of capture pairs, mutate position for misdirection as Eq. 15
13  Repeat step 3-13 until the stopping criterion is satisfied.
```

FIGURE III. PSEUDO CODE OF RGSA

IV. EXPERIMENTAL RESULTS

To evaluate the performance of our algorithm, we apply it to 10 standard benchmark functions [7]. These benchmark functions are divided into unimodal functions (Table 1) and multimodal functions (Table 2). Where f_{opt} represents the optimum value of the function, S represents the search space, n is the number of dimensions for each function. We set all functions' dimension as 30 except F_9 and F_{10}, their dimensions are set as 2 originally.

TABLE I. UNIMODAL TEST FUNCTIONS

Test function	**S**	$\mathbf{f_{opt}}$
$F_1(X) = \sum_{i=1}^{n} x_i^2$	$[-100,100]^n$	0
$F_2(X) = \max_{i\in\{1,..N\}} \lvert x_i \rvert$	$[-100,100]^n$	0
$F_3(X) = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	$[-100,100]^n$	0
$F_4(X) = \sum_{i=1}^{n} i{x_i}^4 + \mathrm{random}[0,1)$	$[-100,100]^n$	0

TABLE II. MULTIMODAL TEST FUNCTIONS

Test function	**S**	$\mathbf{f_{opt}}$
$F_5(X) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$	$[-32.768,32.768]^n$	0
$F_6(X) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	$[-100,100]^n$	0
$F_7(X) = 10n + \sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i)$	$[-5.12,5.12]^n$	0
$F_8(X) = -\frac{1+\cos\left(12\sqrt{\sum_{i=1}^{n} x_i^2}\right)}{0.5\sum_{i=1}^{n} x_i^2 + 2}$	$[-5.12,5.12]^n$	-1
$F_9(X) = 0.5 + \frac{\sin^2(x_1^2 - x_2^2) - 0.5}{[1+0.001(x_1^2 + x_2^2)]^2}$	$[-100,100]^n$	0
$F_{10}(X) = \left(\sum_{i=1}^{5} i\cos\big((i+1)x_1 + i\big)\right)\left(\sum_{i=1}^{5} i\cos\big((i+1)x_2 + i\big)\right)$	$[-5.12,5.12]^n$	-186.7309

We take a comparison of Standard GSA(SGSA) and Particle Swarm Optimization(PSO) with our method Revolving GSA(RGSA). In all comparison cases, we set population size as 100 and iteration number is 1000 for three algorithms.

In SGSA and RGSA, all parameters are same as [6].

In RGSA, we need to set the threshold C, population L for group M and variable k according to section 3.We set $k = 3$, set C and L as Eq. 12 and Eq. 13.

In PSO, $c_1 = c_2 = 2$ and inertia factor ω is decreasing linearly from 0.9 to 0.2[6]

The experiment takes 30 independent runs for each functions, average best-so-far value and standard deviation of best solution for each algorithms are reported to evaluate their performances.

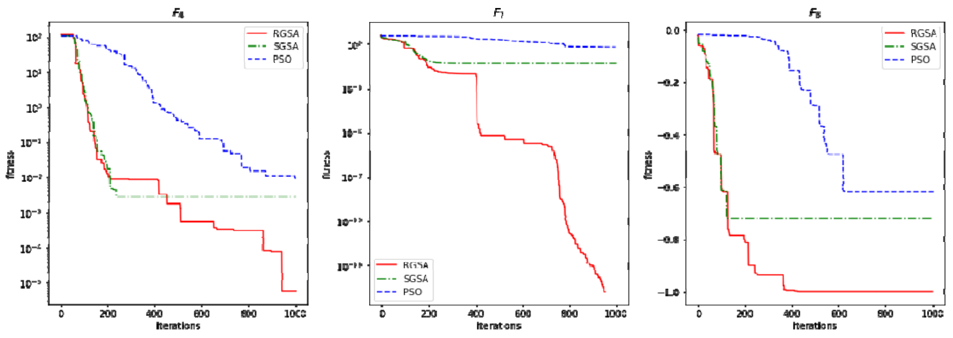
For unimodal functions, the convergence rate of the algorithm is more important than the final result of optimization [6]. Table 3 illustrates experiment results of unimodal functions, in F_1, F_2, SGSA is capable to explore and exploit, while RGSA achieves better and more steady optimization result. In F_3, F_4, SGSA is weak at optimizing while RGSA shows powerful ability and outperforms the other two methods significantly. The convergence result for F_4 is shown in Fig. 4(a).

TABLE III. COMPARISON OF BENCHMARK FUNCTIONS IN TABLE 1

		SGSA	RGSA	PSO
$F_1(X)$	Average best-so-far	4.3177263049934e-18	**1.429936487680128e-18**	1.3344422130474104e-12
	Std best-so-far	6.1071369170286e-19	1.965725671805461e-19	2.009106599719646e-12
$F_2(X)$	Average best-so-far	1.0433990526131e-09	**8.70905854753924e-10**	6.9237216103477
	Std best-so-far	1.9417308424888e-10	1.83027215063579e-10	2.1565241328409
$F_3(X)$	Average best-so-far	29.9530999335856	**6.862353867559747e-11**	401.274391004069
	Std best-so-far	14.651631210782	2.8555190276973854e-10	210.20507900135
$F_4(X)$	Average best-so-far	0.0072756040121764	**5.7807057148991e-05**	2.3071157352334
	Std best-so-far	0.002442448250125	4.316578058835e-05	2.84668957593991

TABLE IV. COMPARISON OF BENCHMARK FUNCTIONS IN TABLE 2

		SGSA	RGSA	PSO
$F_5(X)$	Average best-so-far	1.6125828672386e-09	**9.5754749314854e-10**	7.126224104740e-07
	Std best-so-far	2.4018457109637e-10	1.17363694708085e-10	1.298367431164e-06
$F_6(X)$	Average best-so-far	9.8754347749668e-22	**3.55655776367050e-22**	0.21318770444168
	Std best-so-far	1.702834727044e-22	4.36667401289107e-23	0.76942368215734
$F_7(X)$	Average best-so-far	8.95463151383962	**0**	113.2720682148396
	Std best-so-far	2.46407256828827	0	29.0524834411635
$F_8(X)$	Average best-so-far	-0.70297602710363	**-1.0**	-0.57138999212498
	Std best-so-far	0.073947543018825	1.16441173166909e-16	0.165444665773313
$F_9(X)$	Average best-so-far	0.0008716810987575	**0**	**0**
	Std best-so-far	0.001415161429419	0	0
$F_{10}(X)$	Average best-so-far	-184.84326252598234	-186.6006149939825	**-186.73090883102392**
	Std best-so-far	1.3512149830979387	0.1244059566732892	1.2710574864626037e-14



(a) (b) (c)

FIGURE IV. CONVERGENCE CHARACTERISTICS FOR F_4, F_7,F_8

Multimodal functions having many local minima makes them comparatively harder to optimize. Table 4 illustrates the experiment results of multimodal functions. RGSA outperforms SGSA and PSO in most functions except F_{10}. As Table 4 illustrates, in F_5, F_6, both of SGSA and RGSA are capable to explore and exploit, and RGSA preforms slightly better than SGSA. In F_7, F_8, RGSA significantly outperforms the other two methods and exactly converges to global optimum. In F_9, both of RGSA and PSO get stable global optimum. In F_{10}, the search space is much smaller than other functions, so the exploitation ability will be in charge, and PSO has better solution than SGSA and RGSA. The convergence results for F_7 and F_8 are shown in Fig. 4(b) (c). Moreover, we can conclude from Fig. 4 that RGSA is capable to escape from local minima in the search process.

Results of unimodal and multimodal functions confirm that the way to keep more objects possess the chance to attract other objects is a useful way to improve the ability of GSA. That is because in the early stage, the revolving operator adds stochastic feature to worse agents to search potential districts and enforce the ability of exploration. In the late stage, the revolving operator diversifies search directions within small search space and mutates wrong attempts according to global best agent, which makes the convergence more efficient to enforce the ability of exploitation.

V. SUMMARY

This paper analyzes the problem of $kbest$ selection in GSA, and then proposes RGSA with revolving operator. At the base of $kbest$, RGSA introduces revolving operator and defines group M, which make the agents not included in $kbest$ have opportunity to exert force to other agents. In the early stage, RGSA makes potential districts be searched further to improve the ability of exploration. In the late stage, RGSA enhances convergence rate, by mutating position for wrong capture behaviors, to improve the ability of exploitation. Experiments confirm RGSA is superior to GSA.

ACKNOWLEDGEMENT

This work was supported by the National Key Research and Development Program of China undertook by the Academy of Broadcasting Science (No.2017YFC0806200, No.2017ZX03001027).

REFERENCES

[1] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: a gravitational search algorithm[J]. Information sciences, 2009, 179(13): 2232-2248.

[2] Mirjalili S A, Hashim S Z M, Sardroudi H M. Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm[J]. Applied Mathematics and Computation, 2012, 218(22): 11125-11137.

[3] Marzband M, Ghadimi M, Sumper A, et al. Experimental validation of a real-time energy management system using multi-period gravitational search algorithm for microgrids in islanded mode[J]. Applied energy, 2014, 128: 164-174.

[4] Rezaei M, Nezamabadi-Pour H. Using gravitational search algorithm in prototype generation for nearest neighbor classification[J]. Neurocomputing, 2015, 157: 256-263.

[5] Barani F, Mirhosseini M, Nezamabadi-Pour H. Application of binary quantum-inspired gravitational search algorithm in feature subset selection[J]. Applied Intelligence, 2017, 47(2): 304-318.

[6] Sarafrazi S, Nezamabadi-Pour H, Saryazdi S. Disruption: a new operator in gravitational search algorithm[J]. Scientia Iranica, 2011, 18(3): 539-548.

[7] Qian K, Li W, Qian W. Hybrid Gravitational Search Algorithm Based on Fuzzy Logic[J]. IEEE Access, 2017, 5: 24520-24532.

[8] Mirjalili S, Gandomi A H. Chaotic gravitational constants for the gravitational search algorithm[J]. Applied Soft Computing, 2017, 53: 407-

419.Chen Z, Yuan X, Tian H, et al. Improved gravitational search algorithm for parameter identification of water turbine regulation system[J]. Energy conversion and management, 2014, 78: 306-315.

[9] Mirjalili S, Lewis A. Adaptive gbest-guided gravitational search algorithm[J]. Neural Computing and Applications, 2014, 25(7-8): 1569-1584.

[10] Bi X, Diao P, Xiao J. Gravitational search algorithm with mixed strategy[J]. Systems Engineering and Electronics, 2014, 36(11): 2308-2413