Fireworks Algorithm with Covariance Mutation

Chao Yu and Ying Tan*
The Key Laboratory of Machine Perception and Intelligence (Ministry of Education),
Department of Machine Intelligence, School of Electronics Engineering and Computer Science,
Peking University, Beijing, China, 100871
Email: {chaoyu, ytan}@pku.edu.cn

***Abstract*—Fireworks algorithm is a novel swarm intelligence algorithm for solving optimization problems - the latest versions include the adaptive fireworks algorithm and the dynamic fireworks algorithm. However, the mutation operator in the former algorithm was ineffective, whereas there was no mutation operator available in the latter algorithm. In this paper, a mutation operator is proposed, dubbed as the covariance mutation (CM) operator. The CM operator utilizes the information of the sparks with better fitness values to generate potential sparks for finding the optima of functions with higher possibility. Therefore, we proposed the fireworks algorithm with covariance mutation (FWACM) and compared it with the most advanced fireworks algorithms. The experimental results show that FWACM is a significant improvement for fireworks algorithms.**

I. Introduction

To solve the function optimization problems effectively, fireworks algorithm (FWA) is first proposed in 2010 by Tan and Zhu [1]. This version of FWA is called the conventional FWA. The conventional FWA works very well when the optima are near the original point. The performance of the conventional FWA has been tested and ranked at sixth out of twelve algorithms by Bureerat in 2011 [2]. Yet, the conventional FWA's development is at the infancy stage and have its own weaknesses. Firstly, the explosion amplitude is near zero in some conditions, which creates difficulty in moving forward for the affected algorithm; secondly, the Gaussian mutation has been proven useless in FWA, which lengthens its computational time output, but yields very little result; thirdly, the selection strategy in conventional FWA is deficient. The most time consuming place in the algorithm is the selection strategy, rather than more important explosion operator.

Many scholars quickly pinpointed drawbacks of the conventional FWA and proposed various versions of FWA. In 2013, Zheng, et al. put forward an enhanced fireworks algorithm (EFWA) to improve the conventional FWA in the following five aspects [3]: 1) the minimal explosion amplitude was given to prevent the conventional FWA from producing useless sparks; 2) the explosion offsets were different in each dimension to make the population diverse, whereas in the conventional FWA, the explosion offsets were the same in each dimension. 3) the mapping operator was improved by randomly mapping the sparks back into the feasible space; 4) a new Gaussian mutation operator was introduced by utilizing the information of the current best fireworks; 5) a time saving selection operator was utilized. Moreover, Liu, et al. improved EFWA by balancing the local and global search ability [4]. Zheng, et al. proposed dynamic search FWA (dynFWA) by adjusting the amplitudes of the firework with the best fitness value [5]. Yu, et al. presented a new algorithm for FWA with differential mutation (FWA-DM) [6], [7]. The new algorithm improved the Gaussian mutation by introducing a differential mutation operator and performed well on single objective optimization functions. Li, et al. proposed a new algorithm named adaptive FWA (AFWA), aiming to make the explosion amplitudes adaptable [8]. Zheng, et al. introduced differential operators and created a hybrid FWA [9]. Zhang, et al. studied the biogeography information and proposed a hybrid FWA to solve optimization problems [10]. Yet, the global convergence ability of FWA was analyzed and the time complexity was proven by Liu, et al [11].

FWA and its variants have been applied to many fields successfully [12]. Zheng, et al. successfully utilized FWA to solve the multi-objective optimization problems [13]. Ding, et al. pioneered the process to use FWA on graphic processing unit (GPU) [14]. Gao and Diao presented a cultural fireworks algorithm and used it to design digital filters [15]. Janecek and Tan used FWA to solve the non-negative matrix factorization [16]–[18]. He, et al. utilized FWA to filter digital and information spam [19]. Thus, FWA can be applied to various fields and effectively solve practical problems.

Despite the achievements of FWA in the applications, the algorithm still had flaws. For instance, the Gaussian mutation in the FWA variants, e.g. AFWA and FWA-DM, all largely depended on the premise that the best firework was analyzed. As such, the information of the additional or outlying sparks is wasted. In this paper, the information provided from the better sparks and from the best firework will all be used. Since there is more useful information than from previous proposed FWA variants, it is important to use the information wisely. To reach this goal, a covariance mutation is introduced, designed to improve the performance of FWA.

The rest of the paper is organized as follows. The details of AFWA will be introduced in section II; the mutation operator of AFWA is analyzed in section III, where the disadvantages of the operator are pointed out; section IV proposes the covariance mutation to overcome the shortcomings of the mutation operator in AFWA. Experimental environments and results are given in section V, after which the discussion is given and followed by a summary.

* Prof. Y. Tan is the corresponding author.

978-1-4799-7492-4/15/$31.00

II. Adaptive Fireworks Algorithm

AFWA provided a feasible way to find the optimum of the benchmark functions. AFWA simulates the way fireworks explode in the night sky by producing proximal sparks around each illumination. In AFWA, two kinds of sparks are generated, as 'explosion sparks' and 'Gaussian sparks', respectively. Both the fireworks and the sparks are evaluated by the benchmark functions. A selection strategy is carried out and a few sparks are chosen for the next generation. This process of producing and choosing sparks is continued until the algorithm reaches the maximum function evaluation times or the evaluation value satisfies the accuracy requirement.

To be more specific, AFWA can be represented as explosion operator, mutation operator, mapping rules and selection strategy.

1) Explosion Operator

To begin with, N individuals are generated randomly according with uniform distribution in the search space. These N individuals form the first generation and their fitness values are evaluated right after they have been produced. In AFWA, the N individuals in the first generation and those individuals selected by the selection strategy, are treated as fireworks. Meanwhile, the individuals produced by explosion operator and Gaussian operator are refer to sparks. In explosion operator, each firework explodes randomly within a precise amplitude. For example, in a two-dimensional space, the firework is the center of the explosion and the sparks lie in a circle within an amplitude. To make it more vivid, when the explosion takes place in a 3-dimensional space, it is more likely that the sparks are distributed in a sphere. This phenomena extends to higher dimensions as the fireworks produce sparks in a hypersphere.

2) Mutation Operator

In the conventional FWA, the mutation operator is called as Gaussian mutation. However, the sparks produced by AFWA are not according with Gaussian distribution. The sparks are produced range from less suitable fitting firework to the best firework in the current generation. In this way, the produced sparks have a direction by using the statistical information of the most suitable or fitting firework.

3) Mapping Rules

When evaluating the benchmark functions, the search space is limited. However, as the explosion takes place in all directions within calculated amplitudes, it is possible that the produced sparks lie out of the boundaries. To deal with this kind of problem, the mapping rules are designed. If there are any outlying sparks, a new random spark is generated to replace the one that is out of the boundary. By doing so, all the sparks fall in the feasible space and the efficiency of AFWA is ensured since no sparks are outside the boundary.

4) Selection Strategy

In each generation, a display of sparks is generated by the explosion and mutation operators. In selection strategy, N individuals are selected and passed down to the next generation. In AFWA, the selection strategy is simple and easy to understand. The best individual, selected from both the fireworks and the sparks, is kept for the next generation. The other $N-1$ individuals are selected randomly from the entire population.

The four parts aforementioned are repeated until the termination conditions are met, which are typically the accuracy of the solution and the maximum times of function evaluations.

The operators of AFWA are introduced as follows.

First, the explosion operator needs to be determined. When the fireworks explode, two parameters need to be obtained. These are the number of the sparks and the amplitude of the explosion. In AFWA, the number of sparks is acquired the same as the conventional FWA.

Parameter S_i donates the number of sparks for the i^{th} firework.

$$S_i = \hat{S} * \frac{Y_{\max} - f(x_i) + \varepsilon}{\sum_{i=1}^{N} (Y_{\max} - f(x_i)) + \varepsilon} \tag{1}$$

In the equation, $\hat{S}$ is the total number of sparks in a generation. Parameter Y_{max} is the worst fitness value of all the fireworks and the sparks. Function $f(x_i)$ represents the fitness value for the individual x_i, while the parameter ε is used to avoid the denominator from becoming zero. Besides, there are boundaries for the parameter S_i, as N_{min} and N_{max}, respectively.

In AFWA, the firework with the best fitness value is called the 'core' firework. The amplitudes for the core firework and the other fireworks are different. Let parameter $A(g)$ represents the amplitude of the core firework in generation g and parameter A_i donates the amplitude for the i^{th} firework.

$$A(g+1) = \begin{cases} UB - LB, & g = 0 \quad or \quad \forall f(s_i) < f(X) \\ 0.5 * [\lambda \cdot \|s_i - s^*\|_\infty + A(g)], otherwise \end{cases} \tag{2}$$

In the equation, $A(g+1)$ is the amplitude for the next generation. Parameter UB and LB are the upper and lower bounds for individuals. Parameter s_i denotes the position of any sparks, s^* is the best spark of the current generation and X stands for the core firework. Parameter λ is used for tuning the amplitudes.

$$A_i = \hat{A} * \frac{f(x_i) - Y_{\min} + \varepsilon}{\sum_{i=1}^{N} (f(x_i) - Y_{\min}) + \varepsilon} \tag{3}$$

In the equation, $\hat{A}$ is a parameter controlling the maximum amplitude of the explosion. Parameter Y_{min} stands for the worst fitness value of the current generation. Function $f(x_i)$ represents the fitness value for the individual x_i, and the meaning of parameter ε is the same as aforementioned.

The details of explosion operator are given below.

In the explosion operator, $rand(a, b)$ means to generate a random number with uniform distribution between a and b. Parameter N represents the number of fireworks and parameter D is the number of dimension. Position s_{ij} represents the position of individual s_i in j^{th} dimension and X_{ij} is the

Algorithm 1 Explosion Operator in AFWA

for $i = 1 \to N$ **do**
  for $j = 1 \to D$ **do**
    if $rand(0,1) \leq 0.5$ **then**
      $s_{ij} = X_{ij} + rand(-1,1) \cdot Ampl$
    end if
    if $s_{ij} < LB$ or $s_{ij} > UB$ **then**
      $s_{ij} = LB + rand(0,1) \cdot (UB - LB)$
    end if
  end for
end for

position of the selected firework X_i in its j^{th} dimension. $Ampl$ is the amplitude for the selected firework. $A(g+1)$ is used for the core firework and A_i is used for the other fireworks. Parameter UB and LB represent upper and lower bounds, respectively.

Secondly, the mutation operator is shown. The mutation operator is called 'Gaussian mutation' in FWA. However, the Gaussian distribution in AFWA is not as strong as in FWA, since only a few of random numbers with Gaussian distribution are generated. The mutation operator generates sparks between the best firework and another selected firework in current generation, aiming to utilize the information of the best firework and thus to improve the exploitation of AFWA.

The details of the mutation operator are given below.

Algorithm 2 Mutation Operator in AFWA

for $i = 1 \to NG$ **do**
  for $j = 1 \to D$ **do**
    $s_{ij} = X_{ij} + randn(0,1) \cdot (X_{best,j} - X_{ij})$
    if $s_{ij} < LB$ or $s_{ij} > UB$ **then**
      $s_{ij} = LB + rand(0,1) \cdot (UB - LB)$
    end if
  end for
end for

In the mutation operator, NG is the number of Gaussian mutation sparks and D stands for the number of function dimensions. s_{ij} is the position of the i^{th} sparks in j^{th} dimension, X_{ij} is the position of the selected firework X_i in its j^th dimension. $X_{best,j}$ is the position of the best firework X_{best} in its j^{th} dimension. $randn(a,b)$ is normal distribution with mean a and standard deviation b, while $rand(c,d)$ is uniform distribution and within the range c and d. Parameter UB and LB represent upper and lower bounds, respectively.

Thirdly, to give an overall view of AFWA, the process of AFWA is given in Alg. 3.

AFWA uses an adaptive amplitude for the core firework and overcomes the disadvantages of the conventional FWA. In the conventional FWA, the amplitudes of the core firework are close to zero, which led the explosion operator produces useless sparks and wasted function evaluation times. In AFWA, the amplitudes can adjust themselves and make the explosion operator produce more helpful sparks.

Algorithm 3 The process of AFWA

randomly generates N fireworks according with uniform distribution.
evaluate the fitness values of the N fireworks
while terminate condition not met **do**
  calculate S_i, $A(g+1)$ and A_i for each firework
  generate explosion sparks according with explosion operator
  generate Gaussian sparks according with mutation operator
  evaluate all the fitness values of both kinds of sparks
  keep the best individual and randomly select the other $N-1$ individuals
end while
return the best individual and its fitness value

III. Analysis of the Mutation in AFWA

There are three disadvantages for the mutation operator in AFWA. First of all, only the information of the fireworks is taken into consideration. However, the mutation operator can use the information of the sparks generated by explosion operator. If these valuable information is not used in mutation operator, it will cause a lot of information loss. Secondly, the scale factor in mutation operator is not set properly in AFWA. Even when there is a random normal distribution number that is used for all generation, it is better to use a larger-scale factor at the initial generations to ensure that the exploration ability of AFWA and a smaller factor at the last generations to enhance AFWA's exploitation ability. Thirdly, the mutation operator in AFWA is impractical during the global search, as the sparks generated by mutation operator tend to utilize the best firework while searching the local space. In fact, all the sparks that are generated in mutation operator are attracted by the current best firework. Therefore, the mutation operator focuses on local search ability and doesn't balance both of the global and local search.

The mutation operator in AFWA requires improvement in order to overcome the aforementioned disadvantages. If the information of the sparks and fireworks, rather than fireworks only, can be used, the possibility of finding better sparks is increased. Even if the scale factor is set properly, only the information from the best firework is utilized, whereas allowing an influx of more information can be used to improve the performance of mutation operator. Therefore, it is important to change the mutation operator in AFWA and avoid its shortcomings.

IV. Covariance Mutation

A covariance mutation is proposed to deal with the disadvantages of mutation operator in AFWA. The covariance mutation utilized the information of better sparks in each generation, avoiding to concentration on using only the information for the best spark. The mean value and the covariance matrix of the better sparks are used and the mutation operator generates new sparks according with Gaussian mutation.

First of all, the mean value of the better explosion sparks is calculated. Let λ represents the number of sparks generated from a firework. Parameter μ is the number of better sparks

selected from the λ sparks. For any μ sparks, the mean value m is calculated by following equation.

$$m = \sum_{i=1}^{\mu} w_i x_i, \tag{4}$$

where x_i are the selected better individuals and the index i varied from 1 to μ. Parameter w_i are the weight for each spark and are calculated in Eq. 5.

$$w_i = 1/\mu \quad (i = 1, 2, ..., \mu). \tag{5}$$

Note that m is the mean value of the μ individuals, not the λ individuals.

Secondly, the covariance matrix C is obtained. Equation 6 shows the details of the C.

$$C = \begin{pmatrix} cov(v_1, v_1) & cov(v_1, v_2) & \cdots & cov(v_1, v_D) \\ cov(v_2, v_1) & cov(v_2, v_2) & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathrm{cov}(v_D, v_1) & \cdots & \cdots & cov(v_D, v_D) \end{pmatrix}, \tag{6}$$

where $cov(v_i, v_j)$ is the covariance of vector v_i and v_j. Vector v_i represents the i^{th} dimension of the sparks and D is the number of dimensions.

$$\mathrm{cov}(a, b) = \frac{\sum_{i=1}^{\mu} (a_i - \bar{A})(b_i - \bar{B})}{\mu} \tag{7}$$

In Eq. 7, $\bar{A}$ and $\bar{B}$ represent the mean values of λ sparks in different dimensions, rather than μ sparks. Moreover, the denominator is not $\mu - 1$, which makes the meaning of $cov(a, b)$ slightly different from the covariance.

Thirdly, Gaussian sparks are produced according with Gaussian distribution by mean value m and covariance matrix C. The produced sparks can be called as Gaussian mutation sparks, or Gaussian sparks, or mutation sparks for convenience. This paper calls the sparks as Gaussian sparks.

As shown in Fig. 1, the contour lines are drawn as curves. The better sparks lie within the ellipsoid area of dotted lines and the black dot represents the center of the sparks. If the Gaussian sparks are generated according with $N(m, C)$, they will be produced within the boundary of solid line ellipsoid. The two ellipsoid area are nearly vertical and the possibility of finding useful sparks is increased. It can be seen from Fig. 1 that the Gaussian sparks produced by covariance mutation are now with a more possible direction to find the optima.

The algorithm of the covariance mutation is given in Alg. 4.

To further analyzed the covariance mutation, a simple function $f(x) = \sum_{i=1}^{2} x_i^2$ is taken as an example.

To begin with, the contour lines of function $f(x)$ are drawn in Fig.2(a). Secondly, 50 sparks are generated according with

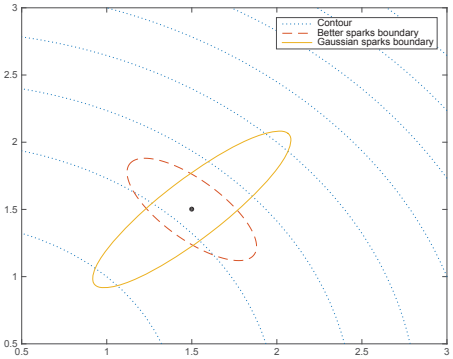

Fig. 1. The Gaussian sparks distribution with $N(m, C)$.

Algorithm 4 Covariance Mutation

1: **for** $i = 1 \rightarrow N$ **do**
2: calculate the mean value m according to Eq. 4
3: obtain the covariance matrix C according to Eq. 6
4: generate Gaussian sparks with $N(m, C)$
5: **end for**

uniform distribution and marked as dots and shown in Fig.2(b), where $1 < x_i < 3 \quad (i = 1, 2)$. Thirdly, the better sparks are illustrated in Fig.2(c). The sparks with better fitness values are marked with black 'x' and the sparks with worse fitness values remain unchanged as dots. Lastly, 30 Gaussian sparks are generated according with Alg.4 and represented as sign '+' in Fig.2(d). Note that a few Gaussian sparks beyond the boundary of Fig.2(d) are not shown, since drawing those sparks will make Fig.2(d) too small to see.

It can be seen from Fig. 2 that the covariance mutation operator produces Gaussian sparks nearly at the direction of the gradient of the function, which makes the new algorithm useful at finding the local optimum. This example can be extended to higher dimensions, such as 30, 50 and 100.

Experiments are carried out in four 30-dimensional functions to see how many better sparks are found by mutation operator and explosion operator. Four types of functions are used, as unimodal, multimodal, hybrid and composition functions. Each type contains one function taken as the function 1, 3, 6 and 9 from the section of experiments. The algorithm runs for 51 times and during each run, the number of better sparks generated by each operator is recorded. Table I shows the average better sparks that are produced by mutation operator and explosion operator, separately.

TABLE I. THE COMPARISON OF EXPLOSION AND MUTATION OPERATORS

Function No.	Explosion Sparks	Gaussian Sparks
1	1595	1095
3	152	159
6	1501	1086
9	60	51

To have a fair comparison, the number of sparks produced by each operator in a generation needs to be considered. When 200 explosion sparks are generated, 30 Gaussian sparks are produced. Therefore, if the better sparks produced by the mutation operator are 3 over 20 compared with explosion oper-

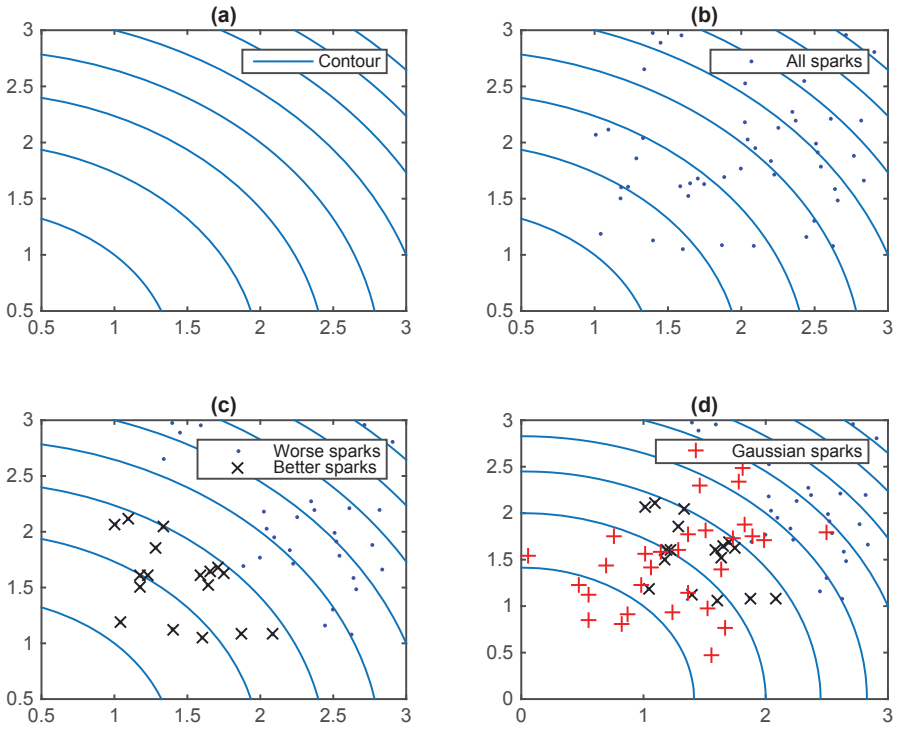

Fig. 2. The process of generating Gaussian sparks by covariance mutation.

ator, the mutation operator is effective. From the experimental results in Table I, the mutation operator is effective on all of the four functions.

V. Fireworks Algorithm with Covariance Mutation

A. FWACM Description

To give an overall look on the new algorithm, the descriptions of FWACM are given in this section. When FWACM begins, N fireworks are generated according with uniform distribution. Their fitness values are first evaluated. Then, the numbers of the sparks for each firework are calculated. For the core firework, the adaptive amplitudes are calculated. For the other fireworks, the amplitudes are calculated by considering the fitness values of each firework. At this time, the explosion sparks are generated and N patches of sparks are produced. For each patch, the mean value and the covariance matrix of better sparks are obtained for mutation operator. In FWACM, the mutation operator is called as covariance mutation. The sparks generated by covariance mutation obey the normal distribution, which is also Gaussian distribution. Therefore, the sparks can be called as Gaussian sparks. In this way, there are N patches and each patch contains one firework, some explosion sparks and several Gaussian sparks. The best individual in each patch is selected for the next generation. The algorithm continues to produce explosion sparks, generates Gaussian sparks and selects the N individuals for the next generation until the termination conditions are met.

To give a summarize of FWACM, the process of FWACM is given below.

Algorithm 5 The process of FWACM

```
randomly generates N fireworks according with uniform distribution.
evaluate the fitness values of the N fireworks
while terminate condition not met do
    calculate S_i, A(g+1) and A_i for each firework
    generate explosion sparks according with explosion operator
    calculate m and covariance matrix C
    generate Gaussian sparks according with N(m, C)
    evaluate all the fitness values of explosion and Gaussian sparks
    select N individuals for the next generation
end while
return the best individual and its fitness value
```

B. FWACM Analysis

In the explosion operator, parameters S_i, $A(g+1)$ and A_i are set the same as AFWA. The parameters S_i and A_i reflect the information exchange of FWACM, whereas $A(g+1)$ is an adaptive amplitude for the core firework and relevant to the amplitude of the $A(g)$ in g generation . Parameter λ is used to tuning the amplitude of the core firework.

To perform the mutation operator, the mean value vector m and covariance matrix C are obtained. The vector m contains D elements, where D is the dimension of the optimization problem. The matrix C is a D-by-D symmetric positive semi-definite matrix, represents a calculated covariance matrix of better sparks. Both m and C are used to generate sparks in covariance mutation.

N individuals are selected for the next generation. The way to select sparks for the next generation is different from AFWA, whereas $N-1$ individuals are selected randomly in AFWA and those individuals are selected from each patch of sparks in FWACM. After circles of generation, a solution will be given.

The running time for FWACM is longer than AFWA and dynFWA. Compared with AFWA, FWACM uses CM operator to replace the GM operator. The CM operator calculates covariance matrix to generate Gaussian mutation sparks, while GM operator produces sparks by a simple arithmetic operation. As a result, the running time for FWACM is longer than AFWA. Moreover, AFWA is more time consuming than dynFWA, as the adaptive amplitudes are obtained and tuned in AFWA, while only a multiplication operation is taken place in dynFWA to control the explosion amplitudes.

VI. Experiments

The benchmark functions are first introduced, followed by the parameter settings. The experimental results are given at last, which include the mean values, the rank of all three algorithms and t-test results between FWACM and AFWA.

A. Benchmark Functions

FWACM is used to find the global optimum values of 15 benchmark functions from the CEC'15 competition [20]. The names and attributes of the functions are as follows.

Unimodal Functions:
1 Rotated High Conditioned Elliptic Function
2 Rotated Bent Cigar Function

Multimodal Functions:
3 Shifted and Rotated Ackley's Function
4 Shifted and Rotated Rastrigin's Function
5 Shifted and Rotated Schwefel's Function

Hybrid Functions:
6 Hybrid Function 1
7 Hybrid Function 3
8 Hybrid Function 5

Composition Functions:
9 Composition Function 1
10 Composition Function 2
11 Composition Function 3
12 Composition Function 4
13 Composition Function 5
14 Composition Function 6
15 Composition Function 7

B. Parameters Settings

The dimension D for the functions is 30. Each experiment runs 51 times and during each run, the function is evaluated for 10000*D times. The actual used parameter values are set as follows. Parameter λ, $\hat{S}$ and $\hat{A}$ are set as 1.3, 200 and 100, as in [8]. The number of the better individuals is set as 50% percent of the number of selected individuals. The population size N is set as 5 and the lower and upper bounds of the number of explosion sparks N_{min} and N_{max} are set as 100 and 4, respectively. The number of Gaussian sparks equals to D.

C. Experimental Results

The experimental platform is Matlab 2014a and the program is running on a Windows 8 operating system.

The mean errors and standard deviations are presented in Table II. The numbers in bold represent the best solution of all three algorithms. Note that the best solutions for function 3 and 7 are obtained dynFWA, though FWACM appears to have the same values when the results are accuracy to two decimal places.

TABLE II. Mean errors of the algorithms

Function No.	dynFWA	AFWA	FWACM
1	8.87E+05	1.18E+06	**8.53E+05**
2	4.89E+03	4.02E+03	**3.48E+03**
3	**2.00E+01**	2.00E+01	2.00E+01
4	1.29E+02	1.46E+02	**1.06E+02**
5	3.55E+03	3.30E+03	**3.22E+03**
6	5.41E+04	5.65E+04	**4.60E+04**
7	**1.44E+01**	1.62E+01	1.44E+01
8	4.66E+04	4.30E+04	**3.46E+04**
9	1.36E+02	1.39E+02	**1.11E+02**
10	**5.21E+04**	5.94E+04	7.72E+04
11	7.60E+02	7.87E+02	**4.84E+02**
12	1.21E+02	1.31E+02	**1.13E+02**
13	2.80E-02	**2.77E-02**	3.66E-02
14	4.50E+04	**4.46E+04**	4.83E+04
15	**1.00E+02**	**1.00E+02**	**1.00E+02**

The ranks of the algorithms are shown in Table III.

TABLE III. The rank of the algorithms

Function No.	dynFWA	AFWA	FWACM
1	2	3	1
2	3	2	1
3	1	2	2
4	2	3	1
5	3	2	1
6	2	3	1
7	1	3	2
8	3	2	1
9	2	3	1
10	1	2	3
11	2	3	1
12	2	3	1
13	2	1	3
14	2	1	3
15	1	1	1
Mean	1.93	2.27	1.53

The experimental results of t-test between AFWA and FWACM are given in Table IV. T-test is used to judge whether two populations are significantly different when both of them follow a normal distribution. The null hypothesis is the means of the two populations are equal. Then a *p-value* is calculated. The null hypothesis is rejected if the *p-value* is lower than a threshold, which is set as 0.05 in this paper. The two populations are significantly different if the null hypothesis is rejected.

VII. Discussion

Table II show that FWACM outperformed the other comparison algorithms. Comparing with AFWA, FWACM performed worse on function 5, 10, 13 and 14. This meant that FWACM performed well on unimodal functions and hybrid functions, yet was worse than AFWA on most composition functions. When compared with dynFWA, FWACM outperformed worse on function 3, 7, 10, 13 and 14. This further proved that FWACM excelled on unimodal functions, as the

TABLE IV. T-TEST ON AFWA AND FWACM

Function No.	H	p
1	1	**0.000889573**
2	0	0.426500506
3	0	0.369812089
4	1	**0.000000418**
5	0	0.551322781
6	0	0.078626171
7	0	0.260082489
8	1	**0.036691109**
9	1	**0.013407029**
10	0	0.082490436
11	1	**0.000000000**
12	1	**0.000360926**
13	1	0.000000081
14	1	0.000824998
15	NaN	NaN

covariance mutation increased the local search ability of the FWACM.

Table III ranked all the three algorithms. It was obvious that with covariance mutation operator, FWACM ranked the first and was the best algorithm. A t-test was made to see if FWACM was significantly better than AFWA on 15 functions. The $p-values$ in bold indicated that FWACM was significantly better than AFWA. The experimental results on 6 functions were found to be improved while on 2 functions were found to be worse. The other function were of no significant statistical differences.

VIII. CONCLUSION

Addressing to overcome the disadvantages of AFWA, a new mutation operator called as covariance mutation is introduced. This operator produces sparks according with Gaussian distribution and helps the new algorithm FWACM better find the optima. The CEC 2015 competition problems are used to test the performance of FWACM. From the experimental results, FWACM outperforms AFWA and dynFWA on most functions. Therefore, the covariance mutation is useful and FWACM is able to solve function optimization problems effectively.

ACKNOWLEDGMENT

This work was supported by the Natural Science Foundation of China (NSFC) under grant no. 61375119, 61170057 and 60875080, and partially supported by National Key Basic Research Development Plan (973 Plan) Project of China with grant no. 2015CB352300.

REFERENCES

[1] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Advances in Swarm Intelligence*, pp. 355–364, Springer, 2010.

[2] S. Bureerat, "Hybrid population-based incremental learning using real codes," in *Learning and Intelligent Optimization*, pp. 379–391, Springer, 2011.

[3] S. Zheng, A. Janecek, and Y. Tan, "Enhanced fireworks algorithm," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pp. 2069–2077, IEEE, 2013.

[4] J. Liu, S. Zheng, and Y. Tan, "The improvement on controlling exploration and exploitation of firework algorithm," in *Advances in swarm intelligence*, pp. 11–23, Springer, 2013.

[5] S. Zheng, A. Janecek, J. Li, and Y. Tan, "Dynamic search in fireworks algorithm," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 3222–3229, IEEE, 2014.

[6] C. Yu, L. Kelley, S. Zheng, and Y. Tan, "Fireworks algorithm with differential mutation for solving the cec 2014 competition problems," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 3238–3245, IEEE, 2014.

[7] C. Yu, J. Li, and Y. Tan, "Improve enhanced fireworks algorithm with differential mutation," in *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*, pp. 264–269, IEEE, 2014.

[8] J. Li, S. Zheng, and Y. Tan, "Adaptive fireworks algorithm," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 3214–3221, IEEE, 2014.

[9] Y. Zheng, X. Xu, H. Ling, and S. Chen, "A hybrid fireworks optimization method with differential evolution operators," *Neurocomputing*, 2012.

[10] B. Zhang, M.-X. Zhang, and Y.-J. Zheng, "A hybrid biogeography-based optimization and fireworks algorithm," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 3200–3206, IEEE, 2014.

[11] J. Liu, S. Zheng, and Y. Tan, "Analysis on global convergence and time complexity of fireworks algorithm," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pp. 3207–3213, IEEE, 2014.

[12] Y. Tan, C. Yu, S. Zheng, and K. Ding, "Introduction to fireworks algorithm," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 4, no. 4, pp. 39–70, 2013.

[13] Y. J. Zheng, Q. Song, and S. Y. Chen, "Multiobjective fireworks optimization for variable-rate fertilization in oil crop production," *Applied Soft Computing*, vol. 13, no. 11, pp. 4253–4263, 2013.

[14] K. Ding, S. Zheng, and Y. Tan, "A gpu-based parallel fireworks algorithm for optimization," in *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, pp. 9–16, ACM, 2013.

[15] H. Gao and M. Diao, "Cultural firework algorithm and its application for digital filters design," *International Journal of Modelling, Identification and Control*, vol. 14, no. 4, pp. 324–331, 2011.

[16] A. Janecek and Y. Tan, "Iterative improvement of the multiplicative update nmf algorithm using nature-inspired optimization," in *Natural Computation (ICNC), 2011 Seventh International Conference on*, vol. 3, pp. 1668–1672, IEEE, 2011.

[17] A. Janecek and Y. Tan, "Swarm intelligence for non-negative matrix factorization," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 2, no. 4, pp. 12–34, 2011.

[18] A. Janecek and Y. Tan, "Using population based algorithms for initializing nonnegative matrix factorization," in *Advances in Swarm Intelligence*, pp. 307–316, Springer, 2011.

[19] W. He, G. Mi, and Y. Tan, "Parameter optimization of local-concentration model for spam detection by using fireworks algorithm," in *Advances in Swarm Intelligence*, pp. 439–450, Springer, 2013.

[20] J. Liang, B. Qu, P. Suganthan, and Q. Chen, "Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization," 2014.