The Effect of Information Utilization: Introducing a Novel Guiding Spark in the Fireworks Algorithm

Junzhi Li, Shaoqiu Zheng, and Ying Tan, *Senior Member, IEEE*

***Abstract*—The fireworks algorithm is a competitive swarm intelligence algorithm which has been shown to be very useful in many applications. In this paper, a novel guiding spark is introduced to further improve its performance by enhancing the information utilization in the fireworks algorithm. The idea is to use the objective function's information acquired by explosion sparks to construct a guiding vector with promising direction and adaptive length, and to generate an elite solution called a guiding spark by adding the guiding vector to the position of the firework. The fireworks algorithm with guiding spark is called the guided fireworks algorithm. Experimental results show that the guiding spark contributes greatly to both exploration and exploitation of the guided fireworks algorithm. The guided fireworks algorithm outperforms previous versions of the fireworks algorithm and other swarm and evolutionary algorithms on a large variety of test functions and it is also a useful method for large scale optimization. The principle of the guiding spark is very simple but efficient, which can be easily transplanted to other population-based algorithms.**

***Index Terms*—Fireworks Algorithm, Information Utilization, Swarm Intelligence, Evolutionary Algorithm, Guiding Spark**

I. Introduction

THE fireworks algorithm (FWA) is a newly proposed swarm intelligence algorithm. It searches for an optimal point in the search space by iterating the explosion operation and the selection operation. In the explosion operation, numerous explosion sparks are generated around the fireworks within certain explosion amplitudes. After that, the fireworks of a new generation are selected from these explosion sparks. In each iteration, much information about the objective function is acquired by these explosion sparks, but it is not fully utilized in previous versions of the FWA.

Some of the recent research on the fireworks algorithm [1], [2] places their focus on the adaptive control of the explosion amplitude. But more importantly, they have also revealed the importance and the effect of utilizing the information about the objective function acquired by these explosion sparks.

In this paper, the information is further utilized to generate guiding sparks (GSs) in the FWA. The position of a GS is calculated by adding to the position of a firework a certain vector called the guiding vector (GV). The GV is the difference between the centroids of two groups of explosion sparks: those with good evaluation values, and those with bad evaluation values. It will be shown both experimentally and theoretically that the GV usually points to a promising area of the search space and its length is adaptive according to the distance away from the optimal point.

The fireworks algorithm with GSs is called the guided fireworks algorithm (GFWA) meaning the search process is guided by GSs as well as by the acquired information about the objective function.

It will be shown in experiments that GSs are usually much more efficient than explosion sparks in the FWA and that the proposed GFWA outperforms state-of-the-art variants of the fireworks algorithm and also some famous swarm and evolutionary algorithms on a wide range of test functions. Moreover, it turns out that the GFWA is also a useful method for large scale optimization problems due to its powerful global optimization capability and linear time complexity.

The GFWA is not the first algorithm to use the population's information to guide search. But the guiding spark provides a new, simple, reasonably designed, theoretically sound and practically effective method to help further improve information utilization in heuristic algorithms, and it can be easily transplanted to a large variety of population-based algorithms.

The rest of this paper is organized as follows. Some related works about information utilization and fireworks algorithms are introduced in Section II. The framework and the operators of the baseline algorithm — the dynamic search fireworks algorithm are introduced in Section III. The proposed guided fireworks algorithm and its theoretical analyses are presented in Section IV. Experimental results are shown in Section V. Finally, Section VI concludes the paper.

II. Related Works

In this section, some related research works are briefly introduced around the issue of information utilization and in the field of the FWA.

Besides, some algorithms also deserve to be discussed because their ideas share some similarities with that of the GFWA. Readers can find these discussions and comparisons in Section IV-G.

A. Information Utilization in Heuristic Algorithms

Typically, most of the swarm algorithms and evolutionary algorithms share the same framework consisting of two steps: new individuals are produced and old individuals are eliminated.

In the first step, new information is acquired; while in the second step, some information about the objective function is abandoned. The performances of these algorithms depend

Junzhi Li, Shaoqiu Zheng and Ying Tan are with the Key Laboratory of Machine Perception (Ministry of Education), Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, 100871, P.R. China. Shaoqiu Zheng is also with the 28th Research Institute of China Electronics, Technology Group Corporation, Nanjing, Jiangsu, China, 210007. Email:{ljz,zhengshaoqiu,ytan}@pku.edu.cn

largely on how and to what extent the information is utilized before it is abandoned.

By information utilization, we mean the usage of the information acquired by a heuristic algorithm in the search process for guiding future behaviors.

Information utilization is the potential drive of most research works in the field of computational intelligence, such as adaptive control [3], fitness approximation [4], estimation of distribution [5], etc. As for the GFWA, the adaptive length of the guiding vector makes the GFWA one of the adaptive control methods, while the direction of the guiding vector estimated using the information of the explosion sparks makes it one of the approximation methods.

The extent, the way and the cost of information utilization are all vital to the performance of a heuristic algorithm. Although it seems very hard to compare the extents between different families of algorithms, at least, in terms of developing a specific heuristic algorithm, its ability to utilize information should be developed as much as possible. In this paper, we are going to give an example and attempt on a recently proposed swarm algorithm, the fireworks algorithm, to show how much a heuristic algorithm can benefit from the idea of information utilization.

B. Related Works of Fireworks Algorithm

Since the FWA was proposed in 2010 [6], it has been shown to be a very competitive algorithm for optimization and has attracted much research interest. The FWA has been successfully applied to many real world problems, including digital filters design [7], nonnegative matrix factorization [8]–[10], spam detection [11], image identification [12], power loss minimization and voltage profile enhancement [13], capacitated vehicle routing problem [14], mass minimisation of trusses with dynamic constraints [15], laser machining processes [16], swarm robotics [17]–[19], clustering [20], multilevel image thresholding [21], RFID network planning [22], multi-satellite control resource scheduling [23], constrained portfolio optimization [24], regional seismic waveform [25], modern web information retrieval [26], gamma-ray spectrum fitting for radioisotope identification [27], de novo motif prediction [28], thermal unit commitment [29], privacy preserving [30], etc.

So far, research on the FWA has concentrated on improving the operators. In one of the most important improvements of the FWA, the enhanced fireworks algorithm (EFWA) [31], the operators of the conventional FWA were thoroughly analyzed and revised. Based on the EFWA, an adaptive fireworks algorithm (AFWA) [1] was proposed, which was the first attempt to control the explosion amplitude without preset parameter by detecting the results of the search process. In [2], a dynamic search fireworks algorithm (dynFWA) was proposed in which a simple dynamic controller of the explosion amplitude was adopted and the Gaussian mutation operator of the EFWA was removed.

Thanks to these works, the operators in the FWA have become much more reasonable and efficient. But more importantly, the direction of further research on the FWA has been clearly revealed. In fact, compared with the conventional FWA, the EFWA which adopts the best-firework-based Gaussian mutation strategy has promoted the information sharing capacity of the fireworks swarm. In the same way, based on the EFWA's static explosion amplitude strategy, the dynFWA and the AFWA introduced the adaptive controller by using the information acquired in the search process. The intrinsic property of these improvements is to raise the information utilization level. Thus, mining the information about the objective function from the evaluated fitness more thoroughly to help guide the search process is one promising and interesting research topic.

III. Overview of the dynFWA

The GFWA is based on the dynFWA because its idea is very simple and it works stably. In this section, we will briefly introduce the framework and the operators of the dynFWA for further discussion.

Without loss of generality, consider the following minimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}), \tag{1}$$

where $\mathbf{x}$ is a vector in the d dimensional Euclidean space. The object is to find an optimal $\mathbf{x}$ with minimal evaluation (fitness) value.

The dynFWA keeps searching for better solutions by the iteration of generating sparks from the fireworks and selection of fireworks among the sparks. Each iteration consists of the following two steps:

1) Explosion operator: Each firework explodes and generates a certain number of explosion sparks within a certain range (explosion amplitude). The numbers of explosion sparks (Eq. (2)) and the explosion amplitudes (Eq. (4)) are calculated according to the qualities of the fireworks. The principle of the calculation is to make better fireworks generate more sparks in smaller ranges in order to conduct exploitation and worse fireworks generate fewer sparks in larger ranges for exploration.

2) Selection operator: Fireworks of the new generation are selected from the candidates including the current fireworks and sparks. In the dynFWA, the best individual among the candidates is selected as a firework of the next iteration firstly, and the other fireworks are selected from the rest of the individuals uniformly randomly.

Besides, if the optimization problem is constrained, there is a mapping operator to map the out-of-bound sparks back into the feasible space. But the mapping operator is not to be discussed in this paper. We refer interested readers to a recent monograph [32].

In the following, the explosion operator of the dynFWA will be described in detail.

For each firework $\mathbf{X}_i$, its explosion sparks' number is calculated as follows:

$$\lambda_i = \hat{\lambda} \cdot \frac{\max_j(f(\mathbf{X}_j)) - f(\mathbf{X}_i)}{\sum_j (\max_k(f(\mathbf{X}_k)) - f(\mathbf{X}_j))}, \tag{2}$$

where $\hat{\lambda}$ is a constant parameter which controls the total number of explosion sparks in one generation.

In each generation, the firework with the best fitness is called the core firework (CF):

$$\mathbf{X}_{\mathrm{CF}} = \arg\min_{\mathbf{X}_i}(f(\mathbf{X}_i)). \tag{3}$$

In the dynFWA, the fireworks' explosion amplitudes (except for the CF's) are calculated just as in the previous versions of the FWA:

$$A_i = \hat{A} \cdot \frac{f(\mathbf{X}_i) - f(\mathbf{X}_{\mathrm{CF}})}{\sum_j (f(\mathbf{X}_j) - f(\mathbf{X}_{\mathrm{CF}}))}, \tag{4}$$

where $\hat{A}$ is a constant parameter which controls the explosion amplitudes generally.

But for the CF, its explosion amplitude is adjusted according to the search results in the last generation.

$$A_{\mathrm{CF}}(t) = \begin{cases} A_{\mathrm{CF}}(1) & t = 1 \\ C_r A_{\mathrm{CF}}(t-1) & f(\mathbf{X}_{\mathrm{CF}}(t)) = f(\mathbf{X}_{\mathrm{CF}}(t-1)) \\ C_a A_{\mathrm{CF}}(t-1) & f(\mathbf{X}_{\mathrm{CF}}(t)) < f(\mathbf{X}_{\mathrm{CF}}(t-1)) \end{cases} \tag{5}$$

where $A_{\mathrm{CF}}(t)$ is the explosion amplitude of the CF in generation t. In the first generation, the CF is the best among all the randomly initialized fireworks, and its amplitude is preset to a constant number which is usually the diameter of the search space. After that, if in generation $t-1$, the algorithm found a better solution than the best in generation $t-2$, the amplitude of the CF will be multiplied by an amplification coefficient $C_a > 1$, otherwise it will be multiplied by a reduction coefficient $C_r < 1$. The best solution in generation $t-1$ is always selected into generation t as the CF, so the right hand conditions in Eq. (5) means whether the best solution found has been improved.

The core idea of this dynamic explosion amplitude is described as follows: if in one generation no better solution is found, that means the explosion amplitude is too long (aggressive) and thus needs to be reduced to increase the probability of finding a better solution, and otherwise it may be too short (conservative) to make the largest progress and thus needs to be amplified. By the dynamic control, the algorithm can keep the amplitude appropriate for the search. That is, the dynamic explosion amplitude of the CF is long in early phases to perform exploration, and is short in late phases to perform exploitation.

Algorithm 1 shows how the explosion sparks are generated for each firework. For each firework, its sparks are generated with a uniform distribution within a hypercube around the firework. Besides, there is a dimension selection mechanism in the explosion operator, where only about half of the explosion sparks' dimensions are different from the firework.

The dynFWA is one of the most recent versions of the FWA, and it has significant advantages over the previous FWAs in terms of the design of the operators. However, the information acquired by the algorithm is still not thoroughly utilized. In the explosion operator, only the fitness value of the best individual is used to calculate the explosion amplitude of the CF (cf. Eq. (5)). In the selection operator, only the best individual is selected certainly as a firework of the new generation, while the rest of the individuals will be most probably abandoned.

Algorithm 1 Generating Explosion Sparks for $\mathbf{X}_i$

Require: $\mathbf{X}_i$, A_i and λ_i
for $j = 1$ **to** λ_i **do**
  for each dimension $k = 1, 2, ...d$ **do**
    sample κ from $\mathcal{U}(0, 1)$
    if $\kappa < 0.5$ **then**
      sample η from $\mathcal{U}(-1, 1)$
      $\mathbf{s}_{ij}^{(k)} \leftarrow \mathbf{X}_i{}^{(k)} + \eta \cdot A_i$
    else
      $\mathbf{s}_{ij}^{(k)} \leftarrow \mathbf{X}_i{}^{(k)}$
    end if
  end for
end for
return all the $\mathbf{s}_{ij}$

IV. Guided Fireworks Algorithm

When a number of explosion sparks are generated by a firework, their positions and fitness values contain a lot of information about the objective function, but it was not fully utilized in previous versions of the FWA. In this section, a guided fireworks algorithm (GFWA) is proposed for improving the performance by utilizing the information provided by the explosion operator. In each generation, a guiding spark (GS) is generated for each firework. The GS is generated by adding to the firework's position a certain vector called guiding vector (GV). To calculate the guiding vector, two things are to be learned from these explosion sparks: a promising direction and a proper step size along this direction. They are learned at the same time by calculating the difference between the centroids of two groups of explosion sparks: those with good evaluation values (top sparks) and those with bad evaluation values (bottom sparks).

A. Principle

GS's position $\mathbf{G}_i$ for firework $\mathbf{X}_i$ is determined by Algorithm 2.

Algorithm 2 Generating the Guiding Spark for $\mathbf{X}_i$

Require: $\mathbf{X}_i$, $\mathbf{s}_{ij}$,$f(\mathbf{s}_{ij})$,λ_i and σ
1: Sort the sparks by their fitness values $f(\mathbf{s}_{ij})$ in ascending order.
2: $\mathbf{\Delta}_i \leftarrow \frac{1}{\sigma\lambda_i}(\sum_{j=1}^{\sigma\lambda_i} \mathbf{s}_{ij} - \sum_{j=\lambda_i-\sigma\lambda_i+1}^{\lambda_i} \mathbf{s}_{ij})$
3: $\mathbf{G}_i \leftarrow \mathbf{X}_i + \mathbf{\Delta}_i$
4: **return** $\mathbf{G}_i$

Notes:

1) For each firework, only one GS is generated.
2) For each firework, only its own explosion sparks are required.
3) If $\sigma\lambda_i$ is not an integer, use $\lceil \sigma\lambda_i \rceil$ instead[1].

Actually in step 1 of Algorithm 2, it is not necessary to sort all the explosion sparks. Only the top and bottom $\sigma\lambda_i$

[1] $\lceil x \rceil$ is the smallest integer larger than x.

explosion sparks' indexes are needed, especially when λ_i is very large. Thus it can be improved by using the quick select algorithm [33] [34] to find the $(\sigma\lambda_i+1)^{th}$ best (worst) explosion spark and then find those who are better (worse) than it. In this way, the algorithm only requires linear time $O(\lambda_i d)$.

In Algorithm 2, the guiding vector $\mathbf{\Delta}_i$ is the difference between the top σ of the sparks' centroid and the bottom σ of the sparks' centroid. Besides, it can also be seen as the mean of $\sigma\lambda_i$ vectors:

$$\mathbf{\Delta}_i = \frac{1}{\sigma\lambda_i}\sum_{j=1}^{\sigma\lambda_i}(\mathbf{s}_{ij} - \mathbf{s}_{i,\lambda_i-j+1}), \tag{6}$$

and each of these vectors points from a bad solution to a good one. If the explosion amplitude is short, i.e., the explosion sparks are crowded around the firework, the GV can be seen as an estimator of the (minus) gradient, although the objective function is not required to be differentiable. While if the explosion amplitude is long, it is expected that the GV points to a promising area of the search space.

There are some reasons why the algorithm uses the top and the bottom population of explosion sparks instead of only the best spark and the worst explosion spark.

Firstly, it is expected that by using the top and bottom populations, their irrelevant values will be cancelled out. Most of the dimensions of the best explosion spark are good, but the rest of them are not, which means to learn from the sole best individual is to learn both its good and bad quality. While, learning from the good population is another thing, as long as the algorithm learns the common qualities of them. Except for their common qualities, other information can be regarded as random noise. This also holds for the bad explosion sparks. In Section IV-B, it will be shown that by using the information of the populations, the learned direction will be more stable and accurate.

The second main concern is the step size. If the minimal point of the objective function is out of the explosion amplitude of the firework, the algorithm is supposed to generate an elite spark which can lead the firework and accelerate the search process. While if the minimal point is already within the explosion amplitude, the step size should not be too long otherwise it cannot contribute to the search. So, the step size should be adjustable, according to the distance away from the minimal point. In the rest of this section, we will also prove that by using the information of the population, the step size does change with the distance automatically.

As will be shown in Section V-A, in practice the performances are better using the top and the bottom populations than using only the best and the worst explosion sparks.

The framework of the guided fireworks algorithm is shown in Algorithm 3.

In each iteration, after the guiding sparks are generated, the candidates' set from which the fireworks of the next generation will be selected includes three kinds of individuals: current fireworks, explosion sparks and guiding sparks. Except for the best candidate, the other candidates are still randomly selected.

Algorithm 3 Guided Fireworks Algorithm

1: Randomly initialize μ fireworks in the potential space.
2: Evaluate the fireworks' fitness.
3: **repeat**
4: Calculate λ_i according to Eq.(2).
5: Calculate A_i according to Eq.(4) and Eq.(5).
6: For each firework, generate λ_i sparks within the amplitude A_i according to Algorithm 1.
7: For each firework, generate guiding sparks according to Algorithm 2.
8: Evaluate all the sparks' fitness.
9: Keep the best individual as a firework.
10: Randomly choose other $\mu-1$ fireworks among the rest of individuals.
11: **until** termination criteria is met.
12: **return** the position and the fitness of the best individual.

B. Accuracy of the Direction

In the following, a specific example will be shown to study the properties of the GV and theoretical analyses on the GFWA will be presented. For the convenience of theoretical analyses, the dimension selection mechanism (line 3,4,7,8 and 9 in Algorithm 1) is not considered in the remaining parts of this section. (And it will be actually removed from the GFWA in the experiments. See Section V-A.) It won't change the essential propositions given below, because all the bounds would only be multiplied by some constants, taking dimension selection into consideration.

Assume[2] $f(\mathbf{x}) = x_1^2$, and the algorithm only adopts one firework $\mathbf{X}$ (thus the subscript i is omitted) with $\mathbf{X}^{(1)} > 0$. That is, at the firework's location, the value of the objective function decreases fastest on direction $[-1, 0, 0, \ldots]$ (i.e., the minus gradient), and doesn't change on direction $[0, \ldots]$. The following theorem shows that the noises on irrelevant directions will be cancelled out.

Theorem 1 (Error Bound of Irrelevant Directions)**.** *Assume $f(\mathbf{x}) = x_1^2$, $\mathbf{s}_j$ are calculated by Algorithm 1 without dimension selection, and $\mathbf{\Delta}$ is calculated by Algorithm 2. For any $k \neq 1$, with probability at least $1-\delta$,*

$$|\mathbf{\Delta}^{(k)}| \leqslant \sqrt{\frac{2}{3\sigma\lambda\delta}}A. \tag{7}$$

The proof is given in Appendix A.

With the number of σ and λ increasing, the mean locations of the top and bottom sparks on insensitive dimensions both gradually concentrate to 0.

The noises on these dimensions cancel each other out and make the $\mathbf{\Delta}$ closer to the real minus gradient. As shown in Fig. 1, the GS is always located near the direction of the minus gradient. The noises are cancelled out, but the useful direction is preserved. Actually, this is an application and example of the law of large numbers [35]. Only with the information of the population can the algorithm get an accurate direction.

[2]Here x_1 means the first dimension of $\mathbf{x}$. While for the notations in the algorithms, such as $\mathbf{X}$ and $\mathbf{\Delta}$, the subscript is the index of the firework, and the subscript with parentheses is the index of dimension.

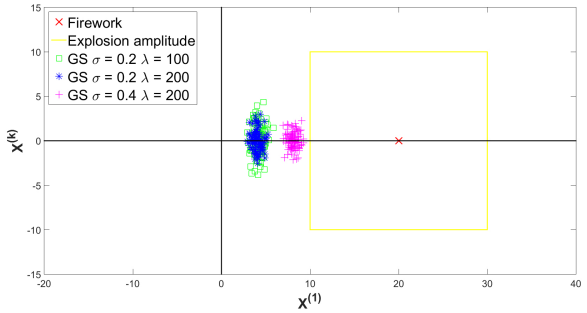

Fig. 1: $f(\mathbf{x}) = x_1^2, \mathbf{X}^{(1)} = 20, A = 10$. Repeated for 100 times for each set of parameters, the GS is always located near the $\mathbf{X}^{(1)}$-axis.

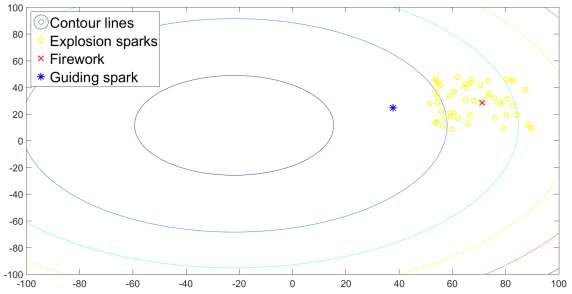

Fig. 2: Exploration on a single unimodal function by the guiding spark

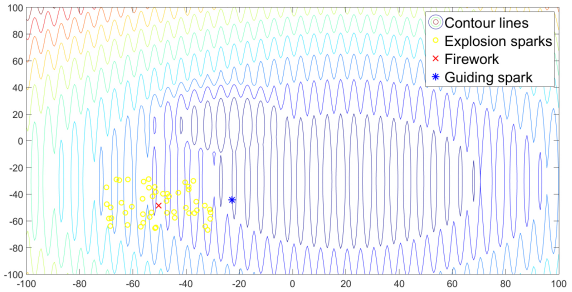

Fig. 3: Exploration on a multimodal function by the guiding spark

Now let's take a closer look at how long $\mathbf{\Delta}$ will be at the most sensitive direction. There are two cases: $\mathbf{X}^{(1)} > A > 0$ (i.e., the minimal point is out of the explosion amplitude) and $A > \mathbf{X}^{(1)} > 0$ (i.e., the minimal point is within the explosion amplitude).

C. Exploration

When the minimal point is out of the explosion amplitude of the firework, the GS is supposed to explore along the promising direction.

In this case, the top sparks are located nearest to the promising area, while the bottom sparks are located farthest from the promising area. So the distance between their centroids, namely the length of GV, is long, and the GS can get out of the explosion amplitude and explore the promising area. This can greatly improve the performance on multimodal functions and also the convergence speed on unimodal functions.

The following theorem shows that the length of GV on the sensitive direction is long.

Theorem 2 (Step Size for Exploration)**.** *Assume $f(\mathbf{x}) = x_1^2$, $\mathbf{s}_j$ are calculated by Algorithm 1 without dimension selection, and $\mathbf{\Delta}$ is calculated by Algorithm 2. If $\mathbf{X}^{(1)} \geqslant A > 0$, $\sigma \leqslant 0.5$, and λ is sufficiently large, then with probability at least $1 - \delta$,*

$$|\mathbf{\Delta}^{(1)}| > (2(1-\sigma)^{1-\sigma}\sigma^{\sigma}\delta^{\frac{1}{\lambda}} - 1)A. \tag{8}$$

The proof is given in Appendix B.

The lower bound increases when λ increases or σ decreases, but the influence of λ is ignorable for a normal confidence level δ. Note that the distribution of $\mathbf{\Delta}^{(1)}$ is independent of the distance to the minimal point (i.e., $\mathbf{X}^{(1)}$), as long as the minimal point is out of the explosion amplitude (i.e., $\mathbf{X}^{(1)} > A$).

Fig. 2 shows how a guiding spark accelerates the search by leading the population on a unimodal function (Function 1 in Table I).

Fig. 3 shows how a guiding spark helps the population escape from the local minimum on a multimodal function (Function 17 in Table I).

D. Exploitation

When the minimal point is within the explosion amplitude of the firework, the length of GV gradually decreases and meanwhile the direction is still kept accurate to search more precisely together with other explosion sparks.

When $A > \mathbf{X}^{(1)} > 0$, it is obvious that $\mathbf{\Delta}^{(1)}$ is not as long as in the former case, because 1) the top sparks are not located near an end of the amplitude and 2) the worst sparks may be located near both ends and thus cancel each other out. It can also be inferred from the limit case: when $\mathbf{X}^{(1)} \to 0$, $\mathbf{\Delta}^{(1)}$ will also concentrate to zero.

Theorem 3 (Convergence of the Step Size with Distance)**.** *Assume $f(\mathbf{x}) = x_1^2$, $\mathbf{s}_j$ are calculated by Algorithm 1 without dimension selection, and $\mathbf{\Delta}$ is calculated by Algorithm 2. If $\mathbf{X}^{(1)} = 0$, with probability at least $1 - \delta$,*

$$|\mathbf{\Delta}^{(1)}| \leqslant \sqrt{\frac{4}{3\sigma\lambda\delta}}A. \tag{9}$$

The proof is given in Appendix C.

The theorem implies that, with $|\mathbf{X}^{(1)}|/A$ decreasing and with sufficiently large $\sigma\lambda$, the length of $\mathbf{\Delta}^{(1)}$ gradually decreases to zero.

As shown in Fig. 4, when $\mathbf{X}^{(1)} > A$, the step size $\mathbf{\Delta}^{(1)}$ is not influenced by the distance and is comparatively large, but when $\mathbf{X}^{(1)} < A$ the step size gradually converges to zero. When the edge of the explosion amplitude approaches the minimal point, GV senses it, and reacts by decreasing the step size.

It may seem ideal if $\mathbf{\Delta}^{(1)}$ also increases with $\mathbf{X}^{(1)}$ even when $\mathbf{X}^{(1)} > A$, so that the algorithm can approach the minimal point as fast as possible, but it is not. Even though the direction of $\mathbf{\Delta}$ is very accurate as been proved, it is accurate only in a small area, because nearly no objective function has the same gradient everywhere. To make matters worse, if the step size is too long, the error of the direction will be amplified enormously. Note that the GS in exploration can be considered as a kind of extrapolation, whose accuracy is always affected by the step size.

It can also be seen in Fig. 4 that the λ itself has almost no influence on the expectation of the step size, but the stability of the step size increases with λ. While larger σ causes shorter step size and better stability. As for the direction, there is little chance for the algorithm to take the wrong direction (i.e., $\mathbf{\Delta}^{(1)} > 0$), not until the distance $|\mathbf{X}^{(1)}|/A$ is very close to zero.

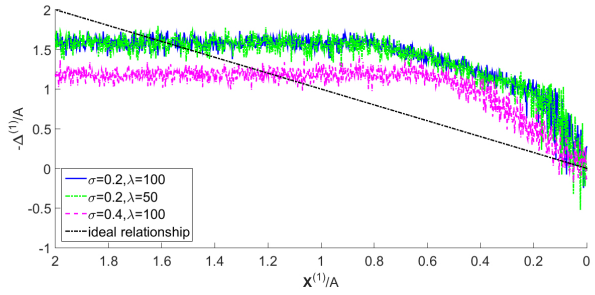

Fig. 4: The relationship between the distance from the optimal point and the step size

Fig. 5 shows how a GS helps to conduct local search for enhancing exploitation on a single modal function (Function 1 in Table I). When the optimal point is within the explosion amplitude of the firework, it means the firework has approached the optimal point, though the algorithm doesn't know where exactly it is. In this case, both top and bottoms sparks are located around the optimal point (though their distances from the optimal point are different), and the distance between two centroids is short. The GS will be most probably located inside the explosion amplitude of the firework, pointing out roughly the direction of the minimal point, meanwhile the length of GV will shrink. In this case, GS works like an explosion spark, but the expected contribution of GS may be slightly larger than the average of explosion sparks, because at least it is more probably located on the correct direction due to the fact that the length of GV converges faster on irrelevant directions than on the relevant direction (compare Eq.(7) and Eq.(9)).

E. How to Choose σ

Generally speaking, any $0 < \sigma < 1$ is possible, but $\sigma > 0.5$ is not suggested as some of the top sparks and the bottom sparks will cancel out.

Larger σ is good at learning the direction of the gradient, while smaller σ may result in a comparatively inaccurate direction. However, smaller σ makes the GS move faster because the algorithm doesn't take into consideration these

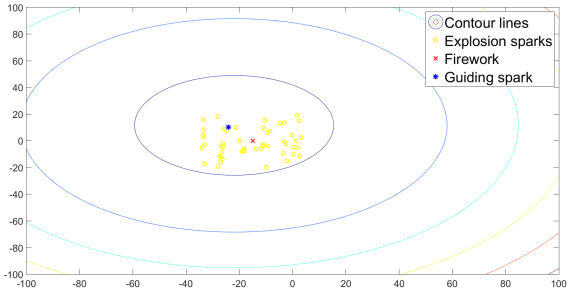

Fig. 5: Exploitation by the guiding spark

not-so-good (bad) explosion sparks. For example, in the above case, when $\mathbf{X}^{(1)} > A > 0$, if $\sigma \to 0.5$, $|E(\mathbf{\Delta}^{(1)})| \to A$ and the GS has little chance to be located out of the explosion amplitude.

Weighting these factors, and according to experimental results (cf. Section V-A), $\sigma = 0.2$ is recommended for most cases empirically.

F. Remarks

The proposed algorithm is a population-based algorithm and is also an information-based algorithm. Generally speaking, the more explosion sparks it adopts, the better it works. Some theoretical analyses have been conducted on GV, which proved its following properties:

1) The noises on the irrelevant dimensions are cancelled out and concentrate to 0 with sufficiently large $\sigma\lambda$, and thus the direction of GV is promising, accurate and stable.
2) If the minimal point is out of the explosion amplitude of the firework, GV will be long at the promising direction and thus be able to contribute to exploration.
3) If the minimal point is within the range of the explosion amplitude, GV will be short at the promising direction and thus be able to contribute to exploitation.

Compared with the previous FWAs, the GFWA utilizes more thoroughly the information about the objective function.

G. Comparison with Other Heuristic Algorithms

There are several typical algorithms that share some similarities with the GFWA and deserve to be mentioned here.

The particle swarm optimization (PSO) [36] is one of the most famous swarm intelligence algorithms. Typically, in the PSO, each particle is attracted by a global/local best position (best known position of the entire/local swarm) and a personal best position (best known position of this particle), which is to some extent similar to the idea of the GFWA — to learn from the good ones and move towards the promising direction. However, their differences are essential:

1) The structures of the FWA and the PSO are extremely different. In the PSO, the number of the particles is fixed, or in other words, there is a bijection from generation to generation. While in the FWA, the fireworks generate different numbers of sparks, from which the fireworks of the next iteration will be selected, and there is no

bijection from generation to generation. As a result, the positions of the fireworks change mainly through the explosion operation and the selection operation while the particles in the PSO move by adding velocity vectors to their current positions.

2) The GV could provide a more accurate direction than the linear combination of the three vectors in the PSO (inertia component, social component and cognitive component), because it integrates the information of numerous explosion sparks around the firework. In Section V-A, experiments show that using only the best one or two individuals is not sufficient to construct a reliable guiding vector. Moreover, the algorithm for generating the GS is deterministic, while the particle's position in the PSO is stochastic, which means the direction of the GV is more stable. The robustness of the PSO is rather based on the cooperation among the particles than the efficiency of each particle.
3) Although the step size in the PSO also changes with different search stages, it is possible it doesn't converge if parameters are not carefully selected [37]. On the contrary, the FWA in general converges [38], and the length of the GV also converges (cf. Theorem 3).

As a variant of the PSO, the Swarm & Queen algorithm [39] also uses the centroid to accelerate the convergence, like the GFWA. But their principles are different. In the Swarm & Queen algorithm, the centroid is itself introduced in the algorithm as an elite solution, while in the GFWA, two centroids are used to calculate a vector which will be added to the firework's position to generate an elite solution and lead the population. The centroid is typically located in the middle of the population whose contribution to exploration is limited, and thus there is a re-initialization mechanism in the Swarm & Queen algorithm. While the guiding spark is able to conduct both exploration and exploitation according to the information of the population.

The differential evolution (DE) [40] is a famous evolutionary algorithm, which also shares some similarities with the GFWA. They both add vectors to individuals to generate new offspring. However, their mechanisms are quite different.

1) In the DE, the vector is the difference between two (or more) randomly chosen individuals. The direction of such a vector is unpredictable. While in the GFWA, the vector is the difference between two centroids (or the mean of several vectors which all point from bad positions to good positions), and is thus with a more clear purpose.
2) In the DE, the vector's length converges with the population approaching the optimal point because the whole population converges. While in the GFWA, the length of the GV converges even when the explosion amplitude is quite large as long as the population approaches the optimal point. Thus the guiding spark is able to conduct exploitation in earlier phases than individuals in DE if its needed.

The artificial bee colony (ABC) algorithm [41] is a another famous swarm intelligence algorithm. The resource distribution (mainly performed by onlooker bees) in the ABC is to some extent similar to that of the FWA, in which positions with good fitness values are searched more thoroughly. But the way individuals move in the ABC is similar to the DE. Since they are both random walk based algorithms and share similar formula for updating positions [42], a similar comparison between the GFWA and the DE can be made between the GFWA and the ABC.

The CMA-ES [43] is a highly developed evolutionary algorithm. In the CMA-ES, quite a few mechanisms are employed to control the direction and the step size. The search direction in the CMA-ES is also led by the good subset of solutions, and the step size mechanisms in the CMA-ES and the GFWA are especially similar: if the directions of several vectors are unified, the step size will be long. However, the frameworks of the FWA and the CMA-ES are different. The fireworks algorithm is a swarm algorithm whose framework allows multiple fireworks' population (along with their sparks) to interact with each other. Many research works have been conducted to enhance the cooperation among fireworks [31], [44], [45]. While the CMA-ES is an evolutionary algorithm, and typically there is only one population. For the CMA-ES, designing restart mechanism for the population has attracted more research interest [46]–[48]. Besides, there are some other different points between the CMA-ES and the GFWA.

1) There is no center in the population of the CMA-ES, but there are fireworks (around which sparks are generated) in the GFWA. As a result, the mean for sampling solutions is calculated (weighted sum) from the solutions in the CMA-ES but selected from the solutions in the GFWA.
2) The explosion sparks are generated within a limited range (explosion amplitude) around the firework. While the solutions in the CMA-ES are generated with a distribution, within an unlimited range (though the probability beyond 3Σ is very low). That's why the exploration ability of the GS is very important in the GFWA.
3) The CMA-ES also senses the promising direction, but the Gaussian distribution is symmetrical, thus it has to search on both ways. While the guiding spark searches only towards the promising direction.
4) The GFWA runs faster. The time complexity of plain CMA-ES is quadratic [49], while the time complexity of the GFWA is linear.

V. Experiments and Analyses

A. Parameter Setting

In order to illustrate the performance of the GFWA, a set of experiments is conducted on the CEC 2013 single objective optimization benchmark suite [50]. This benchmark suite includes unimodal functions, multimodal functions and composition functions, shown in Table I. In the following experiments, the dimensionality of these functions is $D = 30$. All the algorithms are run 51 times for each function and the maximal number of evaluations of each run is $10000D$.

For the two parameters in the dynamic explosion amplitude, we follow the suggestions of the authors [2] and use $C_r = 0.9$,

TABLE I: Test functions of CEC 2013 single objective optimization benchmark suite

	No.	Name
Unimodal Functions	1	Sphere Function
	2	Rotated High Conditioned Elliptic Function
	3	Rotated Bent Cigar Function
	4	Rotated Discus Function
	5	Different Powers Function
Basic Multimodal Functions	6	Rotated Rosenbrocks Function
	7	Rotated Schaffers F7 Function
	8	Rotated Ackleys Function
	9	Rotated Ackleys Function
	10	Rotated Griewanks Function
	11	Rastrigins Function
	12	Rotated Rastrigins Function
	13	Non-Continuous Rotated Rastrigins Function
	14	Schwefel's Function
	15	Rotated Schwefel's Function
	16	Rotated Katsuura Function
	17	Lunacek Bi_Rastrigin Function
	18	Rotated Lunacek Bi_Rastrigin Function
	19	Expanded Griewanks plus Rosenbrocks Function
	20	Expanded Scaffers F6 Function
Composition Functions	21	Composition Function 1 (Rotated)
	22	Composition Function 2 (Unrotated)
	23	Composition Function 3 (Rotated)
	24	Composition Function 4 (Rotated)
	25	Composition Function 5 (Rotated)
	26	Composition Function 6 (Rotated)
	27	Composition Function 7 (Rotated)
	28	Composition Function 8 (Rotated)

TABLE II: Comparison between GFWA-ds and GFWA

	GFWA-ds		GFWA		
F.	Mean	Std.	Mean	Std.	p
1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	NaN
2	7.04E+05	3.15E+05	6.96E+05	2.66E+05	9.68E-01
3	4.85E+07	1.06E+08	3.74E+07	8.65E+07	1.94E-01
4	4.99E-05	7.39E-05	5.02E-05	6.17E-05	8.67E-01
5	**1.26E-04**	**2.62E-05**	1.55E-03	1.82E-04	3.29E-18
6	2.92E+01	2.43E+01	3.49E+01	2.74E+01	2.50E-01
7	7.21E+01	2.69E+01	7.58E+01	2.98E+01	3.88E-01
8	2.09E+01	7.03E-02	2.09E+01	9.11E-02	7.68E-01
9	1.84E+01	4.11E+00	1.83E+01	4.61E+00	9.36E-01
10	6.15E-02	2.90E-02	6.08E-02	3.36E-02	7.15E-01
11	8.70E+01	2.51E+01	**7.50E+01**	**2.59E+01**	5.77E-03
12	1.10E+02	3.29E+01	**9.41E+01**	**3.28E+01**	1.23E-02
13	1.93E+02	4.18E+01	**1.61E+02**	**4.74E+01**	2.39E-04
14	**2.74E+03**	**5.58E+02**	3.49E+03	8.30E+02	1.77E-06
15	3.73E+03	6.52E+02	3.67E+03	6.35E+02	7.28E-01
16	1.19E-01	8.63E-02	1.00E-01	7.13E-02	2.16E-01
17	1.12E+02	2.11E+01	**8.49E+01**	**2.10E+01**	7.40E-10
18	1.11E+02	3.77E+01	**8.60E+01**	**2.33E+01**	1.04E-04
19	5.58E+00	1.27E+00	5.08E+00	1.88E+00	5.15E-02
20	1.38E+01	1.41E+00	**1.31E+01**	**1.09E+00**	5.43E-03
21	3.28E+02	9.81E+01	**2.59E+02**	**8.58E+01**	2.21E-04
22	**3.16E+03**	**5.27E+02**	4.27E+03	8.90E+02	6.52E-10
23	4.41E+03	9.23E+02	4.32E+03	7.69E+02	7.79E-01
24	2.59E+02	1.14E+01	2.56E+02	1.75E+01	3.45E-01
25	**2.81E+02**	**9.35E+00**	2.89E+02	1.34E+01	3.30E-03
26	2.29E+02	5.91E+01	**2.05E+02**	**2.71E+01**	6.85E-03
27	8.38E+02	9.91E+01	8.15E+02	1.22E+02	5.34E-01
28	3.21E+02	1.51E+02	3.60E+02	2.60E+02	6.42E-01

$C_a = 1.2$ in the experiments. There are three other parameters to be set up in the GFWA: μ, $\hat{\lambda}$ and σ.

Before tuning these parameters, it is necessary to firstly test whether or not the dimension selection mechanism (line 3,4,7,8 and 9 in Algorithm 1) is necessary in the GFWA, because it costs some extra time to generate random numbers and it may arouse some problems in the direction of the GV (with the samples' number in each dimension halved). The GFWA with dimension selection mechanism is denoted as GFWA-ds, and the GFWA without dimension selection is denoted as GFWA. In this experiment, $\mu = 1, \lambda = 200$ and $\sigma = 0.2$. The mean errors, standard deviations and p values are shown in Table II. A set of Wilcoxon signed rank tests is conducted to show if their differences are significant (with confidence level at least 95%). The significantly better results are highlighted.

On most of these test functions, the GFWA without dimension selection performs better. So this mechanism is removed from the GFWA in the following experiments.

Secondly, μ is to be tuned based on this benchmark suite. The mean errors and standard deviations using different μ are shown in Table III. The minimal mean error on each function is highlighted. A set of pair-wise Wilcoxon signed rank tests is also conducted. If on a certain test function, a group of results performs significantly better than another, it gets one score. So the maximal sum scores of each μ is $28 \times (4-1) = 84$. The total scores achieved by each μ are also recorded and shown in Table III.

On these test functions, $\mu = 1$ performs the best. Thus in the following experiments, $\mu = 1$ is used.

Finally, a set of experiments is conducted to adapt $\hat{\lambda}$ and σ based on this benchmark suite. In order to select the set of σ and $\hat{\lambda}$ with the best performance, pair-wise Wilcoxon signed rank tests are conducted. If on a certain test function a set of parameters performs significantly better than another, it gets one score. So the maximal sum score of each set of σ and $\hat{\lambda}$ is $28 \times (12-1) = 308$. The total scores are shown in Figure 6.

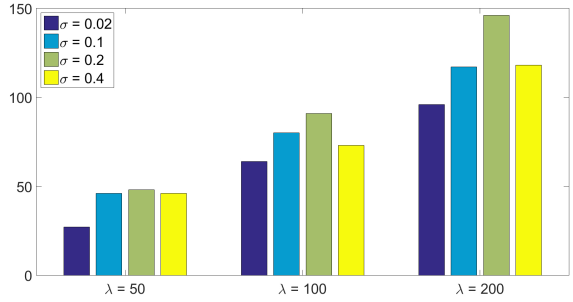

Fig. 6: Scores of 12 sets of parameters

Generally speaking, the performance improves as $\hat{\lambda}$ gets larger. [3] But the performance doesn't fluctuate too much as $\hat{\lambda}$ varies. In real world applications, if the dimensionality is not this high or the maximal evaluation number is very limited, $\hat{\lambda}$ can be set to smaller values, and the GFWA can still work stably.

As for σ, 0.2 is usually the best choice, whatever $\hat{\lambda}$ is. In contrast, $\sigma = 0.02$ usually performs the worst, due to the

[3] In fact, the performance doesn't suffer until $\hat{\lambda}$ is around 1000. But this is because of the large maximal evaluation number and the fact that there are more multimodal functions than unimodal functions in this benchmark suite. $\hat{\lambda}$ larger than 200 may result in too few iterations to find the minimal point in normal applications.

TABLE III: Performances of the GFWA using different μ on CEC2013 benchmark suite

μ	1		3		5		7	
F.	Mean	Std.	Mean	Std.	Mean	Std.	Mean	Std.
1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
2	6.96E+05	2.66E+05	**6.31E+05**	**2.24E+05**	1.08E+06	4.65E+05	1.32E+06	7.24E+05
3	**3.74E+07**	**8.65E+07**	8.16E+07	1.90E+08	6.05E+07	7.24E+07	1.24E+08	2.21E+08
4	**5.02E-05**	**6.17E-05**	1.02E-01	7.93E-02	9.87E+00	5.21E+00	8.95E+01	4.31E+01
5	1.55E-03	1.82E-04	**1.46E-03**	**1.47E-04**	2.01E-03	2.38E-04	2.15E-03	2.99E-04
6	**3.49E+01**	**2.74E+01**	4.17E+01	2.61E+01	4.74E+01	2.83E+01	4.62E+01	2.95E+01
7	**7.58E+01**	**2.98E+01**	8.33E+01	2.47E+01	9.99E+01	2.84E+01	9.90E+01	3.24E+01
8	2.09E+01	9.11E-02	**2.09E+01**	**7.91E-02**	2.09E+01	7.54E-02	2.09E+01	7.43E-02
9	**1.83E+01**	**4.61E+00**	1.92E+01	4.05E+00	1.94E+01	4.46E+00	2.15E+01	3.77E+00
10	6.08E-02	3.36E-02	**5.80E-02**	**2.68E-02**	5.90E-02	3.32E-02	7.40E-02	3.93E-02
11	**7.50E+01**	**2.59E+01**	9.40E+01	2.75E+01	1.04E+02	4.68E+01	1.38E+02	6.11E+01
12	**9.41E+01**	**3.28E+01**	1.02E+02	4.00E+01	9.54E+01	3.01E+01	1.16E+02	4.49E+01
13	**1.61E+02**	**4.74E+01**	1.79E+02	5.38E+01	1.70E+02	4.96E+01	1.96E+02	5.14E+01
14	3.49E+03	8.30E+02	**3.39E+03**	**6.70E+02**	3.58E+03	8.35E+02	3.52E+03	6.40E+02
15	**3.67E+03**	**6.35E+02**	3.68E+03	6.93E+02	3.81E+03	7.74E+02	3.70E+03	6.10E+02
16	**1.00E-01**	**7.13E-02**	1.76E-01	1.31E-01	2.45E-01	2.27E-01	2.35E-01	1.61E-01
17	**8.49E+01**	**2.10E+01**	8.85E+01	3.01E+01	8.97E+01	2.26E+01	9.73E+01	2.57E+01
18	**8.60E+01**	**2.33E+01**	8.93E+01	1.91E+01	8.55E+01	1.97E+01	9.67E+01	2.97E+01
19	**5.08E+00**	**1.88E+00**	5.67E+00	2.15E+00	6.06E+00	2.32E+00	5.90E+00	2.19E+00
20	1.31E+01	1.09E+00	1.33E+01	9.91E-01	**1.28E+01**	**1.05E+00**	1.33E+01	1.02E+00
21	**2.59E+02**	**8.58E+01**	3.05E+02	9.24E+01	3.16E+02	9.96E+01	3.02E+02	9.78E+01
22	**4.27E+03**	**8.90E+02**	4.41E+03	9.10E+02	4.48E+03	1.05E+03	4.79E+03	9.65E+02
23	**4.32E+03**	**7.69E+02**	4.63E+03	6.68E+02	4.48E+03	8.03E+02	4.71E+03	9.07E+02
24	**2.56E+02**	**1.75E+01**	2.60E+02	1.84E+01	2.66E+02	1.90E+01	2.68E+02	1.58E+01
25	2.89E+02	1.34E+01	**2.88E+02**	**1.24E+01**	2.96E+02	1.39E+01	3.00E+02	1.28E+01
26	2.05E+02	2.71E+01	**2.03E+02**	**2.25E+01**	2.06E+02	2.96E+01	2.17E+02	5.14E+01
27	**8.15E+02**	**1.22E+02**	8.57E+02	1.18E+02	8.86E+02	1.18E+02	9.15E+02	1.29E+02
28	3.60E+02	2.60E+02	3.22E+02	1.56E+02	**3.09E+02**	**1.60E+02**	3.93E+02	3.22E+02
Score	**40**		22		10		1	

inaccurate direction and the aggressive step size. Using the top and bottom population of the explosion sparks is a better choice than to use only the best explosion spark and the worst explosion spark.

Thus in the following experiments, $\hat{\lambda} = 200$ and $\sigma = 0.2$ are used.

B. The Efficiency of the Guiding Spark

In order to measure how much the guiding spark contributes to the search on different test functions, we counted how many times a better solution is found by the GS (i.e., $f(\mathbf{G}) < \min\{f(\mathbf{s}_j), f(\mathbf{X})\}$), and how many times a better solution is found by the explosion sparks (i.e., $\min(f(\mathbf{s}_j)) < \min\{f(\mathbf{G}), f(\mathbf{X})\}$). The normalized ratios are shown in Fig. 7.

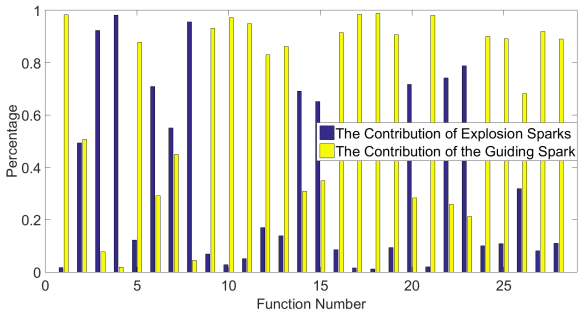

Fig. 7: Normalized contribution comparison between the guiding spark and explosion sparks

Note that in each iteration, there are 200 explosion sparks and only one guiding spark. On most of the 28 functions, the guiding spark's contribution is far greater than 1/200 of the explosion sparks', which implies that the guiding spark is efficient. It can also be seen that, on multimodal functions (6-28), the guiding spark contributes more than on unimodal functions (1-5), which means the guiding spark helps more towards exploration than exploitation.

C. Comparison with Other Algorithms

In order to measure the relative performance of the GFWA, a comparison among the GFWA, other FWAs and typical swarm intelligence and evolutionary algorithms is conducted on the CEC 2013 single objective benchmark suite. The algorithms compared here are described as follows.

1) Artificial bee colony (ABC) [41]: A powerful swarm intelligence algorithm. The results were reported in [51].

2) Standard particle swarm optimization (SPSO) [52]: The most recent standard version of the famous swarm intelligence algorithm PSO. Some results were reported in [53], but the mean errors presented here are calculated from raw data.

3) Differential evolution (DE) [40]: One of the best evolutionary algorithms for optimization. The results were reported in [54], where the DE is combined with a recently proposed constraint-handling strategy to adapt the benchmark.

The raw data of the above three technical reports (#1502, #1534, #1676) can be downloaded from [55]. However the data of the DE is incomplete.

4) Covariance matrix adaptation evolution strategy (CMA-ES) [43]: A developed evolutionary algorithm. The results are based on the code from [56] using default settings.

5) Enhanced fireworks algorithm (EFWA) [31]: See Section II-B. The results are based on the code from [57] using default settings. Comparable results can be found in [1] or [2].

6) Adaptive fireworks algorithm (AFWA) [1]: See Section II-B. The results are based on the code from [58] using default settings. Comparable results can be found in [1].

7) Dynamic search fireworks algorithm (dynFWA) [2]: See Section II-B. The results are based on the code from [59] using default settings. Comparable results can be found in [2].

The mean errors of these 8 algorithms CEC2013's 28 benchmark functions are compared and listed in Table IV. In addition, these algorithms are ranked according to their mean errors on each function, and the average rankings (AR) over the 28 functions are presented at the bottom of Table IV. The minimal mean errors are highlighted[4].

The ABC beats other algorithms on 10 functions (some differences are not significant), which is the most, but performs poorly on other functions. The CMA-ES performs extremely well on unimodal functions, but suffers from premature convergence on some complex functions. In terms of average ranking, the GFWA performs the best among these 8 algorithms on this benchmark suite due to its stability. The DE takes the second place. The performances of the ABC, the AFWA and the dynFWA are comparable.

A set of pair-wise Wilcoxon signed rank tests is also conducted between the GFWA and each of the other algorithms except the DE due to its incomplete data. The numbers of significantly better results (indicated by "win") and significantly worse results (indicated by "lose") of the GFWA on each kind of functions are recorded, shown in Fig. 8.

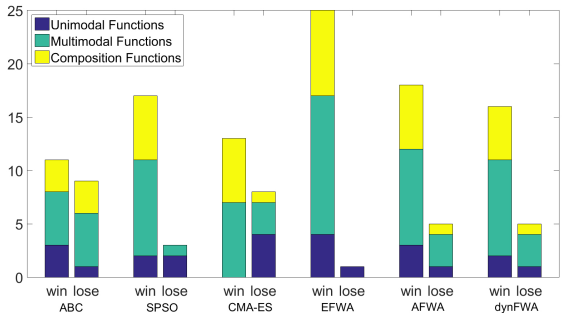

Fig. 8: Numbers of significantly better and worse results of the GFWA compared with other algorithms

In all these pair-wise comparisons, the GFWA wins more than its opponents, especially on multimodal and composition functions, which proves the GFWA a powerful global optimization algorithm.

D. Large Scale Optimization

Nowadays, many applications require algorithms to deal with large scale optimization problems. The computational complexity of the GFWA is linear with both dimensionality and population size, which is convenient in large-scale optimization. Here the GFWA is also tested on CEC 2010 large scale global optimization benchmark [60]. These 20 test functions are shown in Table V, where dimensionality $D = 1000$, group size $m = 50$.

The results of two state-of-the-art large scale optimization algorithms the DECC-G [61] and the MLCC [62] which are specialized in solving large scale problems are adopted for a comparison [63].

Each algorithm is run 25 times for each function and the maximum number of evaluations of each run is 3.0E+06.

The mean errors, standard deviations and average rankings are presented in Table VI. The best results among the three algorithms on these functions are highlighted.

The GFWA doesn't use any a priori knowledge of the evaluation functions (separable/nonseparable, group size, etc.) and treats them totally as black-box problems, and yet the results of the GFWA are still comparable to those of the two algorithms specializing in separable large scale optimization. The GFWA performs well on highly nonseparable functions and outperforms the DECC-G and the MLCC on both two totally nonseparable functions: 19 and 20, which indicates that it is a powerful universal optimization method. It provides an alternative tool to deal with large scale optimization. Yet surely there is still some work to do for its performance improvement on large scale optimization problems.

VI. Conclusion

In this paper, a novel guiding spark is introduced in the fireworks algorithm. The position of the guiding spark is calculated through adding a guiding vector to the position of the firework, which is the difference between the good explosion sparks' centroid and the bad explosion sparks' centroid. It is shown both experimentally and theoretically that the direction of such a guiding vector is promising, accurate and stable, and its step size is adaptive according to the distance away from the optimal point in the search space. In the proposed algorithm, the information about the objective function acquired by these explosion sparks is more thoroughly utilized. Experimental results show that the guiding spark contributes much more than explosion sparks on both unimodal and multimodal test functions. The proposed algorithm is also compared with other typical heuristic algorithms, and it outperforms them on a wide range of test functions in terms of average performance and pair-wise comparisons. Moreover, it is also shown experimentally that the proposed algorithm is a useful method for large scale global optimization. The computational complexity of the proposed algorithm is linear, which allows it to be used in a large variety of real world applications.

The principle of the proposed algorithm is very simple and can be easily adapted to other population-based algorithms. We sincerely hope this work could be an example of how to enhance information utilization in heuristic algorithms and inspire more.

Appendix A
Proof of Theorem 1

Theorem 1 (Error Bound of Irrelevant Directions)**.** *Assume* $f(\mathbf{x}) = x_1^2$, $\mathbf{s}_j$ *are calculated by Algorithm 1 without dimen-*

[4]The precisions of DE's data are not sufficient to calculate some of the rankings. For example, on function 8, maybe the mean error of DE is not the smallest.

TABLE IV: Mean errors and average rankings of 8 algorithms on the CEC2013 benchmark suite

F.	ABC	DE	SPSO	CMA-ES	EFWA	AFWA	dynFWA	GFWA
1	**0.00E+00**	1.89E-03	**0.00E+00**	**0.00E+00**	7.82E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**
2	6.20E+06	5.52E+04	3.38E+05	**0.00E+00**	5.43E+05	8.93E+05	7.87E+05	6.96E+05
3	5.74E+08	2.16E+06	2.88E+08	**1.41E+01**	1.26E+08	1.26E+08	1.57E+08	3.74E+07
4	8.75E+04	1.32E-01	3.86E+04	**0.00E+00**	1.09E+00	1.15E+01	1.28E+01	5.02E-05
5	**0.00E+00**	2.48E-03	5.42E-04	**0.00E+00**	7.90E-02	6.04E-04	5.42E-04	1.55E-03
6	1.46E+01	7.82E+00	3.79E+01	**7.82E-02**	3.49E+01	2.99E+01	3.15E+01	3.49E+01
7	1.25E+02	4.89E+01	8.79E+01	**1.91E+01**	1.33E+02	9.19E+01	1.03E+02	7.58E+01
8	2.09E+01	**2.09E+01**	2.09E+01	2.14E+01	2.10E+01	2.09E+01	2.09E+01	2.09E+01
9	3.01E+01	**1.59E+01**	2.88E+01	4.81E+01	3.19E+01	2.48E+01	2.56E+01	1.83E+01
10	2.27E-01	3.24E-02	3.40E-01	**1.78E-02**	8.29E-01	4.73E-02	4.20E-02	6.08E-02
11	**0.00E+00**	7.88E+01	1.05E+02	4.00E+02	4.22E+02	1.05E+02	1.07E+02	7.50E+01
12	3.19E+02	**8.14E+01**	1.04E+02	9.42E+02	6.33E+02	1.52E+02	1.56E+02	9.41E+01
13	3.29E+02	**1.61E+02**	1.94E+02	1.08E+03	4.51E+02	2.36E+02	2.44E+02	1.61E+02
14	**3.58E-01**	2.38E+03	3.99E+03	4.94E+03	4.16E+03	2.97E+03	2.95E+03	3.49E+03
15	3.88E+03	5.19E+03	3.81E+03	5.02E+03	4.13E+03	3.81E+03	3.71E+03	**3.67E+03**
16	1.07E+00	1.97E+00	1.31E+00	**5.42E-02**	5.92E-01	4.97E-01	4.77E-01	1.00E-01
17	**3.04E+01**	9.29E+01	1.16E+02	7.44E+02	3.10E+02	1.45E+02	1.48E+02	8.49E+01
18	3.04E+02	2.34E+02	1.21E+02	5.17E+02	1.75E+02	1.75E+02	1.89E+02	**8.60E+01**
19	**2.62E-01**	4.51E+00	9.51E+00	3.54E+00	1.23E+01	6.92E+00	6.87E+00	5.08E+00
20	1.44E+01	1.43E+01	1.35E+01	1.49E+01	1.46E+01	**1.30E+01**	1.30E+01	1.31E+01
21	1.65E+02	3.20E+02	3.09E+02	3.44E+02	3.24E+02	3.16E+02	2.92E+02	**2.59E+02**
22	**2.41E+01**	1.72E+03	4.30E+03	7.97E+03	5.75E+03	3.45E+03	3.41E+03	4.27E+03
23	4.95E+03	5.28E+03	4.83E+03	6.95E+03	5.74E+03	4.70E+03	4.55E+03	**4.32E+03**
24	2.90E+02	**2.47E+02**	2.67E+02	6.62E+02	3.37E+02	2.70E+02	2.72E+02	2.56E+02
25	3.06E+02	**2.80E+02**	2.99E+02	4.41E+02	3.56E+02	2.99E+02	2.97E+02	2.89E+02
26	**2.01E+02**	2.52E+02	2.86E+02	3.29E+02	3.21E+02	2.73E+02	2.62E+02	2.05E+02
27	**4.16E+02**	7.64E+02	1.00E+03	5.39E+02	1.28E+03	9.72E+02	9.92E+02	8.15E+02
28	**2.58E+02**	4.02E+02	4.01E+02	4.78E+03	4.34E+03	4.37E+02	3.40E+02	3.60E+02
AR.	4.11	3.43	4.79	5.25	6.71	4.14	4.00	**3.00**

TABLE V: Test functions of CEC2010 large scale global optimization competition

	No.	Name
Separable Functions	1	Shifted Elliptic Function
	2	Shifted Rastrigin's Function
	3	Shifted Ackley's Function
Single-group m-nonseparable Functions	4	Single-group Shifted and m-rotated Elliptic Function
	5	Single-group Shifted and m-rotated Rastrigin's Function
	6	Single-group Shifted and m-rotated Ackley's Function
	7	Single-group Shifted m-dimensional Schwefel's Problem 1.2
	8	Single-group Shifted m-dimensional Rosenbrock's Function
D/2m-group m-nonseparable Functions	9	D/2m-group Shifted and m-rotated Elliptic Function
	10	D/2m-group Shifted and m-rotated Rastrigin's Function
	11	D/2m-group Shifted and m-rotated Ackley's Function
	12	D/2m-group Shifted m-dimensional Schwefel's Problem 1.2
	13	D/2m-group Shifted m-dimensional Rosenbrock's Function
D/m-group m-nonseparable Functions	14	D/m-group Shifted and m-rotated Elliptic Function
	15	D/m-group Shifted and m-rotated Rastrigin's Function
	16	D/m-group Shifted and m-rotated Ackley's Function
	17	D/m-group Shifted m-dimensional Schwefel's Problem 1.2
	18	D/m-group Shifted m-dimensional Rosenbrock's Function
Nonseparable Functions	19	Shifted Schwefel's Problem 1.2
	20	Shifted Rosenbrock's Function

sion selection, and $\mathbf{\Delta}$ *is calculated by Algorithm 2. For any* $k \neq 1$*, with probability at least* $1-\delta$*,*

$$|\mathbf{\Delta}^{(k)}| \leqslant \sqrt{\frac{2}{3\sigma\lambda\delta}}A. \tag{10}$$

Proof: Without loss of generality, assume $\mathbf{X}^{(k)} = 0$. Let $\widehat{\mathbf{\Delta}} \triangleq \frac{1}{\sigma\lambda}(\sum_{j=1}^{\sigma\lambda}\mathbf{s}_j)$, $\breve{\mathbf{\Delta}} \triangleq \frac{1}{\sigma\lambda}(\sum_{j=\lambda-\sigma\lambda+1}^{\lambda}\mathbf{s}_j)$, then

$$\mathbf{\Delta} = \widehat{\mathbf{\Delta}} - \breve{\mathbf{\Delta}}. \tag{11}$$

Note that since $f(\mathbf{x}) = x_1^2$, the values of the sparks on dimension k have no influence on their evaluation values, so $\mathbf{s}_j^{(k)} \sim \mathcal{U}(-A, A)$ and their covariances are zero. Then,

$$E[\widehat{\mathbf{\Delta}}^{(k)}] = E[\frac{1}{\sigma\lambda}\sum_{j=1}^{\sigma\lambda}\mathbf{s}_j^{(k)}] = \frac{1}{\sigma\lambda}\sum_{j=1}^{\sigma\lambda}E[\mathbf{s}_j^{(k)}] = E[\mathbf{s}_j^{(k)}] = 0, \tag{12}$$

TABLE VI: Mean errors, standard deviations and average rankings of the three algorithms on CEC2010's large scale optimization benchmark

	DECC-G		MLCC		GFWA	
F.	Mean	Std.	Mean	Std.	Mean	Std.
1	2.93E-07	8.62E-08	**1.53E-27**	**7.66E-27**	4.22E+07	3.40E+06
2	1.31E+03	3.26E+01	**5.57E-01**	**2.21E+00**	4.74E+03	7.79E+02
3	1.39E+00	9.73E-02	**9.88E-13**	**3.70E-12**	1.31E+01	7.31E+00
4	1.70E+13	5.37E+12	9.61E+12	3.43E+12	**5.70E+11**	**9.49E+10**
5	2.63E+08	8.44E+07	3.84E+08	6.93E+07	**1.48E+08**	**3.34E+07**
6	4.96E+06	8.02E+05	1.62E+07	4.97E+06	**2.15E+01**	**4.01E-02**
7	1.63E+08	1.37E+08	**6.89E+05**	**7.37E+05**	4.73E+06	1.43E+05
8	6.44E+07	2.89E+07	**4.38E+07**	**3.45E+07**	5.52E+07	1.63E+08
9	3.21E+08	3.38E+07	**1.23E+08**	**1.33E+07**	1.72E+08	1.65E+07
10	1.06E+04	2.95E+02	**3.43E+03**	**8.72E+02**	4.62E+03	6.12E+02
11	**2.34E+01**	**1.78E+00**	1.98E+02	6.98E-01	1.09E+02	3.76E+01
12	8.93E+04	6.87E+03	3.49E+04	4.92E+03	**4.75E+03**	**8.08E+02**
13	5.12E+03	3.95E+03	**2.08E+03**	**7.27E+02**	9.66E+05	2.60E+05
14	8.08E+08	6.07E+07	3.16E+08	2.77E+07	**2.17E+08**	**2.53E+07**
15	1.22E+04	8.97E+02	7.11E+03	1.34E+03	**4.83E+03**	**6.20E+02**
16	**7.66E+01**	**8.14E+00**	3.76E+02	4.71E+01	2.86E+02	9.78E+01
17	2.87E+05	1.98E+04	1.59E+05	1.43E+04	**5.93E+04**	**1.30E+04**
18	2.46E+04	1.05E+04	**7.09E+03**	**4.77E+03**	4.11E+04	1.56E+04
19	1.11E+06	5.15E+04	1.36E+06	7.35E+04	**8.06E+05**	**4.74E+04**
20	4.06E+03	3.66E+02	2.05E+03	1.80E+02	**9.82E+02**	**2.76E+01**
AR.	2.4		**1.8**		**1.8**	

$$\begin{aligned}\mathrm{Var}[\widehat{\boldsymbol{\Delta}}^{(k)}] &= E[(\widehat{\boldsymbol{\Delta}}^{(k)} - E(\widehat{\boldsymbol{\Delta}}^{(k)}))^2] \\ &= E[(\widehat{\boldsymbol{\Delta}}^{(k)})^2] \\ &= E[\frac{1}{(\sigma\lambda)^2}(\sum_{j=1}^{\sigma\lambda}\mathbf{s}_j^{(k)})^2] \\ &= \frac{1}{(\sigma\lambda)^2}E[\sum_{j=1}^{\sigma\lambda}(\mathbf{s}_j^{(k)})^2] \\ &= \frac{1}{(\sigma\lambda)^2}\sigma\lambda\frac{(A-(-A))^2}{12} \\ &= \frac{A^2}{3\sigma\lambda}. \end{aligned} \tag{13}$$

So,

$$E[\boldsymbol{\Delta}^{(k)}] = E[\widehat{\boldsymbol{\Delta}}^{(k)} - \breve{\boldsymbol{\Delta}}^{(k)}] = 0, \tag{14}$$

Since the distributions of $\widehat{\boldsymbol{\Delta}}^{(k)}$ and $\breve{\boldsymbol{\Delta}}^{(k)}$ are independent of each other, their covariance is zero.

$$\begin{aligned}\mathrm{Var}[\boldsymbol{\Delta}^{(k)}] &= \mathrm{Var}[\widehat{\boldsymbol{\Delta}}^{(k)} - \breve{\boldsymbol{\Delta}}^{(k)}] \\ &= \mathrm{Var}[\widehat{\boldsymbol{\Delta}}^{(k)}] + \mathrm{Var}[\breve{\boldsymbol{\Delta}}^{(k)}] - 2\mathrm{Cov}[\widehat{\boldsymbol{\Delta}}^{(k)}, \breve{\boldsymbol{\Delta}}^{(k)}] \\ &= \frac{2A^2}{3\sigma\lambda}. \end{aligned} \tag{15}$$

Using Chebyshev's inequality, we have

$$\Pr\{|\boldsymbol{\Delta}^{(k)} - 0| \geqslant l\} \leqslant \frac{2A^2}{3\sigma\lambda l^2} \tag{16}$$

holds for any $l > 0$.

Take $\delta = \frac{2A^2}{3\sigma\lambda l^2}$ and the theorem follows. ■

APPENDIX B
PROOF OF THEOREM 2

Lemma 1 (Order Statistic)**.** *Suppose $V_1, V_2, \ldots, V_n \sim \mathcal{U}(0,1)$, and $V_{(1)}, V_{(2)}, \ldots, V_{(n)}$ are random variables sorting them in ascending order, then*

$$V_{(k)} \sim \mathcal{B}(k, n+1-k), \tag{17}$$

and

$$\Pr\{V_{(k)} < x\} = I_x(k, n+1-k), \tag{18}$$

where

$$I_x(a,b) = \sum_{j=a}^{a+b-1}\frac{(a+b-1)!}{j!(a+b-1-j)!}x^j(1-x)^{a+b-1-j} \tag{19}$$

is the cumulative distribution function of the Beta distribution [64].

The proof of Lemma 1 can be found in [65].

Theorem 2 (Step Size for Exploration)**.** *Assume $f(\mathbf{x}) = x_1^2$, $\mathbf{s}_j$ are calculated by Algorithm 1 without dimension selection, and $\boldsymbol{\Delta}$ is calculated by Algorithm 2. If $\mathbf{X}^{(1)} \geqslant A > 0$, $\sigma \leqslant 0.5$, and λ is sufficiently large, then with probability at least $1-\delta$,*

$$|\boldsymbol{\Delta}^{(1)}| > (2(1-\sigma)^{1-\sigma}\sigma^{\sigma}\delta^{\frac{1}{\lambda}} - 1)A. \tag{20}$$

Proof: Because $f(\mathbf{x}) = x_1^2$ and $\mathbf{X}^{(1)} \geqslant A > 0$, all the top sparks are located on the left of all the other sparks and all the bottom sparks are located on the right of all the other sparks. For any $0 < l < A$,

$$\begin{aligned}\Pr\{\boldsymbol{\Delta}^{(1)} < -l\} &\geqslant \Pr\{\widehat{\boldsymbol{\Delta}}^{(1)} - \mathbf{X}^{(1)} < -l\} \\ &= \Pr\{\frac{1}{\sigma\lambda}\sum_{j=1}^{\sigma\lambda}\mathbf{s}_j^{(1)} - \mathbf{X}^{(1)} < -l\} \\ &\geqslant \Pr\{\mathbf{s}_{\sigma\lambda}^{(1)} - \mathbf{X}^{(1)} < -l\} \\ &= \Pr\{\frac{\mathbf{s}_{\sigma\lambda}^{(1)} - \mathbf{X}^{(1)}}{2A} + 0.5 < 0.5 - \frac{l}{2A}\} \\ &= I_{0.5-\frac{l}{2A}}(\sigma\lambda, \lambda - \sigma\lambda + 1) \\ &= 1 - I_{0.5+\frac{l}{2A}}(\lambda - \sigma\lambda + 1, \sigma\lambda). \end{aligned} \tag{21}$$

Let $x = 0.5 + \frac{l}{2A} \in (0.5, 1)$,

$$\Pr\{\boldsymbol{\Delta}^{(1)} < -l\} \geqslant 1 - \sum_{j=\lambda-\sigma\lambda+1}^{\lambda} \frac{\lambda!}{j!(\lambda-j)!} x^j (1-x)^{\lambda-j} \geqslant 1 - \sigma\lambda \frac{\lambda!}{(\lambda-\sigma\lambda+1)!(\sigma\lambda-1)!} x^{\lambda}. \quad (22)$$

Take $\delta = \sigma\lambda \frac{\lambda!}{(\lambda-\sigma\lambda+1)!(\sigma\lambda-1)!} x^{\lambda}$,

$$x = \sqrt[\lambda]{\frac{\delta(\lambda-\sigma\lambda+1)!(\sigma\lambda-1)!}{\sigma\lambda \cdot \lambda!}}. \quad (23)$$

Using Stirling's approximation [66], when λ is sufficiently large,

$$x \approx \frac{(\lambda-\sigma\lambda+1)^{\frac{\lambda-\sigma\lambda+1}{\lambda}} (\sigma\lambda-1)^{\frac{\sigma\lambda-1}{\lambda}} \delta^{\frac{1}{\lambda}}}{\lambda} \approx (1-\sigma)^{1-\sigma} \sigma^{\sigma} \delta^{\frac{1}{\lambda}} \quad (24)$$

■

APPENDIX C
PROOF OF THEOREM 3

Lemma 2 (Symmetrical Distribution)**.** *If X and Y are two random variables such that $E(X) = E(Y) = 0$, $P(X = x|Y = y) = P(X = -x|Y = y)$ holds for any y and $P(Y = y|X = x) = P(Y = -y|X = x)$ holds for any x,*

$$\mathrm{Cov}[X, Y] = 0. \quad (25)$$

Proof: Firstly, we prove the joint probability density function is even,

$$\begin{aligned} P(X = x, Y = y) &\triangleq p(x, y) \\ &= p(y)p(x|y) \\ &= p(y)p(-x|y) \\ &= p(-x, y) \\ &= p(x, -y) \\ &= p(-x, -y). \end{aligned} \quad (26)$$

Then,

$$\begin{aligned} &\mathrm{Cov}[X, Y] = E[X, Y] - E[X]E[Y] = E[X, Y] \\ &= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} p(x, y)xydxdy \\ &= \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy + \int_{-\infty}^{0}\int_{0}^{+\infty} p(x, y)xydxdy \\ &+ \int_{0}^{+\infty}\int_{-\infty}^{0} p(x, y)xydxdy + \int_{-\infty}^{0}\int_{-\infty}^{0} p(x, y)xydxdy \\ &= \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy + \int_{+\infty}^{0}\int_{0}^{+\infty} p(x, -y)x(-y)dxd(-y) \\ &+ \int_{0}^{+\infty}\int_{+\infty}^{0} p(-x, y)(-x)yd(-x)dy \\ &+ \int_{+\infty}^{0}\int_{+\infty}^{0} p(-x, -y)(-x)(-y)d(-x)d(-y) \\ &= \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy + \int_{+\infty}^{0}\int_{0}^{+\infty} p(x, -y)xydxdy \\ &+ \int_{0}^{+\infty}\int_{+\infty}^{0} p(-x, y)xydxdy + \int_{+\infty}^{0}\int_{+\infty}^{0} p(-x, -y)xydxdy \\ &= \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy - \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy \\ &- \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy + \int_{0}^{+\infty}\int_{0}^{+\infty} p(x, y)xydxdy \\ &= 0. \end{aligned} \quad (27)$$

■

Theorem 3 (Convergence of the Step Size with Distance)**.** *Assume $f(\mathbf{x}) = x_1^2$, $\mathbf{s}_j$ are calculated by Algorithm 1 without dimension selection, and $\boldsymbol{\Delta}$ is calculated by Algorithm 2. If $\mathbf{X}^{(1)} = 0$, with probability at least $1 - \delta$,*

$$|\boldsymbol{\Delta}^{(1)}| \leqslant \sqrt{\frac{4}{3\sigma\lambda\delta}} A. \quad (28)$$

Proof: This theorem is also an example of the law of the large numbers, but here the distributions of the top and bottom sparks are different. Obviously,

$$E[\widehat{\boldsymbol{\Delta}}^{(1)}] = E[\breve{\boldsymbol{\Delta}}^{(1)}] = 0. \quad (29)$$

Since $f(\mathbf{x}) = x_1^2$, the top sparks are those that are closest to the origin. So their variance is smaller than that of uniform distribution,

$$\mathrm{Var}[\widehat{\boldsymbol{\Delta}}^{(1)}] \leqslant \frac{A^2}{3\sigma\lambda}. \quad (30)$$

While the bottom explosion sparks are what's farthest from the origin. We only have $\mathrm{Var}[\mathbf{s}_j^{(1)}] \leqslant A^2$. By Lemma 2, their covariances are 0, so

$$\begin{aligned}\mathrm{Var}[\breve{\mathbf{\Delta}}^{(1)}] &= \mathrm{Var}[\frac{1}{\sigma\lambda}\sum_{j=\lambda-\sigma\lambda+1}^{\lambda}\mathbf{s}_j^{(1)}] \\ &= (\frac{1}{\sigma\lambda})^2\mathrm{Var}[\sum_{j=\lambda-\sigma\lambda+1}^{\lambda}\mathbf{s}_j^{(1)}] \\ &= (\frac{1}{\sigma\lambda})^2\sum_{j=\lambda-\sigma\lambda+1}^{\lambda}\mathrm{Var}[\mathbf{s}_j^{(1)}] \leqslant \frac{A^2}{\sigma\lambda} \end{aligned} \tag{31}$$

Since $f(\mathbf{x}) = x_1^2 = |x_1|^2$, all the sparks are sorted by their absolute values, the distributions of $\widehat{\mathbf{\Delta}}^{(1)}$, $\breve{\mathbf{\Delta}}^{(1)}$ and their conditional distributions are symmetrical about the origin,

$$\mathrm{Cov}[\widehat{\mathbf{\Delta}}^{(1)}, \breve{\mathbf{\Delta}}^{(1)}] = 0. \tag{32}$$

In sum,

$$\begin{aligned}\mathrm{Var}[\mathbf{\Delta}^{(1)}] &= \mathrm{Var}[\widehat{\mathbf{\Delta}}^{(1)} - \breve{\mathbf{\Delta}}^{(1)}] \\ &= \mathrm{Var}[\widehat{\mathbf{\Delta}}^{(1)}] + \mathrm{Var}[\breve{\mathbf{\Delta}}^{(1)}] - 2\mathrm{Cov}[\widehat{\mathbf{\Delta}}^{(1)}, \breve{\mathbf{\Delta}}^{(1)}] \\ &\leqslant \frac{4A^2}{3\sigma\lambda}. \end{aligned} \tag{33}$$

Using Chebyshev's inequality, we have

$$\Pr\{|\mathbf{\Delta}^{(1)} - 0| \geqslant l\} \leqslant \frac{4A^2}{3\sigma\lambda l^2} \tag{34}$$

holds for any $l > 0$. ■

ACKNOWLEDGMENT

The authors would like to thank Ke Ding, Jie Li, Chao Yu, anonymous reviewers and editors for their invaluable comments.

Prof. Y. Tan is the corresponding author. This work was supported by the Natural Science Foundation of China (NSFC) under grant no. 61375119 and Supported by Beijing Natural Science Foundation (4162029), and partially supported by National Key Basic Research Development Plan (973 Plan) Project of China under grant no. 2015CB352302.

REFERENCES

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

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

[3] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *Evolutionary Computation, IEEE Transactions on*, vol. 3, no. 2, pp. 124–141, 1999.

[4] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft computing*, vol. 9, no. 1, pp. 3–12, 2005.

[5] P. Larranaga, "A review on estimation of distribution algorithms," in *Estimation of distribution algorithms*. Springer, 2002, pp. 57–100.

[6] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," in *Advances in Swarm Intelligence*, ser. Lecture Notes in Computer Science, Y. Tan, Y. Shi, and K. Tan, Eds. Springer Berlin Heidelberg, 2010, vol. 6145, pp. 355–364. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-13495-1_44

[7] 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.

[8] 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. IEEE, 2011, pp. 1668–1672.

[9] ——, "Swarm intelligence for non-negative matrix factorization," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 2, no. 4, pp. 12–34, 2011.

[10] ——, "Using population based algorithms for initializing nonnegative matrix factorization," in *Proceedings of the second international conference on Advances in swarm intelligence*, ser. ICSI'11. Springer-Verlag, 2011, pp. 307–316.

[11] 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*. Springer, 2013, pp. 439–450.

[12] S. Zheng and Y. Tan, "A unified distance measure scheme for orientation coding in identification." in *Information Science and Technology, 2013 IEEE Congress on*. IEEE, 2013, pp. 979–985.

[13] A. Mohamed Imran and M. Kowsalya, "A new power system reconfiguration scheme for power loss minimization and voltage profile enhancement using fireworks algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 62, pp. 312–322, 2014.

[14] N. H. Abdulmajeed and M. Ayob, "A firework algorithm for solving capacitated vehicle routing problem." *International Journal of Advancements in Computing Technology*, vol. 6, no. 1, 2014.

[15] N. Pholdee and S. Bureerat, "Comparative performance of meta-heuristic algorithms for mass minimisation of trusses with dynamic constraints," *Advances in Engineering Software*, vol. 75, pp. 1–13, 2014.

[16] D. Goswami and S. Chakraborty, "A study on the optimization performance of fireworks and cuckoo search algorithms in laser machining processes," *Journal of The Institution of Engineers (India): Series C*, pp. 1–15, 2014.

[17] Z. Zheng and Y. Tan, "Group explosion strategy for searching multiple targets using swarm robotic," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 821–828.

[18] Z. Zheng, J. Li, J. Li, and Y. Tan, "Improved group explosion strategy for searching multiple targets using swarm robotics," in *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 246–251.

[19] ——, "Avoiding decoys in multiple targets searching problems using swarm robotics," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 784–791.

[20] X. Yang and Y. Tan, "Sample index based encoding for clustering using evolutionary computation," in *Advances in Swarm Intelligence*. Springer, 2014, pp. 489–498.

[21] M. Tuba, N. Bacanin, and A. Alihodzic, "Multilevel image thresholding by fireworks algorithm," in *Radioelektronika (RADIOELEKTRONIKA), 2015 25th International Conference*. IEEE, 2015, pp. 326–330.

[22] M. Tuba, N. Bacanin, and M. Beko, "Fireworks algorithm for RFID network planning problem," in *Radioelektronika (RADIOELEKTRONIKA), 2015 25th International Conference*. IEEE, 2015, pp. 440–444.

[23] Z. Liu, Z. Feng, and L. Ke, "Fireworks algorithm for the multi-satellite control resource scheduling problem," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, 2015, pp. 1280–1286.

[24] N. Bacanin and M. Tuba, "Fireworks algorithm applied to constrained portfolio optimization problem," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, 2015, pp. 1242–1249.

[25] K. Ding, Y. Chen, Y. Wang, and Y. Tan, "Regional seismic waveform inversion using swarm intelligence algorithms," in *Evolutionary Computation (CEC), 2015 IEEE Congress on*. IEEE, 2015, pp. 1235–1241.

[26] H. A. Bouarara, R. M. Hamou, A. Amine, and A. Rahmani, "A fireworks algorithm for modern web information retrieval with visual results mining," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 6, no. 3, pp. 1–23, 2015.

[27] M. Alamaniotis, C. K. Choi, and L. H. Tsoukalas, "Application of fireworks algorithm in gamma-ray spectrum fitting for radioisotope identification," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 6, no. 2, pp. 102–125, 2015.

[28] A. Lihu and t. Holban, "De novo motif prediction using the fireworks algorithm," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 6, no. 3, pp. 24–40, 2015.

[29] L. K. Panwar, S. Reddy, and R. Kumar, "Binary fireworks algorithm based thermal unit commitment," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 6, no. 2, pp. 87–101, 2015.

[30] A. Rahmani, A. Amine, R. M. Hamou, M. E. Rahmani, and H. A. Bouarara, "Privacy preserving through fireworks algorithm based model for image perturbation in big data," *International Journal of Swarm Intelligence Research (IJSIR)*, vol. 6, no. 3, pp. 41–58, 2015.

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

[32] Y. Tan, *Fireworks Algorithm*. Springer, 2015.

[33] C. A. R. Hoare, "Algorithm 65: find," *Communications of the ACM*, vol. 4, no. 7, pp. 321–322, 1961.

[34] https://en.wikipedia.org/wiki/Quickselect.

[35] http://en.wikipedia.org/wiki/Law_of_large_numbers.

[36] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1. New York, NY, 1995, pp. 39–43.

[37] F. Van Den Bergh, "An analysis of particle swarm optimizers," Ph.D. dissertation, University of Pretoria, 2006.

[38] 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*. IEEE, 2014, pp. 3207–3213.

[39] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 3. IEEE, 1999.

[40] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[41] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of global optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[42] P. Civicioglu and E. Besdok, "A conceptual comparison of the cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms," *Artificial Intelligence Review*, vol. 39, no. 4, pp. 315–346, 2013.

[43] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*. IEEE, 1996, pp. 312–317.

[44] B. Zhang, Y. Zheng, M. Zhang, and S. Chen, "Fireworks algorithm with enhanced fireworks interaction," *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, vol. PP, no. 99, 2015.

[45] S. Zheng, J. Li, A. Janecek, and Y. Tan, "A cooperative framework for fireworks algorithm," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. PP, no. 99, pp. 1–1, 2015.

[46] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, vol. 2. IEEE, 2005, pp. 1769–1776.

[47] I. Loshchilov, M. Schoenauer, and M. Sebag, "Alternative restart strategies for CMA-ES," in *Parallel Problem Solving from Nature-PPSN XII*. Springer, 2012, pp. 296–305.

[48] V. V. De Melo and G. Iacca, "A modified covariance matrix adaptation evolution strategy with adaptive penalty function and restart for constrained optimization," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7077–7094, 2014.

[49] R. Ros and N. Hansen, "A simple modification in CMA-ES achieving linear time and space complexity," in *Parallel Problem Solving from Nature–PPSN X*. Springer, 2008, pp. 296–305.

[50] J. Liang, B. Qu, P. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," 2013.

[51] M. El-Abd, "Testing a particle swarm optimization and artificial bee colony hybrid algorithm on the CEC13 benchmarks," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 2215–2220.

[52] M. Clerc, "Standard particle swarm optimization, from 2006 to 2011," *Particle Swarm Central*, 2011.

[53] M. Zambrano-Bigiarini, M. Clerc, and R. Rojas, "Standard particle swarm optimisation 2011 at CEC-2013: A baseline for future PSO improvements," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 2337–2344.

[54] N. Padhye, P. Mittal, and K. Deb, "Differential evolution: Performances and analyses," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*. IEEE, 2013, pp. 1960–1967.

[55] http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/CEC2013/Results-of-22-papers.zip.

[56] https://www.lri.fr/~hansen/purecmaes.m.

[57] http://www.cil.pku.edu.cn/research/fwa/code/matlab_efwa.zip.

[58] http://www.cil.pku.edu.cn/research/fwa/code/AFWA-dim30.zip.

[59] http://www.cil.pku.edu.cn/research/fwa/code/matlab_dynFWA.zip.

[60] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC2010 special session and competition on large-scale global optimization," *Rapport technique, University of Science and Technology of China (USTC), School of Computer Science and Technology, Nature Inspired Computation and Applications Laboratory (NICAL), Héféi,© Anhu© i, China*, 2009.

[61] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Information Sciences*, vol. 178, no. 15, pp. 2985–2999, 2008.

[62] ——, "Multilevel cooperative coevolution for large scale optimization," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*. IEEE, 2008, pp. 1663–1670.

[63] http://nical.ustc.edu.cn/cec10ss.php.

[64] http://en.wikipedia.org/wiki/Beta_distribution.

[65] J. E. Gentle, *Computational statistics*. Springer, 2009, vol. 308.

[66] http://en.wikipedia.org/wiki/Stirling%27s_approximation.

Junzhi Li received his B.S. at Department of Machine Intelligence and minor degree of Philosophy at Department of Philosophy and of Religious Studies in 2013 from Peking University. He is currently a Ph.D. candidate, at Key Laboratory of Machine Perception (Ministry of Education) and Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. His research interests include Evolutionary Computation, Artificial Neural Networks and Machine Learning.

Shaoqiu Zheng received Beng degree in Department of Acoustic from Northwestern Polytechnical University in June 2010, and PhD in Department of Machine Intelligence at Peking University in June 2015. Currently, he is an engineer at the 28th Research Institute of China Electronics Technology Group Corporation. His research interests include Evolutionary Computation, Swarm Intelligence, Machine Learning, Biometrics and Pattern Recognition.

Ying Tan is a full professor and PhD advisor at the School of Electronics Engineering and Computer Science of Peking University, and director of Computational Intelligence Laboratory at Peking University (PKU). He received his BEng, MS, and PhD from Southeast Univ., in 1985, 1988, and 1997, respectively. He is the inventor of Fireworks Algorithm (FWA). He serves as the Editor-in-Chief of International Journal of Computational Intelligence and Pattern Recognition (IJCIPR), the Associate Editor of IEEE Transaction on Cybernetics (Cyb), the Associate Editor of IEEE Transaction on Neural Networks and Learning Systems (NNLS), International Journal of Artificial Intelligence (IJAI), International Journal of Swarm Intelligence Research (IJSIR), etc. He also served as an Editor of Springers Lecture Notes on Computer Science (LNCS) for more than 12 volumes, and Guest Editors of several referred Journals, including Information Science, Softcomputing, Neurocomputing, IJAI, IJSIR, B&B, CJ, IEEE/ACM Transactions on Computational Biology and Bioinformatics (IEEE/ACM TCBB). He is a member of Emergent Technologies Technical Committee (ETTC), Computational Intelligence Society of IEEE since 2010. He is a senior member of IEEE and ACM and a senior member of the CIE. He is the founder and chair of the ICSI International Conference series. He was the general chair of joint general chair of 1st&2nd BRICS CCI, program committee co-chair of WCCI 2014, etc. His research interests include computational intelligence, swarm intelligence, data mining, pattern recognition, intelligent information processing for information security. He has published more than 260 papers in refereed journals and conferences in these areas, and authored/co-authored 6 books and 10 chapters in book, and received 3 invention patents.