JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 9, No. 2, June 2015, pp. 39-50

An Optimization Algorithm with Novel Flexible Grid: Applications to Parameter Decision in LS-SVM

Weishang Gao*
Information and Engineering College, Dalian University, Dalian, China
gao.weishang@hotmail.com

Cheng Shao
Institute of Advanced Control, Dalian University of Technology, Dalian, China
cshao@dlut.edu.cn

Qin Gao
School of Control Science and Engineering, Dalian University of Technology, Dalian, China
onlygaoqin@gmail.com

Abstract
Genetic algorithm (GA) and particle swarm optimization (PSO) are two excellent approaches to multimodal optimization problems. However, slow convergence or premature convergence readily occurs because of inappropriate and inflexible evolution. In this paper, a novel optimization algorithm with a flexible grid optimization (FGO) is suggested to provide adaptive trade-off between exploration and exploitation according to the specific objective function. Meanwhile, a uniform agents array with adaptive scale is distributed on the gird to speed up the calculation. In addition, a dominance centroid and a fitness center are proposed to efficiently determine the potential guides when the population size varies dynamically. Two types of subregion division strategies are designed to enhance evolutionary diversity and convergence, respectively. By examining the performance on four benchmark functions, FGO is found to be competitive with or even superior to several other popular algorithms in terms of both effectiveness and efficiency, tending to reach the global optimum earlier. Moreover, FGO is evaluated by applying it to a parameter decision in a least squares support vector machine (LS-SVM) to verify its practical competence.

Category: Smart and intelligent computing

Keywords: Optimization algorithm; Swarm intelligence; Evolutionary computation

I. INTRODUCTION

Over the last few years, evolutionary algorithms (EAs) have proven to be a promising approach to complex multimodal optimization problems [1-3]. These algorithms share common methods, exploring the unknown regions near the present optimum conditions. Evolution strategies are general, nature-inspired heuristics for search and optimization, and are especially useful for optimization in the presence of noise [4]. A classic example is the

Open Access http://dx.doi.org/10.5626/JCSE.2015.9.2.39 http://jcse.kiise.org

Received 25 Aug 2014; **Revised** 01 Jan 2015; **Accepted** 30 Mar 2015
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

crossover and mutation of the genetic algorithm (GA) [5-7], which is based on the present superior survivors and retains most of the present optimum gene sequence in the chromosomes of the next generation. However, the gene encoding and decoding in genetic operators probably causes extreme change in the decision space, making GA's agents too diverse to improve the convergence rate. Another typical example of rapid convergence is particle swarm optimization (PSO) [8, 9]. The exploration of each particle is guided by both the global best agent (gBest) and the personal best agent (pBest). Inertia weight is used in PSO to regulate the population diversity and avoid premature convergence to some extent. However, once the inertia weight and other parameters are determined, the evolution feature of PSO will only be fit for particular problems. Thus, it is difficult to set the parameter to an appropriate value because of the complex optimization environment. Many improved algorithms based on the original genetic algorithm (GA) [10, 11], PSO [12, 13] and other EAs [14, 15] have been developed. A simple genes exchange local search policy is suggested in [16] to improve the performance of the standard canonical GA. Several initializations in different zones of the search space are applied in [17] to improve the diversity of particles. A controllable probabilistic particle swarm optimization (CPPSO) algorithm is introduced in [18] based on Bernoulli stochastic variables and a competitive penalized method. A novel genetic algorithm that is non-revisiting by remembering every position it has previously searched is proposed in [19]. A vendor-managed inventory (VMI) model is developed in [20] for PSO to find a near optimum solution. Combining PSO and other intelligent measures is also common in recent papers [21-23]. Accelerating parallel PSO via GPU was proposed in [24]. In many cases, the modifications can be seen as algorithmic components that provide an improved performance, especially performing a better trade-off between exploration and exploitation. Additionally, other types of EAs, such as an immune algorithm, artificial bee colony algorithm (ABC) and artificial fish swarm algorithm (AFSA), have been proposed in many approaches to deal with exploration-exploitation. Furthermore, co-evolutionary algorithms with current popular EAs provide feasible and effective approaches to special problems [25, 26]. However, it is still difficult to estimate an optimal parameter combination [27] that includes the number of individuals [28] necessary for effective exploration-exploitation of the solution space. This difficulty implies that inflexible evolution may not be appropriate in a real-world application. Therefore, new EAs that can adjust their evolution dynamically with particular population structures are suggested such as the rain forest algorithm (RFA) [29] and the bidirectional dynamic diversity EA [30]. The self-adaptive process [31-33] is important for computational optimization algorithms in various applications [34]. To give a simpler adaptive approach, a novel optimization algorithm with flexible grid optimization (FGO) is suggested in this paper to provide an adaptive trade-off between exploration and exploitation according to the information feedback of the specific objective function. The current study proposes a novel dynamic retractive evolution guided by two types of dominant subregions. First, a flexible grid-based agent swarm is suggested in FGO. All the agents are uniformly distributed on the grid, which provides the most effective exploration on the range of the present grid. Second, the FGO algorithm updates the agents' positions iteratively by zooming and panning the grid. The position of each agent is interlocked by one cell of the flexible grid. In the cell, each agent can freely allocate its specific position. Thus, it is convenient and efficient to calculate the new positions of all the agents only by determining the region and the density of next grid. Third, the region of the next grid is divided by two types of subregions which are designed according to the dominance centroid (DC) and fitness center (FC) proposed in this paper. These two subregions will be used alternately in the algorithm to increase the convergence speed and avoid falling into local optimum. Fourth, the density of the next grid (or agents swarm) is based on the emergence of the optimum. To estimate the number of agents required, a fanciful but effective strategy is suggested. The number of agents will be increased when the speed of emergence reduces, and vice versa. Finally, this flexible grid provides the dominant zooming and panning for the agents' swarm, and whenever the sampling information is fed back to the evolution operator, the region and the density of the grid will be readjusted according to the feedback. Thus, the exploration-exploitation can meet the need of the application environment as much as possible.

The remainder of this paper is organized as follows. Section II describes the proposed algorithm, the FGO, in detail. It includes agents' distribution, subregion decision, and density decision for FGO. To verify the performance of FGO, Section III presents the results obtained with the optimization experiments on four benchmark problems. Moreover, the sensitivity analysis of FGO's parameters is mainly discussed in this section. In Section IV, we apply FGO to the parameter decision in a least squares support vector machine (LS-SVM) and obtain excellent results. Finally, a conclusion to the results is presented in Section V.

II. THE PROPOSED ALGORITHM

In this section, the proposed FGO algorithm with a novel adaptive grid is discussed in detail. It is composed of agents' distribution, subregion decision, and density decision throughout the entire evolution process. In the FGO, DC and FC are suggested and serve as dominance guiding to exploration-exploitation. Compact iterative learning is also offered to provide information fusion and make the evolution more appropriate for the unknown

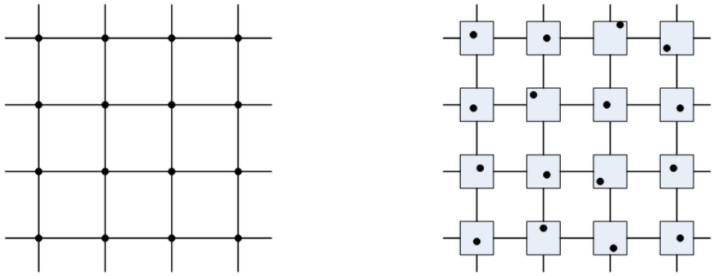


Fig. 1. Agents' distribution. (a) Distribution at intersections and (b) distribution in cells.

objective environment in application.

A. Agents' Distribution

The distribution of the agents' swarm is based on a flexible grid in FGO. The grid divides one decision space into several equal areas with vertical and horizontal lines. Each intersection of the vertical and horizontal lines can represent a sampling distribution of one agent as shown in Fig. 1(a). However, this method will restrict agents' distribution excessively. To improve the population diversity, a crossing cell is proposed in this paper as shown in Fig. 1(b). Thus, the position of each agent is interlocked by one cell instead of one intersection of the flexible grid. In the cell, each agent can freely allocate its specific position as shown by the scattered black spots in Fig. 1(b). In a practical application, the coordinates of agents will be determined by the intersection in a flexible grid, and the FGO operator will then introduce a certain range of random fluctuations.

The changing of agents' distribution between two iterations is based on the zooming and panning of the grid. The scale of agents probably changes dynamically according to the number of cells. Zooming will make agents converge to the center of the present grid for exploitation, and panning will make the agents move to another more dominant area for exploration. Increasing or decreasing the number of cells will adjust the degree of exploration-exploitation. The size of every cell will be set uniformly to adapt to the density of the present grid. A larger interval among gridlines will provide larger cells, which makes FGO's population more diverse and exploration stronger at an early stage of optimization. On the contrary, a smaller interval among gridlines will reserve smaller cells, which restrains the FGO's agents near each intersection and maintain steady exploitation. Thus, after a full exploration, the grid will be guided and shrunk to a very small space, and all the agents will converge to the dominant region. This converging process can be carried out repeatedly to ensure the accuracy of the optimization result. The adaptive changing of the grid is regulated by the FGO operator, which processes the feedback of sampling information in real time. This will ensure that the region and the density of this gird will meet the actual needs of exploration-exploitation in different areas, which is why we call it a flexible grid.

B. Subregion Decision

The subregion suggested here refers to the region of the flexible grid in the next iteration. By designing a subregion for the flexible grid, FGO can carry out grid zooming and panning simultaneously. The evolution of FGO's agents is carried out by a series of subregions throughout the iterations of several generations. Therefore, it is reasonable to plan a subregion considering the guide of the potential dominant agents. DC and FC are proposed here to determine the potential guides rather than the best agents at present. The DC, shown as formula (1), refers to the centroid of the present optimum agents. If only one optimum agent is present, the DC at present is the coordinate of this optimum agent. Obviously, the DC might guide the evolution towards a local optimum. In this case, we add another guiding agent, the FC, shown as formula (2), to adjust the convergence direction.

The formula for calculating the coordinate of DC is shown below:

$$X_{DC} = \left(\begin{bmatrix} \max\limits_{x \in x_1}(x) \\ \max\limits_{x \in x_2}(x) \\ \vdots \\ \max\limits_{x \in x_d}(x) \end{bmatrix} + \begin{bmatrix} \min\limits_{x \in x_1}(x) \\ \min\limits_{x \in x_1}(x) \\ \vdots \\ \min\limits_{x \in x_1}(x) \end{bmatrix} \right) / 2 \tag{1}$$

where x_i refers to the coordinate in the ith dimension of the present optimum agents, and d refers to the dimensions of decision variables. The formula used to calculate the coordinate of FC is as follows:

$$X_{FC} = \sum_{i=1}^{n} \left[X_i \times \frac{F_i - F_{\min}}{\sum\limits_{j=1}^{n} \left(F_j - F_{\min} \right)} \right] \tag{2}$$

where n refers to the population at present, X_i refers to

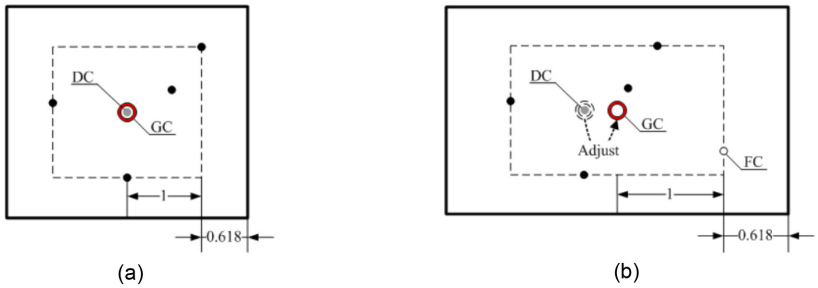


Fig. 2. Subregion decision. (a) Subregion subject to DC and (b) subregion adjusted by FC. DC: dominance centroid, FC: fitness center, GC: grid center.

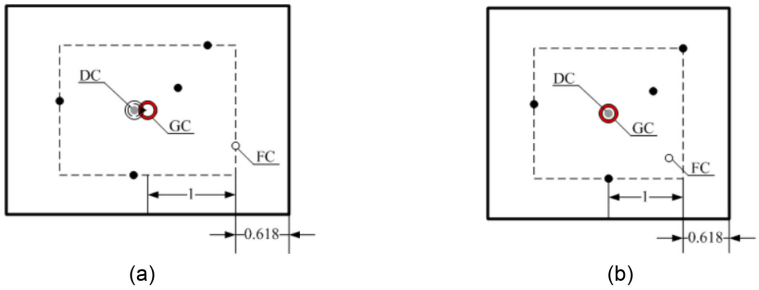


Fig. 3. Subregion adjusted by nearby FC. (a) Adjusting with FC near the bounds and (b) adjusting with FC in the bounds. The bounds refer to the bounds of the present optimum agents. DC: dominance centroid, FC: fitness center, GC: grid center.

one coordinate of all the agents appearing at the present generation, F_i, F_j, and F_{min} refer to the fitness of agent i, the fitness of agent j, and the minimum fitness of present agents, respectively.

Two types of subregion division strategies can be used: one to design only around the DC and another to design around both the DC and the FC. The first strategy involves first dividing the present grid along with the edges that consist of the upper and lower bounds of the present optimum agents. The range of this sectional area is then extended by 0.618 times as shown in Fig. 2(a), and a subregion is designed for the next grid. Obviously, the center of the next grid will be the present DC if we choose this strategy to design the subregion. The second strategy involves first dividing the present grid along with the edges that consist of the upper and lower bounds of the present optimum agents and FC. The range of this sectional area is then also extended by 0.618 times as shown in Fig. 2(b), and another subregion is designed for the next grid. Obviously, the grid center in the next generation will be adjusted away from the DC by the FC. Usually, the subregion obtained by the second division strategy is larger than the one obtained by the first division strategy, especially when the guiding agents fall into local optimum. Thus, the subregion obtained by the first division strategy owns a faster convergence property, while the subregion obtained by the second division strategy can carry out effective exploration outside the convergence area, which is very important for obtaining the global optimum.

In the algorithm flow of FGO, these two types of subregion decision functions will be called separately according to the actual situation. However, whenever the coordinates of both DC and FC are determined, the operator will compare the fitness of these two special positions. If the fitness of DC is not better than that of FC, then FC will adjust the subregion dividing for two succeeding iterations in terms of the second subregion division strategies. This suggests that when there is no significant superiority near the DC, FGO will carry out more exploration for the global optimum. In addition, when the FC is nearby or in the bounds of the present optimum agents, the subregion will mostly divide around the DC as shown in Fig. 3 because the potential dominant areas implied by DC and FC coincide approximately. This indicates that the convergence guided by DC is stronger than that guided by FC. This is also why we need to enhance the adjusting guided by FC in the process of FGO.

C. Density Decision

The density of the grid plays a major role in adaptively assigning the degree of exploration-exploitation according to the actual situation. The density decision proposed in this paper is based on the progress of optimization. Successive progresses between the three latest iterations are compared by FGO to analyze the present situation on the requirement of population as shown in formula (3),

$$f_1\left(\overline{F}_{k-2},\overline{F}_{k-1},\overline{F}_k\right)=\begin{cases}1, \ if\ \ \overline{F}_k-\overline{F}_{k-1}<\overline{F}_{k-1}-\overline{F}_{k-2},\\ 0, \ if\ \ \overline{F}_k-\overline{F}_{k-1}\geq\overline{F}_{k-1}-\overline{F}_{k-2}.\end{cases} \quad (3)$$

where $\overline{F}_k$ refers to the minimum fitness found by FGO until the kth iteration or generation in a minimization problem. Function f_1 is named the situation function, the value of which indicates whether the current population meets the requirement in the current optimization area. 1 and 0 refer to 'True' and 'False', respectively, in the logical operations.

When the situation function, f_1, obtains a value of 'True', FGO will consider that there are successive accelerated progresses and the density of the next grid can be decreased according to formula (4) so that some redundant population will be simplified. If the situation function, f_1, obtains a value of 'False', FGO will consider that there are successive decelerated progresses or no successive progresses and the density of the next grid should be increased according to formula (5). In this case, there are two possibilities. One possibility is that the exploration is relatively inadequate with regard to the features of the present sampling region. If so, a denser grid will be required in the next iteration in order to avoid missing some better potential dominant area. The other possibility is that the exploitation is relatively inadequate in a local or global optimum region. If so, a denser grid should be provided in the next iteration in order to finish the present convergence as soon as possible. In view of the possibility of making the wrong judgment of an unknown environment, the upper and lower bounds of the grid density are also suggested as shown in formulas (4) and (5).

Usually, the density of the next grid can be adjusted by updating the number of mesh lines when the subregion is determined. The number of mesh lines is closely interrelated with the number of agents and the update rules are shown as follows:

$$N_{k+1}=\begin{cases}(1-\alpha)\times N_k+\alpha\times N_{\min}, \ for\ f_1\left(\overline{F}_{k-2},\overline{F}_{k-1},\overline{F}_k\right)=1; \quad (4)\\ (1-\alpha)\times N_k+\alpha\times N_{\max}, \ for\ f_1\left(\overline{F}_{k-2},\overline{F}_{k-1},\overline{F}_k\right)=0. \quad (5)\end{cases}$$

where N_k is a vector that plans the number of mesh lines in each dimension and α is a learning factor which will affect the recognized degree of FGO for the feedback information. $N_{\min}$ and $N_{\max}$ refer to the permissible minimum and maximum numbers, respectively, of mesh lines in each dimension. $N_{\min}$ and $N_{\max}$ are vectors with the same dimensions of N_k and dynamically set the upper and lower bounds, respectively, of the grid density according to the changing subregion.

D. Process of FGO

The steps of the FGO algorithm are explained as follows:

Step 1: Set the value of manipulative parameters such as α, $N_{\min}$, and $N_{\max}$.

Step 2: Generate a grid that covers the entire feasible region with random numbers between $N_{\min}$ and $N_{\max}$ of meshes.

Step 3: Distribute agents randomly in the cells on the intersection of mesh lines and obtain the fitness of each agent.

Step 4: Determine whether to explore or exploit in a subregion, and if so, continue on; otherwise jump to Step 10.

Step 5: Determine the coordinates of DC and FC, then determine whether it is necessary to adjust the subregion with FC according to the fitness of DC and FC. If so, jump to Step 7; otherwise continue.

Step 6: Plan the subregion by using the first strategy (DC-centered) described in subsection II-B, and then jump to Step 8.

Step 7: Plan the subregion by using the second strategy (FC-adjusted) described in subsection II-B.

Step 8: Plan the density of the next grid in the subregion by using the method provided in subsection II-C.

Step 9: Generate the next grid in the subregion and randomly distribute new agents in new cells for the next generation. Then jump to Step 4.

Step 10: Determine whether to reset the grid to a new initial state. If so, jump to Step 2; otherwise end the program.

III. BENCHMARK PROBLEMS

The four benchmark problems frequently employed in the published literature are adopted in this paper to examine the performance of FGO by comparing with other popular algorithms in terms of both effectiveness and efficiency. Table 1 gives details of each benchmark function and shows the search scope selected by the comparison task later in this section. GA and PSO, of which the parameter setting is shown in Table 2, are chosen as comparisons to effectively illustrate the competence of FGO due to their popularity. As FGO can assign multiple cells at the same time with the grid, the samples' distribution process is faster; population diversity is improved by the flexible grid and cells, which help FGO to exit the local optimum earlier and find the global optimum more rapidly and accurately. The sampling density is adjusted with the feedback information of the actual situation, which provides an adaptive exploration-exploitation for FGO and expands the application of the proposed algorithm.

A. Ackley Problem

The Ackley problem is a minimization problem. Originally, this problem was defined for two dimensions, but the problem has been generalized to N dimensions.

Table 1. Benchmark problems adopted in this paper

Name	Expression	Search scope	Optimum
Ackley	$f(X) = -20 \times e^{-0.2 \times \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} - e^{\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i)} + e + 20$	$-13 \le x_i \le 16$, $i = 1, 2$	min $f(X) = 0$, at $X = \mathbf{0}$
Eggholder	$f(X) = -(x_2+47)\sin\left(\sqrt{\left\lvert x_2 + \frac{x_1}{2} + 47 \right\rvert}\right) - x_1 \sin\left(\sqrt{\lvert x_1 - (x_2+47) \rvert}\right)$	$-512 \le x_i \le 512$, $i = 1, 2$	min $f(X) = -959.6407$, at $X = [512, 404.2319]$
Schaffer	$f(X) = 0.5 + \frac{\sin^2(\sqrt{x^2+y^2}) - 0.5}{[1+0.001(x^2+y^2)]^2}$	$-11.3 \le x_i \le 10.7$, $i = 1, 2$	min $f(X) = 0$, at $X = \mathbf{0}$
Schwefel	$f(X) = 418.9829d - \sum_{i=1}^{d} x_i \sin(\sqrt{\lvert x_i \rvert})$	$-500 \le x_i \le 500$, $i = 1, 2$	min $f(X) = 0$, at $x_i = 420.9687$, $i = 1, \ldots, d$

Table 2. Comparison of parameter setting of GA, PSO, and FGO

Algorithm	Population size	Iterations	Other settings
GA	100-150	5000 (MAX)	Probability factor: 0.01-0.1, Encoding length: 15-25
PSO	50-100	2000 (MAX)	$\mathbf{v}_{i,d} \leftarrow \omega \mathbf{v}_{i,d} + \phi_p r_p(\mathbf{p}_{i,d} - \mathbf{x}_{i,d}) + \phi_g r_g(\mathbf{g}_d - \mathbf{x}_{i,d})$, $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$, ω: 0.1-0.9, ϕ_p: 1-1.5, ϕ_g: 1-1.5
FGO	25-225	1000 (Outer×Inner)	$N_{k+1} = \begin{cases} (1-\alpha) \times N_k + \alpha \times N_{min}, \text{ for } f_1(\tilde{F}) = 1, \\ (1-\alpha) \times N_k + \alpha \times N_{max}, \text{ for } f_1(\tilde{F}) = 0. \end{cases}$ $\tilde{F}: \tilde{F}_{k-2}, \tilde{F}_{k-1}, \tilde{F}_k$ $\alpha: \frac{0.618}{1.618}$, $N_{min}: 5$, $N_{max}: 15$

GA: genetic algorithm, PSO: particle swarm optimization, FGO: flexible grid optimization.

The optimum solution of the problem is the vector $X = [0, \cdots, 0]$ with $f(X) = 0$ as shown in Table 1. In order to define an example of this function, we need to provide the dimension of the problem (N). The low dimensional case, $N = 2$, is considered here for ease of experiment design, observation, and analysis. The function distribution of the Ackley problem with two-dimensional decision variables is shown in Fig. 4(a). Obviously, the local optima are tightly packed over the entire decision space.

Fig. 5(a) compares the results of FGO with the other two methods on the benchmark function of Ackley. Obviously, FGO touches the global optimum first and more rapidly than both GA and PSO. After about 0.005 seconds of exploration, FGO finds the global dominant area first and converges to the global optimum very quickly. This illustrates the exploration efficiency with the flexile grid when dealing with problems of which the outline is very similar to a convex optimization. This is because the uniform agents distributed by cells in the grid can quickly focus on a smaller dominant space for further digging. It is very important for evolutionary algorithms to make rapid progress even in the process of exploration; otherwise, the searching will be blinded. On the other hand, the rapid convergence also illuminates the superiority of the subregion dividing strategies suggested in this paper when dealing with the process of exploitation in a dominant space.

B. Eggholder Problem

The Eggholder function is a difficult function to optimize, because of the large number of local minima and the complex distribution as shown in Fig. 4(b). The function is usually evaluated on the square $x_i \in [-512, 512]$, for all $i = 1, 2$ as shown in Table 1. The global minimum is $f(X) = -959.6407$ with optimum decision variable of $X = [512,$

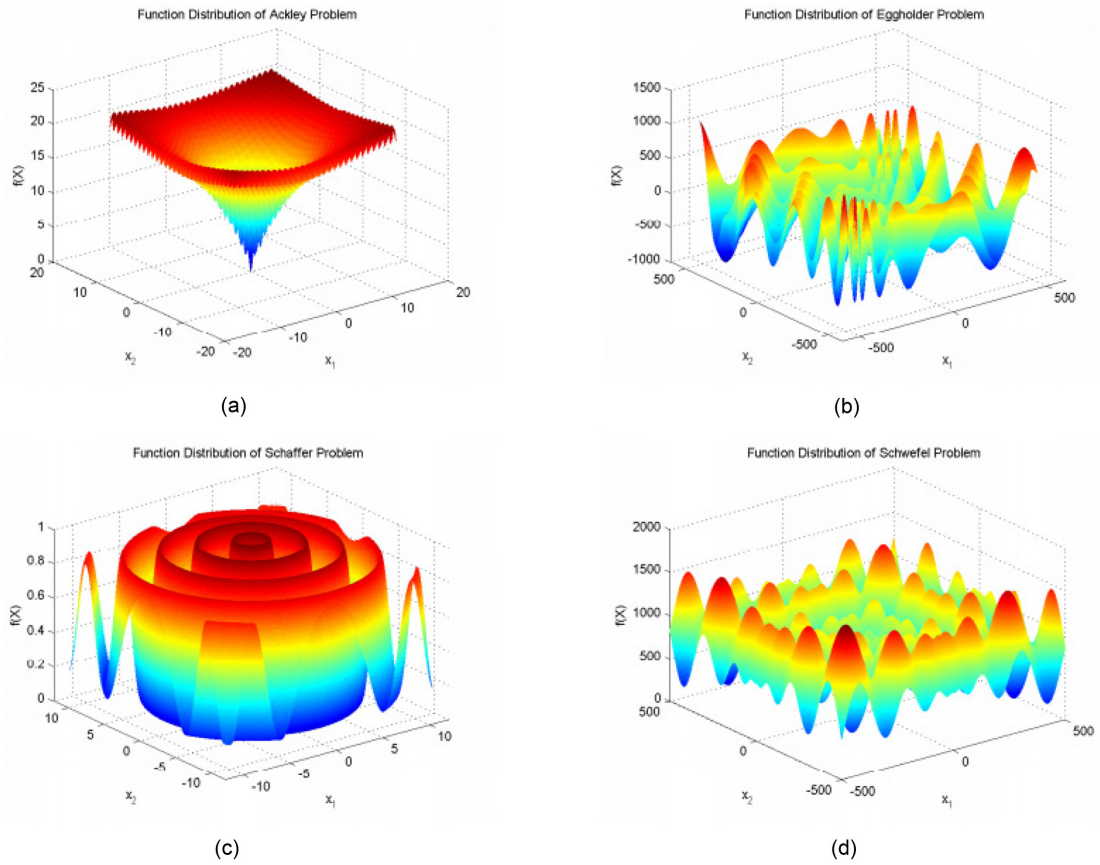


Fig. 4. Function distribution of benchmark functions: (a) Ackley problem, (b) Eggholder problem, (c) Schaffer problem, and (d) Schwefer problem.

404.2319]. This type of marginal optimum will introduce further difficulty to the optimization algorithm. Most of the local optima are very attractive and deceptive, which is appropriate as a valid method of testing the exploration competence of evolutionary algorithms.

Fig. 5(b) compares the results of FGO with the other two methods on the benchmark function of Eggholder. Both FGO and PSO show excellent efficiency in the first 0.02 seconds and find the global optimum $f(X) = -959.6407$. On the other hand, FGO discovered a more dominant area almost at the beginning and continued exploring until about 0.006 seconds, and then converged to the global optimum. PSO required about 0.015 seconds more time than FGO in the process of exploration. This experiment demonstrates that the efficiency of optimizing can be enhanced by the effective exploration of FGO.

C. Schaffer Problem

The two-dimensional Schaffer's function has a single global optimal solution $f(0,0) = 0$ surrounded by infinite local optima and the difference between global optima and the sub-optima is very small. The function distribution of the Schaffer problem is shown in Fig. 4(c). It is the circular peaks and valleys around the global minimum point that trap many methods in one of the local optima areas.

Fig. 5(c) compares the results of FGO with the other two methods on the benchmark function of Schaffer. At the beginning of optimizing, all of these algorithms were trapped in the local optima areas of which the objective functions are distributed near 0.01. However, FGO requires very little time to break through the barriers of circular peaks, and converges to the global optimum rapidly during the first 0.0125 seconds. Both PSO and GA require significantly more time to explore these circular peaks and valleys.

D. Schwefel Problem

The Schwefel function is complex, with many local minima that almost mix up the global optimum. The plot of Fig. 4(d) shows the two-dimensional form of the function. The function is usually evaluated on the square $x_i \in$

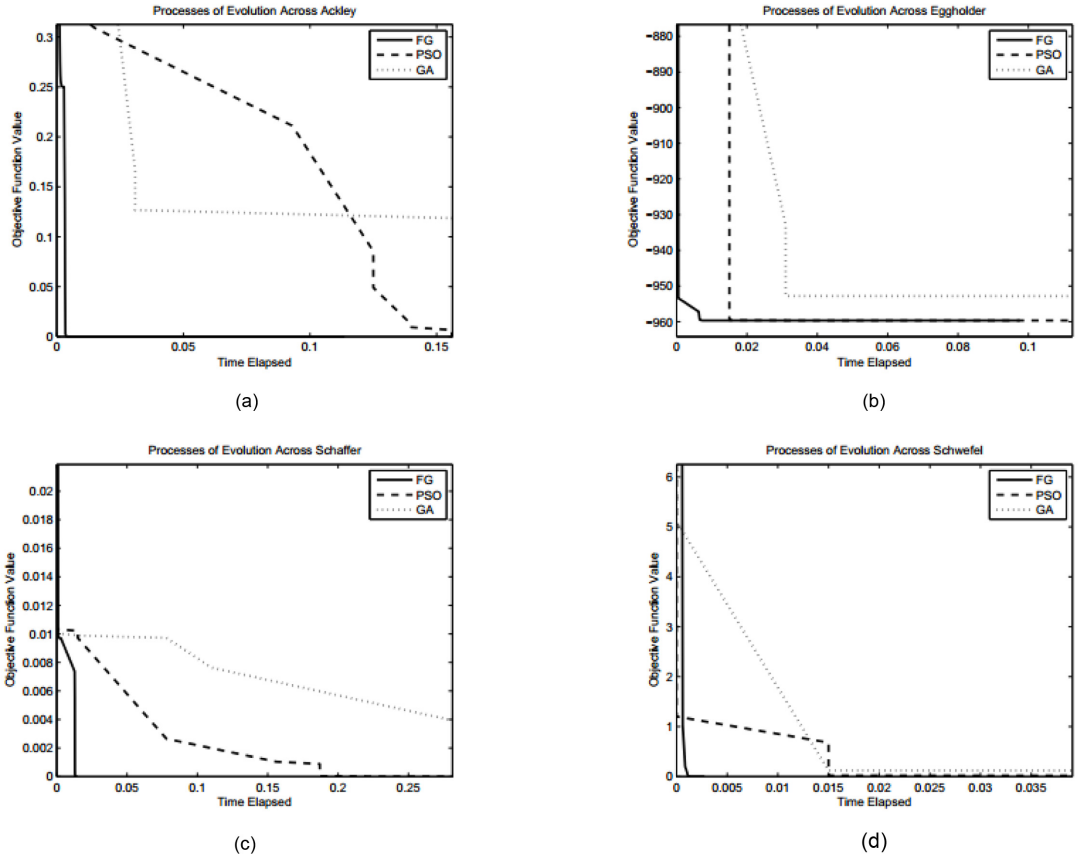


Fig. 5. Processes of evolution across benchmark functions. (a) Ackley function, (b) Eggholder function, (c) Schaffer function, and (d) Schwefer function. FG: flexible grid, PSO: particle swarm optimization, GA: genetic algorithm.

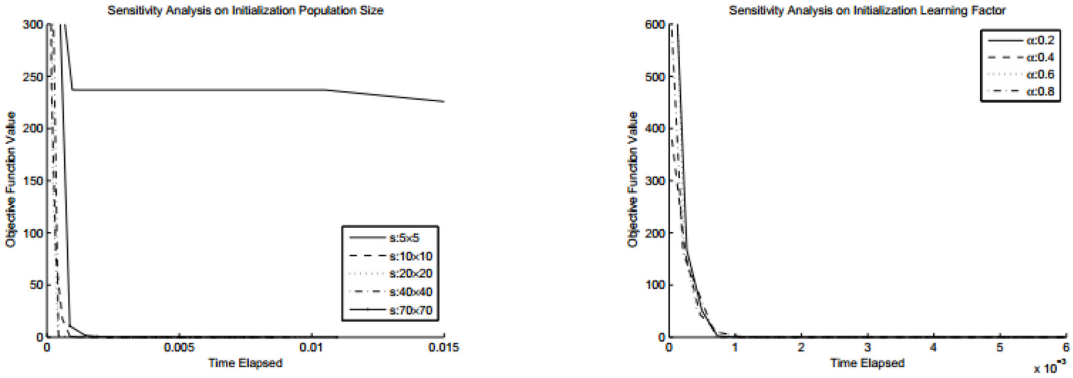
[−500, 500], for all $i = 1, 2$ as shown in Table 1. The optimum solution of this problem is the vector $X = [420.9687, 420.9687]$ with $f(X) = 0$. It is almost a type of marginal optimum that is more difficult to discover.

Fig. 5(d) compares the results of FGO with the other two methods on the benchmark function of Schwefel. Although PSO and GA start from a better situation, FGO discovers the global dominant area first and exploits it quickly during the first 0.00125 seconds. This result reveals that FGO will carry out exploitation very efficiently once it explores a new dominant area, which indicates that FGO's agents can quickly respond to the complex environment and adjust exploration-exploitation adaptively and rapidly.

E. Sensitivity Analysis

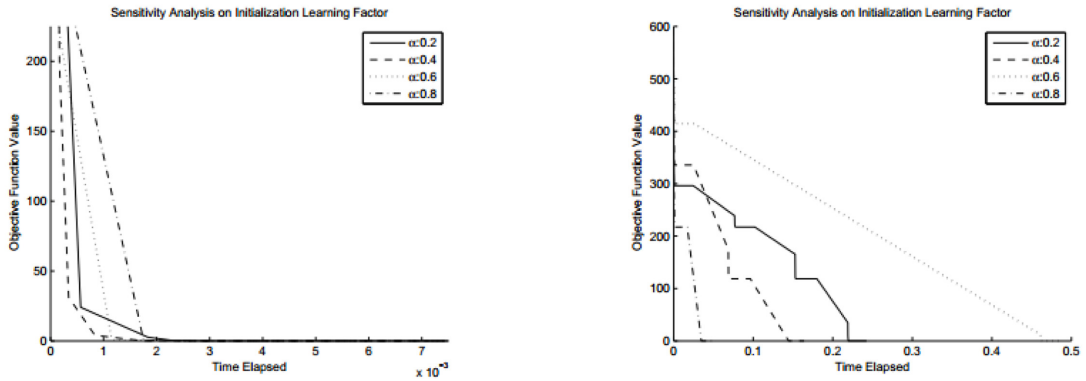
Sensitivity analysis of FGO is very important for the application of the algorithm in practice. When we use a specific algorithm to solve a particular problem, the main problem is to select the set of parameters that will achieve the best effect. This is because most algorithms are sensitive to certain parameters to some extent in the process of optimization, especially for populations. Without attention, this will affect its performance during the optimizing and affect its ability to perform correctly. If one algorithm is less sensitive to a certain parameter, it will be more convenient to set an appropriate parameter to make the algorithm obtain the required effectiveness and efficiency, and vice versa. The following sensitivity analysis is carried out considering the effect of population size and learning factor on FGO across the Schwefel benchmark function.

The evolution curves shown on the left in Fig. 6 demonstrate that FGO is not sensitive to initialization population size between 10 × 10 and 70 × 70. The upper and lower bounds are set as 3 × 3 and 100 × 100, respectively. When the population size is set as 5 × 5, FGO will require more time to exit local minima because of the fewer exploring agents. However, it will improve considerably when we choose 10 × 10 agents as the initialization swarm. When the population size increases to 70 × 70, the evolution speed will be somewhat affected. This is because the increased number of agents will mean that the FGO operators will require more time in processing the sampling information. Thus, the initialization popula-



(a) (b)

Fig. 6. Sensitivity analysis for flexible grid optimization. (a) Sensitivity analysis on initialization population size and (b) sensitivity analysis on initialization learning factor.



(a) (b)

Fig. 7. Sensitivity analysis for flexible grid optimization with fewer agents. (a) Sensitivity analysis with agent of 10×10 and (b) sensitivity analysis with agent of 5×5.

tion size has a relatively wide interval for the user to select, as long as the setting number is not too small.

The evolution curves shown on the right in Fig. 6 indicate that FGO is also not sensitive to the initialization learning factor in a large interval between 0.2 and 0.8. The initialization population size is set as 20×20, which is the same as the choice interval mentioned above in this experimental comparison. We should note that these results do not suggest that the learning factor has no use for FGO because the initialization population almost meets the requirement of this problem. The learning factor plays an important role in adjusting the number of agents when the population does not meet the requirement of the exploration/exploitation situation. The effect of the learning factor can be clearly reflected when the initialization population size cannot satisfy the requirement to a significant degree. The evolution curves shown in Fig. 7 support the above views and reveal that the unreliability resulting from an overly small initialization population size can be avoided by adjusting the number of agents dynamically and adaptively with different learning factors in FGO.

IV. APPLICATION EXAMPLE

FGO is evaluated by applying it to the parameter decision in LS-SVM to verify its practical competence in this section. The fitness function is constructed with the sum of squares of the difference between the predicted values and actual values on some special aspects, such as zeros, bounds, monotony, and period in special parts of the function domain. In fact, most engineering problems contain some special state points or properties that have already been identified. We can use this identified information to construct the contrast trial, which will be more convincing. The identified information used in the following case includes some special zeros and the result shows that these special zeros can guide FGO effectively to find parameters that are more appropriate for LS-SVM.

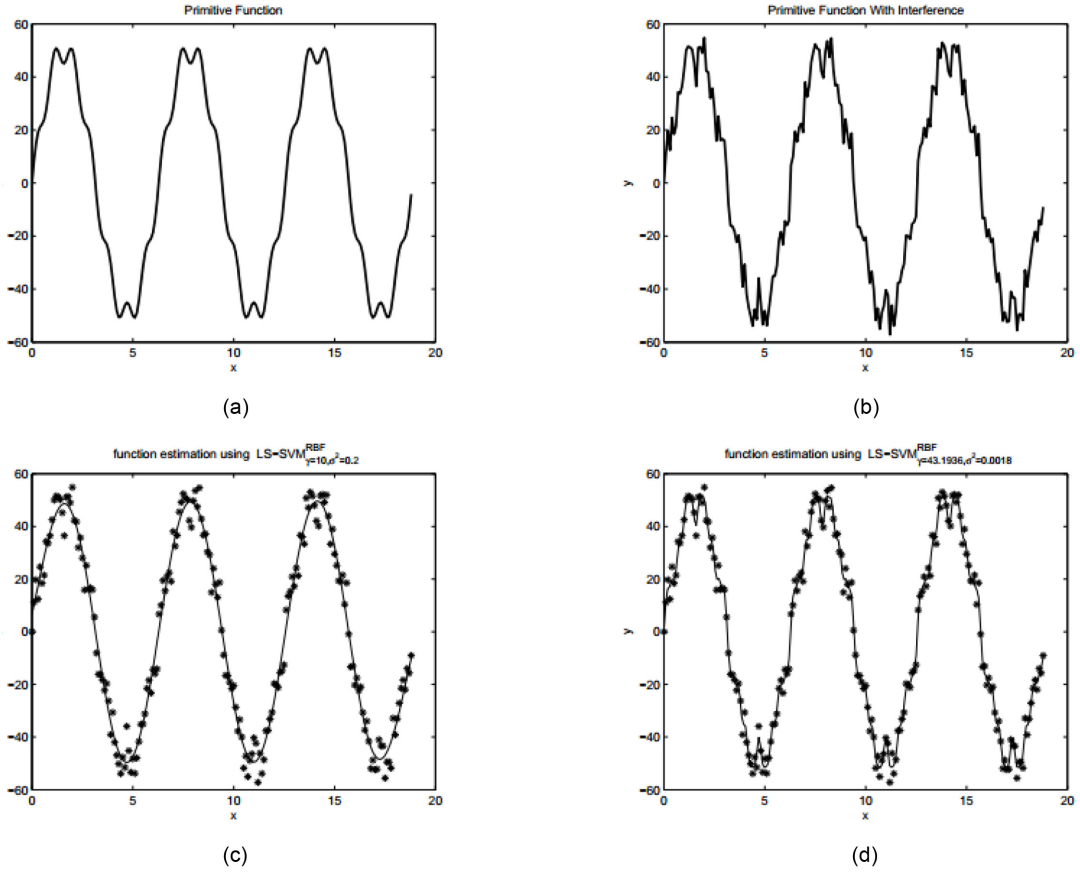


Fig. 8. Parameter optimization in least squares support vector machine (LS-SVM) with flexible grid optimization. (a) Primitive function, (b) primitive function with interference, (c) experiential regression, and (d) optimal regression.

Only six zeros, $X = [0; \pi; 2\pi; 3\pi; 4\pi; 5\pi]$, are used here to construct the fitness function for LS-SVM with FGO. The fitness function is given as follows:

$$f(\gamma, \sigma^2) = \|Y_r - Y\|, \tag{6}$$

where Y_r refers to the regressed values at the six zeros, X, by using the parameters of current γ, σ^2; Y_t refers to the real values at the six zeros, where $Y_t = [0; 0; 0; 0; 0; 0]$. The parameters setting of FGO are given as follows: $\alpha = 0.618$, $\text{DivNum}_{\text{Min}} = [10; 10]$, $\text{DivNum}_{\text{Max}} = [20; 20]$, $\text{LoopNum}_{\text{A}} = 10$, $\text{LoopNum}_{\text{B}} = 10$.

The test function plotted as in Fig. 8(a) is given first as a reference. Disturbance is added to the primitive function to simulate the real engineering condition, which is shown in Fig. 8(b). The asterisks in both Fig. 8(c) and 8(d) represent the same group of samples taken from Fig. 8(b). The regression curve shown in Fig. 8(c) is acquired by using LS-SVM with experiential parameters of $\gamma = 10$ and $\sigma^2 = 0.2$, while the other regression curve shown in Fig. 8(d) is acquired by using LS-SVM with FGO. Apparently, the optimal parameters found by FGO can guide LS-SVM to perform better and acquire excellent results that are closer to the real value. This result also provides support for the effectiveness of identified information in constructing a fitness function.

V. CONCLUSION

This paper presented a novel optimization algorithm, FGO, which consists of a flexible grid and is able to provide an adaptive trade-off between exploration and exploitation across complex objective functions. By examining the performance of FGO on four benchmark functions, the uniform agents array with adaptive scale has been demonstrated to be helpful to increase the speed of the calculation. In addition, the subregion division strategies guided by DC and FC have been demonstrated to be effective in enhancing evolutionary diversity and convergence rate. We also discussed the sensitivity of FGO's parameters and found that the parameter selection space is very wide for FGO to be able to take advantage of its own superiority, which indicates that this algorithm has good generalization ability. The application example of LS-SVM with FGO shows an excellent regression, which verifies, as expected, that FGO is practical.

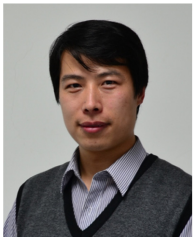
ACKNOWLEDGMENTS

The paper is supported by the State High-Technology Development Project of China with Grant No. 2014AA041802-2.

REFERENCES

1. A. W. Mohamed and H. Z. Sabry, "Constrained optimization based on modified differential evolution algorithm," *Information Sciences*, vol. 194, pp. 171-208, 2012.
2. M. Pelikan, D. E. Goldberg, and F. G. Lobo, "A survey of optimization by building and using probabilistic models," *Computational Optimization and Applications*, vol. 21, no. 1, pp. 5-20, 2002.
3. C. von Lcken, B. Barn, and C. Brizuela, "A survey on multi-objective evolutionary algorithms for many-objective problems," *Computational Optimization and Applications*, vol. 53, no. 3, pp. 707-756, 2014.
4. D. V. Arnold and H. G. Beyer, "A comparison of evolution strategies with other direct search methods in the presence of noise," *Computational Optimization and Applications*, vol. 24, no. 1, pp. 135-159, 2003.
5. S. Saha and S. Bandyopadhyay, "A new point symmetry based fuzzy genetic clustering technique for automatic evolution of clusters," *Information Sciences*, vol. 179, no. 19, pp. 3230-3246, 2009.
6. Y. Tominaga, Y. Okamoto, S. Wakao, and S. Sato, "Binary-based topology optimization of magnetostatic shielding by a hybrid evolutionary algorithm combining genetic algorithm and extended compact genetic algorithm," *IEEE Transactions on Magnetics*, vol. 49, no. 5, pp. 2093-2096, 2013.
7. K. Deb and S. Srivastava, "A genetic algorithm based augmented Lagrangian method for constrained optimization," *Computational Optimization and Applications*, vol. 53, no. 3, pp. 869-902, 2012.
8. C. C. Lin, "Dynamic router node placement in wireless mesh networks: a PSO approach with constriction coefficient and its convergence analysis," *Information Sciences*, vol. 232, p. 294-308, 2013.
9. J. Fernandez-Martinez and E. Garcia-Gonzalo, "Stochastic stability analysis of the linear continuous and discrete PSO models," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 3, pp. 405-423, 2011.
10. H. Mabed, A. Caminada, and J. K. Hao, "Genetic tabu search for robust fixed channel assignment under dynamic traffic data," *Computational Optimization and Applications*, vol. 50, no. 3, pp. 483-506, 2011.
11. B. A. Sawyerr, M. M. Ali, and A. O. Adewumi, "A comparative study of some real-coded genetic algorithms for unconstrained global optimization," *Optimization Methods and Software*, vol. 26, no. 6, pp. 945-970, 2011.
12. M. K. Dhadwal, S. N. Jung, and C. J. Kim, "Advanced particle swarm assisted genetic algorithm for constrained optimization problems," *Computational Optimization and Applications*, vol. 58, no. 3, pp. 781-806, 2014.
13. Y. J. Wang, "Improving particle swarm optimization performance with local search for high-dimensional function optimization," *Optimization Methods and Software*, vol. 25, no. 5, pp. 781-795, 2010.
14. C. Luo, S. L. Zhang, and B. Yu, "Some modifications of low-dimensional simplex evolution and their convergence," *Optimization Methods and Software*, vol. 28, no. 1, pp. 54-81, 2013.
15. T. Aittokoski and K. Miettinen, "Efficient evolutionary approach to approximate the Pareto-optimal set in multiobjective optimization, UPS-EMOA," *Optimization Methods and Software*, vol. 25, no. 6, pp. 841-858, 2010.
16. M. H. Lim, Y. Yuan, and S. Omatu, "Efficient genetic algorithms using simple genes exchange local search policy for the quadratic assignment problem," *Computational Optimization and Applications*, vol. 15, no. 3, pp. 249-268, 2000.
17. A. El Dor, M. Clerc, and P. Siarry, "A multi-swarm PSO using charged particles in a partitioned search space for continuous optimization," *Computational Optimization and Applications*, vol. 53, no. 1, pp. 271-295, 2012.
18. Y. Tang, Z. Wang, and J. A. Fang, "Controller design for synchronization of an array of delayed neural networks using a controllable probabilistic PSO," *Information Sciences*, vol. 181, no. 20, pp. 4715-4732, 2011.
19. S. Y. Yuen and C. K. Chow, "A genetic algorithm that adaptively mutates and never revisits," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 454-472, 2009.
20. J. Sadeghi, S. Sadeghi, and S. T. A. Niaki, "Optimizing a hybrid vendor-managed inventory and transportation problem with fuzzy demand: an improved particle swarm optimization algorithm," *Information Sciences*, vol. 272, pp. 126-144, 2014.
21. S. Wang and J. Watada, "A hybrid modified PSO approach to VaR-based facility location problems with variable capacity in fuzzy random uncertainty," *Information Sciences*, vol. 192, pp. 3-18, 2012.
22. C. W. Ahn, J. An, and J. C. Yoo, "Estimation of particle swarm distribution algorithms combining the benefits of PSO and EDAs," *Information Sciences*, vol. 192, pp. 109-119, 2012.
23. W. P. Lee and Y. T. Hsiao, "Inferring gene regulatory networks using a hybrid GA-PSO approach with numerical constraints and network decomposition," *Information Sciences*, vol. 188, pp. 80-99, 2012.
24. Y. Hung and W. Wang, "Accelerating parallel particle swarm optimization via GPU," *Optimization Methods and Software*, vol. 27, no. 1, pp. 33-51, 2012.
25. L. N. Xing, Y. W. Chen, and K. W. Yang, "Multi-population interactive coevolutionary algorithm for flexible job shop scheduling problems," *Computational Optimization and Applications*, vol. 48, no. 1, pp. 139-155, 2011.
26. Y. Yang and X. Yu, "Cooperative coevolutionary genetic algorithm for digital IIR filter design," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, pp. 1311-1318, 2007.
27. M. Baz, B. Hunsaker, and O. Prokopyev, "How much do we pay for using default parameters?," *Computational Optimization and Applications*, vol. 48, no. 1, pp. 91-108, 2011.
28. A. Cassioli, M. Locatelli, and F. Schoen, "Dissimilarity

measures for population-based global optimization algorithms," *Computational Optimization and Applications*, vol. 45, no. 2, pp. 257-281, 2010.
29. W. S. Gao, C. Shao, and Q. Gao, "Pseudo-collision in swarm optimization algorithm and solution-rain forest algorithm," *Acta Physica Sinica*, vol. 62, no. 19, article id. 190202, 2013.
30. W. Gao, C. Shao, and Y. An, "Bidirectional dynamic diversity evolutionary algorithm for constrained optimization," *Mathematical Problems in Engineering*, vol. 2013, article id. 762372, 2013.
31. W. W. Hager and H. Zhang, "Self-adaptive inexact proximal point methods," *Computational Optimization and Applications*, vol. 39, no. 2, pp. 161-181, 2008.
32. M. Al-Baali and H. Khalfan, "A combined class of self-scaling and modified quasi-newton methods," *Computational Optimization and Applications*, vol. 52, no. 2, pp. 393-408, 2012.
33. C. Audet, J. E. Dennis Jr, and S. Le Digabel, "Globalization strategies for mesh adaptive direct search," *Computational Optimization and Applications*, vol. 46, no. 2, pp. 193-215, 2010.
34. A. Nahapetyan and P. Pardalos, "Adaptive dynamic cost updating procedure for solving fixed charge network flow problems," *Computational Optimization and Applications*, vol. 39, no. 1, pp. 37-50, 2008.

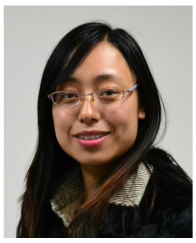


Weishang Gao

Weishang Gao received his Ph.D. degree and M.E. degree from the School of Control Science and Engineering, Dalian University of Technology, Dalian, China in 2009 and 2015, respectively. His research interests include evolutionary computation, swarm intelligence optimization, and intelligent control. He is the corresponding author of the paper.

Chen Shao

Chen Shao is currently a professor at the Institute of Advanced Control, Dalian University of Technology, Dalian, China. His research interests include modeling and control of complex industrial processes, integrated optimization control of continuous industrial processes, robust adaptive control of nonlinear systems, multiobjective optimization control technology, and intelligent control theory and method.



Qin Gao

Qin Gao is currently pursuing her Ph.D. degree in Control Science and Engineering from Dalian University of Technology, Dalian, China. She received her B.S. degree in automation from North China University of Water Conservancy and Hydropower, Zhengzhou, China in 2007. Her research interests include bio-inspired robotics, locomotion control, and models of snake-like robots.