Improvement of Particle Swarm Optimization

K. Kawakami and Z. Meng
Department of Electrical Engineering, Fukuoka University, Japan

Abstract— A new technique titled "Particle Refresh" and a hybridization with conjugate gradient method are introduced to particle swarm optimization (PSO). The former charges power to inactive particle to improve the recovery ability of PSO after trapping on a local solution, and as a result, it becomes easy to choose suitable values for control-parameters to keep high performance for diverse objective functions. On the other hand, the point of the latter is how to determine the changeover timing between a conjugate gradient method and a PSO algorithm.

1. INTRODUCTION

There are cases when an optimization is required in the field of electromagnetic engineering, such as optimizing design parameters, resolving a nonlinear inverse scattering problem as an optimization problem, and so on. Usually, an objective function has multiple local solutions due to the function complexity or errors in approximately calculation, and a global applicable optimization method is necessary. Particle swarm optimization (PSO) [1] has achieved considerable success as a global optimization method with a wide range of applicability, requiring no prior information, being appropriate for diverse objective functions, and offering the ability of recovery even after trapping on a local solution. However, PSO has the following problems.

① Difficult to use

Several parameters are needed for controlling the algorithm. Unless these parameters are set appropriately, search efficiency drops significantly. There are, however, no clear rules for setting the parameters, and almost all users have considerable difficulty in setting them without prior experience in parameter tuning.

② Slow convergence

Global optimization methods do not utilize differential information, so that the convergence is usually slower than that of a gradient descent method.

In this paper, a new technique "Particle Refresh" and a hybridization with a conjugate gradient method are introduced to improve the conventional PSO algorithm.

2. THE ALGORITHM OF PARTICLE SWARM OPTIMIZATION

A PSO algorithm employs a swarm composed of multiple particles, and estimates each particle at each moment $t\,(t = 0, 1, 2, \ldots)$ by using the value of the objective function at the particle's current position $\boldsymbol{x}_i(t) \in D$, where i and D denote the number of the particle and search space, respectively. Each particle has its velocity $\boldsymbol{v}_i(t)$. The velocity at moment t depends on the velocity at moment $(t-1)$, the position $\boldsymbol{x}_i(t)$, the best position $\boldsymbol{p}_i(t)$ found up to now by the article, and the best position $\boldsymbol{g}(t)$ found up to now by the swarm:

$$\boldsymbol{v}_i(t) = c_1 \boldsymbol{v}_i(t-1) + c_2 \left\{\boldsymbol{p}_i(t) - \boldsymbol{x}_i(t)\right\} + c_3 \left\{\boldsymbol{g}(t)\boldsymbol{x}_i(t)\right\}. \tag{1}$$

Here c_1 is a constant, c_2 and c_3 is selected randomly at the moment by

$$c_k = c_{\max} alea(0,1), \quad k = 2, 3, \tag{2}$$

where $alea(0,1)$ denotes a random number uniformly distributed in $[0,1]$. Then the position of particle at moment $(t+1)$ is defined by

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t). \tag{3}$$

The algorithm of PSO can be described as follows:

① Select the size of the swarm n (how many particles), a threshold E (if possible) and a time limit T to stop the algorithm when the estimation of any particle e_i becomes $e_i \geq E$ or time t becomes $t \geq T$; let $t = 0$; randomly locate n particles in the search space and set an initial velocity to each particle according to the size of the search space.

② Get estimation $e_i\,(i = 1, 2, \ldots, n)$ for each particle by the objective function, and update $\boldsymbol{p}_i$ and $\boldsymbol{g}$.

③ Stop the algorithm and output $\boldsymbol{g}$ as its solution if one of the following "stop conditions" is satisfied, or go to the next step otherwise.

(a) the best estimation $e_{\text{best}} \geq E$;

(b) $t \geq T$;

③ Let $t = t + 1$ and update $\boldsymbol{v}_i$, $\boldsymbol{x}_i$ by (1), (3).

④ Go to step ②.

3. EFFECTS OF C_1 AND C_{MAX} ON ALGORITHM PERFORMANCE

Obviously, c_1 is an "inertial coefficient" of the particle. If $c_1 > 1$, a particle can power itself and the algorithm will not converge to a solution. Therefore, viewed in the light of algorithm convergence, c_1 should be smaller than one, and particles will converge to $(\boldsymbol{g} + \boldsymbol{p}_i)/2$ as a result of a damping motion. A PSO algorithm with a large c_1 turns to an advantage in recovery after trapping on a local solution and disadvantage in convergence rate. On the other hand, $c_{\max}$ is a "gravitational coefficient". The gravitation pulls particles to $\boldsymbol{g}$ or $\boldsymbol{p}$ to improve searching efficiency. However, the algorithm will diverge if $c_{\max}$ is too large.

Certainly, high performance can be obtained when the "inertia" and the "gravitation" work in cooperation with each other. Some textbooks recommend that user should select values for c_1 and $c_{\max}$ in pairs, such as "0.7,1.47", "0.8,1.62", and so on. To investigate the effects of c_1 and $c_{\max}$ minutely, two numerical experiments are considered.

The first one is to maximize the objective function given by (4):

$$\Omega = \prod_{i=1}^{2} \frac{\sin(x_i)}{(x_i)}, \quad -10 \leq x_i \leq 10, \quad i = 1, 2. \tag{4}$$

A three-dimensional graphical representation of (4) is given by Figure 1. We change c_1 from 0.1 to 0.6 and $c_{\max}$ from 1 to 2.4, and estimate the performance by the times of objective function calculated by the algorithm until the solution is found out. Because PSO is stochastic in nature and a single test does not validate the characteristic we made 1000 runs for each test of a pair values of c_1 and $c_{\max}$, and show the average calculation times (ACT) in Figure 2. It is shown that there is a region for c_1 and $c_{\max}$ in which the PSO keeps high performance. If c_1 or $c_{\max}$ is set to a value out of the region, the algorithm will lose its performance or even fails in the searching.

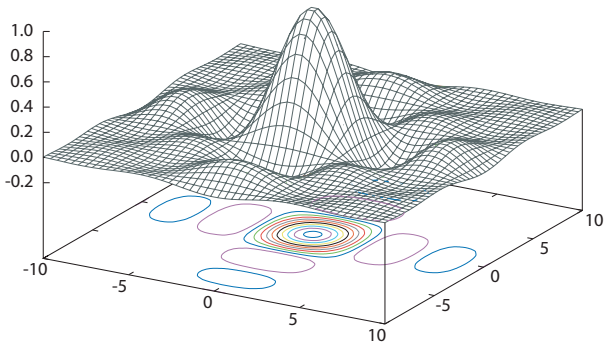

Figure 1: Three-dimentional representation of (4).

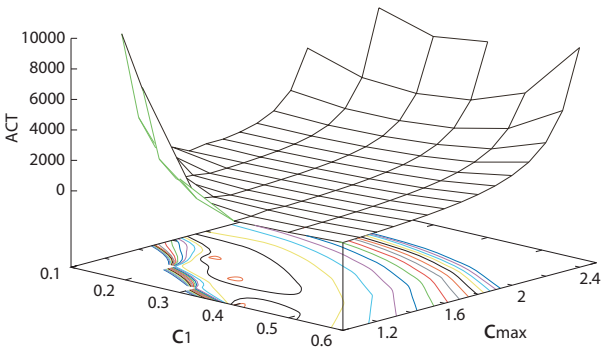

Figure 2: Estimation of the PSO applied to (4).

The second experiment is to minimize the objective function given by (5):

$$\Omega = x_1 \sin(4x_1) + x_2 \sin(2x_2), \quad 0 \leq x_i \leq 10, \quad i = 1, 2. \tag{5}$$

There are many local minima in the search space, as shown in Figure 3. Similar to that for experiment 1, we show the ACT in Figure 4. The region in which the PSO keeps high performance

exists still, but becomes very narrow and almost does not double with the region shown in Figure 2 for (4). The results show that the conventional PSO may fail in the searching or lose its performance if c_1 and $c_{\max}$ are not set suitably, and the suitable values depend on the objective function. General values working well for any objective function do not exist.

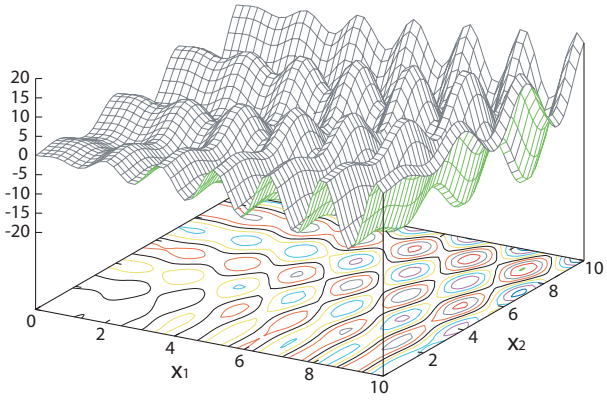

Figure 3: Three-dimentional representation of (5).

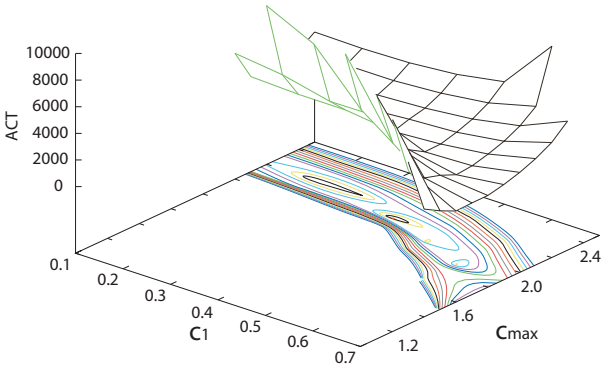

Figure 4: Estimation of the PSO applied to (5).

4. REFRESH OPERATOR

Why are the suitable regions for (4) and (5) different from each other?

For an objective function like (4) that the algorithm does almost not trap on a local minimum, it becomes more important for PSO to focus the search attention to the neighborhood of the current best solution. Therefore, a "convergence-mode" where c_1 and $c_{\max}$ are small can get high performance in experiment 1.

However, if a convergence-mode PSO is applied to optimize an objective function with many local minima like (5), the algorithm will trap on a local minimum easily. As a result, the algorithm may fail or lose its performance. In fact, many particles lose their power and stop at a local minimum in many cases when the c_1 and $c_{\max}$ are set to small values in experiment 2. If all of the particles stop at a local minimum, the algorithm will fail in the searching.

Because a standstill particle is useless for the algorithm, we introduce a new operator titled "Particle Refresh" (PR) to PSO algorithm shown in Section 2 after the step ③:

③$^+$ when $|\boldsymbol{v}_i| < \varepsilon$, randomly re-locate the particle and clear $\boldsymbol{p}_i$, where ε denotes a small enough value.

The operation will power standstill particles and reinforce the algorithm to avoid trapping on a local minimum.

The numerical experiments results applying a PSO with PR to (4) and (5) are shown in Figures 5 and 6.

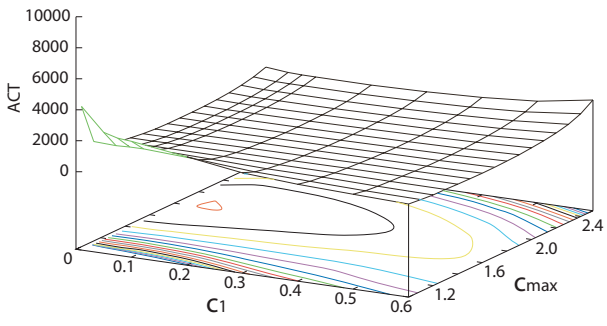

Figure 5: Estimation of the PSO with PR applied to (4).

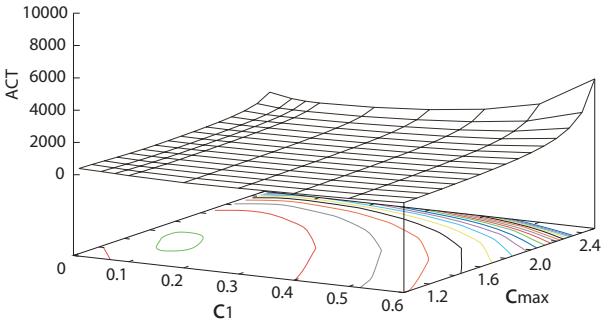

Figure 6: Estimation of the PSO with PR applied to (5).

Obviously, the suitable region of c_1 and $c_{\max}$ in Figures 5 and 6 become widely and partly double with each other, comparison with those in Figures 2 and 4. In other words, it becomes easy to

choose suitable values for c_1, $c_{\max}$ to keep high performance for different type objective functions. The PSO becomes Easy-To-Use.

5. DISCUSSION ABOUT HYBRIDIZING WITH A GRADIENT METHOD

In order to improve the search performance, many hybridizations between a gradient descent method and an evolution algorithm, such as a conjugate gradient method (CGM) and a genetic algorithm, have been reported. One of the points for the hybridizations is how to determine the changeover timing between the two methods, because it is difficult to judge if the current best solution $\boldsymbol{g}$ is in the neighborhood of the global minimum. We try to hybridize a CGM (using numerically differential information) with a PSO. The algorithm is based on the PSO algorithm shown in Sections 2 and 4, but inserted the following step after the step ②.

②$^+$ If one of the following conditions is satisfied, start CGM at $\boldsymbol{g}$, update $\boldsymbol{g}$ with the solution of the CGM, and return to the PSO ③

- CGM has not been applied up to now and $\boldsymbol{g}(t) = \boldsymbol{g}(t-1) = \boldsymbol{g}(t-2)$;
- CGM has been applied one or more than one times and $|\boldsymbol{g}(t) - \boldsymbol{g}(t-1)|$ is larger than an allowable margin of error;

The ACTs of 1,000 runs where the hybrid method is employed to (4) and (5) are shown in Table 1. The hybridization method shows higher performance than the PSOs in minimizing (5), but is almost not effective in maximizing (4). In fact, the above conditions for starting the CGM are almost not satisfied and the CGM had almost not been used in maximizing (4). We need a rational ground to construct the conditions for starting the CGM. A technique that can make use of the information of the objective function obtained by the algorithm at that time to roughly image the basins of attraction of local minima, as reported in [2], is necessary for a hybrid method.

Table 1: A comparison of ACT between the cases of using standard PSO, PSO with refresh operator and the hybridization method.

Objective Function	standard PSO	standard PSO + PR	standard PSO + PR + CGM
(4)	498	261	262
(5)	1521	535	365

6. CONCLUSIONS

A new operator titled "Particle Refresh" proposed in Section 4 is effective in improving particle swarm optimization (PSO) algorithm keeping high performance for different type objective functions. Hybridization between a PSO and a conjugate gradient method is tested and the results show that how to determine the changeover timing between the methods is very important. We are going to introduce a technique that can make use of the information of the objective function obtained by the algorithm to roughly image the basins of attraction of local minima, as reported in [2], to solve the problem.

REFERENCES

1. Clerc, M., *Particle Swarm Optimization*, ISTE Ltd., London, 2006.
2. Meng, Z., "Autonomous genetic algorithm for functional optimization," *Progress In Electromagnetic Research*, PIER 72, 253–268, 2007.