A Modified Particle Swarm Optimization Algorithm for Global Optimizations of Inverse problems

Shafi Ullah Khan[1], Shiyou Yang[1,2], Luyu Wang[1], Lei Liu[1], Wei Zhang[1]

[1] College of Electrical Engineering, Zhejiang University, 310027, Hangzhou, China, [2]eesyyang@zju.edu.cn

Particle Swarm Optimization (PSO) is a population based stochastic search algorithm inspired from the natural behavior of bird flocking or fish schooling. Due to its easiness in numerical implantations, PSO is used to solve a wide range of inverse problems. However, a PSO is often trapped into local optima while dealing with complex and real world problems. To tackle this problem, a new modified PSO is presented by introducing a mutation mechanism and using dynamic algorithm parameters. The experimental results on different case studies show that the proposed PSO obtain the best results among the tested algorithms.

***Index Terms*—Dynamic inertia weight, dynamic learning factors, mutation operator, PSO.**

I. A MODIFIED PSO

MOST inverse problems in electrical engineering involve optimizations of a multimodal cost function. As traditional deterministic methods are unable to find the global solutions of these kinds of problems, a lot of efforts have been devoted to the study of stochastic and heuristic algorithms in the last couple of decades. In this context, a wealth of stochastic methods, commonly called evolutionary algorithms (EA) including, for example, simulated annealing method, genetic algorithm, tabu search method, ant colony algorithm, as well as particle swarm optimization, have all been proposed and used successfully to solve typical electromagnetic design problems. As a consequence, EAs have become the standards and paradigms for solving inverse problems. However, so far there is no any universal evolutionary algorithm which is equally successful for all engineering problems, and it is essential to keep the diversity of the EAs.

PSO is a population based stochastic optimal algorithm stimulated by the collective behavior of bird flocking or fish schooling for finding the most favorable solution of an optimal problem. Due to its meta-heuristic characteristics, PSO is well-suited for solving complex and multi-dimensional problems. As a result, different variants of PSOs have been proposed and applied in many research areas. However, the main drawback of existing PSOs is their premature convergence. In this regard, a modified PSO is proposed. For space limitations, the details including the terms and parameter definitions on a general PSO are referred to [1] and the references therein, and only the proposed modifications will be explained in this digest.

A. Introduction of a Mutation Operation

In the original PSO, in a particular dimension, if the p_{best} and g_{best} are trying to guide the current particle to different direction, this particle will fluctuate in the feasible space. On the other hand, if the p_{best} and g_{best} have the same direction as that of the present particle, the particle will follow the same evolution direction. In such situations, the particles cannot escape from the local optimum if g_{best} is far away from the global optimal point. To address this issue, a mutation operation is introduced. For this goal, another best particle, P_{best1}, is defined to motivate the particles to move in different directions during the optimization process to guarantee the particles to jump out local optimum and to easily explore the global optimal position.

Moreover, to use the dynamic information gathered from the current population to guide the searches in the mutation operation, a threshold value of the population is firstly introduced and defined as

$$U = K \times \frac{\sum_{i=1}^{N} pop\_value_i}{N} \tag{1}$$

where, pop_value_i is the fitness value of particle i, N is the size of the swarm, K is a user defined constant which is set to be 0.2 after comprehensive numerical experiments.

Secondly, the current population is divided into two subpopulations according to this threshold value. The particles whose fitness values are smaller than this threshold are consisted of a bad subpopulation, and the reminders of a good subpopulation. Thirdly, one will randomly select a best particle as previously mentioned, and named as P_{best1}, from all of the P_{best} particles. This P_{best1} will be used to design the following proposed mutation mechanism.

Step 1: A particle is randomly generated in the current search space using the below mechanism

$$Y_j = rand(a_j(t), b_j(t)) \tag{2}$$

where $a_j(t)$ and $b_j(t)$ are, respectively the lower and upper bounds of the decision parameters in the j^{th} dimension, *rand* () is a uniform random number in the specified interval, t is the index of generations.

Step 2: *A* trial particle, $X_J^* = \{x_1^*, x_2^*, x_3^*.....x_d^*\}$, is then generated from

$$X_J^* = (1+b_j)P_{best_1} - b_j y_j\,, \tag{3}$$

where b_j is a constant number. By using an error and trial approach, b_j is set to 0.5.

Step 3: One compares the trial particle with the worst one in the current population. If the trial particle is better than the worst one, the worst particle will be replaced by the trial one; otherwise, the worst particle is survived in the same position.

B. Dynamic Parameters Setting

To keep a good balance between exploration and exploitations searches, dynamic algorithm parameters are proposed and used.

As demonstrated in the literature, a large inertia weight will favor exploration searches and a small one will bias exploitation searches [2]. Therefore, it is essential to use a dynamic inertia weight. In this paper, the maximum and minimum values of the inertia weight are, respectively, 0.05 and 0. Moreover, to increase the diversity of the particles, the inertia weight should have some randomness characteristics. Based on these observations, one proposes a random updating formula for the inertia weight as

$$W = K \times rand() \tag{4}$$

where K =1/N, $rand()$ is a random parameter uniformly distributed in [0,1].

The two learning factors, the social constant, c_1, and cognitive constant, c_2, are other two important parameters deciding the performances of a PSO. A larger social constant as compare to the cognitive constant has the possibility to move the particles prematurely and, as a result, all the particles coverage to local optimum, In contrast, higher cognitive constant compared to the social constant will results in excessive wandering of the individuals. In order to bias the particles escaping from local optimum and coverage to the global optimal solution, the aforementioned two learning constants are updated dynamically by using

$$c_1 = 1 + rand() \tag{5}$$

$$c_2 = e - rand() \tag{6}$$

II. Numerical Examples

A. Function Tests

To test the proposed modified PSO algorithm, some well known benchmarks of mathematical test functions are firstly solved, and its performances are compared to the basic PSO (BPSO), GPSO [3], and IPSO [4]. For a fair comparison, all algorithms use the same parameters. Moreover, only the numerical results on the function defined below are reported for space limitations. The function is defined as

$$f = \sum_{i=1}^{n} |x_i|^{i+1} (x_i \in [-10,10]) \tag{7}$$

The global minimum point of this function is at x_i=0 with the function value of 0. In the numerical experiment, the dimension of the decision parameters is 30. To evaluate the average performance of an algorithm, every method runs 50 times, and the mean values, f_{mean}, of the final solutions for 50 random runs of different algorithms are compared in Table I. To give an intuitive impression about the convergent performances of an algorithm, the convergent characteristics of a typical run for each algorithm are depicted in Fig. 1. Obviously, the proposed MPSO outperforms other three PSOs in both solution quality (objective function) and the convergence speed.

TABLE I
MEAN VALUES OF THE FINAL SOLUTIONS FOR 50 RANDOM RUNS OF DIFFERENT ALGORITHMS

Algorithm	BPSO	IPSO	GPSO	Proposed MPSO
f_{mean}	2.40×10^{9}	5.36×10^{-4}	3.54×10^{-10}	9.68×10^{-228}

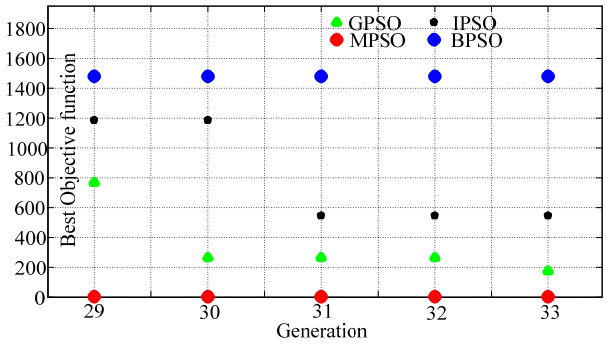

Fig 1. The convergence characteristics of different algorithms.

B. Application

The proposed algorithm is then used to solve the TEAM Workshop problem 22 [5],[6]. Also, the aforementioned 4 algorithms are used to solve this case study for performance comparisons, and the mean values of the final solutions for 10 random runs of each algorithm are tabulated in Table II. Again, these numerical results on this case study demonstrate that the proposed MPSO outperforms other three well recognized variants of PSOs in terms of both solution quality and convergence speed.

TABLE II
MEAN VALUES OF THE FINAL SOLUTIONS FOR 10 RANDOM RUNS OF DIFFERENT ALGORITHMS FOR APPLICATION

Algorithm	BPSO	IPSO	GPSO	Proposed MPSO
f_{mean}	0.1356	0.1278	0.1123	0.0929

In summary, from the numerical results on the two case studies, it is clear that the performance of the proposed MPSO is significantly better than other three ones in terms of both solution quality and convergence performance.

III. References

[1] J. Kennedy, and R.C. Eberhart, "Particle swarm optimization, " Proc. *IEEE Int. Conf. Neural Network*, 1995, pp.1942-1948.

[2] R.C. Eberhart, and Y. H. Shi, "Tracking and optimizing dynamic system with particle swarm," *Congress on Evolutionary Computational*, 2001, vol.1, pp, 94-100.

[3] J. J. Jamian, M. N. Abdullah, H. Mokhlis, M. W. Mustafa, and A. H. A. Bakar, "Global particle swarm optimization for high dimension numerical function analysis," *Journal of Applied Mathematics*, vol. 2014, pp.1-14, Article ID 329193, doi.10.1155/2014/329193,2014.

[4] T.Y. Lee, and C-L .Chen ,"Unite commitment with probabilistic reserve: an IPSO approach," *Energy Conversion and Management*, vol,48, no2, pp.486-493, 2007.

[5] TEAM optimization benchmark problem 22 [online], available at: http://www.compumag.org/jsite/team.html.

[6] S. L. Ho, Shiyou Yang, Guangzheng Ni, and Jin Huang, "A Quantum-Based Particle Swarm Optimization Algorithm Applied to Inverse Problems, " *IEEE Trans. Magn.*, vol. 49, pp. 2069-2072, May 2013.