9

A learning algorithm based on PSO and L-M for parity problem

Guangyou Yang, Daode Zhang, and Xinyu Hu
School of Mechanical Engineering
Hubei University of Technology
Wuhan,430068
P. R. China

1. Introduction

The Back-propagation network(BP network) is the most representative model and has wide application in artificial neural network(J. L. McClelland, D. E. Rumelhart & the PDP Research Group). Owing to the hidden layer and learning rules in the BP network and the Error Back-propagation algorithm, the BP network can be used to recognize and classify nonlinear pattern(Zhou zhihua, Cao Cungen, 2004). Currently, the applications include handwritings recognition, speech recognition, text - language conversion, image recognition and intelligent control. As the BP algorithm is based on gradient descent learning algorithm, it has some drawbacks such as slow convergence speed and easily falling into local minimum, as well as poor robustness. In the last decade, a series of intelligent algorithms, which is developed from nature simulation, are got wide attention, especially the global stochastic optimization algorithm based on the individual organisms and groups makes a rapid development and gets remarkable achievements in the field of engineering design and intelligent control. The most famous are genetic algorithm, the PSO algorithm (Particle Swarm Optimization, PSO), etc. In this chapter, the research focuses on the integration of the improved PSO algorithm and the Levenberg-Marquardt (L-M) algorithm of neural network, and its application in solving the parity problem, which enhances the optimization property of the algorithm, and solves the problems such as slow convergence speed and easily falling into local minimum.

2. Particle Swarm Optimization (PSO)

2.1 Standard Particle Swarm Optimization

Dr. Kennedy and Dr. Eberhart proposed the PSO algorithm in 1995(Kennedy, & Eberhart, 1995), which derived from the behavior research of flock foraging, and the research found out that the PSO theory can be applied to the function optimization, then it was developed into a universal optimization algorithm gradually. As the concept of PSO is simple and easy to implement, at the same time, it has profound intelligence background, the PSO algorithm attracted extensive attention when it was first proposed and has become a hot topic of

research. The search of PSO spreads all over the solution space, so the global optimal solution can be easily got, what is more, the PSO requires neither continuity nor differentiability for the target function, even doesn't require the format of explicit function, the only requirement is that the problem should be computable. In order to realize the PSO algorithm, a swarm of random particles should be initialized at first, and then get the optimal solution through iteration calculation. For each iteration calculation, the particles found out their individual optimal value of *pbest* through tracking themselves and the global optimal value of *gbest* through tracking the whole swarm. The following formula is used to update the velocity and position.

$$v_{id}^{k+1} = w \cdot v_{id}^{k} + c_1 \cdot rand \cdot (p_{id} - x_{id}^{k}) + c_2 \cdot rand \cdot (p_{gd} - x_{id}^{k}) \quad (1)$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \quad (2)$$

In the formula (1) and (2), i=1, 2, …, m, m refers to the total number of the particles in the swarm; d=1, 2,…, n, d refers to the dimension of the particle; v_{id}^{k} is the No. d dimension component of the flight velocity vector of iteration particle i of the No. k times. x_{id}^{k} is the No. d dimension component of the position vector of iteration particle i of the No. k times; p_{id} is the No. d dimension component of the optimization position (*pbest*) of particle i ; p_{gd} is the No. d dimension component of the optimization position (*gbest*) of the swarm; w is the inertia weight; *c1,c2* refer to the acceleration constants; rand() refers to the random function, which generates random number between [0, 1]. Moreover, in order to prevent excessive particle velocity, set the speed limit for V_{max}, when accelerating the particle velocity into the level: $v_{id} > V_{max}$, set $v_{id} = V_{max}$; In contrast, on the condition of $v_{id} < -V_{max}$, set $v_{id} = -V_{max}$.
The specific steps of the PSO algorithm are as follows:

(1) Setting the number of particles m, the acceleration constant *c1,c2*, inertia weight coefficient w and the maximum evolution generation *Tmax*, in the n-dimensional space, generating the initial position *X(t)* and velocity *V (t)*of m-particles at random.
(2) Evaluation of Swarm *X(t)*
   i. Calculating the fitness value *fitness* of each particle.
   ii. Comparing the fitness value of the current particle with its optimal value *fpbest*. If *fitness* < *fpbest,* update *pbest* for the current location, and set the location of *pbest* for the current location of the particle in the n-dimensional space.
   iii. Comparing the fitness value of the current particle with the optimal value *fGbest* of the swarm. If *fitness* < *fGbest,* update *gbest* for the current location, and set the *fGbest* for the optimal fitness value of the swarm, then the current location *gbest* of the particle is referred to as the optimal location of the swarm in the n-dimensional space.
(3) In accordance with the formula (1) and (2), updating the location and velocity of the particles and generating a new swarm *X(t+1)*.

(4) Checking the end condition, if meet the end condition, then stop optimizing; Otherwise, $t=t+1$ and turn to step (2).

In addition, the end condition is referred to as the following two situations: when the optimizing reaches the maximum evolution generation *Tmax* or the fitness value of *gbest* meets the requirement of the given precision.

2.2 Improved Particle Swarm Optimization Algorithm

The PSO algorithm is simple, but research shows that, when the particle swarm is over concentrated, the global search capability of particle swarm will decline and the algorithm is easy to fall into local minimum. If the aggregation degree of the particle swarm can be controlled effectively, the capability of the particle swarm optimizing to the global minimum will be improved. According to the formula (1), the velocity v of the particle will become smaller gradually as the particles move together in the direction of the global optimal location *gbest*. Supposed that both the social and cognitive parts of the velocity become smaller, the velocity of the particles will not become larger, when both of them are close to zero, as $w<1$, the velocity will be rapidly reduced to 0, which leads to the loss of the space exploration ability. When the initial velocity of the particle is not equal to zero, the particles will move away from the global optimal location of *gbest* by inertial movement. When the velocity is close to zero, all the particles will move closer to the location of *gbest* and stop movement. Actually, the PSO algorithm does not guarantee convergence to the global optimal location, but to the optimal location *gbest* of the swarm(LU Zhensu & HOU Zhirong, 2004). Furthermore, as shown in the formula (2), the value of the particle velocity also represents the distance of particle relative to the optimal location *gbest*. When the particles become farther from the *gbest*, the particle velocity will be greater, on the contrary, when the particles become closer to the *gbest*, the velocity will be smaller gradually. Therefore, as shown in the formula (1), by means of the extreme variation of the swarm individual, the velocity of the particles can be controlled in order to prevent the particles from gathering at the location *gbest* quickly, which can control the swarm diversity effectively. Known from the formula (1), when the variability measures are taken, both the social and cognitive parts of each particle velocity are improved, which enhances the particle activity and increases the global search capability of particle swarm to a large extent. The improved PSO(MPSO) is carried out on the basis of standard PSO, which increases the variation operation of optimal location for the swarm individual. The method includes the following steps:

(1) Initializing the position and velocity of particle swarm at random;

(2) The value *pbest* of the particle is set as the current value, and the *gbest* for the optimal particle location of the initial swarm；

(3) Determining whether to meet the convergence criteria or not, if satisfied, turn to step 6; Otherwise, turn to step 4;

(4) In accordance with the formula (1) and (2), updating the location and velocity of the particles, and determining the current location of *pbest* and *gbest*；

(5) Determining whether to meet the convergence criteria or not, if satisfied, turn to step 6; Otherwise, carrying out the optimal location variation operation of swarm individuals according to the formula (3), then turn to step 4;

$$new\_pbest_d = pbest_d(1+\eta\sigma) \tag{3}$$

(6) Outputting the optimization result, and end the algorithm.

In the formula (3), the parameter σ refers to random number which meets the standard Gaussian distribution, the initial value of the parameter η is 1.0, and set $\eta=\beta\eta$ every 50 generations, where the β refers to the random number between [0.01, 0.9]. From above known, the method not only produces a small range of disturbance to achieve the local search with high probability, but also produces a significant disturbance to step out of the local minimum area with large step migration in time.

2.3 Simulation and Result Analysis of the Improved Algorithm

2.3.1 Test Functions

The six frequently used Benchmark functions of the PSO and GA(genetic algorithm) (Wang Xiaoping & Cao Liming, 2002)are selected as the test functions, where the Sphere and Rosenbrock functions are unimodal functions, and the other four functions are multimodal functions. The Table 1 indicates the definition, the value range and the maximum speed limit *Vmax* of these Benchmark functions, where: x refers to real type vector and its dimension is n, x_i refers to the No. i element.

name	Function	Initialization Range	V_{max}
Sphere	$f_1(x) = \sum_{i=1}^{n} x_i^2$	$(-1000,1000)^n$	1000
Rastrigrin	$f_2(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	$(-5.12,5.12)^n$	10
Griewank	$f_3(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$(-600,600)^n$	600
Rosenbrock	$f_4(x) = \sum_{i=1}^{n} (100x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	$(-30,30)^n$	100
Ackley	$f_5(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + e$	$(-30,30)^n$	30
Schaffer	$f_6(x) = 0.5 + \frac{(\sin\sqrt{x^2+y^2})^2 - 0.5}{(1.0 + 0.001(x^2+y^2))}$	$(-5.12,5.12)^n$	1

Table 1. Benchmark functions

2.3.2 Simulation and Analysis of the Algorithm

In order to study the property of the improved algorithm, the different performances are compared between the standard PSO and the improved PSO (mPSO) for Benchmark functions, which adopt linear decreased inertia weight coefficient. The optimal contrast test is performed on the common functions as shown in Table 1. For each algorithm, the maximum evolution generation is 3000, the number of the particles is 30 and the dimension is 10, 20 and 30 respectively, where the dimension of Schaffer function is 2. As for the inertia weight coefficient w, the initial value is 0.9 and the end value is 0.4 in the PSO algorithm, while in the mPSO algorithm, the value of w is fixed and taken to 0.375.The optimum point of the Rosenbrock function is in the position X=1 in theory, while for the other functions, the optimum points are in the position X=0 and the optimum value are $f(x)$= 0. The 50 different optimization search tests are performed on different dimensions of each function. The results are shown in Table 2, where the parameter Avg/Std refers to the average and variance of the optimal fitness value respectively during the 50 tests, iterAvg is the average number of evolution, Ras is the ratio of the number up to target value to the total test number. The desired value of function optimization is set as 1.0e-10, as the fitness value is less than 10e-10, set as 0.

		PSO			mPSO		
Fun.	Dim	Avg/Std	iterAvg	Ras	Avg/Std	iterAvg	Ras
f1	10	0/0	1938.52	50/50	0/0	340.18	50/50
	20	0/0	2597.24	50/50	0/0	397.64	50/50
	30	0/0	3000	1/50	0/0	415.02	50/50
f2	10	2.706/1.407	3000	0/50	0/0	315.24	50/50
	20	15.365/4.491	3000	0/50	0/0	354.10	50/50
	30	41.514/11.200	3000	0/50	0/0	395.22	50/50
f3	10	0.071/0.033	3000	0/50	0/0	294.32	50/50
	20	0.031/0.027	2926.94	8/50	0/0	343.42	50/50
	30	0.013/0.015	2990.52	13/50	0/0	370.06	50/50
f4	10	13.337/25.439	3000	0/50	8.253/0.210	3000	0/50
	20	71.796/175.027	3000	0/50	18.429/0.301	3000	0/50
	30	122.777/260.749	3000	0/50	28.586/0.2730	3000	0/50
f5	10	0/0	2197.40	50/50	0/0	468.08	50/50
	20	0/0	2925.00	47/50	0/0	532.78	50/50
	30	0.027/0.190	3000	0/50	0/0	562.60	50/50
f6	2	0.0001/0.002	857.58	47/50	0/0	67.98	50/50

Table 2. Performance comparison between mPSO and PSO for Benchmark problem

As shown in Table 2, except for the Rosenbrock function, the optimization results of the other functions reach the given target value and the average evolutionary generation is also very little. For the Schaffer function, the optimization test is performed on 2-dimension, while for the other functions, the tests are performed on from 10 dimensions to 30 dimensions. Compared with the standard PSO algorithm, whether the convergence accuracy or the convergence speed of the mPSO algorithm has been significantly improved, and the mPSO algorithm has excellent stability and robustness.

In order to illustrate the relationship between the particle activity and the algorithm performance in different algorithms, the diversity of particle swarm indicates the particle activity. The higher the diversity of particle swarm is, the greater the particle activity is, and the stronger the global search capability of particles is. The diversity of particle swarm is represented as the average distance of the particles, which is defined by Euclidean distance, and the distance L refers to the maximum diagonal length in the search space; The parameters of S and N represent the population size and the solution space dimension, respectively; p_{id} refers to the No.d dimension coordinate of the No.i particle; $\overline{p_d}$ is the average of the No.d dimension coordinate of all particles, so the average distance of the particles is defined as followed:

$$d(t) = \frac{1}{sL} \cdot \sum_{i=1}^{s} \sqrt{\sum_{j=1}^{N} (p_{ij} - \overline{p_d})^2} \tag{4}$$

For the 30-D functions (Schaffer function is 2-D), the optimal fitness value and particles' average distance are shown in Fig.1-6, which indicates the optimization result contrast of the mPSO and PSO algorithm performed on different functions.

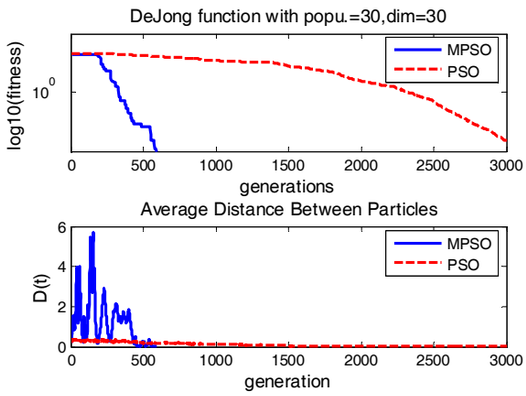

Fig. 1. Minima value and particles' average distance for 30-D Sphere

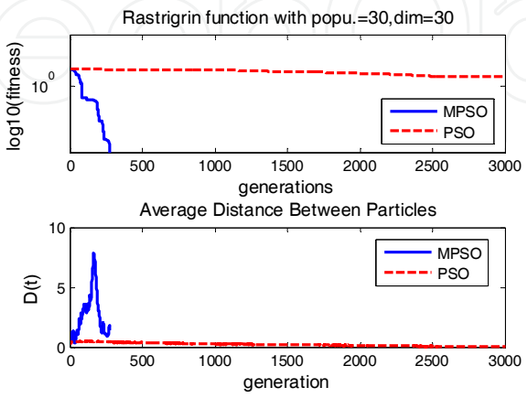

Fig. 2. Minima value and particles' average distance for 30-D Rastrigin

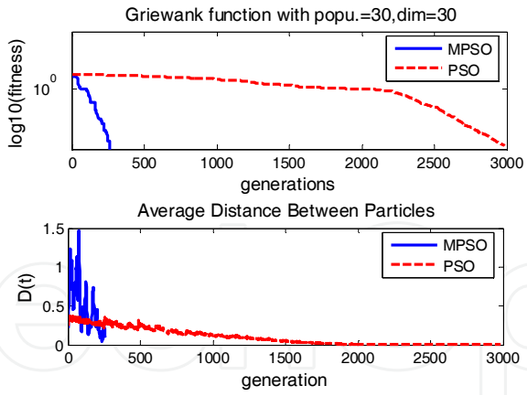

Fig. 3. Minima value and particles' average distance for 30-D Griewank

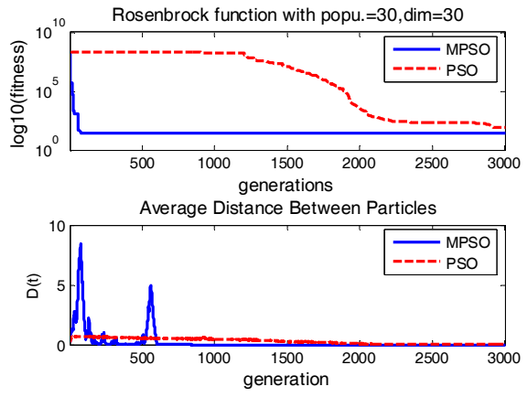

Fig. 4. Minima value and particles' average distance for 30-D Rosenbrock

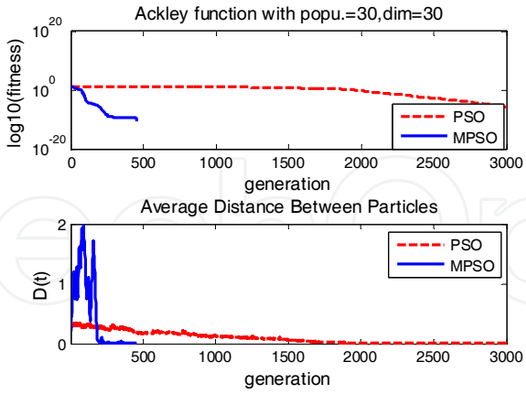

Fig. 5. Minima value and particles' average distance for 30-D Ackley

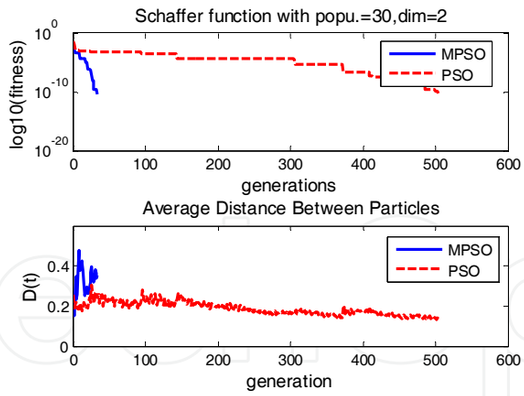

Fig. 6. Minima value and particles' average distance for 2-D Schaffer

As can be seen from the Figure 1-6, except for the Rosenbrock function, the average distance of particle swarm varies considerably, which indicates the particle's high activity as well as the good dynamic flight characteristic, which can also be in favor of the global search due to the avoidance of local minimum. When the particle approaches the global extreme point, the amplitude of its fluctuation reduces gradually, and then the particle converges quickly to the global extreme point. The mPSO algorithm has demonstrated the high accuracy and fast speed of the convergence. Compared with the corresponding graph of PSO algorithm in the chart, the the particles' average distance of the PSO algorithm decreases gradually with the increase of evolution generation, and the fluctuation of the particles is weak, and the activity of the particles disappears little by little, which is the reflection of the algorithm performance, i.e., it means slow convergence speed and the possibility of falling into local minimum. As weak fluctuation means very little diversity of particle swarm, once the particles fall into local minimum, it is quite difficult for them to get out. The above experiments, performed on the test functions, show that: the higher the diversity of particle swarm is, the greater the particle activity is, and the better the dynamic property of particle is, which result in stronger optimization property. Therefore, it is a key step for the PSO to control the activity of the particle swarm effectively. Besides, from the optimization results of mPSO algorithm shown in Table 2, it can be seen that, except for the Rosenbrock function, not only the mean of the other functions has reached the given target value, but also the variance is within the given target value, which shows that the mPSO algorithm has high stability and has better performance than the PSO algorithm. In addition, the chart has also indicated that, for the optimization of Rosenbrock function, whether the mPSO or the PSO algorithm is applied, the particles have high activity at the beginning, then gather around the adaptive value quickly, after which the particle swarm fall into the local minimum with the loss of its activity. Though the optimization result of mPSO for Rosenbrock function is better than the standard PSO algorithm, it has not yet got out of the local minimum. Hence, further study is needed on the optimization of PSO for Rosenbrock function.

3. BP Network Algorithm Based on PSO

3.1 BP Neural Network

Artificial Neural Network (ANN) is an engineering system that can simulate the structure and intelligent activity of human brain, which is based on a good knowledge of the structure

and operation mechanism of the human brain. According to the manner of neuron interconnection, neural network is divided into feedforward neural network and feedback neural network. According to the hierarchical structure, it is separated into single layer and multi-layer neural network. In terms of the manner of information processing, it is separated into continuous and discrete neural network, or definitive and random neural network, or global and local approximation neural network. According to the learning manner, it is separated into supervision and unsupervised learning or weight and structure learning. There are several dozens of neural network structures such as MLP, Adaline, BP, RBF and Hopfield etc. From a learning viewpoint, the feedforward neural network (FNN) is a powerful learning system, which has simple structure and is easy to program. From a systemic viewpoint, the feedforward neural network is a static nonlinear mapping, which has the capability of complex nonlinear processing through the composite mapping of simple nonlinear processing unit.

As the core of feedforward neural network, the BP network is the most essential part of the artificial neural network. Owing to its clear mathematical meaning and steps, Back-Propagation network and its variation form are widely used in more than 80% of artificial neural network model in practice.

3.2 BP Network Algorithm Based on PSO

The BP algorithm is highly dependent on the initial connection weight of the network, therefore, it has the tendency of falling into local minimum with improper initial weight. However, the optimization search of the BP algorithm is under the guidance (in the direction of negative gradient), which is superior to the PSO algorithm and other stochastic search algorithm. There is no doubt that it provides a method for the BP optimization with derivative information. The only problem is how to overcome the BP algorithm for the dependence of the initial weight. The PSO algorithm has strong robustness for the initial weight of neural network (Wang Ling, 2001). By the combination of the PSO and BP algorithm, it could improve the precision, speed and convergence rate of BP algorithm, which makes full use of the advantage of the PSO and BP algorithm, i.e., the PSO has great skill in global search and BP excels in local optimization.

Compared with the traditional optimization algorithm, the feedforward neural network has great differences such as multiple variables, large search space and complex optimized surface. In order to facilitate the PSO algorithm for BP algorithm in certain network structure, the weight vector of NN is used to represent FNN, and each dimension of the particles represents a connection weights or threshold value of FNN, which consists of the individuals of the particle swarm. To take one input layer, a hidden layer and an output layer of FNN as an example, when the number of input nodes was set as R, the number of output nodes was set as $S2$ and the number of hidden nodes was set as $S1$, the dimension N of particles can be obtained from the formula (5):

$$N=S1*(R+1)+S2*(S1+1)+S3*(S2+1) \tag{5}$$

The dimension of the particles and the weight of FNN can be obtained by the following code conversion:

```
for i=1:S1
    w1(i,:)=Swarm(iPopindex,R*(i-1)+1:R*(i-1)+R);
    c1(i)=Swarm(iPopindex,S1*R+i);
end
b1=c1';
for i=1:S2
  w2(i,:)=Swarm(iPopindex,S1*(R+1)+S1*(i-1)+1:S1*(R+1)+S1*(i-1)+S1);
    c2(i)=Swarm(iPopindex,S1*(R+1)+S2*S1+i);
end
b2=c2';
```

Where, iPopindex refers to the serial number of the particles.

When training the BP network through PSO algorithm, the position vector X of particle swarm is defined as the whole connection weights and threshold value of BP network.. On the basis of the vector X, the individual of the optimization process is formed, and the particle swarm is composed of the individuals. So the method is as follows: at first, initializing the position vector, then minimize the sum of squared errors (adaptive value) between the actual output and ideal output of network, and the optimal position can be searched by PSO algorithm, as shown in the following formula (6):

$$J = \sum_{i=1}^{N} \sum_{k=1}^{c} (T_{ik} - Y_{ik})^2 \tag{6}$$

Where: N is the sample number of training set; T_{ik} is the ideal output of the No. k output node in the No. i sample; Y_{ik} is the actual output of the No. k output node in the No. i sample; C is the number of output neuron in the network.

The PSO algorithm is used to optimize the BP network weight (PSOBP), the method includes the following main steps:

(1) The position parameter of particle can be determined by the connection weights and the threshold value between the nodes of neural network.
(2) Set the values range [Xmin, Xmax] of the connection weights in neural network, and generate corresponding uniform random numbers of particle swarm, then generate the initial swarm.
(3) Evaluate the individuals in the swarm. Decode the individual and assign to the appropriate connection weights (including the threshold value). Introduce the learning samples to calculate the corresponding network output, then get the learning error E, use it as the individual's adaptive value.
(4) Execute the PSO operation on the individuals of the swarm
(5) Judge the PSO operation whether terminate or not? No, turn to step (3), Otherwise, to step (6).

(6) Decode the optimum individual searched by PSO and assign to the weights of neural network (include the threshold value of nodes).

3.3 FNN Algorithm Based on Improved PSO

The improved PSO (mPSO) is an algorithm based on the optimal location variation for the individual of the particle swarm. Compared with the standard PSO, the mPSO prevents the particles from gathering at the optimal location *gbest* quickly by means of individual extreme variation of the swarm, which enhances the diversity of particle swarm.

The algorithm flow of FNN is as follows:

(1) Setting the number of hidden layers and neurons of neural network. Determining the number m of particles, adaptive threshold e, the maximum number $Tmax$ of iterative generation; acceleration constants $c1$ and $c2$; inertia weight w; Initializing the P and V, which are random number between [-1, 1].

(2) Setting the iteration step t=0; Calculating the network error and fitness value of each particle according to the given initial value; Setting the optimal fitness value of individual particles, the individual optimal location, the optimal fitness value and location of the particle swarm.

(3) while($J_g > e$ & $t < T_{max}$)

for i = 1 : m

Obtaining the weight and threshold value of the neural network from the decoding of x_i and calculating the output of the neural network, compute the value of J_i according to the formula (6):

if $J_i < J_p(i)$ $J_p(i) = J_i$; $p_i = x_i$; *end if*

if $J_i < J_g$ $J_g = J_i$; $p_g = x_i$; *end if*

end for

(4) for i=1:m

Calculating the v_i and x_i of particle swarm according to the PSO;

end for

(5) Execute the variation operation on the individual optimal location of the swarm according to the formula (3).

(6) *t=t+1;*

(7) end while

(8) Result output.

3.4 BP NN Algorithm Based on PSO and L-M

Because the traditional BP algorithm has the following problems: slow convergence speed, uncertainty of system training and proneness to local minimum, the improved BP algorithm is most often used in practice. The Levenberg-Marquardt (L-M for short) optimization algorithm is one of the most successful algorithm among the BP algorithms based on derivative optimization. The L-M algorithm is developed from classical Newton algorithm by calculating the derivative in terms of the nonlinear least squares. The iterative formula of LM algorithm is as follows(Zhang ZX, Sun CZ & Mizutani E, 2000):

$$\theta_{n+1} = \theta_n - \eta(J_n^T J_n + \lambda_n I_n)^{-1} J_n^T r_n \tag{7}$$

Where, I is the unit matrix, λ is a non-negative value. Making use of the changes in the amplitude of λ, the method varies smoothly between two extremes, i.e., the Newton method (when $\lambda \to 0$) and standard gradient method (when $\lambda \to \infty$). So the L-M algorithm is actually the combination of standard Newton method and the gradient descent method, which has the advantages of both the latter two methods.

The main idea of the combination algorithm of PSO and L-M (PSOLM algorithm) is to take the PSO algorithm as the main framework. Firstly, optimize the PSO algorithm, after the evolution of several generations, the optimum individual can be chosen from the particle swarm to carry out the optimization search of L-M algorithm for several steps, which operates the local depth search. The specific steps of the algorithm is as follows:

(1) Generate the initial particle swarm X at random, and $k = 0$.
(2) Operate the optimization search on X with the PSO algorithm.
(3) If the evolution generation k of PSO is greater than the given constant dl, chose the optimal individual of particle swarm to carry out the optimization search of L-M algorithm for several steps.
(4) Based on the returned individual, reassess the new optimal individual and global optimal individual by calculating according to PSO algorithm.
(5) If the target function value meets the requirements of precision ε, then terminate the algorithm and output the result; otherwise, $k = k + 1$, turn to step (2).

The above PSO algorithm is actually the particle swarm optimization algorithm (MPSO) by means of the optimal location variation of individual, and the particle number of particle swarm is 30, $c1=c2=1.45$, $w=0.728$.

4. Research on Neural Network Algorithm for Parity Problem

4.1 XOR Problem

Firstly, taking the XOR problem (2 bit parity problem) as an example to discuss it. The XOR problem is one of the classical questions on the NN learning algorithm research, which includes the irregular optimal curved surface as well as many local minimums. The learning sample of XOR problem is shown in Table 3.

Sample	Input	Output
1	00	0
2	01	1
3	10	1
4	11	0

Table 3. Learning sample of XOR

Different network structures result in different learning generations of given precision10^{-n} (where: n is the accuracy index). In this part, there is a comparison between the learning generations and the actual learning error. The initial weight ranges among [-1, 1] in BP network and conducted 50 random experiments.

As shown in Table4, it displays the experimental results of 2-2-1 NN structure. The activation functions are S-shaped hyperbolic tangent function (Tansig), S-shaped logarithmic function (Logsig) and linear function (Purelin) respectively, and the learning algorithms include the BP, improved BP (BP algorithm with momentum, BPM) and BP based on the Levenberg-Marquardt (BPLM). Judging from the results for XOR problem, as the number of the neurons in the hidden layer is 2, the BP and improved BP (BPM, BPLM) can't converge completely in 50 experiments.

It can also be seen that the performance of the improved BP is better than that of the basic BP, as for the improved BP, the BPLM performs better than BPM. In addition, the initial value of the algorithm has great influence on the convergence property of BP algorithm, so is the function form of the neurons in the output layer.

XOR Problem		BP		BPM		BPLM	
Activation function	Hidden layer	Tansig	Tangsig	Tansig	Tangsig	Tansig	Tangsig
	Output layer	Purelin	Logsig	Purelin	Logsig	Purelin	Logsig
NN structure： 2-2-1		56%	0	60%	0	72%	0

Table 4. Convergence statistics of BP, BPM and BPLM (Accuracy index n=3)

The Table 5 shows the training results under different accuracy indices. The activation functions are Tansig-purelin and tansig-logsig respectively, and the NN algorithms include the BPLM and the PSO with limited factor (cPSO, Clerc, M., 1999). It can be indicated that the basic PSO, which is applied to the BP network for XOR problem, can't converge completely, either. In such circumstance, the number of the neurons in the hidden layer is 2.

XOR learning	Accuracy index	BPLM		cPSO			
		Tansig-purelin		Tansig-purelin		Tansig-logsig	
		Average iteration number	Ras	Average iteration number	Ras	Average iteration number	Ras
Network stucture 2-2-1	3	16.36	36	71.00	35	24.71	26
	6	20.84	40	145.94	36	64.88	25
	10	13.76	38	233.36	36	43.52	25
	20	25.99	8	461.13	38	68.21	26

Table 5. BP training results of BPLM, cPSO, and mPSO

Besides, for the BP and the improved BP algorithm, it has never converged in the given number of experiments when the activation function of output layer in NN is Logsig, while

the form of activation function has relatively minor influence on the PSO algorithm. It can be seen from the table that the form of activation function has certain influence on the learning speed of NN algorithm based on PSO, and the learning algorithm, which adopts Tangsig-Logsig function converges faster than that adopts Tangsig-Purelin function.
The Table 6 shows the optimization results of the PSOBP and PSOBPLM algorithm, which are the combination of MPSO and standard BP (PSOBP) as well as the combination of MPSO and BP algorithm based on L-M (PSOBPLM) respectively. As seen in Table 6, for the given number of experiments, the optimization results of the algorithms have all achieved the specified target value within the given iteration number.

XOR	**PSOBP**			**PSOBPLM**		
Accuracy index	Average iteration number		Mean time (s/time)	Average iteration number		Mean time (s/time)
	PSO	BP		PSO	BP	
3	11.06	379.15	3.74	21	2.67	0.72
6	11.12	517.35	5.35	21	3.8	0.75
10	11.46	910.5	8.31	21	4.73	0.73
20	12.3	1578.55	13.37	23.07	20.13	1.97

Table 6. BP optimization results of PSOBP and PSOBPLM algorithm

In addition, the Table 6 has also displayed the average iteration number and the mean time of PSO and BP algorithm under different accuracy indices in 50 experiments. As shown in Table 3, the algorithm of PSO combined with BP or LM has good convergence property, which is hard to realize for single BP (including BPLM) or PSO algorithm. It's especially necessary to notice that the combination of the PSO and LM algorithm brings about very high convergence speed, and the algorithm of PSOBPLM converges much faster than PSOBP algorithm under the condition of high accuracy index. For example, when the network structure is 2-2-1 and the accuracy index is 10 and 20 respectively, the relevant mean time of PSOBP algorithm is 8.31 and 13.37, while for the PSOBPLM algorithm, the mean time is reduced to 0.73 and 1.97. Obviously, the PSOBPLM algorithm has excellent speed performance.

4.2 Parity Problem

The parity problem is one of the famous problems in neural network learning and much more complex than the 2bit XOR problem. The learning sample of parity problem consists of 4-8 bit binary string. When the number of 1 in binary string is odd, the output value is 1; otherwise, the value is 0. When the PSO (including the improved PSO) and PSOBP algorithm are applied to solve the parity problem, the learning speed is quite low and it is impossible to converge to the target value in the given iteration number. The PSOBPLM algorithm, proposed in this article, is applied to test the 4-8bit parity problem. The network structure of 4bit parity problem is 4-4-1, and the activation function of both hidden layer and output layer are Tansig-logsig, the same is with the activation function of NN for 5-8bit

parity problem, and the parameter of NN for 5-8bit parity problem can be got from that of NN for 4bit parity problem by analogy. For each parity problem, 50 random experiments are carried out. The Table 7 shows the experimental result of the PSOBPLM algorithm for 4-8bit parity problem under various accuracy indices. In the Table 7, the Mean, Max and Min represent the average iteration number, the maximum and minimum iteration number, respectively. The number below the PSO and BP column represents the iteration number needed by the corresponding algorithm.

net:4-4-1;							
Accuracy index	Mean		Max		Min		Mean time (s/time)
	PSO	LM	PSO	LM	PSO	LM	
3	21.07	67.60	22	489	21	12	1.15
6	21.10	80.77	22	424	21	11	1.19
10	21.17	114.5	22	699	21	14	1.31
20	25.23	405.73	35	1414	21	18	4.62
net:5-5-1;							
3	50.10	99.27	51	532	50	16	1.49
6	50.07	103.17	52	1019	50	19	1.58
10	50.13	143.57	51	557	50	12	1.84
20	53.77	371.07	65	1960	50	27	5.21
net:6-6-1;							
3	50.23	208.93	52	1103	50	23	2.97
6	50.13	204.47	51	591	50	34	2.58
10	50.50	334.57	53	1281	50	42	3.81
20	53.77	944.73	65	3069	50	49	10.72
net:7-7-1;							
3	50.27	267.5	51	708	50	29	4.66
6	50.27	279.7	51	686	50	35	4.64
10	50.33	278.67	52	1067	50	32	4.53
20	52.77	748.57	59	2206	50	57	11.69
net:8-8-1;							
3	50.23	273.53	52	1066	50	56	7.98
6	50.43	391.63	51	803	50	78	8.29
10	51.63	387.27	54	1388	51	71	10.71
20	54.83	1225.47	63	3560	51	65	30.43

Table 7. Result of PSOBPLM algorithm for 4-8 bit parity problem

As seen in Table 7, the integration of the PSO and L-M algorithm can solve the parity problem. The PSOBPLM algorithm makes full use of the advantage of the PSO and L-M algorithm, i.e., the PSO has great skill in global search and the L-M excels in local optimization, which compensate their own drawback and have complementary advantages. So the PSOLM algorithm has not only a good convergence, but also fast optimization property.

5. Conclusion

As a global evolutionary algorithm, the PSO has simple model and is easy to achieve. The integration of the PSO and L-M algorithm makes full use of their own advantage, i.e., the PSO has great skill in global search and the L-M excels in local fast optimization, which could avoid falling into local minimum and find the global optimal solution for the parity problem effectively. Meanwhile, the PSOBPLM algorithm has better efficiency and robustness. The only shortage of the algorithm is that it needs the derivative information, which increases the algorithm complexity to some extent.

6. References

Clerc, M. (1999). The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 1999)*, pp. 1951-1957, Washington, DC, 1999, IEEE Service Center, Piscataway, NJ.

Eberhart, R. C. & Kennedy, J. (1995). A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39-43, Nagoya, Japan, 1995, IEEE Service Center, Piscataway, NJ.

James L. McClelland, David E. Rumelhart & the PDP Research Group, (1986). *Parallel Distributed Processing*. MIT press, Cambridge, MA

Kennedy, J. & Eberhart, R. C. (1995). Particle swarm optimization, *Proceedings of IEEE International Conference on Neutral Networks*, IV, pp. 1942-1948, Perth, Australia, 1995, IEEE Service Center, Piscataway, NJ.

LU Zhensu & HOU Zhirong (2004). Particle Swarm Optimization with Adaptive Mutation, *ACATA ELECTRONICA SINICA*, Vol. 33, No. 3, 416-420.ISSN: 3972-2112

Wang Ling (2001). *Intelligent Optimization Algorithms With Applications*, Tsinghua University Press, ISBN: 7-302-04499-6, Beijing

Wang Xiaoping & Cao Liming (2002). *Genetic Algorithm-Its Theory,Application and Software Realization*, Xian Jiaotong University Press, ISBN: 7560514480, Xian

Zhang ZX, Sun CZ & Mizutani E (2000). *Neuro-Fuzzy and Soft Computing*, Xian Jiaotong University Press, ISBN: 7560511872, Xian

Zhou Zhihua, Cao cungen (2004). Neural networks and its application, Tsinghua University Press, ISBN: 7302086508, Beijing

Stochastic Control

Edited by Chris Myers

ISBN 978-953-307-121-3
Hard cover, 650 pages
Publisher Sciyo
Published online 17, August, 2010
Published in print edition August, 2010

Uncertainty presents significant challenges in the reasoning about and controlling of complex dynamical systems. To address this challenge, numerous researchers are developing improved methods for stochastic analysis. This book presents a diverse collection of some of the latest research in this important area. In particular, this book gives an overview of some of the theoretical methods and tools for stochastic analysis, and it presents the applications of these methods to problems in systems theory, science, and economics.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen