Multilayer Perceptron Learning with Particle Swarm Optimization for Well Log Data Inversion

Kou-Yuan Huang
Department of Computer Science
National Chiao Tung University
Hsinchu, Taiwan
kyhuang@cs.nctu.edu.tw

Liang-Chi Shen
Department of Electrical & Computer Engineering
University of Houston
Texas, USA
liang.shen@gmail.com

Kai-Ju Chen
National Chiao Tung University
Department of Computer Science
Hsinchu, Taiwan
chenkaiju@gmail.com

Ming-Che Huang
National Chiao Tung University
Department of Computer Science
Hsinchu, Taiwan
toolwell@gmail.com

***Abstract*—Well log data inversion is important for the inversion of true formation. There exists a nonlinear mapping between the measured apparent conductivity (C_a) and the true formation conductivity (C_t). We adopt the multilayer perceptron (MLP) to approximate the nonlinear input-output mapping and propose the use of particle swarm optimization with mutation (MPSO) algorithm to adjust the weights in MLP. In the supervised training step, the input of the network is the measured C_a and the desired output is the C_t. MLP with optimal size 10-9-10 is chosen as the model. We have experiment in simulation and real data application. In simulation, there are 31 sets of simulated well log data, where 25 sets are used for training, and 6 sets are used for testing. After training the MLP network, input Ca, then C_t' can be inverted in testing process. Also we apply it to the inversion of real field well log data. The result is acceptable. It shows that the proposed MPSO algorithm in MLP weight adjustments can work on the well log data inversion.**

Keywords- apparent conductivity (C_a); true formation conductivity (C_t); multilayer perceptron (MLP); particle swarm optimization with mutation (MPSO);

I. INTRODUCTION

Well logging is a technique that uses the instruments to measure the formation information. We invert the true formation parameters from the measured information. There is a nonlinear mapping between them. Martin et al. [1] used the neural network with conjugate gradient and Levenberg-Marquardt (LM) algorithms for well log data inversion. Huang et al. [2] [3] used the higher order multilayer perceptron and radial basis function network for well log data inversion. But the methods could get the local minimum solution. Goswami et al. [4] applied the differential evolution algorithm. Szucs and Civan [5] applied the simulated annealing algorithm. But they had complex computation.

Particle swarm optimization (PSO) was proposed by Kennedy [6] that searched the global optimal solution by a population of particles distributed over the search space. It was an evolutionary technique and used cooperative mechanism to find the solution. PSO with mutation (MPSO) was proposed in [7]. The PSO algorithm had the disadvantage that it might get trapped at the local minimum if velocity was large. To solve the problem, MPSO algorithm used the mutation that the mutated particles provided a chance to escape from the local minimum.

We adopt multilayer perceptron (MLP) with MPSO algorithm in this study. MLP was able to approximate the nonlinear input-output mapping. The adjustment of weights based on the gradient descent method may get the local minimum, so we use PSO global optimization algorithm to train the weights in MLP. The supervised system of MLP with MPSO is presented in Fig. 1 for training and testing of well log data inversion.

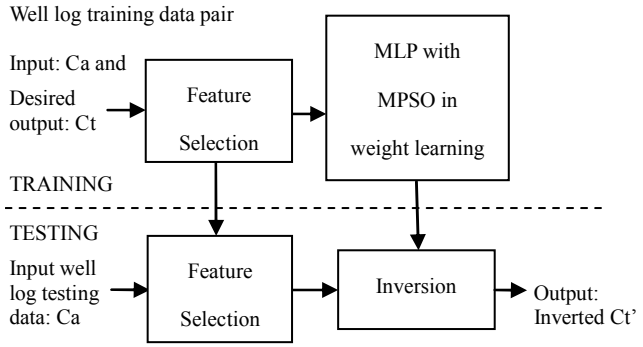

Fig. 1. System of MLP with MPSO in training and testing of well log data inversion.

Section II is the MLP weight learning with MPSO. Section III is the procedures of training and testing. Section IV is the experimental results. Finally, section V is the conclusions.

II. MLP WEIGHT LEARNING AND MPSO

PSO simulates the behavior of flocks of birds [6]. Initially we set up m particles. One particle in PSO is one MLP network. A two-layer MLP using MPSO learning is shown in Fig. 2. The input value of the first layer $x^1 = [x_1, x_2, \cdots, x_I, 1]^T$ is C_a and the output value in the output layer is the inverted C_t'. The output value of hidden node j is

$$s_j = \sum_{i=1}^{I+1} w_{ji} x_i \tag{1}$$

The output value of output node k is

$$s_k = \sum_{j=1}^{J+1} w_{kj} o_j \tag{2}$$

The activation function is a sigmoidal transfer function. So, the outputs of the second layer, o_j, and the third layer, o_k, are

$$o_j = f(s_j) = \frac{1}{1+e^{-s_j}} \tag{3}$$

$$o_k = f(s_k) = \frac{1}{1+e^{-s_k}} \tag{4}$$

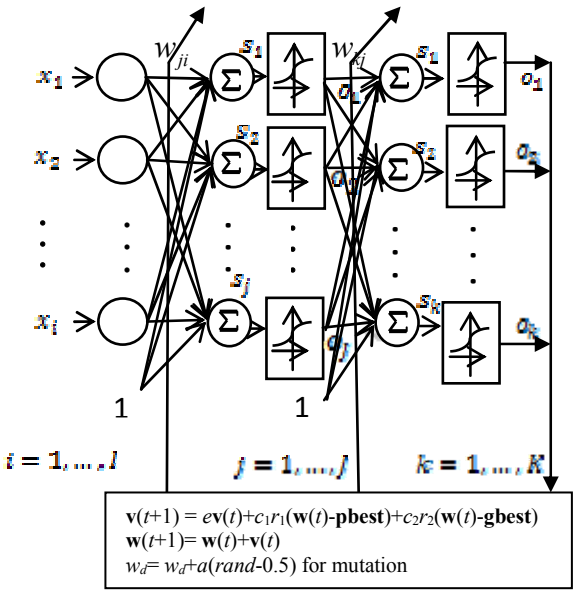

Fig. 2. Two-layer perceptron.

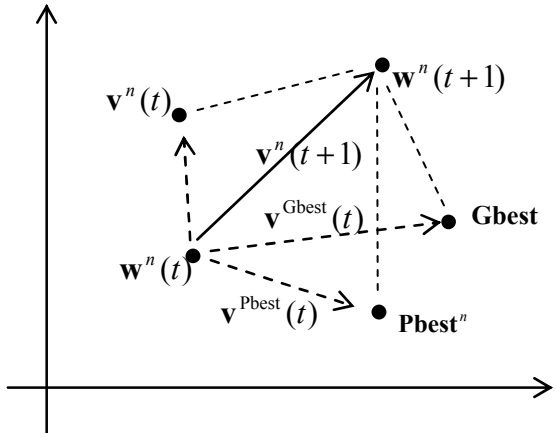

Fig. 3. The movements of a particle in MPSO.

We concatenate the two-layer weighting coefficients of MLP to be a vector as a particle in MPSO. A particle is represented as

$$\mathbf{w}^n = [w_{1,1}, \ldots, w_{j,i}, \ldots, w_{J,I+1}, w_{1,1}, \ldots, w_{k,j}, \ldots, w_{K,J}, w_{K,J+1}],$$

where $n=1, \ldots, m$, n is the index of particle, m is the number of particles, $w_{j,i}$ and $w_{k,j}$ are the weighting coefficients in MLP shown in Fig. 2. The weighting vector $\mathbf{w}$ is equivalent to position in MPSO.

Each particle has a velocity vector

$$\mathbf{v}^n = [v_{1,1}, \ldots, v_{j,i}, \ldots, v_{J,I+1}, v_{1,1}, \ldots, v_{k,j}, \ldots, v_{K,J}, v_{K,J+1}],$$

which is also a velocity vector in MPSO. N is the length of $\mathbf{w}$ and $\mathbf{v}$.

Fig. 3 shows the movements of a particle in MPSO and we want to find the point with minimum fitness value in the search space [6]. Weighting vector adjustment in one MLP is equivalent to the movement of a particle in MPSO.

We have two important weighting vectors $\mathbf{Pbest}^n$ and **Gbest** that affect the global minimum solution. $\mathbf{Pbest}^n$ is the best weighting coefficient vector that the particle itself has found. **Gbest** is the best weighting coefficient vector of all particles so far.

The velocity vector and weighting vector are adjusted by

$$\mathbf{v}^n(t+1) = e\,\mathbf{v}^n(t) + c_1 r_1 (\mathbf{w}^n(t) - \mathbf{Pbest}^n) + c_2 r_2 (\mathbf{w}^n(t) - \mathbf{Gbest}), \tag{5}$$

$$\mathbf{w}^n(t+1) = \mathbf{w}^n(t) + \mathbf{v}^n(t+1) \qquad n = 1, \ldots, m \tag{6}$$

where e is the inertia weight, c_1 and c_2 are the acceleration constants, r_1 and r_2 are random numbers uniformly distributed over [0, 1]. In (5), the new velocity is composed of three terms. The first term is the inertia of the previous velocity. The second term is that the particle $\mathbf{w}^n(t)$ points toward its personal best weighting coefficient vector, $\mathbf{Pbest}^n$. And the third term is that the particle $\mathbf{w}^n(t)$ points toward the best weighting coefficient vector of all particles so far, **Gbest.** Formula (6) is the weight

vector adjustment, that the updated velocity is added to the current weighting coefficient vector $\mathbf{w}^n(t)$.

Initially, we have m particles. Randomly generate the n-th particle with $\mathbf{v}^n(0)$ and $\mathbf{w}^n(0)$. And set $\mathbf{w}^n(0)$ to the personal best weighting coefficients $\mathbf{Pbest}^n$. The fitness function is the sum of the squared error (SSE) and is defined as

$$\mathrm{SSE} = \frac{1}{2}\sum_{q=1}^{Q}\sum_{k=1}^{K}(d_{qk} - o_{qk})^2 \tag{7}$$

where Q is the number of training samples and K is the number of output nodes. d_{qk} is the desired output of the k^{th} output node of the q^{th} training sample and o_{qk} is the real output of the k^{th} output node of the q^{th} training sample. Each particle has a fitness value in (7). $\mathbf{Pbest}^n$ and **Gbest** use the (7) in the calculation of fitness value.

Mutation is the process to possibly lead some of the swarms away from the current position by changing the mutated particle's element. For mutation, the first step is the mutation of the particles. Ten percent of particles are selected to mutate with uniform random probability. At the second step, from mutated particle, one element w_d with uniform random probability 1/N is mutated and the selected element mutates according to (8).

$$w_d = w_d + a(rand - 0.5) \tag{8}$$

where a is a constant that controls the amplitude of the mutated component, *rand* is the uniform random number distributed over [0, 1]. Fig. 4 shows the flowchart of MPSO algorithm for weight adjustments in MLP.

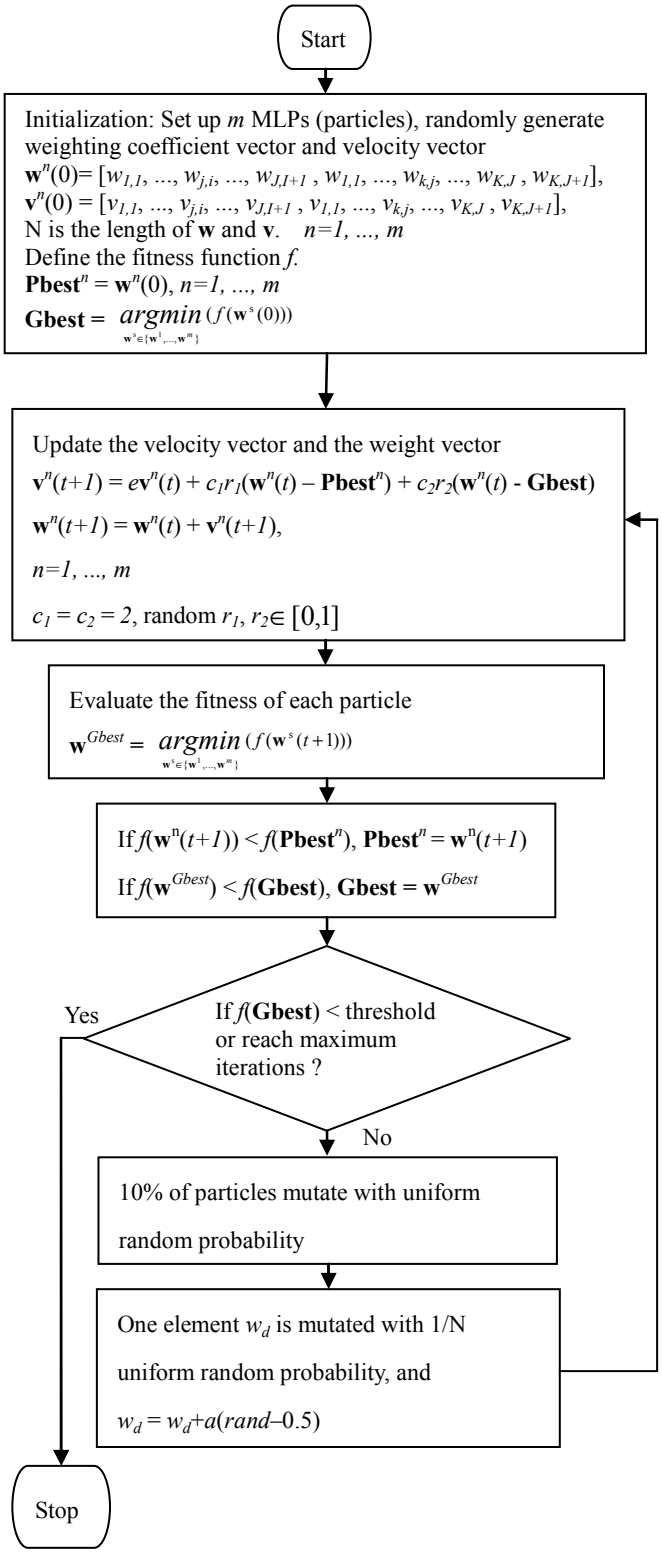

Fig. 4. Flowchart of MPSO for weight adjustments in MLP.

III. PROCEDURES OF TRAINING AND TESTING

We follow the procedures of training and testing in Fig. 1. There are 31 synthetic well log datasets that are calculated by the electromagnetic theory. Each simulated well log is from

490 to 589.5 feet depth and the sample interval is 0.5 feet. Hence, there are 200 C_a and the correspondents C_t in each simulated well log. The 1st to the 25th simulated datasets are used for training, and others, 26th to 31st, are used for testing.

In the training stage, we select 25 well log datasets as training where the input of the network is C_a and the desired output is C_t. Weight adjustments are trained by MPSO algorithm. By the theorem developed by Mirchandani and Cao [8] and the conclusions in [2] and [3], we choose MLP with optimal size 10-9-10. In training, we take and skip every 10 C_a data as the input and every 10 C_t data as the desired output. That is, when the input pattern from the 1st to the 10th C_a data is fed into the input layer, the next pattern fed into the input layer is the 11th to the 20th C_a data. Hence, for 25 well log datasets, the number of training patterns is 25×200/10 = 500.

After the convergence of training process, we adopt mean absolute error (MAE) to test 6 well log datasets from the 26th to the 31st well logs. The input data are every 10 C_a, and the inverted output data are every 10 C_t'.

$$\mathrm{MAE} = \frac{1}{QK}\sum_{q=1}^{Q}\sum_{k=1}^{K}| d_{qk} - o_{qk} | \qquad (9)$$

After the convergence of training, we also apply the MLP to the inversion of the real field well log data.

IV. EXPERIMENTAL RESULTS

A. Results of simulated well log data inversion

We use this 10-9-10 MLP model with MPSO algorithm in the experiments. Table 1 shows the used parameters. The stop iteration is set to 20,000. Fig. 5 shows SSE of the best particle versus iteration in the training process. The SSE of the best particle after 20,000 iterations is 0.04589. Fig. 6 shows the 26th well log data, the dot line is C_a and the solid line is C_t. Fig. 7 shows the C_a and C_t of the 31st well log data. Fig. 8 and Fig. 9 show the inverted C_t' and desired C_t of the 26th and the 31st well logs, respectively. The dot line is the input C_a and solid line is the inverted C_t'. Table 2 shows the MAE error from the 26th to the 31st well logs. From the experiments, the average MAE of each point in 6 well log datasets between the inverted C_t' and the desired C_t is 0.004194. It shows that MLP with MPSO gets a good result for well log data inversion. Table 3 compares the average of MAE error of 6 well log data inversion between radial basis function network (RBF) models [3] and MLP with MPSO. From the comparison, MLP with MPSO has the minimum average MAE.

Table 1. Parameters used in MLP learning with MPSO.

m	N	*e* (decreasing linearly)	c_1	c_2	*a*
20	199	0.9-0.4	2	2	1

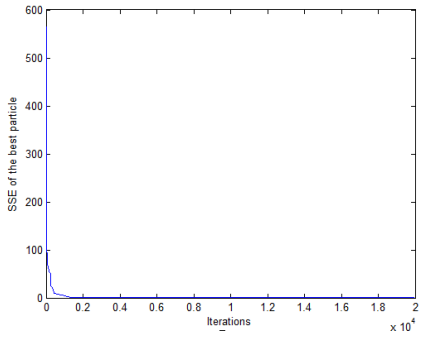

Fig. 5. SSE of the best particle vs. iteration in 10-9-10 MLP.

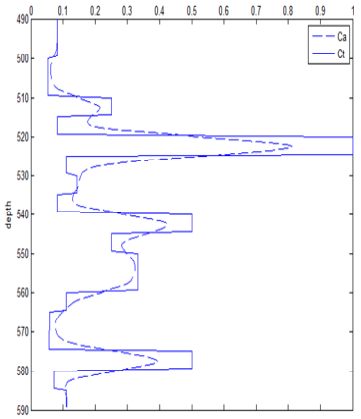

Fig. 6. Ca and Ct of the 26th well log.

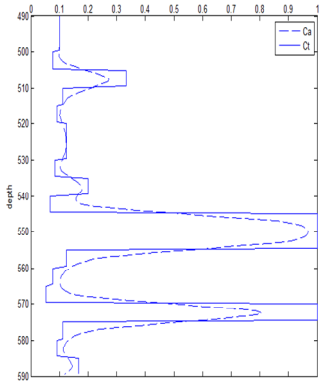

Fig. 7. Ca and Ct of the 31st well log.

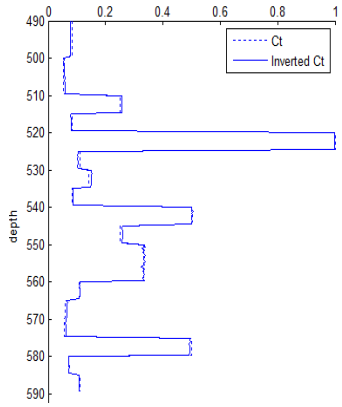

Fig. 8. Inverted Ct' and desired Ct at the 26th well log.

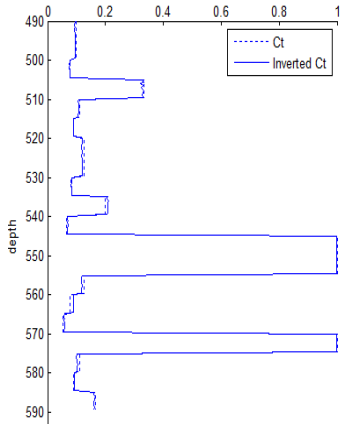

Fig. 9. Inverted Ct' and desired Ct at the 31st well log.

Table 2. The MAE between input and inverted output from the 26th to the 31st well log data.

Well log data	MAE of well log data inversion
#26	0.004651
#27	0.004244
#28	0.004574
#29	0.003551
#30	0.003882
#31	0.004260
Average MAE	0.004194

Table 3.Comparison of average of mean absolute error of 6 well log data inversion between RBF models and MLP with MPSO.

Model	Network size	Number of training patterns	Average MAE of 6 well log data inversion
Modified two-layer RBF	10-27-10	500	0.048003
Modified three-layer RBF	10-27-9-10	500	0.046625
MLP with MPSO	10-9-10	500	0.004194

B. Results of real well log data inversion

After the training of the 10-9-10 network, we also apply it for the inversion of real field well log data. Real field log contains 2,210 data from depth 5,577.5 to 6,682 feet and the sample interval is 0.5 feet. Fig. 10 shows the real field data C_a. The input data to the 10-9-10 MLP is the real conductivity. Fig. 11 shows the inverted C_t'. Dot line is the measured real conductivity C_a and solid line is the inverted C_t'. Because there is no desired C_t in real well log data, we calculate the MAE error of each point between the input C_a and the inverted C_t' output, and get 0.05283.

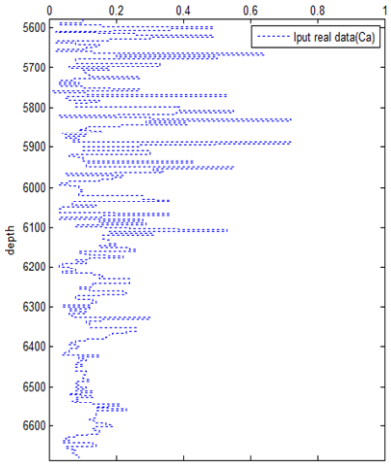

Fig. 10. Real field data with depth from 5,577.5 to 6,822 feet.

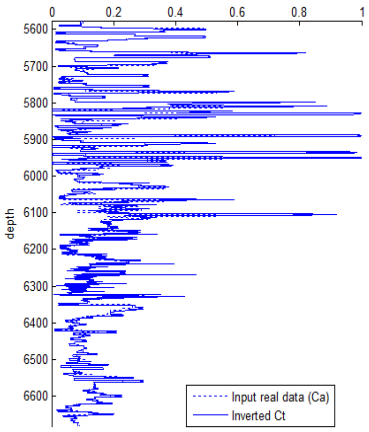

Fig. 11. Input Ca and inverted Ct' at real data.

V. CONCLUSIONS

MLP with MPSO algorithm for weight training is proposed. MLP is able to approximate the nonlinear input-output mapping. The global optimization property of MPSO algorithm can avoid the local minimum during the training in MLP. In the experiments, 31 simulated well log datasets are used for supervised training and testing. The optimal network size 10-9-10 is used as the MLP model. After training of the MLP network, we apply it for the inversion of the real field well log data. The experimental result is acceptable. It shows that the proposed method can work on the well log data inversion.

ACKNOWLEDGEMENT

This work was supported in part by the National Science Council, Taiwan, under NSC100-2221-E-009-139.

REFERENCES

[1] L. S. Martin, D. Chen, T. Hagiwara, R. Strickland, G. Gianzero, and M. Hagan, "Neural network inversion of array induction logging data for dipping beds," Society of Professional Well Log Analysts, 42nd Annual Logging Symposium, Paper U, pp. 1-11, Jun 17-20, 2001.

[2] K. Y. Huang, L. C. Shen, and C. Y. Chen, "Higher Order Neural Networks for Well Log Data Inversion," IEEE International Joint Conference on Neural Networks, pp. 2545-2550, 2008.

[3] K. Y. Huang, L. C. Shen, and L. S. Wang, "Radial basis function network for well log data inversion," IEEE International Joint Conference on Neural Networks, pp. 1093-1098, 2011.

[4] J. C. Goswami, R. Mydur, P. Wu, and D. Hwliot, "A robust technique for well-log data inversion," IEEE Transactions on Antennas and Propagation, Volume 52, Issue 3, pp. 717-724, March 2004.

[5] P. Szucs and F. Civan, "Multi-layer well log interpretation using the simulated annealing method," Journal of Petroleum Science and Engineering, Volume 14, Issue 3-4, pp. 209–220, May 1996.

[6] J. Kennedy and R. C. Eberhart, "Particle Swarm Optimization," IEEE International Conference on Neural Networks, Piscataway, Volume 4, pp.1942-1948, 1995.

[7] A. Stacey, M. Jancic, and I. Grundy, "Particle swarm optimization with mutation," IEEE CEC Conference on Evolutionary Computation, Volume 2, pp. 1425–1430, 2003.

[8] G. Mirchandani and W. Cao, "On hidden nodes for neural nets," IEEE Transactions on Circuits and Systems, Volume 36, No. 5, pp. 661-664, May 1989.