Nonlinear System Identification with On-Line Learning Self Organization Neural Networks

Chi-Y-Feng Perng, *Student Member IEEE* and Yung-Yaw Chen, *Member IEEE*
Dept. of Electrical Engineering, Lab 202, National Taiwan University
Taipei, Taiwan, R.O.C.

Abstract - It is important to identify the dynamics of systems for control. There exists two approaches to identify the unknown system. One is parameters identifications and the other is functional methods. Self Organization Neural Network (SONN) derived from GMDH works as a functional method. The original SONN needs a lot of previous input and output data to build up the model. We proposed a modification on SONN such that SONN can do on-line learning to identify the unknown system without gathering a large amount of data at first. In this paper, section 1 introduces the original concepts of SONN. Section 2 gives the details of our modified SONN and explains why we would develop the on learning algorithm. Section 3 would show the simulation results. And we will make a conclusion in section 4.

1. Introduction

Neural networks have been developed for nearly 40 years and have been recognized as powerful schemes for learning. Many researchers in various fields such as expert system, artificial intelligence, etc., adopted neural nets to solve difficult problems. For examples, Kohonen layer is used for optimization and patterns matching. Associative memory, a recursive nets, is a good method for data storage. Backpropagation has a capable ability to deal with pattern recognition. Neural networks also are appropriate in control fields. In fact, backpropagation has been adopted successfully in control, especially in system identification [5].

But we all know that neural networks have three main demerits: (1) no promising of convergence, (2) complex computation, and (3) no on-line learning capability. For instances, Backpropagation needs a lot of prior data to construct the weights of synapses. No one can promise that the nets could reach the optimal goal. The nets also have no ability to do on-line learning and it needs a lot of time to calculate the updating weights. In control, on-line learning

identification ability is important because the complex nonlinearity of control plant could deduce a unpredictable behavior instantly. You have to detect it soon to make a suitable compensation such that the system would not break down.

For solving this problem, we suggest a improving model of neural networks to do on-line identification. It is derived from Group Method Data Handling (GMDH) [2][3][4]. Tenorio [1] named it: Self Organization Neural Networks (SONN). In their papers, the neural networks also used off-line learning. We used the same structure of SONN but adjust the learning algorithm to give it on-line learning ability and speed up the learning rate without losing its performance.

2. On Line Learning Algorithm of SONN

This method, GMDH, originally called the polynomial theory of complex systems, was first introduced in October, 1971. Astrom and Eykhoff pointed out that the identification problem can be transformed into the optimization problem, if the criterion of equivalence is defined in terms of the error function. A general form to represent systems, both linear and nonlinear, is the Kolmogorov-Gabor polynomial shown below:

$$y = a_0 + \sum_i a_i x_i + \sum_{ij} a_{ij} x_i x_j + \dots \quad (1)$$

where y is the output and x is the input of the system.

The other important theory about polynomials identification problem is Weierstrass Theorem [9]. Let $C([a,b])$ be the space of continuous real valued function defined on the interval $[a,b]$. Then any function in the interval can be approximated arbitrarily closely by a polynomial. We can use several kinds of polynomials to construct a complex model. GMDH and SONN both utilize the multiplication of quadratic form of polynomials to approximate a complex model. They need a lot of data to construct a linear matrix equation:

$$X^*A = Y \quad (2)$$

Where X is the input data and Y is output data. If the number of data is n . And A is composed of m unknown coefficients. Then A is an $m \times 1$ vector, X is an $n \times m$ matrix and Y is an $n \times 1$ vector. (Assume this is a MISO system.) Calculate the coefficients of A by pseudo inverse manipulation. We can prove that when pseudoinverse is adopted the error of real Y between Ax is the smallest.

SONN adopts the same structure as GMDH. They both have three main parts: (1) Node Generation, (2) Training, (3) Searching and Canceling. Node Generation is responsible for generating nodes by combination. Training procedure would produce the best coefficients of polynomials in one node. Searching part could delete the worse nodes and ascertains the structure of the net. SONN changes the fixed nodes of GMDH into floating nodes. GMDH's number of nodes is fixed. SONN uses a randomly procedure to produce nodes. GMDH uses error square sum as a measure function and SONN utilizes a function MDL (Minimal Description Length) to choose the best node.

$$MDL = -\log P(z|\hat{\theta}) + 0.5k \log N \quad (3)$$

Z means a sequences of observations which is characterized by probability density function $p_z(\theta)$. $\hat{\theta}$ is the maximum likelihood estimate of θ . k is the number of parameters used in the model, and N is the number of observations. This formula is difficult to calculate, and it will also increase the computing time. SONN suggests another formula to measure the performance of the node.

$$SEC = 0.5N \log S_N^2 + 0.5k \log N \quad (4)$$

(4) is the modification rule of (3) called SEC (Structure Estimation Coefficient)-MDL. GMDH would admonish the worse nodes but SONN don't abandon these worse nodes. It uses a simulated annealing process to decide whether the nodes reserved or not. SONN's quadratic forms have following 5 forms.

$$y = \left\{ \begin{array}{l} a_0 + a_1x_1 + a_2x_2 \\ a_0 + a_1x_1 + a_2x_2 + a_3x_1x_2 \\ a_0 + a_1x_1 + a_4x_1^2 \\ a_0 + a_2x_2 + a_5x_2^2 \\ a_0 + a_1x_1 + a_2x_2 + a_3x_1x_2 + a_4x_1^2 + a_5x_2^2 \end{array} \right\} \quad (6)$$

Because it also considers the number of parameters used. The fewer number of parameters used is the

better. According to the number of nodes and default temperature (in simulated annealing) , it will decide to grow up continuously or not. Except the above condition, SONN's and GMDH's structures are the same. In MRAC indirect adaptive theory, there exists an on-line identifier. Use the information acquainted from the identifier to design the error function and the control law. Originally, this identifier is a differential equation. The error function is dominated by the differential equation and will converge to zero if you give it better designing and estimating. But it has a fatal demerits : the speed of convergence is too slow. From the SONN, we find this neural networks can learn the behavior of the model more quickly. But we need an on-line identifier not an off-line sight-seeing. If the neural networks can identify complex systems without previous learning, using it in an adaptive controller is a good idea.

Since there are at most six parameters to be decided, it needs only six pair of input-output data to get them and generate a new node. And when the seventh pair of input-output data come in, we can use them to detect the node's performance. Use SEC-MDL to decide whether keep it or not. At the same time ,it can predict the eighth output and the new node also has to join the competition with the older nodes. Continue this process, we seemly will get the good approximation of an on-line ID.

After simulation, we find that in the first 50 steps this nets could track and predict the behavior of the model well. But when it exceeds 50 steps, the result get worse and worse. Maybe using only 6 data is not enough because the pairs of data could be dependent. We increase the number to eight to diminish the correspondence between the input-output pairs. And get the following algorithm:

Algorithm of 8-pairs data

At time (t+7)

- Use input-output pair acquired from (t) to (t+7) to generate the parameters of the new node
- Use the best node that have ever been to predict the next step output

At time (t+8)

- Calculate the new node's output at time(t+8)
- Compare the new node's and the best node's output with the actual output at (t+8)
- Adjust the nodes structure
- Repeat the above step until reaching the limitation

At the time when stop growing

- Count the times of the winning node
- Choosing the real winner with the most winning time as the output node

Using the algorithm, We get the following good simulating results.

3.Simulation Results

Examples 1:

$$u(t)=\sin(\pi t/180)$$

$$y(t)=[1.2\cos(0.3y(t-2))+1.7(0.8y(t-1))-2.4\sin(1.7u(t))+1]*0.2+u(t)$$

Fig. 1 Identification result and system output

Fig. 2 Identification Error

Example 2:

$$u(t)=\sin(\pi t/180)$$

$$y(t)=0.6y(t-2)+0.3y(t-1)+F(u(t))$$

$$F(x)=0.6\sin(\pi x)+0.3\sin(3\pi x)+0.1\sin(5\pi x)$$

Fig. 3 Identification result and system output

Fig. 4 Identification error

In fact, at the beginning, we use 6 data pairs not eight. But when the number of time step is over 40, the prediction gets worse and worse. That stands for the data we used are not enough to generate good SONN to predict the behaviors of the models. On the other hand, even the polynomials of higher order still can't approximate the system well because of lacking of data. For improving the identification performance, increasing the data of generating nodes is necessary. So we adopts the 8-pair data algorithm. From the simulation results, we can find that SONN has a much better performance than ever.

In order to prove the SONN has really learned the behavior of the system, we change the input of the two examples:

Example 1:

$$u(t)=1.02, \text{ if } \sin(\pi t/180) > 0$$

$$=-1.02, \text{ else}$$

fig. 5 Target and identification result

fig. 6 Error between target and ID result

Example 2:

$$u(t)=1.02, \text{ if } \cos(2\pi*t/200)>0 \\ = -1.02, \text{ else.}$$

fig. 7 Target and identification result

fig. 8 Error between target and ID result

From the above results, SONN have actually handled the behaviors of these two nonlinear system.

SONN is a nets seeming alive, and it will grow up increasingly if there exists no limit. So a up bound is needed. When the nodes' number reached the up bound, we must stop it to grow up. If number of nodes grows up to a large number, the memory needed must be large enough. If the number of the node is too small, it will have a worse performance because the order of polynomials is not high enough to approximate a nonlinear system. Some criterions to evaluate the number of the nodes of the system are necessary. Now, for simplifying the simulation, the number of nodes in our SONN is a constant.

Computation time is still a serious problem in the simulations. The time to compute the output of the plant takes about 10 minutes in 400 time steps with PC-486. Comparing it with the off-line SONN, and the backpropagation, it indeed becomes faster. But in the real time system, it is still too slow. In the algorithm, the cost of the time suffer from two parts : tracing back and matrix inversions. In fact, this problem can be treated as a least square problem. Pseudo inverse is not the unique solution. QR decomposition is also applicable.

4. Conclusion

On-line learning SONN did have a good capability to identify the complex nonlinear systems without a lot of priory data of the systems but SONN also suffer the same demerits of the neural networks. Time is a critical factor in modern control theory. We still need to approve the on-line learning rate such that we can use it as a real time controller.

Reference

- [1] "Self-Organizing Networks for Optimum Supervised Learning", M. F. Tenorio, W. T. Lee, *IEEE Tran. on Neural Networks*, pp. 100-110, Mar. 1990
- [2] "Polynomial Theory of Complex Systems", A. G. Ivakhnenko, *IEEE Tran. on SMC*, pp. 364-378, Oct. 1971
- [3] "Sequential GMDH Algorithm and Its Application to River Flow Prediction", S. Ikeda, M. Ochiai, Y. Sawaragi, *IEEE Tran. on SMC*, pp.473-479, Jul. 1976
- [4] "A learning Identification Algorithm and Its Application to Enviromental System", J. J. Duffy, M. A. Franklin, *IEEE Tran on SMC*, pp. 226-239, Mar. 1975
- [5] "Identification and Control of Dynamical Systems Using Neural Networks", K. Narendra, K. Parthasarathy, *IEEE Tran on Neural Networks*, pp. 4-27, Mar. 1990
- [6] "Maximum Entropy as a Special Case of the Minimum Description Length Criterion", M. Feder, *IEEE Tran on Information Theory*, pp.847-849, Nov. 1986
- [7] "Neural Networks for System Identification", S. R. Chu, R. Shoureshi, M. Tenorio, *IEEE Control Mag.*, pp.31-35, Apr. 1990
- [8] "Backpropagation Neural Networks for Nonlinear Self-tuning Adaptive Control", F. C. Chen, *IEEE Control Mag.* pp. 44-48, Apr. 1990
- [9] "The Stone-Weierstrass Theorem and Its Application to Neural Networks", N. E. Cotter, *IEEE Trans. on Neural Networks*, Vol 1, No. 4, DEC, 1990, pp.290-295