USING OF LEVENBERG-MARQUARDT METHOD IN IDENTIFICATION BY NEURAL NETWORKS

Ing. Jiří DOHNAL, Doctoral Degree Programme (2)
Dept. of Control and Instrumentation, FEEC, BUT
E-mail: dohnalj@feec.vutbr.cz

Supervised by: Prof. Petr Pivoňka

ABSTRACT

The last square method enables us relatively quickly find required parameters, however, we are limited by the choice of the suitable sampling period. We finds other methods of identification. One possibility is to use the neural network. This paper deals with the comparison of the most used learning algorithms neural networks, mainly the Levenberg-Marquardt algorithm is discussed.

1 INTRODUCTION

Identification of dynamic systems is essential for adaptive control. The definition of the adaptive system can be formulated as follows: Adaptive control systems adjust the parameters or configuration of one part of the system (controller) to the changes of the parameters or configuration of another part of the system (controlled system) so that an optimal behaviour of the whole system is ensured on the basis a of the chosen criterion. Obviously, we have to obtain information on the dynamic behaviour of the whole system - to identify it. One approach is the monitoring of the system characteristics, refinement and thus eliminating potential changes [1].

A frequently used identification method is the least-squares method. Its advantage is fast convergence of the model parameters, and storage of previous input values u, and system outputs y. The main negative feature of the methods is the computation of unreal hypothetical estimates of parameters at a short sampling period T_0. A short sampling period, however, is desirable for control. At a shorter sampling period the introduced defect is more easily controlled [2]. A shorter sampling period is therefore desirable, and other identification techniques must be sought. The neural network seems to be a desirable solution because of its adaptation characteristics.

2 NEURAL APPROACH

A formal neuron has n real inputs $x_{1,}...,x_n$. The inputs are evaluated using corresponding real synaptic weights $w_{1,}...,w_n$ defining their „throughput“. A neuron

transforms input data into output data based on the transfer function. Individual neurons can be arranged to form a neural network – the neurons are interconnected so that a neuron output is an input to multiple neurons. The number of neurons and their interconnections in the network determine the neural network architecture. The so-called feedforward networks are used in control technology to implement controllers or, for example, to identify process parameters. In a linear model of a process it seems beneficial to use only a single neuron with a linear transfer function for identification.

The advantageous and distinguishing feature of neural networks is their ability to learn. The network in the adaptive mode abstracts and generalizes the function character in the process of learning from training patterns. The learning algorithm is an optimization method capable of finding weight coefficients and thresholds for a given neural network and a training set. There are a number of learning algorithms. Those that are used most frequently are the backpropagation (BP) algorithm and the Levenberg - Marquardt (LM) algorithm.

Batch Training

The preparation of an appropriate training set is one of the major factors affecting the final training of the network. The data contained in it must sufficiently cover the problem area. A discreet description of the dynamic system can be obtained using the differential equation

$$y(k) = b_1 u(k-1) + \ldots + b_{nb} u(k-nb) - a_1 y(k-1) - \ldots - a_{na} y(k-na) \qquad (1)$$

where $y(k)$ is the output parameter value in the k-th sampling moment.
Only the previously obtained values are used (ARX model). The identification should result is obtaining the same response of the identified configuration y and model y_m to the initiation signal u. The method of preparing the training set is apparent from equation (1). In step k the vector of previous models $X = [u(k\text{-}1), \ldots u(k\text{-}nb), \text{-y}(k\text{-}1), \ldots \text{-y}(k\text{-}na)]$ is submitted to the network input, and trained to the current configuration response $d = y(k)$. If the network is to be used for on-line identification (control), it must be trained to more than one training pattern X, d. The batch-training principle is to create batch of p elements. For system order $n = 2$ structure of batch is

$$X = \left[X_1, \ldots, X_p\right] = \begin{bmatrix} u(n+1) & \cdots & u(p+n) \\ u(n) & & u(p+n-1) \\ -y(n+1) & & -y(p+n) \\ -y(n) & \cdots & -y(p+n-1) \end{bmatrix}$$

$$d = \left[d_1, d_2, \ldots, d_p\right] = \left[y(n+2), y(n+3), \ldots, y(p+n+1)\right] \qquad (2)$$

In time $t = 0$ s data collection starts, $p+n+1$ steps take place (on the condition $nb = na = n$). When the batch is full, the network is trained to in this manner prepared training set. The older pattern (X_1, d_1) is removed in the next sample point and new pattern (X_{p+1}, d_{p+1}) is added to set.

2.1 BACKPROPAGATION ALGORITHM (BP)

This algorithm is based on minimizing the error of the neural network output compared to the required output. The required function is specified by the training set (a sequence of input / required network output pairs). The error of network E relative to the training set is

defined as the sum of the partial errors of network E_k relative to the individual training patterns and depends on network configuration w

$$E = \sum_{k=1}^{p} E_k = \frac{1}{2} \sum_{k=1}^{p} \sum_{j \in Y} \left(y_j - d_{kj} \right)^2 \qquad (3)$$

where p - number of available patterns, E_k - partial network error, Y - set of output neurons.
The new configuration in time $t > 0$ is calculated as follows

$$w_{ji}(k) = w_{ji}(k-1) - \alpha \frac{\partial E}{\partial w_{ji}} + \beta \left[w_{ji}(k-1) - w_{ji}(k-2) \right] \qquad (4)$$

where $0 < \alpha < 1$ is the speed of learning, β is the momentum [3]

The speed of training is dependent on the set constant α. If a low value is set, the network weights react very slowly. On the contrary, high values cause divergence - the algorithm fails**.** Therefore the parameter α is set experimentally. If the neural network is to be used as a model in an adaptive system, for real industrial process control, the divergence must be prevented. In order to avoid it, the algorithm is often modified, the parameter α can be adjusted in the progress of training in dependence on the network error E. The neural network is submitted the training set patterns. The instantaneous error $E(w(k))$, $\partial E(w(k))/\partial w(k)$, is determined, and a new weight configuration $w(k+1)$, then $E(w(k+1))$ are calculated. Now, we have to find out if the network training error was reduced. If

$$E(w(k+1)) < E(w(k)) \qquad (5)$$

is fulfilled, the new configuration of network weights is accepted, the value of parameter α is increased. Otherwise constant α is decreased and configuration $w(k+1)$ is recalculated.

2.2 LEVENBERG-MARQUARDT ALGORITHM (LM)

This algorithm is a variant of the Gauss-Newton optimization method. The new configuration of weights in step $k+1$ is calculated as follows

$$w(k+1) = w(k) - (J^T J + \lambda I)^{-1} J^T \varepsilon(k) \qquad (6)$$

The Jakobi's matrix for single neuron can be written as follows:

$$J = \begin{bmatrix} \frac{\partial \varepsilon_1}{\partial w_1} & \cdots & \frac{\partial \varepsilon_1}{\partial w_n} & \frac{\partial \varepsilon_1}{\partial w_0} \\ \vdots & & \vdots & \vdots \\ \frac{\partial \varepsilon_p}{\partial w_1} & \cdots & \frac{\partial \varepsilon_p}{\partial w_n} & \frac{\partial \varepsilon_p}{\partial w_0} \end{bmatrix} = \begin{bmatrix} x_{1_1} & \cdots & x_{n_1} & 1 \\ \vdots & & \vdots & \vdots \\ x_{1_p} & \cdots & x_{n_p} & 1 \end{bmatrix} \qquad (7)$$

where is w - vector of the weights, w_0 – bias of neuron, ε - error vector (the difference between the actual and the required value of the network output for the individual pattern).

Parameter λ is modified based on the development of error function E. Should the step cause a reduction of E, we accept it. Otherwise we change parameter λ, reset the original value and recalculate $w(k+1)$ [4].

3 SINGLE NEURON

In a linear system identification, an single neuron with linear transfer function can be used. Simulation experiments were performed for a number of transfer functions.

An example can be identification of system (form in plane Z for $T_0 = 0.1$ s)

$$G(z) = \frac{0.0003857z^{-1} + 0.000372z^{-2}}{1 - 1.8949z^{-1} + 0.8958z^{-2}} = \frac{B(z)}{A(z)} \qquad (8)$$

The system is initiated by signal u_1 (rectangular pulses – amplitude 2,-1; period 5 s), The output of the process is measured in the sampling period 0.1 s. The batch contains 40 patterns. In training by means of the BP method modification using adaptive training constants α, β was performed. Initial values were: $\alpha = 0.01$, $\beta = 0$, $\lambda = 0.001$.

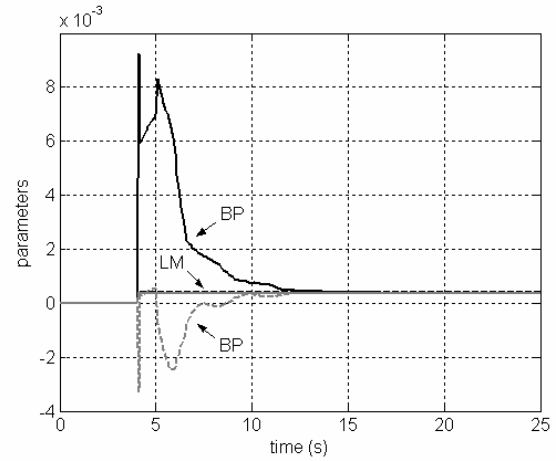

Fig. 1: *Curve of parameters B(z)*

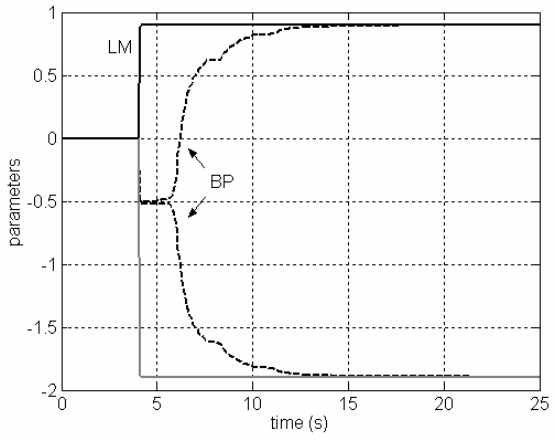

Fig. 2: *Curve of parameters A(z)*

The LM algorithm places a quadratic function around the point and proceeds towards its minimum. The sought-for optimum w^* can be established with great accuracy using fewer steps, although at the expense of increased calculation complexity. In one step a much greater number of calculation operations need to be made compared to a single step in BP. Because of its speed and accuracy, only algorithm LM was used in further experiments.

4 PHYSICAL MODEL IDENTIFICATION

For verification of algorithms on physical models communication between the MATLAB/Simulink environment and the programmable logic controller B&R (series 2005) was applied (using the PVI interface). PLC acts as an input/output card only. In synchronous instants the PVI client can read or record into internal variables of PLC mapped at its inputs and outputs [5].

When working with real systems we can measure data with only limited accuracy. It is due to the signal transition in A/D, D/A converters with the step function of parameters. The calculation accuracy is therefore not in the order of 10 or more valid digits, but a maximum of 4 valid digits [6]. Consequently, the identified parameters are not stabilized, but oscillate, which is undesirable. One of the possible approaches is stopping the identification in the instant when the prediction model error e drops below the set limit.

Fig.3 shows the curve of the physical model response (approximate value of time constants $T_1 = 10$ s, $T_2 = 1$ s) and the neural model to the initiation signal selected as in the previous case.

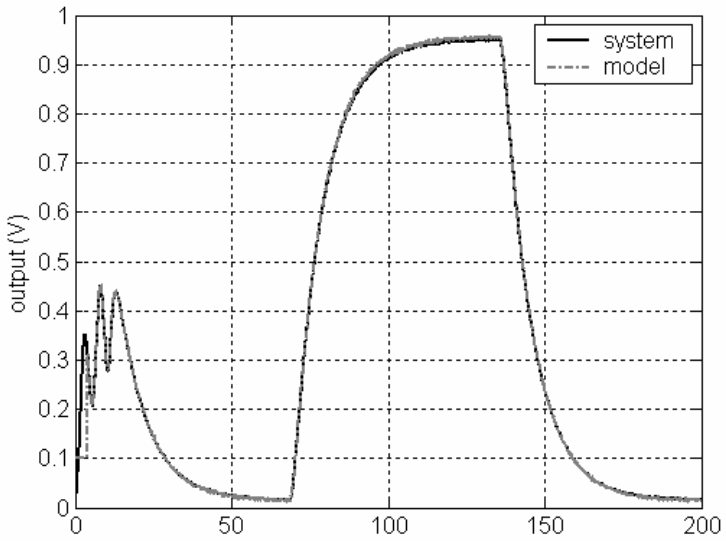

Fig. 3: *Physical model identification (by LM)*

It was proven by experiments that an single neuron can be used even in noise-affected systems.

5 CONCLUSION

The neural network can be used for progress identification of dynamic systems even if a reduced sampling period is applied. Better results were obtained using the LM algorithm although it was more demanding in terms of calculation complexity within a single step. It was proved that when an single neuron is used an industrial PLC can identify a noise-affected linear model with sufficient accuracy.

ACKNOWLEDGEMENT

The paper has been prepared with the support of the research plan CEZ: MSM 260000013.

REFERENCES

[1] Bobál, V., Böhm, J., Prokop, R., Fessl, J.: Practical Aspects of Self-Tuning Controllers, Algorithms and Implementation. Brno, VUTIUM 1999, in Czech

[2] Švancara, K., Pivoňka, P.: Closed Loop On-Line Identification Based on Neural Networks in Adaptive Optimal Controller. In Proc. of Zittau Fuzzy Colloquium, Zittau, 2002, pp. 218-223

[3] Šíma, J., Neruda, R.: Theoretical issue of neural networks. MATFYZPRESS 1996, in Czech

[4] Hristev, R.M.: The ANN Book, GNU Public Licence, 1998

[5] Veleba, V.: Neural adaptive controller. Diploma Thesis, UAMT, VUT 2003, in Czech

[6] Pivoňka, P: Digital control engineering, UAMT, textbook VUT Brno 2002, in Czech