JELENKA B. SAVKOVIĆ–STEVANOVIĆ

Department of Chemical Engineering, Faculty of Technology and Metallurgy, Belgrade

SCIENTIFIC PAPER

159.953.4+681.5.017:510.644

AUTOMATED LEARNING AND CONTROL USING SPEEDUP AND NEURAL NETWORKS

This paper studies complex dynamic neural network learning models. Backpropagation was used to train a neural network for dynamic simulation and control of a chemical stirred tank. The generalized delta rule algorithm was used to train the network minimizing the sum of squares of the residual. It was assumed that a historical database of plant inputs and outputs is available. A Pseudo Random Binary Sequence–PRBS was used as a disturbance. For training the database the 1% PRBS signal was superimposed upon its steady state value from SPEEDUP simulation. Once a trained neural network model was available, it then used in real time learning and pH control. The examined inverse and standard neural network models controllers achieved better performance than a conventional PI controller. Complex Internal Model Control – IMC achieved the best results in control and local stability. The obtained models in this paper improve noisy handling and reduce process variability. Some of these systems can be used for self – maintaining non–linear, multivariable models and day to day troubleshooting.

Neural networks have been successfully used to identity dynamical systems such as deterministic chaos. Artificial neural networks are composed of many simple computation elements, nodes, locally interacting across very low bandwidth channels or connections. Each neurod performs a non–linear transformation on its inputs. The architecture of these models is specified by the node characteristics, network topology and learning algorithm. Often the neurodes are organized into distinct layers, input (perception), hidden or associative (conceptual), and output (action) layers. When the layers are connected in a feedforward way, so that signals are passed from the input to hidden to output layers without feedback or communication within a layer, the net is called a multilayered perception. Various architectures of neural networks have been suggested in the literature [1–29]. Neural networks in all categories address the need for rapid computation robustness and adaptability. Neural networks in all categories address the need for rapid computation robustness and adaptability. Neural models require fewer assumptions and less precise information about the modeled system than some more traditional techniques. The design of a neural system requires advance specification of relatively few fixed parameters. Many network variables can be set or reset by the data being processed, thus adapting to the specific needs of the current system state. Neural networks have been used in many engineering problems [2, 7, 17–21, 24–29]. Neural networks are useful as modules in larger systems interacting particularly well with fuzzy logic and expert systems [4–6, 22–24]. The careful combination of techniques produces the most useful robust systems. Incorporation of neural network modules is an option that should be considered by simultanists in all application fields.

Since non–linear governing relationships can be handled by artificial neural nets, the nets may offer a cost effective approach to dynamic behavior simulation and control of non–linear process systems. Although detailed quantitative models are presumably not available, there is a lot of qualitative information about the structure of the process system and the potential relationship among the inputs and outputs, in constructing neural networks with an internal structure. Neural networks can serve as simulators trained from the observed behavior of the process. In this paper neural network architectures on three distinct layers (input, hidden and output) are connected by feedforward weighted connections. The function of the input layer is to distribute the values of the input parameters into the net. Neural networks can be trained to recognize patterns in process data [18]. Using plant historical data, they can be taught to predict how a process will respond based on past response, and thereby create a process model [25]. In this paper the process model performs automated advanced learning to improve self maintenance, dynamic simulation and control.

This paper shows the advantages neural networks controllers have over conventional PI controllers from SPEEDUP. The neural network complex control structures achieved the best performance in control and local stability.

1. TRAINING A FEEDFORWARD NEURAL NETWORK

Although network systems really support automated learning, the link between learning and parsimony is the development of a structure for characterizing an active goal hierarchy that represents a logical and coherent cognitive theory within the some model [26]. When a learning process is expressed in hierarchical form, the relationship that exist between goals and subgoals provides a basis for relating the

Author address: J. Savković-Stevanović, Faculty of Technology and Metallurgy, University of Belgrade, Karnegijeva 4, 11000 Belgrade, Yugoslavia, e–mail: savkovic@elab.tmf.bg.ac.yu
Paper received: April 24, 2000
Paper accepted: May 20, 2000

overall goal based system performance to specific assumptions about the viability and contribution of the supporting subgoals. In this form the system is a full expression of some control theory in that the system relationship with the environment, as expressed in a set of feasible state conditions, can be related either to overall system performance measures or to the relationships and the subgoals that support them. The automated closed loop learning model is shown in Fig. 1.

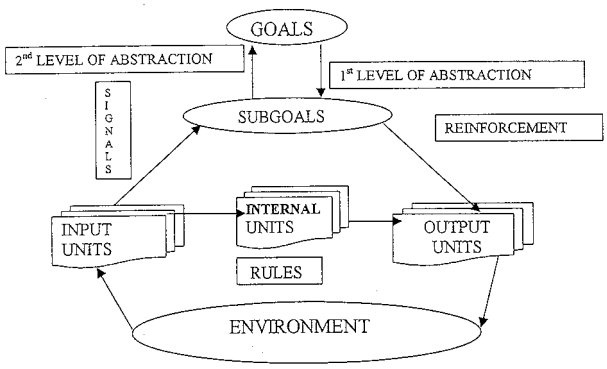

Figure 1. Automated closed loop learning model

The backpropagation algorithm adjusts the weights in a feedforward neural network consisting of several layers and a output layer. The goal is to teach the network to associate specific output states, called target states, to each of several input states. Having learned the fundamental relationships between inputs and outputs, the neural network can produce the correct output for a new previously unseen input. The backpropagation learning algorithm does not need too much computation time to obtain the correct weights if the training data is of small size. In other words, if the relationship between inputs and outputs is simple, a set of weights is easy to obtain. Most often for complex problems the relationship between inputs and outputs cannot be completely represented with a small amount of training data sets. Therefore, the computation time required for backpropagation is very large.

A popular configuration of neural networks for backpropagation is a totally feedforward net [12, 19, 28]. In feedforward nets inputs feed up through hidden layers to an output layer. Each neuron forms a weighted sum, the inputs from previous layers to which it is connected adds a threshold value and produces a non–linear function of this sum as its output value. The output value serves as the input to the next layer to which to the neuron is connected, and the process is repeated until output values are obtained for the neurons in the output layer.

Thus, each neuron performs:

$$y(p,j) = f\,[\sum_i w(i,j)\, x(p,j) - \theta(j)] \qquad (1)$$

where w(i,j) is the weight from neuron i to neuron j, w(i,j) can be a positive or negative real number and θ(j) is the threshold of the jth neuron, p means input–output vector pairs of the pth pattern, The f(x) is a non–linear activation function, often chosen to be of a sigmoidal form:

$$f(x) = 0.5\,[1+\tanh(x)] \qquad (2)$$

is used in this analysis where tanh is the hyperbolic tangent. If d(p,j) are the desired outputs and y(p,j) are the outputs obtained from the output layer for the pth pattern, neural networks are trained by minimizing the error function:

$$E = \min_w J = \sum_{p=1} \sum_{i=1} [d(p,i) - y(p,i)]^2$$

where i indexes the number of neurons in the output layer, and p means the pth input pattern of the training set is presented on the input layer.

Minimizing the sum of squares of errors is not always the best way of training a neural net, but for some applications it suffices [3, 20]. The backpropagation algorithm by the generalized delta rule – GDR as a kind of gradient descent method was used [13, 17–20].

The gradient descent method was described by the following equations (4–9), the commonly used steepest descents procedure in minimizing E is to change w(i,j) and θ(j) by Δ∂w(i,j) and Δθ(j), where:

$$\Delta w(i,j) = \frac{\partial E}{\partial w(i,j)}\,\eta \qquad (4)$$

$$\Delta\theta(j) = \frac{\partial E}{\partial \theta(j)} \qquad (5)$$

where η is the learning rate.

Such an approach requires using all of the input–output pairs to calculate the gradient. Backpropagation also uses gradient information to change the weights. However, the gradient is calculated only with respect to one input–output pair at a time. Initially the weights in the net are randomized.

After simplification Δw(i,j) and Δθ(j) can be expressed as:

$$\Delta w(i,j) = \eta \sum_p \delta(p,j)\, y(p,i) + a\Delta w(p-1,i,j) \qquad (6)$$

$$\Delta\theta(j) = \eta \sum_p \delta(p,j) \qquad (7)$$

where

$$\delta(p,j) = [d(p,j) - y(p,j)]\, y(p,j)\, [1-y(p,j)] \qquad (8)$$

if the jth neuron is in the output layer, and

$$\delta(p,j) = y(p,j)\, [1-y(p,j)] \sum_k \delta(p,k)\, w(j,k) \qquad (9)$$

if the jth neuron is the hidden layer and k are the overall neurons in the layer above neuron j.

The algorithm based on the GDR – method was developed in previous paper [19].

2. NEURAL NETWORK CONTROLLER DESIGN

This work considers an important application in process engineering control of a non–linear dynamic process for controller design [16, 17]. Control is an essential part of any plant for most chemical processes where the mathematical models of the process are not accurate, and apply only to small operating domain. The conventional methods of control system design are not so reliable. The mathematical model of the process is not assumed to be known.

The feasibility of using neural nets for learning a non–linear dynamic model from plant input–output data, as well as control of a process with highly non–linear characteristics was investigated. The neural network directly computes the control variable in order to make the plant output follow the desired set point. Given this goal the learning objective consists in modeling the functionalities between the input and output of the plant. These input can be randomly generated, but they must preferably cover all the input domains.

The learning phase must allow the network to model the inverse dynamics of the plant and the computational function of the neural networks in the learning phase is:

$$u(k) = f\,[y(k+1), y(k), y(k-n+1); u(k-n+1)] \qquad (10)$$

After successful learning the neural network can be integrated in a feedback control loop (Fig. 2). The input unit which coded the future state of the plant $[y(k+1),..., y(k+i)]$ using the learning phase must know how to represent the future desired set–point $[s(k+1),.., s(k+i)]$ and the computational function of the neural controller in the control phase becomes:

$$u(k) = f\,[s(k+1), y(k), y(k-n+1); u(k-n+1)] \qquad (11)$$

The structure of direct inverse control is shown in Fig. 2.

The forward dynamic model:

$$y(k+1) = f\,[y(k),..., y(k-n+1), u(k),... u(k-n+1)] \qquad (12)$$

where u(k) and y(k) are input and output vectors at the time instance and f is an unknown function. A scheme of standard control is shown in Fig. 3. The minimization control function is given as:

$$\min J = \sum_{i=1} \left\{ [ynn(k+1) - y(k+1)]^2 + [u(k) - u(k-1)]^2 \right\} \qquad (13)$$

The input of the network were the values of the state variables and the manipulative variable. The

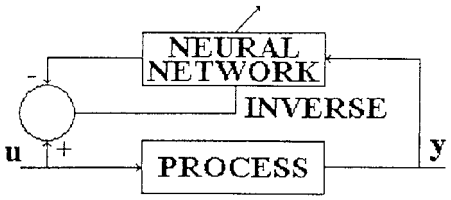

Figure 2. Direct inverse control

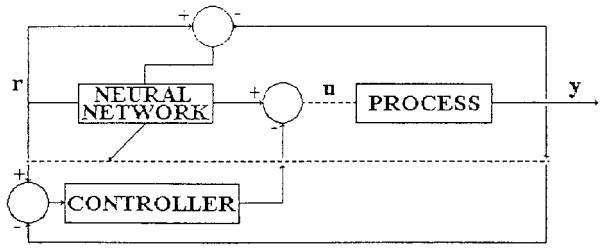

Figure 3. Standard control

desired output (target) were the values of the system state variables at the end of the sampling time and the control variable. Alkali flow was the manipulated variable and pH was the control variable.

4. CASE STUDY

The dynamic response of pH in a Chemical Stirred Tank – CST as a case study. The pH control loop is shown schematically in Fig. 4. The CST has two input streams, one containing alkali F(2) and the other acid,

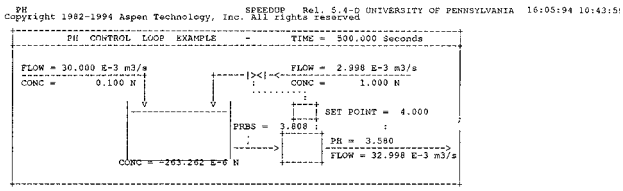

Figure 4. pH control loop of a CST

F(1). A dynamic model for the pH in the tank can be obtained using the approach presented in SPEEDUP and the CST model unit was used from the SPEEDUP Library Casebook. By writing material balances on the alkali and total acid and assuming that the acid base equilibrium and electroneutrality relationships hold, one obtains:

total acid balance

$$F(1)c(1) - [F(1) + F(2)]\,c = V\frac{dc}{dt} \qquad (14)$$

alkali ion balance

$$F(2)\,c(2) - [F(1) + F(2)]\,c = V\frac{dc}{dt} \qquad (15)$$

The parameters for the CST considered are given in Table 1. After 20 s the acid flow is increased by 30.00 10^{-4} m^3/s, as a simple step unit disturbance, and the pH response with a PI controller from SPEEDUP (reset rate = 210 s, time delay = 12s, and gain=0.6) is shown in Fig 5.

HAC equalibrium

$$\frac{[AC^-]\,[H^+]}{[HAC]} = K_a \qquad (16)$$

water equalibrium:

$$[H^+]\,[OH^-] = K_w \qquad (17)$$

Table 1. CSTR parameters

Parameters used in the simulation	Value
Volume of the tank V	5 m^3
Flow rate of acid F(1)	30.00 x 10^{-3} m^3/s
Steady state flow rate of alkali F(2)	30.00 x 10^{-4} m^3/s
Steady state pH	4.0
Concentration of acid c(1)	0.100 N
Concentration of alkali c(2)	1.000 N
Initial flow rate of acid F(1)	25.00 x 10^{-3} m^3/s
Initial flow rate of alkali F(2)	2.49 x 10^{-4} m^3/s

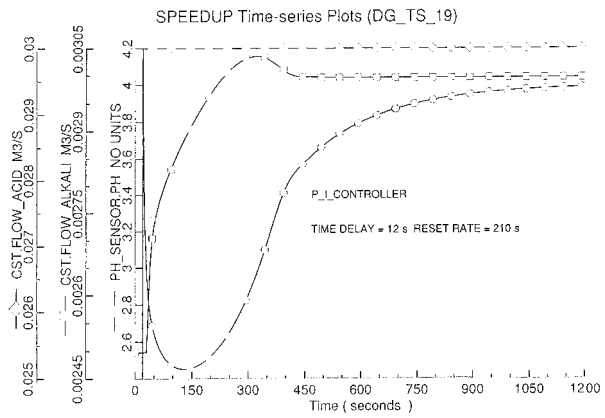

Figure 5. pH control loop response without random disturbance with a PI controller (reset rate = 210 s, time delay 12 s, gain 0.6)

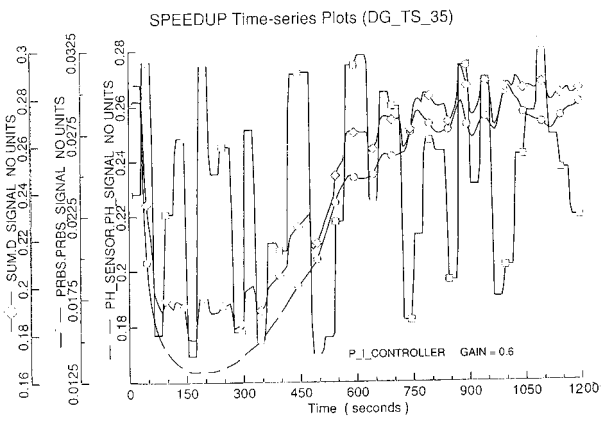

Figure 6. pH response used for training for PRBS random disturbance with varying amplitude (select = 1.0 and seed = 10000) on the alkali flow with a PI controller

The training data base is a 1% PRBS random disturbance with varying amplitude superimposed upon its steady state value. The steady state value is pH = 4.0 and the sample time 30 s. Fig. 6 shows the pH response used for training. In the pH model equations (15) and (16) are highly non–linear algebraic equations. Thus, one can view the model as having highly non–linear

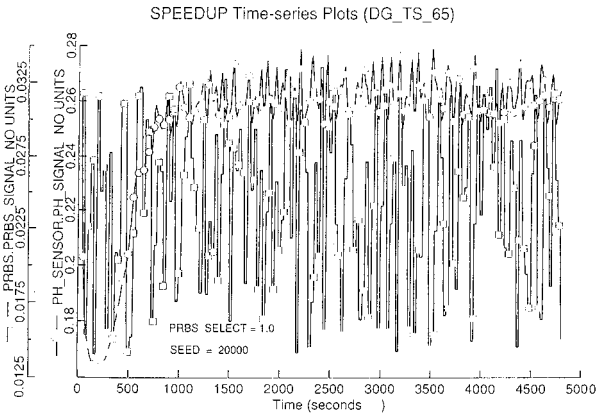

Figure 7. pH response used for training for PRBS random disturbances with varying amplitude (select = 1.0 and seed = 20000) on the alkali flow with a PI controller

steady state characteristics. At the pH set point of 4.0 a 1% PRBS random disturbance signal with varying amplitude (select = 1.0 and seed = 10000) is added on F(2) alkali flow, and the pH response is shown in Fig. 6.

Fig. 7 shows the pH response on a 1% PRBS random disturbance with varying amplitude (select=1.0, seed = 20000) added on the alkali flow, with a PI controller (reset rate = 210 s time delay = 12 s, and gain = 0.6).

4. NEURAL NETWORK DYNAMIC MODEL LEARNING

Various sizes were tried for the input layer. Different numbers of nodes in the hidden layer were simulated with improvements and verification.

The input to the network consisted of a moving window of pH and F(2) values. The creation of a window is taken as the current time t_o. Past and present values of the pH and F(2), as well as future values of F(2) are fed to the net. The output from the net is the pH in the future. It should be emphasized here that since the used database was historical, the future values (relative to the center of the window) are known and they can be used for training. A time step of 0.4 min was used. At the beginning of the training process the window is placed at the beginning of the data base. The first five pH and ten F2 values are an input to the network, NN (15x20x5) (Fig. 8).

An alternative approach to using neural networks for control involves determining the inverse process model as shown in Fig. 9. The combination of different control structures in an internal model control is shown in Fig. 10.

The desired network output are the pH values at t = – to + 1,2,3,4 and 5 Δt's in the future. After the first presentation of data, the window moves Δt down the database. Again the current and past four pH and future five F(2) values are input to the net. This process is continued until the end of the database. The database repeats the process by starting at the beginning of the

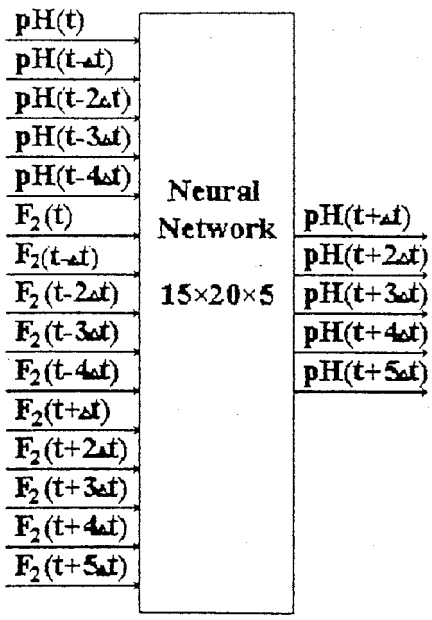

Figure 8. Considered neural network input and output signals

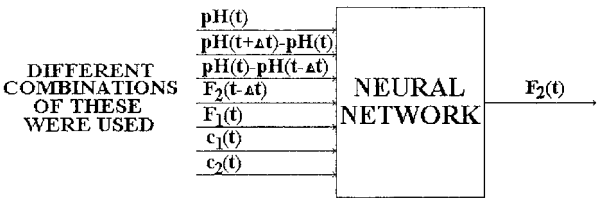

Figure 9. Considered input output signals of the inverse neural network model

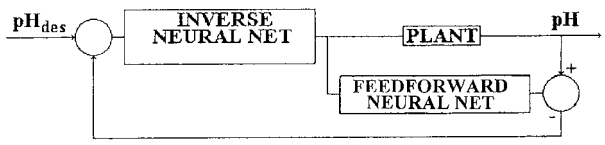

Figure 10. Neural networks IMC complex structure

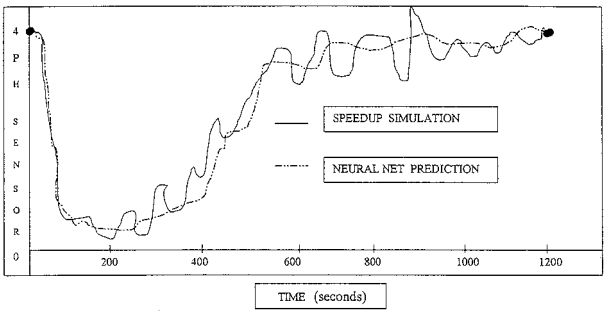

Figure 11. Neural networks pH response prediction with inverse model learning for PRBS random disturbances with varying amplitude (select = 1.0 and seed = 10000) on the alkali flow

database. During training the learning rate factor η was lowered from 0.3 to 0.15. The momentum factor α is 0.5. After many cycles through the database, the net converges as determined by the fact that the weights achieve fixed values.

Since the network weights are initially randomized, the net does a very poor job in its prediction at first. After convergence the net gives god predictions. Although the network predicts pH at one to five time steps into the future, the agreement between the predicted and actual pH (Fig. 11) at the other four time steps is as good as the one time step.

As can be seen the neural network model is close to the actual pH. Also, corrupting the pH outputs with noise presents no problem for the network. Fig. 11 shows the neural network prediction and SPEEDUP pH response on a PRBS disturbance with varying amplitude (Fig. 6).

5. NEURAL NETWORK CONTROL

This section discusses the results of neural network controller design for process control. Once a trained backpropagation model is available, then it can be used in a straight forward approach for process control. Figs. 2 and 3 present a schematic diagram showing how neural network models are used.

The manipulated variables in this paper are F(2)(m), n=k to k+m, after M time steps the F(2)'s are held constant. Typically, several future F(2)'s would be calculated, but only the first F(2)(k), would be implemented. Constraints on the size of F(2) move and high and low values for F(2) and pH can be incorporated into the problem. An alternative approach to using neural networks for control involves high and low values for F(2) and pH can be incorporated into the problem. One could switch the future pH's (outputs) with the future F(2)'s (inputs) and develop an inverse plant model referred to in Fig. 9. Such a net would predict what rates are necessary to achieve a desired pH. This approach to modeling was tried on the data in Fig. 7 using 10 inputs, 15 hidden and output neurons. As can be seen, the neural network dynamic model does a good job of learning the inverse process model. As expected, the networks input with F2(t–Δt) and pH(t) – pH(t–Δt) in addition to pH(t+Δt) – pH(t) and one state variables showed good control performance. The errors in the controlled variable were dependent on the size for different alternatives of the input state variables, in this case oscillations occurred in the control signal.

Once the inverse model is available, then it can be used in an IMC structure, as shown in Fig. 10. One last point should be made about using neural net dynamic models for control. The nature of backpropagation modeling is such that it lends itself naturally to an adaptive implementation. Many different schemes are possible. For example, one could have two networks

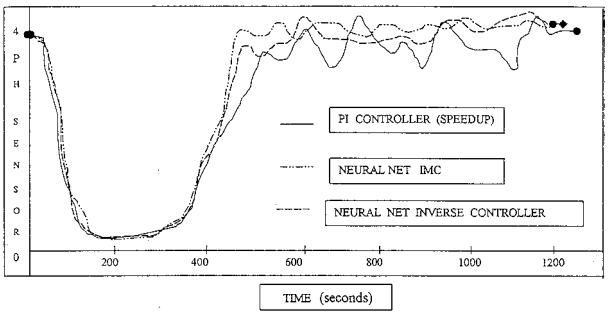

Figure 12. pH responses with neural networks internal model controller, inverse model controller and PI controller for PRBS random disturbance with varying amplitude (select = 1.0 and seed = 10000) on the alkali flow

and use one for control and one for learning. Both networks would start out the same and they would be converged on a historical database. The net used for control would be frozen. The other net would continue to learn as new data became available.

Such learning could only take place when there was sufficient information in the plant inputs and outputs. Other combinations are also possible. Fig. 10 shows the IMC strategy using a (10x10x5) inverse neural model and (15x10x1) feedforward neural network model. The obtained results show the best control performance (Fig 12) of this complex structure.

Fig. 12 shows the dynamic response of a pH controlled CST for a PRBS random disturbance with varying amplitude (Fig. 6) with a conventional PI controller, neural network inverse controller and neural networks internal model controller. The best control is achieved by the complex structure of a neural network internal model controller. The illustrated results in Fig. 12 show the advantages of neural network controllers over a PI controller.

6. CONCLUSION

This work demonstrates the advantages of neural network controllers over PI controller from SPEEDUP in local control and stability. Neural network internal control improved process control in comparison with neural network inverse and forward examined models.

The backpropagation dynamic modeling approach was illustrated on a pH CST that was forced with a 1% PRBS input. The neural network controllers performed well for the process studied. Feedforward neural networks have been shown to be capable of controlling a state variable in pH control. The non–linearities in the process require non–linear properties in the controller realized by hidden nodes with non–linear activation functions in the network. The non–stationary characteristics of the process are handled by feeding information of the state variables, and not only the control error to the network. These are the major advantages of the neural controller compared to the well established control algorithm of a conventional PI controller. The target for process control is normally to achieve minimal error in the controlled variable, however, the network is trained to produce a control signal as near to the required one as possible, which may not lead to a minimum in the control error.

Compared to control based on a mathematical model of the process, the network has certain advantages. In a neural controller, the structure of the model is formed from training data and little a priori knowledge is required. This can save time–consuming modeling of the process. On the other hand, enough representative data to learn the process behavior is required and available knowledge of the process behavior can not be included into the network directly. The combination of a plant inverse neural controller and a standard neural controller compensate for stationary control errors because the IMC structure shows the best results.

This paper contributes the requirement of real time learning in neural modeling and building more robust and accurate process control models. A neural network controller goes on line to deliver the bottom lines benefits of neural network modeling. The advantages of neural modeling are easier, faster, less expensive model building, time control loops, the diagnosis of difficult operating conditions, the search for new operating points without interrupting production. These systems reduce costs, variability and pollution.

Acknowledgment

The author wishes to express her gratitude to the University of Pennsylvania, Department of Chemical Engineering, USA and the Scientific Research Funds of Serbia and Yugoslavia.

NOTATION

c(1) – concentration of acid
c(2) – concentration of alkali
d – desired output
E – minimizing error function
F(1) – acid flow
F(2) – alkali flow
p – number of patterns
pH – desired pH
J – minimization control function
x – input
y – calculated output
ynn – neural net calculated learning goal
V – volume of tank
u – manipulated variable
w – weight on each branch

Greek symbols

α – momentum term
δ – factor defined in eqs.(8) and (9)
η – learning rate
θ – threshold

REFERENCES

[1] M. Bavarian, IEEE Control System Magazine, April (1988), 152.

[2] N. Bhat and T.J. Mc Avoy, Computers chem. Engng. **14** (1990) 573.

[3] A.B. Bulsari and H. Saxen, Computer–Oriented Process Engineering, Process Technology, **10** (1991) 23.

[4] A.B. Bulsari and H. Saxen, Fuzzy logic inferencing using a specially designed neural network architecture, Proc. of the Int. Symp. On Artificial Intelligence Applications and Neural Network, Zurich, Switzerland (1991) 57–60.

[5] A.B. Bulsari and A. Kraslawski, Application of artificial neural networks for fuzzy simulation of a chemical reactor, Proc. of the 35th SIMS Simulation Conference, Kongesberg, Norway, June 1993, 115–124.

[6] C.J. Harris, M. Brown, K.M. Bossley, D.J. Mills, F. Ming, Engng Applic. Artif. Intell., **9** (1996) 1.

[7] A.B. Bulsari and S. Palosaari, Simulating the dynamics of an adsorption column with limited measurements of state using artificial neural network, Proc. of the 35th SIMS Simulation Conference, Kongsberg, Norway, June 1993 97–106.

[8] R. Dale, D.F. Seborg, T.F. Edgar, D.A. Mellichamp, Process Dynamic and Control, John Wiley & Sons Inc. (1989)

[9] R.D. DeVeaux, D.C. Psichogios, L.H. Ungar, Computers chem. Engng. **17** (1993) 819.

[10] J.L. Dirion, M. Cabasud, M.V. LeLann and Gassamata, Computers Chem. Engng., **14** (1990) 573.

[11] J.C. Hoskins and D.M. Himmelblau, Computers Chem. Engng., **12** (1998) 881.

[12] W.J.M. Epping and G. Nitters, A neural network for analysis and improvement of gas well production, Proc. of the SCS '90 – Summer Computer Simulation Conf., Calgary, Canada, July 16–18 (1990) 329–334.

[13] W.P. Jones and J. Hoskins, Backpropagation. A generalized delta learning rule, Byte, Oct., (1987) 155.

[14] L.C. McCullough L.C., Simulation, **58** (1992) 327–332.

[15] M.L. Padgett and T.A. Roppel, Simulation, **58** (1992) 295–305.

[16] D.C. Psihogios and L.H. Unger, Ind. Engng. Chem. Res., **30** (1991) 2564–2573.

[17] J. Savković–Stevanović, Computers Chem. Engng., **20** (1996) 925–930.

[18] J. Savković–Stevanović, Neural networks in dynamic simulation and control of chemical systems, EANN'95 – Inter. Conf. on Engng. Applic. of Neural Networks, Aug. 21–25, Helsinki (1995) 618–625.

[19] J. Savković–Stevanović, Computers Chem. Engng., **18** (1994) 1149–1155.

[20] J. Savković–Stevanović, J. Computers Chem. Engng., **17** (1993) 411–416.

[21] J. Savković–Stevanović, M. Vico–Stevanović, S. Jorgačević, N. Tica, An artificial neural network for variable estimation and state identification in process fermentation, Proc. of the SCS'93 – Summer Computer Simulation Conf., Boston, USA, July, 19–21 (1993) 277–282.

[22] J. Savković–Stevanović, A. fuzzy–neural network controller, Proc. of the 10th Inter. Conf. on Math and Computer Modeling and Sci. Computing, Boston, USA, Juli, (1995) 5–8.

[23] J. Savković–Stevanović, Neurofuzzy modular modeling and control of a distillation plant, Proc. of the ESM'99 – 13th Europ. Sim. Multiconf., Modeling & Simulation a Tool for the Next millennium, Warsaw, Poland, June 1–4 (1999).

[24] J. Savković–Stevanović, Neural–fuzzy controller an Industrial distillation plant, Proc. of the IES'99 – 10th Sem. and Symp. on Inf. and Expert Systems in the Process Inds., p. 35–38, Belgrade, Yugoslavia, Oct., 29–30 (1999) 35–38.

[25] J. Savković–Stevanović, Journal of System. Engng., **4** (1996) 605–612.

[26] J. Savković–Stevanović, J., Neural network dynamic learning and controllers design, Proc. of the IES'99 – 10th Sem. and Symp. on Inf. and Expert Systems in Process Inds., Belgrade, Yugoslavia, p. 1–34, Oct., 29–30 (1999).

[27] Z. Tang, C. Almeida and P. Fishwick, Simulation, **57** (1991) 303–310.

[28] V. Venkatasubramanian, V. Vaydianathan and Z. Zamanoto, Computers Chem. Engng., **14** (1990) 699.

[29] X.L. Zhong, Lewis J., Nagy – N, F.L. Engng. Applic. Artif. Intell., **9** (1996) 83.

IZVOD

AUTOMATSKO UČENJE I UPRAVLJANJE POMOĆU SPEEDUP-a I NEURONSKIH MREŽA

(Naučni rad)

Jelenka Savković–Stevanović,
Tehnološko–metalurški fakultet, Univerzitet u Beogradu, Jugoslavija

Ovaj rad proučava složene modele neoronskih mreža za dinamičko učenje. Iterativno obučena neuronska mreža korišćena je za dinamičku simulaciju i kontrolu hemijskog rezervoara sa idealnim mešanjem. Opšti algoritam delta pravila je korišćen za obuku mreže minimizirajući sumu kvadrata odstupanja. Pretpostavljena je dostupnost prethodne baze ulazno/izlaznih podataka postrojenja. Kao poremećaj korišćen je generator slučajnih binarnih signala PRBS. Za bazu podataka koja se obučava 1% PRBS je dodavan na vrednosti stacionarnog stanja is SPEEDUP simulacija. Kada jedanput obučen model neuronske mreže stoji na raspolaganju onda je on korišćen za učenje u realnom vremenu i za regulaciju pH u rezervoaru sa mešanjem. Ispitivani standardni i inverzni modeli regulatora neuronskih mreža pokazuju bolje parametre regulacije nego klasični PI regulator. Složeni model samo regulacije – IMC pokazuje najbolje rezultate u regulaciji i lokalnoj stabilnosti. Dobiveni modeli u ovom radu poboljšavaju otklanjanje šumova i smanjenje variranja procesa. Neki od ovih sistema mogu se koristiti i za samoodržavanje nelinearnih multivarijabilnih modela regulatora i permanentno otklanjanje smetnji.

Ključne reči: PRBS poremećaj • pH kontrola • SPEEDUP simulacija • Kontrola neuronskim mrežama • IMC struktura neuronske mreže •
Key words: PRBS disturbance • pH control • SPEEDUP simulation • Neural networks control • Neural network IMC structure •