Neural Model-Predictive Control of Distributed Parameter Crystal Growth Process

Masaru Ishida and Jixian Zhan
Research Laboratory of Resources Utilization, Tokyo Inst. of Technology, Yokohama 226, Japan

A neural model-predictive control (NMPC) method is proposed in which a multilayer neural network represents the forward process model. This method is based on the deepest descent optimization algorithm. The calculation is completely in the domain of neural network.

A distributed parameter anthracene crystal growth process is used as an example to demonstrate the proposed method. Good results are obtained by simulation tests using both three-layer and two-layer networks.

Proposed NMPC Method

The case of no-time delay multiinput multioutput (MIMO) process is considered, assuming N input and N output variables. A neural network with one hidden layer is constructed to represent a discrete-time forward process model as shown in Figure 1, where u represents manipulated variables and y controlled variables.

The above neural network is first trained with process input-output data to represent process model. Then this same neural model is utilized to determine the optimal values of manipulated variables by the optimizer shown in Figure 2. The optimization algorithm is expressed as follows:

Let t_i and $y_{m,i}$ $(i=1,..., N)$ respectively be target values and actual outputs of output neurons of $1, \ldots, N$, the error is defined as:

$$E = \sum_{i=1}^{N} e_i = \sum_{i=1}^{N} \frac{1}{2} (y_{m,i} - t_i)^2 \tag{1}$$

Linden and Kindermann (1989) developed a method of inversion for multilayer network. The basic idea is to minimize the error defined above by adjusting the input values while keeping the network weights unchanged. The control method proposed in this work is based on the same idea as Linden and Kindermann's method.

At sampling time k, taking the setpoint values of controlled variables at next sampling time of $(k+1)$ as target outputs, the error is redefined as:

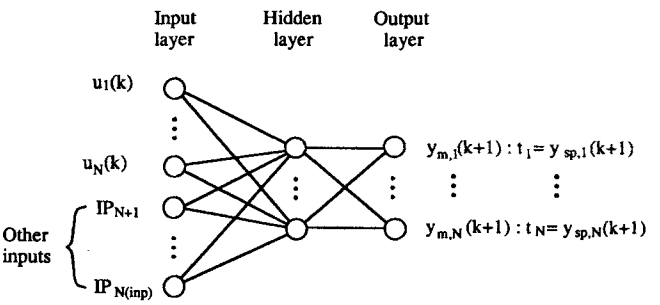

Figure 1. Neural network representing forward model of a MIMO process.

$$E = \sum_{i=1}^{N} e_i = \sum_{i=1}^{N} \frac{1}{2} [y_{\text{pred},i}(k+1) - y_{sp,i}(k+1)]^2 \tag{2}$$

where $y_{sp,i}(k+1)$ is the setpoint of ith controlled variable at time $k+1$, and $y_{\text{pred},i}(k+1)$ is compensated prediction value. The plant $y_i(k)$/model $y_{m,i}(k)$ mismatch is compensated as follows:

$$d_i(k) = y_i(k) - y_{m,i}(k) \tag{3}$$

$$y_{pred,i}(k+1) = y_{m,i}(k+1) + d_i(k) \qquad \text{for all } i = 1, \ldots, N \tag{4}$$

According to the deepest descent method, the manipulated variable at time k, $u_j(k)$ is calculated repeatedly as:

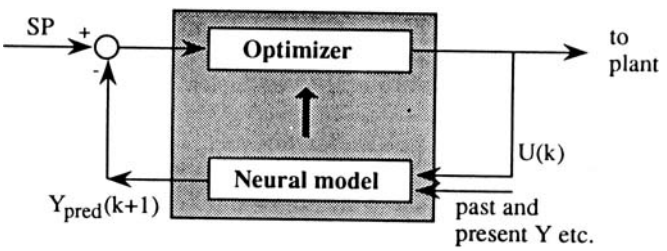

Figure 2. Neural model predictive control strategy.

Correspondence concerning this article should be addressed to M. Ishida.

$$u_j^{new}(k) = u_j(k) - \eta \frac{\partial E}{\partial u_j(k)} \qquad (j = 1, \ldots, N) \quad (5)$$

where η is the step size of the steepest descend method. According to Eqs. 2 through 4 we obtain the partial differential term in the above equation as:

$$\frac{\partial E}{\partial u_j(k)} = \sum_{i=1}^{n} (y_{\text{pred},i}(k+1) - y_{sp,i}(k+1)) \frac{\partial y_{\text{pred},i}(k+1)}{\partial u_j(k)}$$

$$= \sum_{i=1}^{N} (y_{\text{pred},i}(k+1) - y_{sp,i}(k+1)) \frac{\partial y_{m,i}(k+1)}{\partial u_j(k)} \quad (6)$$

The last partial differential term $\partial y_{m,i}(k+1)/\partial u_j(k)$ is calculated by chain operation to the neural network.

To apply Eq. 5, initial values of $u_j(k)$ are necessary, and a simple initialization method is to let $u_j(k) = u_j(k-1)$, $(j = 1, \ldots, N)$. By substituting Eq. 6 into Eq. 5, the new value of $u_j(k)$ $(j = 1, \ldots, N)$ can be calculated. This calculation is carried out repeatedly until an acceptable value of error defined by Eq. 2 is achieved. Constraints on manipulated variables, like $u_{j,\max}$, $u_{j,\min}$ and $|u_j(k+1) - u_j(k)|_{\max}$ can be simply handled during the calculation. When any one of $u_j(k)$ reaches its limitation value, this manipulated variable is then set to equal to this value, but the adjustment for the other manipulated variables is continued, until an optimal solution is found. Then the control action is implemented, and the same calculation is carried out for the next sampling interval. Here the critical point is the optimizer by which manipulated variables are determined. Although in this work, only one-step predictive control is demonstrated, the method proposed here can be extended to multistep predictive control.

Nahas et al. (1992) pointed out that for the case with a single variable to be optimized, it is superior to use Newton's algorithm than to use the backpropagation algorithm and they chose the former for their optimization calculation of the neural network internal model control. However, in this article, for the optimization of multivariables, backpropagation algorithm is preferred.

Application to Distributed Parameter Crystal Growth Process

Process description

A distributed parameter anthracene crystal growth process is adopted with little modification from Ito and Katoh (1994) as an application example. As shown in Figure 3, the temperature distribution along the ampoule tube is controlled by five electric heaters with the tube fixed. The size of the ampoule tube is 7 mm in diameter and 36 mm in length. The desired temperature profile for this process is shown in Figure 4. The temperature profile with a slope G moves from bottom ($z = 0$) to top ($z = 1$) at speed R. For the purpose of simulation, the heat balance for this process is described as follows:

$$\frac{\partial x(z,t)}{\partial t} = \frac{\partial^2 x(z,t)}{\partial z^2} + p(z,t) - l(z,t) \qquad (0 \le z \le 1) \quad (7)$$

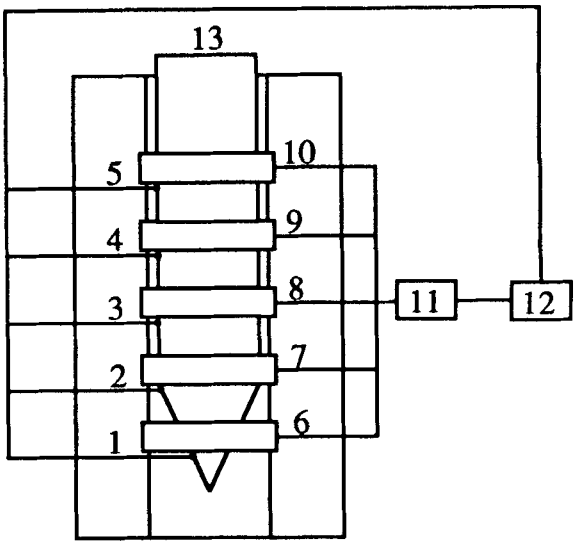

Figure 3. The system.

(1) – (5) = Thermocouples, (6) – (10) = heater; (11) = thyristor; (12) = NMPC controller; (13) = ampoule tube.

with initial and boundary conditions as:

$$x(z,0) = x_0(z) \quad (8)$$

$$\left.\frac{\partial x}{\partial t}\right|_{z=0} = \left.\frac{\partial x}{\partial t}\right|_{z=1} = 0 \quad (9)$$

where l, p, t, x and z are dimensionless heat loss, heat produced by the heaters, time, temperature, and height, respectively.

Equation 7 is discretized by a step-by-step method (Jenson and Jeffreys, 1977) with initial and boundary conditions 8 and 9. The temporal step-size for discretization is 6s and spatial step-size is 36/50 = 0.72 mm.

Neural network control with the proposed method

Manipulated variables are the heat supplies by the five electric heaters, and the controlled variables are the anthracene temperatures at the centers of the regions corresponding to the heaters.

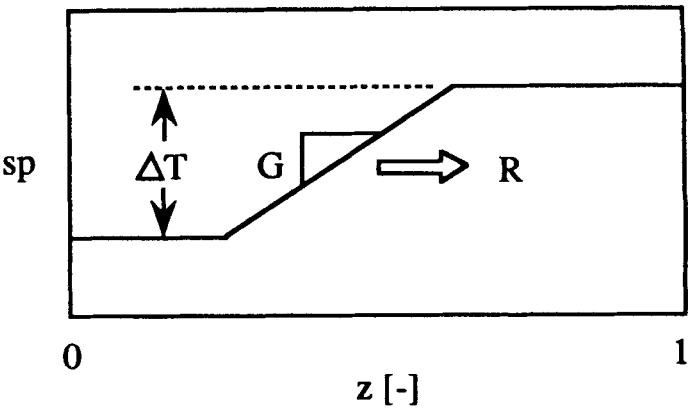

Figure 4. Desired temperature profile.

In applying the proposed neural network method, all the controlled variables, temperatures y_1 to y_5, and manipulated variables, u_1 to u_5, are scaled to the range of [0,1] by:

$$y_i = (T_i - T_{\min})/(T_{\max} - T_{\min}) \quad (i = 1, \ldots, 5) \quad (10)$$

$$u_j = (q_j - q_{\min})/(q_{\max} - q_{\min}) \quad (j = 1, \ldots, 5) \quad (11)$$

where T_i is temperature and q_j is heat supply rate by the heaters. $T_{\min}$, $T_{\max}$ and $q_{\min}$ are used merely for normalization and $q_{\max}$ is the constraint on the manipulated variables q_j ($j = 1$ to 5). The constraint on changing speed of manipulated variables is $|q_j(k) - q_j(k-1)|_{\max} = q_{\max}/6$. y and u are proportional to x and p in Eqs. 7 through 9, respectively.

Training of the Neural Network. A three-layer network with 10 input, 5 hidden and 5 output units is constructed. The inputs and outputs are as follows:

$$\text{Inputs: } u_1(k), u_2(k), u_3(k), u_4(k), u_5(k),$$
$$y_1(k), y_2(k), y_3(k), y_4(k), y_5(k)$$
$$\text{outputs: } y_1(k+1), y_2(k+1), y_3(k+1), y_4(k+1), y_5(k+1)$$

where k represents present time. The sampling interval is chosen to be $\Delta t = 30$s, so $k+1$ means 30 s after the present time.

To identify the process model, the neural network is trained off-line first of all. To train the neural network, six sets of training data are obtained by discretization of Eq. 7. Among them, five are obtained by randomly changing one of the five manipulated variables, u_j ($j = 1, \ldots, 5$) while keeping the others unchanged. The sixth set is obtained by changing all of them randomly. The network is trained with all the data sets 5,000 times in total; then the trained network is used as forward model for control. The training data set No. 4 and the trained results are illustrated in Figure 5. The trained results demonstrate that the neural prediction is fairly good. We can also see the interaction effect of u_4 to y_3 and y_5.

Control. As shown in Figure 4, the desired temperature profile is moving with time. The control results are shown in Figure 6. The bold lines in the y-time plot represent the setpoints, and the controlled results of y_1 through y_5 are given by different legends. All of the initial temperature values are set to be equal. Then y_1, the bottom part temperature of the ampoule tube, begins to decrease. The temperature descending profile moves upward gradually to $y_2, \ldots, y_5$. Finally, all of them are kept constant at the lower value. When we notice the change of $u_2, \ldots, u_5$, it can be found that when they start to change, all of them increase to some degree before decreasing. This is the correct reaction to cancel the interaction effect from the neighboring u_j ($j = 1, \ldots, 4$) to keep the setpoint $y_{sp,i}$ ($i = 2, \ldots, 5$).

Figure 7 shows the control results of a two-layer network. The algorithm is the same as that of the three-layer network. Because the example process is a linear process, a two-layer network is good enough for the neural modeling and the whole calculation is even simpler.

We also used controlled results (controlled process input-output data) to train the network after each run, but as the off-line model identification had been performed almost perfectly, the controller did the job well enough from the first run and the controller performance did not vary much.

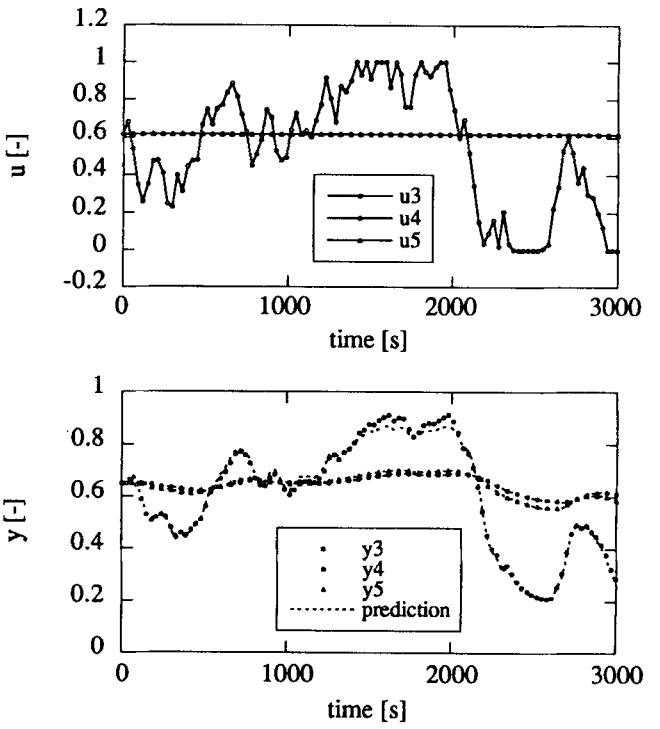

Figure 5. Training data and trained results for set No. 4 (changing u_4 and keeping other u's unchanged).

In this work, NEC PC-9801BA with 40MHz CPU is used. For the calculatioin of the 5 manipulated variables, $u_1(k)$

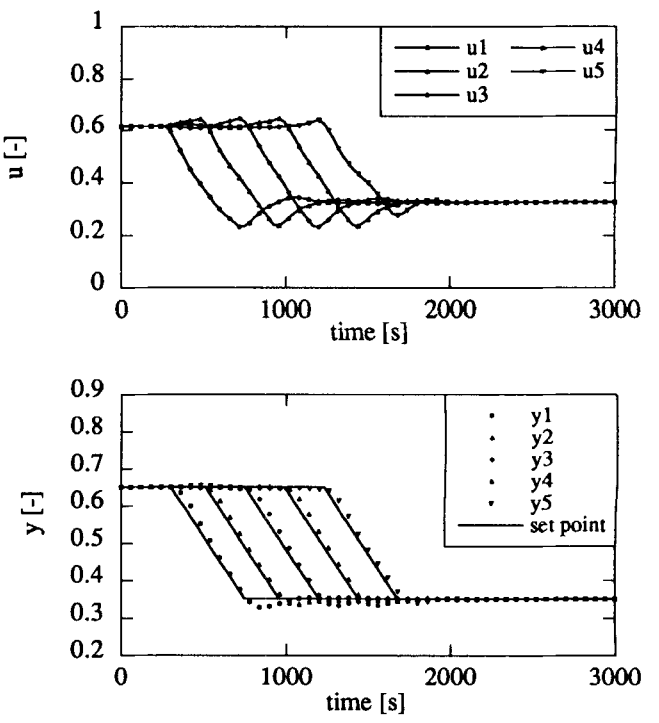

Figure 6. Control result with a 3-layer network.

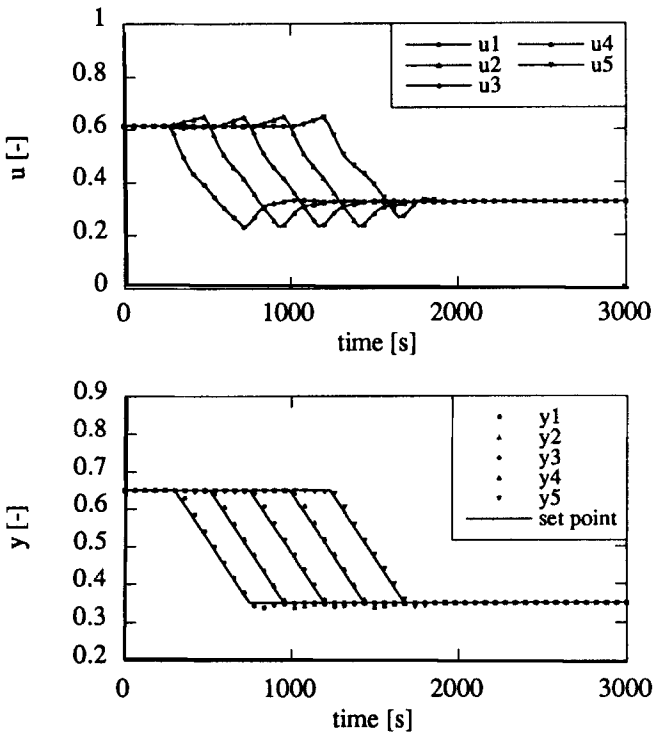

Figure 7. Control result with a 2-layer network.

through $u_5(k)$, during one sampling interval, about 0.2 s is needed for the three-layer network (10 input, 5 hidden and 5 output units) with 20 times iteration based on Eq. 5 with $\eta = 0.5$.

Conclusions

A NMPC method is proposed in which a mulitlayer neural network represents the forward process model. The calculation of the optimal values of manipulated variables is carried out according to the trained network. The optimization is based on the deepest descent algorithm, and the calculation is completely in the domain of neural network. A linear distributed parameter anthracene crystal growth process is used as an example to demonstrate the proposed method. Good results are obtained by simulation tests using both three-layer and two-layer networks.

The proposed NMPC method has the following advantages:

(1) For many chemical processes it is difficult to obtain process models, but neural networks can "learn" process dynamics from available input-output data. Hence, we can use trained neural networks as process models.

(2) The proposed method can be easily extended to multistep predictive control. Hence, it possesses the advantages which a conventional MPC method possesses, like constraint handling ability.

(3) The calculation algorithm is easy to understand and any process engineer can master the control method proposed here.

(4) The characteristics of some processes may change with time, but this will not cause big problems for the present neural model-based control because the neural network possesses on-line learning ability.

Further work is necessary to extend the proposed NMPC method to nonlinear MIMO processes with different time lag values. The comparison with other model-based control methods (like the control method for nonlinear processes using polynomial ARMA model proposed by Hernandez and Arkun (1993)) is also necessary.

Literature Cited

Ito, Y., and N. Katoh, "A New Bridgman Process for Single Crystal Growth Based on Control of Distributed Parameter System," *Kagaku Kogaku Ronbunshu*, **20**, 288 (1994).

Hernandez, E., and Y. Arkun, "Control of Nonlinear Systems Using Polynomial ARMA Models," *AIChE J.*, **39**, 446 (1993).

Jenson, V. G., and G. V. Jeffreys, *Mathematical Methods in Chemical Engineering*, 2nd ed., Academic Press, p. 413 (1977).

Linden, A., and J. Kindermann, "Inversion of Multilayer Nets," IEEE Int. Joint Conf. on Neural Networks, Washington, p. 425 (June 1989).

Nahas, E. P., M. A. Henson, and E. E. Seborg, "Nonlinear Internal Model Control Strategy for Neural Network Models," *Computers Chem. Eng.*, **16**, 1039 (1992).

Manuscript received July 5, 1994, and revision received Oct. 31, 1994.