Predictive Control With Neuro-compensation For A Class of Nonlinear Processes

Furong Gao, Fuli Wang* and Mingzhong Li
Department of Chemical Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Phone: +852-2358-7139, Fax: +852-2358-0054
email: kefgao@ust.hk

ABSTRACT: Many processes in the chemical industry have modest nonlinearities. Linear dynamics play a dominant role in governing the process output behaviour in the interesting operating range, but the linearization errors are significant. Linear based control will give a poor performance, while a nonlinear based control will result in computation complexity. For this type of processes we propose to model them with a composite model consisting of a linear model (LM) and a multilayered feedforward neural network (MFNN). The LM is used to capture the linear dynamics, while the MFNN is employed to approximate the LM's residual errors, i.e., the process nonlinearities. Effective off-line and on-line identification algorithms are proposed for the identification of the composite model. With this model structure, it is shown that a simple analytical predictive control law can be formulated to control a nonlinear process. A simulation example is also given to illustrate the effectiveness of the model identification and the proposed predictive control.

KEYWORDS: Model predictive control; nonlinear processes; neural network.

1. Introduction

Model predictive control (MPC), an intuitive and effective control strategy, has attracted much research attention [1-3], primarily due to its design feature of the minimization of a cost penalizing future deviations from an output or state reference trajectory. Linear model predictive control (LMPC), a widely recognized method, has been successfully applied to many chemical processes [4-6]. Despite of the fact that most chemical processes are inherently nonlinear, they have been "adequately" controlled with a linear control design over a small operating range. Today, demands for tighter environmental regulation, better energy utilization, higher product quality and more production flexibility have made process operations more complex with a larger operating range. Consequently, predictive control based on linear model may not always yield satisfactory results.

The interest now has been increasingly shifting toward designing predictive controllers based on nonlinear models [7-9]. Neural networks, having an inherent ability to approximate an arbitrary nonlinear function, have become an attractive means for modeling nonlinear processes. Several neural-network-based predictive control algorithms for nonlinear processes have been proposed [10,11]. However, with nonlinear process models, two major drawbacks arise in MPC. First, it is now difficult to obtain the control law in an analytical form, and the control sequence must be obtained via nonlinear optimization routine, resulting in a higher computational effort and potential numerical stability problem. Second, the abundant theory and experience accumulated for the design and the tuning of linear controllers cannot easily be used for the design and tuning of nonlinear controllers.

A well-known method to simplify a nonlinear model is the use of the Jacobian linearization, but it is only valid in a certain operating range. To make a process model to be valid for a broader operating range than the Jacobian linearization, a process model consisting of a linear model (LM) plus a multilayered feedforward neural network model (MFNN) is proposed to model a nonlinear process, in which the LM represents the Jacobian linearization of the process, while the MFNN approximates the LM's modeling error, i.e., the nonlinearity of the process. This paper

* on leave from Northeastern University , P. R. China.

concerns with the model identification and the MPC design for nonlinear processes in which the LM can represent the dominant dynamics in the interesting operating range. With this form of the model, we will show that predictive controllers can be designed based on the LM, and the outputs of the MFNN, which represent the process nonlinearities, can be viewed as measurable disturbances and are eliminated through "feedforward" control. As a result, the existing linear predictive control techniques can be directly applied to yield an analytical control law. Furthermore, the process nonlinearities are compensated with the aid of the MFNN, this strategy can be expected to be applicable to a broad process operating range, without any intensive computation for the control determination.

2. Process modeling and controller design

In this section, a class of single-input-single-output nonlinear processes is first modeled, followed by derivation of the predictive controller based on this model. As stated earlier, it is assumed that the process nonlinearity is modest, i.e., a linear model can represent the dominant process dynamics in the interesting operating range, and at the same time, the linearization errors are significant. A composite model consisting of a linear model (LM) and a multilayered feedforward neural network (MFNN) is used to model such a process. Suppose that an input sequence $\{u(k)\}$ is presented to the process and that the corresponding output sequence $\{y(k)\}$ is measured. The following can then be used to model the nonlinear input-output mapping from $\{u(k)\}$ to $\{y(k)\}$:

$$y(k+1) = x(k)^T q_1 + NN(x(k), q_2) + x(k+1) \tag{1}$$

In this equation, $x(k)^T q_1$ is a linear model (LM) with parameter vector q_1 and input vector $x(k) = [y(k), y(k-1), ..., y(k-n+1), u(k), u(k-1), ..., u(k-m)]^T$, $NN(x(k), q_2)$ is a multilayered feedforward neural network (MFNN) with weight vector q_2 and input vector $x(k)$, and $x(k+1)$ is the modeling error. The LM represents the linearization of the process in the interesting operating range, while the MFNN approximates the linearization error, i.e., the process linearity. The good nonlinear mapping ability of the MFNN can make the modeling error, $x(k+1)$, to be sufficiently small by properly determining the parameter vector q_1 and the network weight vector q_2.

The process model of Eq. (1) can be rearranged as

$$A(q^{-1})y(k) = B(q^{-1})u(k-1) + j(k-1) + x(k) \tag{2}$$

where $A(q^{-1}) = 1 + a_1 q^{-1} + ... + a_n q^{-n}$ and $B(q^{-1}) = b_0 + b_1 q^{-1} + ... + b_m q^{-m}$ with $a_i = -q_{1,i}\ (i = 1, ..., n)$ and $b_j = q_{1,n+1+j}\ (j = 0, 1, ..., m)$; $q_{1,i}$'s $(i = 1, ..., n+m+1)$ are the components of the parameter vector q_1; $j(k)$ is defined as $j(k) = NN(x(k), q_2))$.

In a predictive control, predictive equations should be developed to predict the process outputs. To do so, introducing the following polynomial equations:

$$1 = F_i(q^{-1})A(q^{-1}) + q^{-i}G_i(q^{-1}) \qquad i \geq 1 \tag{3}$$

where $F_i(q^{-1}) = 1 + f_{i,1}q^{-1} + ... + f_{i,i-1}q^{-i+1}$ and $G_i(q^{-1}) = g_{i,0} + g_{i,1}q^{-1} + ... + g_{i,n-1}q^{-n+1}$, then we have

$$y(k+i) = H_i(q^{-1})u(k+i-1) + G_i(q^{-1})y(k) + F_i(q^{-1})j(k+i-1) + F_i(q^{-1})x(k+i) \tag{4}$$

where $H_i(q^{-1}) = F_i(q^{-1})B(q^{-1})$. Defining

$$H_i(q^{-1}) = H_{i,1}(q^{-1}) + q^{-i}H_{i,2}(q^{-1}) \tag{5}$$

where $H_{i,1}(q^{-1}) = h_{i,1}^0 + h_{i,1}^1 q^{-1} + ... + h_{i,1}^{i-1} q^{-i+1}$ and $H_{i,2}(q^{-1}) = h_{i,2}^0 + h_{i,2}^1 q^{-1} + ... + h_{i,2}^{m-1} q^{-m+1}$ then Eq. (4) can be written as:

$$\begin{aligned} y(k+i) = {} & H_{i,1}(q^{-1})u(k+i-1) + H_{i,2}(q^{-1})u(k-1) + G_i(q^{-1})y(k) \\ & + F_i(q^{-1})j(k+i-1) + F_i(q^{-1})x(k+i) \end{aligned} \tag{6}$$

The polynomials $F_i(q^{-1})$, $G_i(q^{-1})$, $H_{i,1}(q^{-1})$ and $H_{i,2}(q^{-1})$ can be calculated by the following recursive formulas:

$$F_{i+1}(q^{-1}) = F_i(q^{-1}) + q^{-i}g_{i,0} \tag{7}$$

$$G_{i+1}(q^{-1}) = q[G_i(q^{-1}) - g_{i,0}A(q^{-1})] \tag{8}$$

$$H_{i+1,1}(q^{-1}) = H_{i,1}(q^{-1}) + q^{-i}(h_{i,2}^0 + g_{i,0}b_0) \tag{9}$$

$$H_{i+1,2}(q^{-1}) = q[H_{i,2}(q^{-1}) + g_{i,0}B(q^{-1}) - h_{i,2}^0 - g_{i,0}b_0] \tag{10}$$

with the initial values $F_1(q^{-1}) = 1$, $G_1(q^{-1}) = q[1 - A(q^{-1})]$, $H_{1,1}(q^{-1}) = b_0$ and $H_{1,2}(q^{-1}) = q[B(q^{-1}) - b_0]$.

From Eq. (6), the i-step-ahead prediction of the process output can be obtained by neglecting the unknown quantity $F_i(q^{-1})x(k+i)$, which is a linear combination of the future model errors. Then the predictive equations take the form:

$$\begin{aligned}\hat{y}(k+i) = H_{i,1}(q^{-1})u(k+i-1) + H_{i,2}(q^{-1})u(k-1) + G_i(q^{-1})y(k) \\ + F_i(q^{-1})\hat{j}(k+i-1)\end{aligned} \tag{11}$$

where $\hat{j}(k+i-1)$, the prediction of $j(k+i-1)$, can be calculated by replacing the future process outputs with their corresponding predictions. The predictions $\hat{y}(k+i)$ $(i = 1,2,...)$ can be recursively calculated using Eq. (11).

Long range predictive control strategies minimize a quadratic criterion consisting of future process output predictions and control inputs in a receding horizon sense. A commonly used quadratic criterion is as follows:

$$J = \frac{1}{2}\sum_{i=1}^{M}\{[w(k+i) - \hat{y}(k+i)]^2 + l[u(k+i-1) - u(k+i-2)]^2\} \tag{12}$$

where $w(k)$ is the desired process output, M is the prediction horizon and $l \geq 0$ is a weighting factor. The control is to minimize the criterion (12) in a receding horizon sense, resulting in a sequence of optimized control moves over the control horizon $u(k), u(k+1), ..., u(k+M-1)$. Only the first control move in the sequence, $u(k)$, is actually implemented. At the next instant, the procedure is repeated and $u(k+1)$ is determined in the same fashion.

A common practice in reducing the complexity of the optimization computation is to assume a constant control over the prediction horizon, i.e., $u(k+i) = u(k)$ $(i \geq 1)$. In this case, the prediction equation (11) reduces to

$$\hat{y}(k+i) = H_{i,1}(1)u(k) + H_{i,2}(q^{-1})u(k-1) + G_i(q^{-1})y(k) + F_i(q^{-1})\hat{j}(k+i-1) \tag{13}$$

and Eq. (12) becomes

$$J = \frac{1}{2}\sum_{i=1}^{M}[w(k+i) - \hat{y}(k+i)]^2 + \frac{1}{2}l[u(k) - u(k-1)]^2 \tag{14}$$

The definition of $\hat{j}(k)$ indicates that $\hat{j}(k+i-1)$ $(i = 1,2,...M)$ is a function of $u(k)$, and this makes the minimization of J with respect to $u(k)$ a nonlinear optimization problem. An iterative computation is commonly needed to obtain a numerical solution of a nonlinear optimization. For a real-time control, iterative computation is not preferred due to the time constraint. A simplification is proposed to cope with this problem. Notice that we have placed a constraint on the term $u(k) - u(k-1)$, the change between two consecutive control moves. This implies that $u(k)$ can not be significantly different from $u(k-1)$. $\hat{j}(k+i-1)$ $(i = 1,2,...M)$ can , therefore, be approximately determined by replacing the current control $u(k)$ with the previous control $u(k-1)$. The approximates of $\hat{j}(k+i-1)$ $(i = 1,2,...M)$, denoted by $\hat{j}_0(k+i-1)$, can be computed prior to the optimization as:

$$\hat{j}_0(k+i-1) = NN(\hat{x}(k+i-1), q_2) \tag{15}$$

where $\hat{x}(k+i-1) = [\hat{y}_0(k+i-1), ..., \hat{y}_0(k+i-n), \hat{u}(k+i-1), ..., \hat{u}(k+i-m-1)]^T$ with

$$\hat{u}(k+j) = \begin{cases} u(k-1) & j \geq 0 \\ u(k+j) & j < 0 \end{cases} \tag{16}$$

and $\hat{y}_0(k+j) = y(k+j)$ for $j \le 0$. $\hat{y}_0(k+i)$ $(i = 1,2,...,M)$, the quasi-i-step-ahead predictions, can be obtained by replacing $u(k)$ with $u(k-1)$ in Eq. (13), i.e.,

$$\hat{y}_0(k+i) = H_{i,1}(1)u(k-1) + H_{i,2}(q^{-1})u(k-1) + G_i(q^{-1})y(k) + F_i(q^{-1})\hat{j}_0(k+i-1)$$
$$i = 1,...,M \tag{17}$$

Substituting Eq. (15) into Eq. (13) results in the optimization of Eq. (14) to be linear with the analytic solution of

$$u(k) = u(k-1) + (l + \sum_{i=1}^{M} H_{i,1}(1)^2)^{-1} \overline{H}_1(q^{-1})(w(k+M) - \hat{y}_0(k+M)) \tag{18}$$

where

$$\overline{H}_1(q^{-1}) = H_{M,1}(1) + H_{M-1,1}(1)q^{-1} + ... + H_{1,1}(1)q^{-M+1} \tag{19}$$

It can be seen from Eq. (15) that $\hat{j}_0(k+i-1)$'s $(i = 1,2,...,M)$ are the outputs produced by the MFNN with the input $\hat{x}(k+i-1)$. The process nonlinearity, characterized by $j(k+i-1)$, is compensated in the control law by the term $\hat{j}_0(k+i-1)$.

In the case where the controlled process dynamics are unknown, i.e., q_1 and q_2 in Eq. (1) are the unknown parameter vectors, an identification method needs to be developed to estimate them. This will be discussed in the next section.

3. Identification algorithm

Given a set of process input-output data $\{ y(k), x(k-1); k = 1,2,..., N \}$, the parameter vectors q_1 and q_2 can be estimated by minimizing the following cost function:

$$I(q_1, q_2) = \sum_{i=1}^{N} (y(i) - x(i-1)^T q_1 - NN(x(i-1), q_2))^2 \tag{20}$$

This is a typical nonlinear optimization problem, and it can be solved by many standard nonlinear optimization methods, e.g., gradient descent method. A more effective and simpler algorithm, however, can be formulated by making use of the unique structure of the process model (1). We propose the following iterative optimization procedure for the estimation of q_1 and q_2. First, make an initial guess on q_2, say, q_2^0, and minimize $I(q_1, q_2^0)$ to obtain the optimal q_1^1; With the above optimized q_1^1, $I(q_1^1, q_2)$ is then minimized to obtain the optimal q_2^1. The above steps can be repeated, in theory, until the modeling error $x(k)$ is sufficiently small. As the process nonlinearity is modest, it can be expected that the output of the MFNN will be relatively small, implying that the optimized weight vector, q_2, is in the neighborhood of the origin. $q_2^0 = 0$ is, therefore, a well-guessed initial. Extensive simulations indicate that with the initial $q_2^0 = 0$, the results after one-round of optimization (i.e., $q_1 = q_1^1$ and $q_2 = q_2^1$) can yield satisfactory model parameters. Further iterations generate only little improvement and thus are not necessary in reality.

The above can be summarized into the following off-line and on-line estimation procedures.

Off-line estimation algorithm:

(i) Set $q_2 = 0$ in Eq. (20), and minimizing Eq. (20) with respect to q_1 produces the optimum value of q_1, denoted by q_1^*. This results in:

$$q_1^* = [X^T X]^{-1} X^T Y \tag{21}$$

where

$$Y = [y(1), y(2),..., y(N)]^T \text{ and } X = \begin{bmatrix} x(0)^T \\ x(1)^T \\ \vdots \\ x(N-1)^T \end{bmatrix}$$

Alternatively, q_1^* can be determined equivalently by the recursive least square algorithm:

$$q_1(i) = q_1(i-1) + K(i)[y(i) - x^T(i-1)q_1(i-1)] \quad (22a)$$

$$K(i) = P(i-1)x(i-1)[x^T(i-1)P(i-1)x(i-1)+1]^{-1} \quad (22b)$$

$$P(i) = [I - K(i)x^T(i)]P(i-1) \quad (22c)$$

$$i = 1,2,..., N$$

with $P(0) = a\ I$ (a is a big positive real number) and $q_1(0) = 0$. Then the optimum value of q_1 can be obtained by setting $q_1^* = q_1(N)$.

(ii) Set $q_1 = q_1^*$ in Eq. (20), and minimize Eq. (20) with respect to q_2. A number of optimization schemes can be used to find the optimum q_2, the gradient descent method is used here for simplicity. The optimum q_2, denoted by q_2^*, is calculated iteratively:

$$q_2(t) = q_2(t-1) - h\frac{\partial J}{\partial q_2(t-1)}$$

$$= q_2(t-1) + h\sum_{i=1}^{N}[y_N(i) - NN(x(i-1), q_2(t-1))]\frac{\partial NN(x(i-1), q_2(t-1))}{\partial q_2(t-1)}$$

$$q_2(0) = 0;\ t = 1,2,...,T \quad (23)$$

where $y_N(i) = y(i) - x^T(i-1)q_1^*$ and h is the learning rate. The derivative of the network output with respect to the weight vector q_2 can be calculated by the well-known backproporgation (BP) algorithm [12]. The iteration is repeated until the cost function I is less than a prespecified value, or the maximum allowable iterations T is reached. Suppose the iteration stops at $t = t^* (\le T)$, then set $q_2^* = q_2(t^*)$.

On-line estimation algorithm:

On-line estimation may be used to capture in real-time the changes in the process dynamics. Assume a pair of new process input-output data $\{y(k), x(k-1)\}$ is available at time instant k, then the parameter vectors q_1 and q_2 are updated by the following formulas.

(i)
$$q_1(k) = q_1(k-1) + K(k)[y_L(k) - x^T(k-1)q_1(k-1)] \quad (24a)$$

$$K(k) = P(k-1)x(k-1)[x^T(k-1)P(k-1)x(k-1)+1]^{-1} \quad (24b)$$

$$P(k) = [I - K(k)x^T(k)]P(k-1), \quad P(0) = a\ I \quad (24c)$$

where $y_L(k) = y(k) - NN(x(k-1), q_2(k-1))$. The initials $q_1(0)$ and $q_2(0)$ are provided by the off-line estimation algorithm, i.e., $q_1(0) = q_1^*$ and $q_2(0) = q_2^*$.

(ii)
$$\bar{q}_2(t) = \bar{q}_2(t-1) + h[y_N(k) - NN(x(k-1), \bar{q}_2(t-1))]\frac{\partial NN(x(k-1), \bar{q}_2(t-1))}{\partial \bar{q}_2(t-1)}$$

$$\bar{q}_2(0) = q_2(k-1), \quad t = 1,2,...,T^* \quad (25a)$$

$$q_2(k) = \bar{q}_2(T^*) \quad (25b)$$

where $y_N(k) = y(k) - x^T(k-1)q_1(k)$, and T^* is the number of iterations in each sampling period.

The proposed model structure allows the LM's parameters to be estimated in the recursive least square form, good adaptive ability of the algorithm can therefore be expected when used in an online fashion.

4. Simulation

In this section, simulations are conducted to illustrate the validity of the proposed identification and control algorithms. The nonlinear process

$$y(k+1)=\frac{2.5y(k)y(k-1)}{1+y(k)^2+y(k-1)^2+y(k-2)^2}+0.4y(k-3)+u(k)+1.1u(k-1) \qquad (26)$$

is used to generate input-output data for the simulations.

Model identification. In the simulation, the input vector is set to be $x(k)=[y(k),y(k-1),y(k-2),y(k-3),u(k),u(k-1)]^T$. The MFNN is determined to be a three layered feedforward neural network with six inputs, 10 hidden units and one output. The hidden layer neurons use the sigmoidal activation function while the input and output layer neurons use a linear activation function. Two hundred pairs of input-output training data are first generated with a random input of uniform distribution from -1 to 1, and the data are subsequently used for the identification of the LM and the training of the MFNN using the off-line estimation algorithm proposed in the preceding section. The initial value of the covariance matrix is set to $P(0)=10^3 I$. The network learning rate and iteration number are set to $h=0.001$ and $T=5000$, respectively. The process output, the LM output, and the LM plus MFNN output, generated by the same input, are shown in Figure 1, and the LM's modeling error and the MFNN output are shown in Figure 2. It can be clearly seen that the LM can capture the dominant part of the process dynamics, but with a significant modeling error. The MFNN can approximate the LM's modeling error, i.e., the process nonlinearity, quite well, as shown in Figure 2. This demonstrates that a nonlinear process can be modeled by a LM together with a MFNN to a satisfactory accuracy.

To illustrate that iterations will not produce significant improvement of the model performance, the off-line estimate algorithm is executed once more by setting the optimized parameter as the initial, i.e., $q_2^0=q_2^*$. The new solution $q_1=[0.1299,-0.1402,0.2413,0.3574,\ 1.0001,0.9683]^T$ is not significantly different from the 1st iteration result $q_1=[0.1276,-0.1370,0.2336,0.3517,1.0002,0.9702]^T$, and the same is also true for q_2. Therefore, a reasonably good model can be obtained without the necessity of iteration for the off-line modeling.

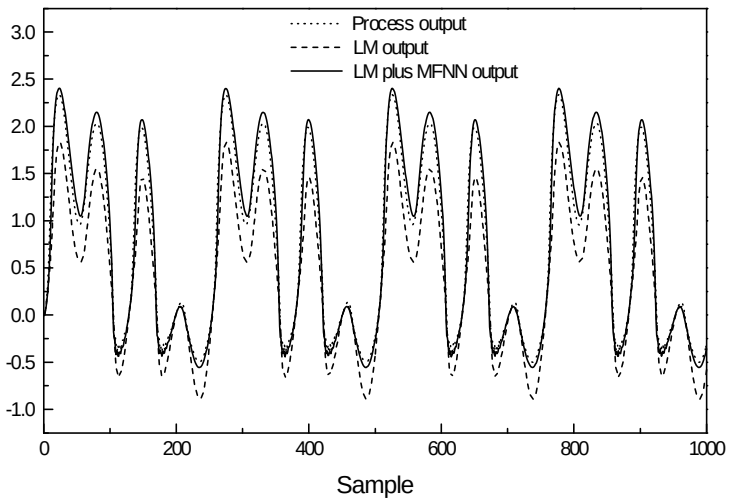

Figure 1: Model validation

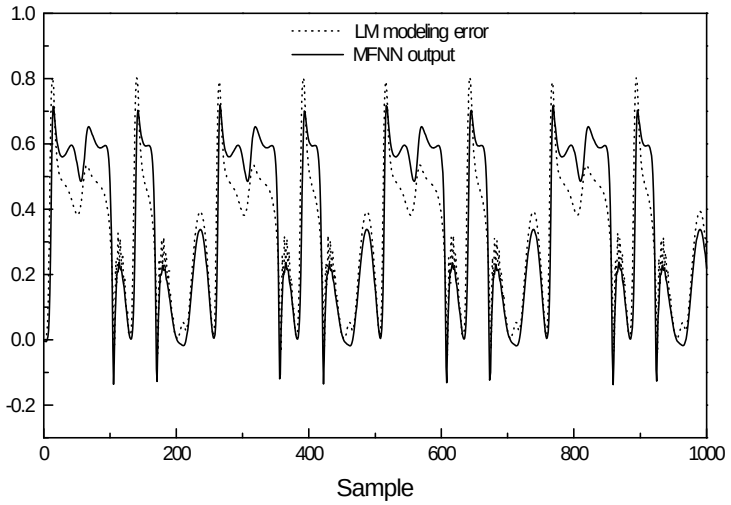

Figure 2: MFNN nonlinearity approximation

Predictive control. To test the proposed control algorithm, it is applied to the process (26), and its control performance is compared to that of the linear model-based predictive control method (LMPC) (i.e., Eqs (17)-(19) with $\hat{j}_0(k+i-1)=0$). The model parameters q_1 and q_2 are updated on-line using the proposed on-line estimation algorithm with the initials determined by the off-line estimation. The initial covariance matrix $P(0)$ and the network learning rate h are set to be the same as the model identification case, and the iteration number T^* is set to be 1 for simplicity. The prediction horizon M and the weighting factor l in the control law (18) are set to be 4 and 0.01, respectively. The control is to track a composite set-point profile consisting of a ramp, a step change, and sinusoidal signals. The tracking results are shown in Figure 3 for the proposed algorithm (solid line) and the LMPC (dashed line). The proposed algorithm tracks the set-point profile closely, without offset and oscillation, indicating a much better control than that of the LMPC. The above simulation demonstrates that the proposed algorithm can be effectively used to control a class of nonlinear processes.

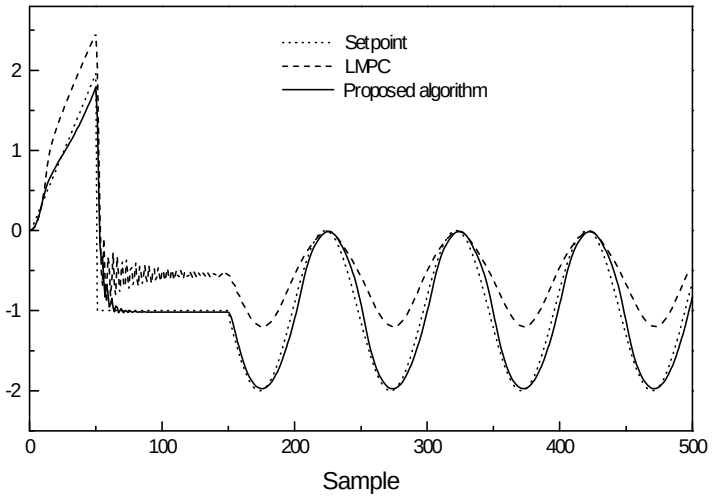

Figure 3: Tracking results of the proposed algorithm and the LMPC

5. Conclusion

An effective analytical predictive control law is presented for a class of nonlinear processes. A LM plus a MFNN is proposed to model a nonlinear process, together with the corresponding identification strategy. With this model representation, a simple analytic predictive control law was derived. Adaptive predictive control, if necessary, can be easily implemented for the process via the proposed on-line estimation algorithm. The simulations show that the proposed LM plus MFNN model structure is suitable for modeling a class of nonlinear processes, and that the proposed predictive control algorithm has a good control performance. It is believed that the proposed algorithm will be useful for many chemical processes where the nonlinearity is modest, and intensive computation is not allowed.

References

(1) Clarke, D. W.; C. Mohtadi, C.; Tuffs, P. S. Generalized Predictive Control- Part 1: Basic Algorithm. Automatica. 1987, 23,137.

(2) Byun, D. G.; Kwon, W. H. Predictive Control: A Review and Some New Stability Results. IFAC Model Based Process Control, GA, 1988.

(3) Muske, K. R.; Rawlins, J.B. Model predictive Control with Linear Models. AIChE J. 1993, 39, 262.

(4) Richalet, J.; Rault, A.; Testud, L.; Papon, J. Model Predictive heuristic Control: Application to Industrial Processes. Automatica 1978, 14, 413.

(5) Culter, C. R.; Ramaker, B. L. Dynamic Matrix Control: a Computer Control Algorithm. The 86th Meeting of The AIChE, Houston, April 1979.

(6) Carica, C. E.; Morari, M. Internal model Control. A Unifying Review and Some New results. Ind. Eng. Chem. Res. 1982, 21, 308.

(7) Wright, G. T.; Edgar, T. F. Nonlinear Model Predictive Control of a Fixed-bed water-gas Shift reactor: An Experimental study. Comput. chem. Eng. 1994, 18, 83.

(8) Peng, C. Y.; Jang, S. S. Fractal Analysis of Time-Series Rule Based Model and Nonlinear Model Predictive control. Ind. Eng. Chem. Res. 1996, 35, 2261.

(9) Arpad, B.; Ferenc, S.; Tibor, C. Convolution Model based Predictive Controller for a Nonlinear Process. Ind. Eng. Chem. Res. 1999, 38, 154.

(10) Jose, R. N.; Wang, H. A Direct Adaptive Neural-Network Control for Unknown Nonlinear Systems and Its Application. IEEE Trans. on Neural Network. 1998, 9, 27.

(11) Tan, Y.; Keyser, R. Neural Network-Based Adaptive Predictive Control. Proceedings of Advances in Model-Based Predictive Control Conference. Oxford, England, 1993.

(12) Narendra, K. S.; and Parthasarathy, K. Identification and Control of Dynamical Systems Using Neural Networks. IEEE Trans. On Neural Networks. 1990, 1, 4.