Batch Process Modelling and Optimal Control Based on Neural Network Models[1)]

Jie Zhang

(*School of Chemical Engineering & Advanced Materials, University of Newcastle, Newcastle upon Tyne NE*1 7*RU UK*)

(E-mail: jie.zhang@newcastle.ac.uk)

Abstract This paper presents several neural network based modelling, reliable optimal control, and iterative learning control methods for batch processes. In order to overcome the lack of robustness of a single neural network, bootstrap aggregated neural networks are used to build reliable data based empirical models. Apart from improving the model generalisation capability, a bootstrap aggregated neural network can also provide model prediction confidence bounds. A reliable optimal control method by incorporating model prediction confidence bounds into the optimisation objective function is presented. A neural network based iterative learning control strategy is presented to overcome the problem due to unknown disturbances and model-plant mismatches. The proposed methods are demonstrated on a simulated batch polymerisation process.

Key words Batch processes, neural networks, model generalisation, optimal control, iterative learning control, polymerisation

1 Introduction

Batch processes are suitable for the agile manufacturing of high value added products, such as specialty polymers, pharmaceuticals, and bio-products[1]. In contrast to continuous processes, batch processes have strong nonlinear behaviour and always operate in transient states. A further difficulty in batch process control is that product quality variables usually cannot be measured on-line and can only be obtained through laboratory analysis after a batch has finished. The main objective in batch process control is to produce a maximum amount of high quality product while under safe process operations. The calculation of optimal control policy requires an accurate process model capable of predicting the product quality at the end of a batch, *i.e.*, the model should be capable of providing accurate long range predictions. These characteristics make the control of batch processes much more complex than the control of a continuous process and novel non-traditional techniques are required[1].

The core in optimal batch process control is an accurate model cable of providing accurate long range predictions of product quality. In earlier works on the optimal control of batch processes, mechanistic models are usually used[2,3]. However, mechanistic models of complex batch processes are usually difficult to develop, especially in responsive processes regarding changing market needs and in multi-product manufacturing. The time taken to develop phenomenological process models has tended to limit the manufacturing applications of mechanistic model based optimal control strategies.

To overcome the difficulties in developing mechanistic models, neural network models based upon process operational data have been widely proposed. Neural networks have been shown to be capable of approximating any continuous non-linear functions[4,5] and have been applied to batch process modelling and control[6∼9]. Neural networks for non-linear process modelling can be broadly divided into two categories: static networks, including multi-layer feed forward neural networks and radial basis function networks, and dynamic networks which include globally recurrent neural networks[10], locally recurrent neural networks[11,12], Elman networks[13] and dynamic filter networks[14]. Static neural networks can provide quite accurate one-step-ahead predictions and are suitable for situations where short-range predictions are the main focus. Such networks are relatively easy to build. Dynamic neural networks are more appropriate for the building of long range prediction models and providing multi-step-ahead predictions[10∼12]

This paper is organised as follows. Section 2 presents reliable neural network modelling through combination of multiple neural networks. Section 3 presents a simulated batch polymerisation process.

1) Supported by UK EPSRC (grants GR/N13319 and GR/R 10875)
Received March 1, 2004; in revised form July 30, 2004

Reliable optimal control of batch processes is given in Section 4. Section 5 presents a neural network model based iterative learning control strategy. Section 6 draws some concluding remarks.

2 Reliable neural network model

An important requirement for industrial applications of neural network model is that the model should possess good accuracy and robustness, *i.e.*, the model should have good generalisation capability and reliability. The accuracy and robustness of neural network models are usually determined by the training data and training methods. When the amount of training data is not sufficient, neural network training tends to over-fit the measurement noise within the training data, leading to large generalisation errors. Ideally, a large amount of training data should be available in order to build accurate neural network models. In industrial processes, data for process variables are usually abundant since they can be automatically measured and recorded by process control computers. However, data for product quality variables are usually obtained from off-line laboratory analysis and, hence, are usually not abundant. Product quality variables are usually sampled at a large sampling time.

Neural network model accuracy and robustness are also affected by network training methods. Since many network training methods are gradient based, different initial weights and different stopping criteria can lead to different network generalisation capability. Some techniques for improving neural network generalisation have been reported recently. One of the techniques is training with regularisation[5,15,16]. The purpose of adding a regularisation term is to prevent unnecessarily large network weights that can lead to large errors on unseen data. Another method for improving network generalisation is to introduce an "early stopping" mechanism in gradient based network training. Data for building a neural model is divided into a training data set and a testing data set. During the training process, both network errors on the training and the testing data sets are monitored. Network training is stopped when the testing error cannot be further reduced. "Early stopping" is an implicit way to implement regularisation[17].

Neural network generalisation can also be enhanced through a parsimonious network structure. A parsimonious network can be obtained in two ways. One is to first train a network with many hidden nodes and then reduce the number of hidden nodes through network pruning[18,19]. The other one is to start from a small number of hidden neurons and then gradually add on more hidden neurons through a sequential orthogonal training method[11,12].

Another very effective method for improving neural network generalisation capability is to first develop a group of neural networks and then combine them[20~25]. Mixture models such as Adaptive Mixture of Experts[22] and Hierarchical Mixture of Experts[26] use the divide and conquer approach where a mixture of experts compete to gain responsibility in modelling the output in a given input region. The system's output is obtained as a linear combination of the experts' output and the combination weights are computed as a function of the inputs. The different experts are usually trained on a single data set simultaneously by minimising a combined cost function and the final combination of the experts is determined by a gating module which is constructed in the same training session. Wolpert proposed a stacked generalisation technique for combining models to reduce the model generalisation errors[24]. A novel feature of stacked generalisation is that it attempts to simultaneously solve the problem of model selection and estimation of model combinations to improve model prediction. Cho and Kim used fuzzy logic to combine multiple neural networks[20] and showed significantly improved performance in handwriting character recognition. Breiman proposed a new method for aggregating multiple models using bootstrap re-samples of the training data, known as Bagging[27]. A set of bootstrap re-sampled data represent a sub-space of the original training data set. Models are developed on different bootstrap re-sampled data sets then combined together. The models can be neural network models[25] or other models such as principal component regression (PCR) or partial least square (PLS) models[28]. Since the models developed on bootstrap re-samples model the same relationship between the process inputs and outputs, they are usually highly correlated. Zhang et al. proposed using PCR to combine the individual networks and overcome the problem of severe correlations among the individual models[25].

Fig. 1 shows an aggregated neural network. The output of the aggregated neural network is a weighted combination of the outputs from the individual networks.

$$f(X) = \sum_{i=1}^{n} w_i f_i(X) \tag{1}$$

where $f(X)$ represents an aggregated neural network model, $f_i(X)$ is the ith neural network, w_i is the weight for combining the i-th neural network, X is a vector of neural network model inputs.

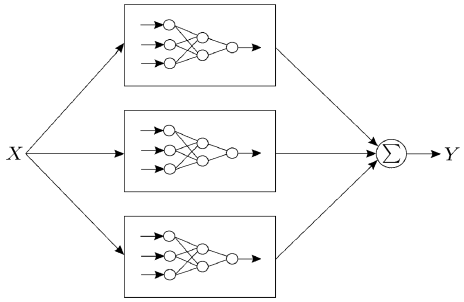

Fig. 1 An aggregated neural network

Let y be the expected model output and $\hat{y}$ the output of the i-th neural network. Then the outputs from all the n networks can be represented by the following matrix:

$$\hat{Y} = [\hat{y}_1 \quad \hat{y}_2 \quad \cdots \quad \hat{y}_n] \tag{2}$$

where each column represents a neural network model. The vector of the aggregated neural network model output, $\hat{y}_{agg}$, can be written as

$$\hat{y}_{agg} = \hat{Y}w = w_1\hat{y}_1 + w_2\hat{y}_2 + \cdots + w_n\hat{y}_n \tag{3}$$

The combination weights obtained from linear regression can be expressed as:

$$w = (\hat{Y}^{\mathrm{T}}\hat{Y})^{-1}\hat{Y}^{\mathrm{T}}y \tag{4}$$

Due to the severe correlation among the individual networks, $\hat{Y}^{\mathrm{T}}\hat{Y}$ will be singular or very close to singular. Therefore, the combination weights obtained from Eq.(4) will be very sensitive to computation errors or measurement noise.

The author proposed a method using PCR to obtain the combination weights in order to overcome this problem[25]. The matrix $\hat{Y}$ can be decomposed as a summation of several matrices of rank one:

$$\hat{Y} = t_1p_1^{\mathrm{T}} + t_2p_2^{\mathrm{T}} + \cdots + t_np_n^{\mathrm{T}} \tag{5}$$

In the above equation, t_i and p_i are, respectively, the i-th score vector and the i-th loading vector. The score vectors are mutually orthogonal and so as the loading vectors, which are furthermore of unit length. The first loading vector, p_1, defines the direction of the largest data variation. The first score vector, t_1, also known as the first principal component, represents the projection of $\hat{Y}$ on p_1. Therefore, the first principal component, $t_1 = \hat{Y}p_1$, is such a linear combination of the columns in $\hat{Y}$ that it can maximally explain the data variations. The second principal component, $t_2 = \hat{Y}p_2$, is such a linear combination of the columns in $\hat{Y}$ that it can maximally explain the data variations excluding those represented by the first principal component and is orthogonal to the first principal component. Principal components are arranged in the order of data variations that they can explain. Since the columns of $\hat{Y}$ are usually highly correlated, the first a few principal components can explain the major data variations in $\hat{Y}$.

The aggregating weights can then be obtained using PCR. Suppose that the first k principal components are used and they are represented by $T_k(T_k = \hat{Y}P_k)$, where $P_k = [p_1\ p_2\ \ldots\ p_k]$; then the aggregated neural network can be represented as:

$$\hat{y}_{agg} = T_k\theta = \hat{Y}P_k\theta \tag{6}$$

The least square estimation of the parameter θ can be obtained as:

$$\theta = (T_k^{\mathrm{T}} T_k)^{-1} T_k^{\mathrm{T}} y = (P_k^{\mathrm{T}} \hat{Y}^{\mathrm{T}} \hat{Y} P_k)^{-1} P_k^{\mathrm{T}} \hat{Y}^{\mathrm{T}} y \tag{7}$$

Thus the aggregating weights, w, obtained from PCR are given by

$$w = P_k \theta = P_k (P_k^{\mathrm{T}} \hat{Y}^{\mathrm{T}} \hat{Y} P_k)^{-1} P_k^{\mathrm{T}} \hat{Y}^{\mathrm{T}} y \tag{8}$$

The procedure for building an aggregated neural network model can be summarised as follows. First generate several replications of the original data for building a neural network model using bootstrap re-sampling[29]. Then develop a neural network model on each of the replications. Finally combine these individual neural network models using PCR.

In addition to possessing good generalisation capability, another advantage of aggregated neural network models is that they can provide prediction confidence bounds. Bootstrap re-sampling can be used to estimate the standard model prediction errors[29,30]. The prediction confidence bounds can be obtained from the standard model prediction errors. Model prediction confidence bounds offer additional information about the model predictions. Wider confidence bounds indicate that the associated model predictions are less reliable and vice versa. Process operator can accept or reject a particular model prediction based on the model prediction confidence bounds.

Tibshirani compares several methods for computing neural network model prediction confidence bounds[30] and points out that the bootstrap method is better than other methods. The bootstrap method for calculating model prediction confidence bounds can be summarised as:

Step 1. Generate B samples, each one of size n drawn with replacement from the n training observations $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. Denote the b-th sample by $\{(x_1^b, y_1^b), (x_2^b, y_2^b), \ldots, (x_n^b, y_n^b)\}$.

Step 2. For each bootstrap sample $b = 1, 2, \ldots, B$, train a neural network model. Denote the resulting neural network weights by W^b.

Step 3. Estimate the standard error of the ith predicted value by

$$\left\{ \frac{1}{B-1} \sum_{b=1}^{B} [y(x_i; W^b) - y(x_i; \cdot)]^2 \right\}^{1/2}$$

where $y(x_i; \cdot) = \sum_{b=1}^{B} y(x_i; W^b)/B$.

Step 4. The 95% confidence bounds can be obtained by taking plus and minus 1.96 times the standard error of the mean of the predicted value.

3 A batch polymerisation process

The simulated batch polymerisation reactor studied here is based on a pilot scale polymerisation reactor installed at the Department of Chemical Engineering, Aristotle University of Thessaloniki, Greece. The reaction is the free-radical solution polymerisation of methyl methacrylate (MMA) with a water solvent and benzoyl peroxide initiator. A schematic diagram of the reactor is shown in Fig. 2. The reactor is provided with a stirrer for thorough agitation of the reacting mixture. Heating and cooling of the reacting mixture is achieved by circulating water at an appropriate temperature through the reactor jacket. The reactor temperature is controlled by a cascade control system consisting of a primary PID and two secondary PI controllers. The reactor temperature is measured through a temperature sensor (TT) and fed back to the primary controller whose output is taken as the setpoint of the two secondary controllers. The two secondary controllers manipulate the cold water (CW) and the hot water (HW) flow rates so that the mixed water is circulated through the reactor jacket at appropriate temperatures. The temperature of the circulating water at the jacket exit is measured through a temperature sensor (TT) and fed back to the two secondary controllers.

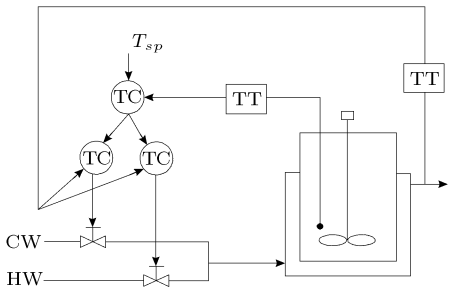

Fig. 2 A batch polymerisation process

A detailed mathematical model covering reaction kinetics and heat and mass balances has been developed for the bulk polymerisation of MMA[31,32]. Based on this model, a rigorous simulation program was developed and used as the real process to generate polymerisation data under different batch operating conditions and to test the developed control strategies.

4 Reliable optimal control of batch process

A problem in batch process optimal control is that, due to the presence of model-plant mismatches, the optimal control policy obtained from a model may not give optimal performance when applied to the actual process. Ruppen *et al.* proposed a method where a discrete probability distribution of some uncertain model parameters in a mechanistic model is assumed and used in a differential/algebraic optimisation problem[33]. A difficulty associated with this technique is that a mechanistic model needs to be first developed and the distribution of uncertain model parameters needs to be identified. Terwiesch *et al.* proposed a technique to optimise a probabilistic measure of success that accounts for possible uncertainties or variations in model parameters[34]. This technique also requires the probability distribution of uncertain model parameters. The technique is suitable for situations where a small number of parameters in a mechanistic model are uncertain and their probability distributions can be identified. An empirical data based model, such as a neural network model, usually contains a large number of model parameters and it is generally very difficult to identify the probability distributions of these model parameters.

The author proposed a method to improve the reliability of empirical model based batch process optimal control by incorporating the model prediction confidence bounds into the objective function[35]. The modified objective function penalises wide model prediction confidence bounds so that the obtained optimal control policy would give good control performance when applied to the actual process, *i.e.*, the obtained control policy is reliable.

In the batch polymerisation process, the batch duration is usually no more than 180 minutes. Here let the possible batch duration be from 60 to 80 min. Since polymer quality variables are typically difficult to be measured on-line, usually only a few samples of quality variables can be made within a batch. Here we assume that monomer conversion, number average molecular weight, and weight average molecular weight are measured every 20 min starting from 60 min into the reaction. Hence, there are up to 7 samples of polymer quality measurements within a batch. The control variables considered here are the initial reactor temperature setpoint for the time interval [0 min, 40 min] and the reactor temperature setpoints at the following time intervals [40 min, 60 min], [60 min, 80 min], ..., and [160 min, 180 min]. These reactor temperature setpoints form a control trajectory for the reactor.

Here an aggregated neural network is used to model the process and the model is of the following form:

$$Y(t_N) = f(I_0, U(t_N)) \tag{9}$$

where $Y(t_N) = [Conv(t_N)Mn(t_N)Mw(t_N)]^{\mathrm{T}}$, $U(t_N) = [T_{sp0}T_{sp1}T_{sp2}\dots T_{spN}]^{\mathrm{T}}$, T_{sp0} to T_{spN} are the trajectory of reactor temperature setpoints in the time interval $[0, t_N]$, I_0 is the initial initiator weight, and $Conv(t_N)$, $Mn(t_N)$, and $Mw(t_N)$ are the monomer conversion, the number average molecular weight, and weight average molecular weight at time t_N respectively. In the above model, t_N can take one of the following values: 60, 80, 100, 120, 140, 160, and 180 minutes. When using this model in optimal control, each of these values is considered as a possible batch ending time and the best batch ending time is selected based on the optimisation results.

Here simulated process data from 50 batches were produced. The control policies for these batches were obtained from Monte Carlo simulation. Noises were added to the data to represent the effect of measurement noise. Bootstrap re-sampling was used to produce 30 replications of these data. On each replication of the data, 7 neural networks were developed to model the polymer quality variables at the 7 possible batch ending times. Each network is a single hidden layer network with 10 hidden neurons. Network weights were initialised as random numbers within $(-0.1, 0.1)$ and networks were trained using Levenberg-Marquardt optimisation algorithm[36] with regularisation and "early stopping". The developed networks were combined using PCR. Further 20 batches of data were produced to evaluate the aggregated network model.

Fig. 3 gives the sum of squared errors (SSE) of individual networks on the training and testing

data and on the unseen validation data. It can be seen that the individual networks give inconsistent performance on the training and testing data and on the unseen validation data. This indicates that the unreliability and poor generalisation capability of the individual networks. The minimum SSE of the individual networks on the training and testing data is 18.0 while that on the unseen validation data is 19.0. The SSE of the aggregated network on the training and testing data is 9.8 while that on the unseen validation data is 13.8. Therefore, the aggregated neural network significantly enhanced model prediction accuracy.

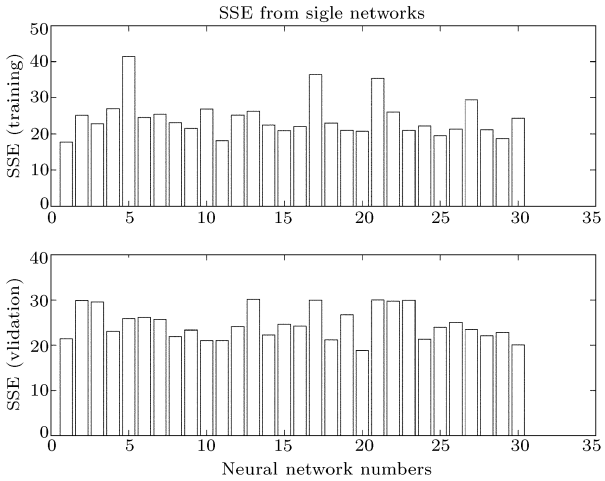

Fig. 3 Errors of individual networks

In order to enhance the reliability of neural network model based control, model prediction confidence bounds are introduced in the optimisation objective function as follows:

$$\begin{aligned} \min_{\boldsymbol{U},t_f}\ & J = [1 - Conv(t_f)]^2 + wt_f + \lambda^{\mathrm{T}}\sigma_e(t_f) \\ \text{s.t. } & 0.85 \leqslant Mn(t_f)/Mnd(t_f) \leqslant 1.15 \\ & 2 \leqslant Pd(t_f) \leqslant 3 \\ & 324 \leqslant u_i \leqslant 352, \quad 1 \leqslant i \leqslant 8 \end{aligned} \tag{10}$$

where $\boldsymbol{U} = [u_1 \quad u_2 \quad \ldots]^{\mathrm{T}}$ is a vector of control actions (reactor temperature setpoints), u_1 is the reactor temperature setpoint for the time interval [0 min, 40 min], u_2 to u_8 are the reactor temperature setpoints for the time intervals [40 min, 60 min], ..., and [160 min, 180 min], $Conv$ is the predicted monomer conversion, t_f is the batch ending time, w is a weighting factor for batch duration ($w = 0.0001\ \mathrm{min}^{-1}$ in this study), $\sigma_e = [\sigma_1\ \sigma_2\ \sigma_3]^{\mathrm{T}}$ is a vector of standard prediction errors in Mn, Mw, and $Conv$ respectively, $\lambda = [\lambda_1\ \lambda_2\ \lambda_3]^{\mathrm{T}}$ is a vector of weightings for σ_e, Mn is the predicted number average molecular weight, Mnd is the desired value of Mn, and $Pd(= Mw/Mn)$ is the polydispersity of the polymer.

This optimisation objective intends to maximise the monomer conversion with reduced batch time. Higher conversion means efficient utilisation of raw materials while short batch time represents more product being produced within a given time. The last term in the objective function is for penalising wide model prediction confidence bounds leading to reliable optimal control.

Since model-plant mismatches are unavoidable, the constraints on product quality in Eq.(10) should be soft constraints, which can be violated to certain extents. Here the hard constraints that must be satisfied are selected as:

$$0.8 \leqslant Mn(t_f)/Mnd(t_f) \leqslant 1.2 \tag{11}$$

$$1.8 \leqslant Pd(t_f) \leqslant 3.2 \tag{12}$$

The differences between soft and hard constraints represent a back off reflecting the extent of model-plant mismatches. If the model-plant mismatches are large, then the back off should be larger and vice versa.

Since the aggregated neural network model can only predict product quality at 20 min interval starting from 60 min into the reaction, here the possible batch ending time is selected from one of the following values: 60, 80, ..., and 180 minutes. When solving the optimisation problem, each of the possible batch ending time is used as the final batch time and the following optimisation problem is solved:

$$\begin{aligned} \min_{U} \; & J = [1 - Conv(t_f)]^2 + wt_f + \lambda^{\mathrm{T}} \sigma_e(t_f) \\ \text{s.t. } & 0.85 \leqslant Mn(t_f)/Mnd(t_f) \leqslant 1.15 \\ & 2 \leqslant Pd(t_f) \leqslant 3 \\ & 324 \leqslant u_i \leqslant 352, \quad 1 \leqslant i \leqslant 8 \end{aligned} \tag{13}$$

Then the objective functions at the 7 possible batch ending times are compared. The batch ending time corresponding to the lowest objective function values is selected as the best batch ending time.

Here Mnd is selected as 1.9×10^5g/mol representing a particular polymer product. Optimal reactor temperature control policy is obtained through solving this optimisation problem using the sequential quadratic programming (SQP) method.

The following cases were studied: Case 1 – $\lambda^{\mathrm{T}} = [0 \quad 0 \quad 0]$; Case 2 – $\lambda^{\mathrm{T}} = [0.05 \quad 0.05 \quad 0.05]$; Case 3 – $\lambda^{\mathrm{T}} = [0.1 \quad 0.1 \quad 0.05]$; Case 4 – $\lambda^{\mathrm{T}} = [0.15 \quad 0.15 \quad 0.05]$; Case 5 – $\lambda^{\mathrm{T}} = [0.2 \quad 0.2 \quad 0.05]$; and Case 6–single neural network model based optimal control. In the single neural network based optimal control, the optimisation objective function is similar to Eq.(10) but without the term for penalising wide model prediction confidence bounds. The single neural network used is the first network of the 30 individual networks. It can be seen from Fig. 3 that this network is among the best of the individual networks.

Fig. 4 shows the minimal objective function values for the 6 cases at different batch ending times. It can be seen from Fig. 4 that the best batch duration is 100 min in all case studies. Fig. 5 gives the control policy (reactor temperature setpoint) and reactor temperature in Cases 1, 5, and 6. It can be seen that the control policies are quite different. Fig. 6 shows the standard prediction errors in Case 1 to Case 5. Fig. 7 shows the actual (mechanistic model simulated) and neural network model predicted product quality variables in Case 1 to Case 6. The horizontal dotted lines and dash-dotted lines in Fig. 7 represent the hard and soft constraints respectively. The effect of penalising wide prediction confidence bounds is clearly seen from Figs. 6 and 7. In Case 1, due to no penalisation, the model prediction confidence bounds under the optimal control policy are quite wide. This means that the model predictions under this optimal control policy are not reliable and so is the optimal control policy. This is further verified by the simulation results given in Fig. 7. With this control policy implemented on the actual process (*i.e.* on the mechanistic model based simulation), the product quality can neither satisfy the hard constraints nor the soft constraints. Under this optimal control policy, the network model predicted Mn, Pd, and $Conv$ are 1.85×10^5g/mol, 3.0, and 0.92 respectively. However, the actual Mn, Pd, and $Conv$ under this "optimal" control are 1.39×10^5g/mol, 3.71, and 0.89 respectively. This indicates that the neural network model predicted performance is not realised on the actual process and, therefore, the optimal control policy is not reliable.

As the weightings on the standard prediction errors are increased, the aggregated neural network model prediction confidence bounds are becoming narrower (Fig. 6), and the actual final product quality moves toward the constraints (Fig. 7). In Case 2, the hard constraints on Mn are satisfied but those on Pd are still not satisfied. In Case 3, as the weighting on the standard prediction errors is further increased, the hard constraints on both Mn and Pd are satisfied. With the weightings on the standard prediction errors increased, the soft constraints on Mn and Pd are satisfied in Case 5. It can be seen from Fig. 6 that when the weightings on the standard prediction errors are increased to those of Case 3, the reductions in standard prediction errors become less significant. Therefore, the weights in Case 3 can be selected as the appropriate weights.

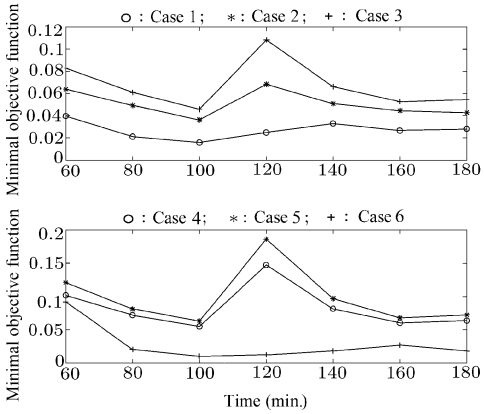

Fig. 4 Minimal objective function values at different batch ending times

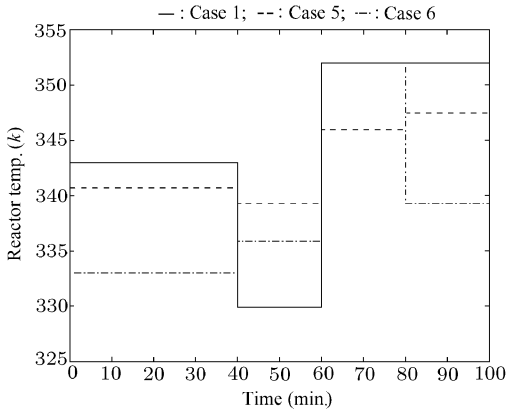

Fig. 5 Control policy and reactor temperature in Cases 1, 5 and 6

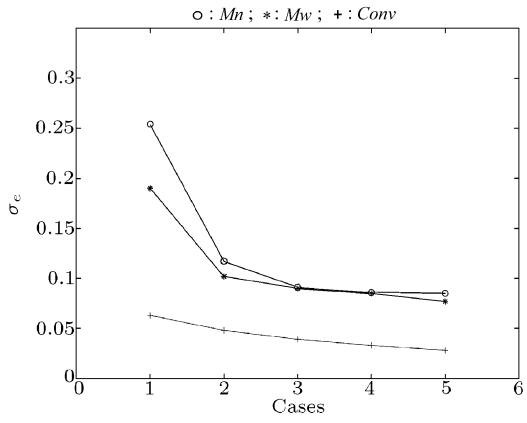

Fig. 6 Standard prediction errors in Case 1 to Case 5

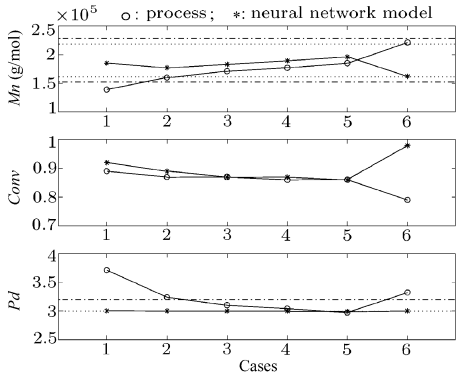

Fig. 7 Actual and neural network model predicted product quality variables

It can be seen from Fig. 7 that large differences exist between the single neural network predicted values and the actual values. The single neural network predicted final Mn is on the lower limit of the soft constraint, but the actual final Mn exceeds the upper limit of the soft constraint. Figs. 8 to 10 show, respectively, the trajectories of $Conv$, Mn, and Pd under the control profiles in Cases 1, 5, and 6. The horizontal dotted lines and dash-dotted lines in Figs. 9 and 10 represent the hard and soft constraints respectively. It can be seen from Fig. 10 that the values of Pd at the batch end in Cases 1 and 6 significantly exceed its hard constraints. This indicates that the optimal control in Cases 1 and 6 can lead to off specification products. Therefore these two "optimal" control policies are not optimal at all.

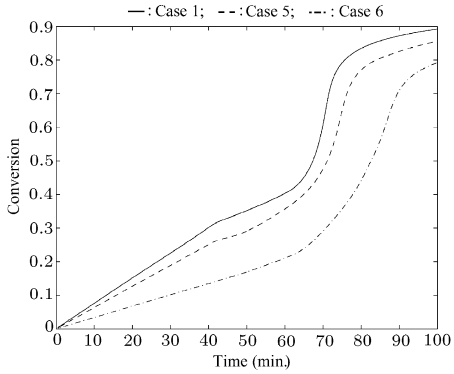

Fig. 8 Monomer conversions in Cases 1, 5 and 6

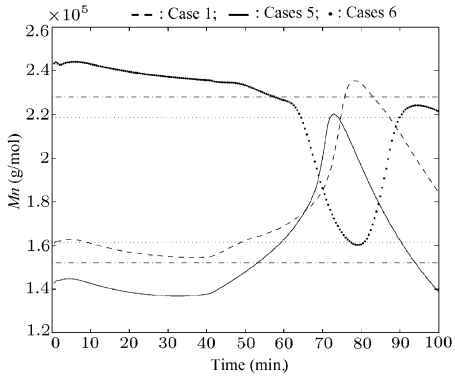

Fig. 9 Mn in Cases 1, 5 and 6

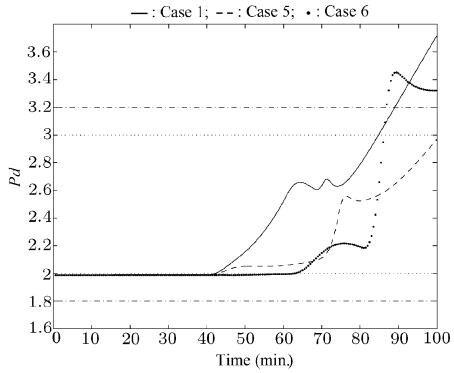

Fig. 10 Pd in Cases 1, 5 and 6

5 Iterative learning control of product quality

Due to the existence of model-plant mismatches and unknown disturbances, the optimal control policy calculated from a neural network model may not be optimal when applied to the actual process. The repetitive nature of batch processes allow the information of previous batch runs to be used to improve the performance of later batch runs.

The first order Taylor series expansion of Eq.(9) around a nominal control profile can be expressed as

$$\hat{y}(t_f) = f_0 \frac{\partial f}{\partial u_1} \triangle u_1 + \frac{\partial f}{\partial u_2} \triangle u_2 + \cdots + \frac{\partial f}{\partial u_N} \triangle u_N \tag{14}$$

where $\triangle u_1$ to $\triangle u_N$ are deviations in the control profile from the nominal control profile.

For the k-th batch, the actual product quality can be written as the model prediction plus an error term

$$\hat{y}_k(t_f) = \hat{y}_k(t_f) + e_k \tag{15}$$

where $y_k(t_f)$ and $\hat{y}_k(t_f)$ are, respectively, the actual and model predicted final product quality, and e_k is model prediction error.

The model prediction for the $(k+1)$-th batch can be represented using the first order Taylor series expansion based on the k-th batch:

$$\hat{y}_{k+1}(t_f) = \hat{y}_k(t_f) + \left.\frac{\partial f}{\partial u_1}\right|_{U_k} (u_1^{k+1} - u_1^k) + \cdots \left.\frac{\partial f}{\partial u_N}\right|_{U_k} (u_N^{k+1} - u_N^k) = \hat{y}_k(t_f) + G^{\mathrm{T}} \triangle U^{k+1} \tag{16}$$

where

$$\triangle U^{k+1} = [\triangle u_1^{k+1} \ \triangle u_2^{k+1} \ \ldots \ \triangle u_N^{k+1}]^{\mathrm{T}}$$

$$G^{\mathrm{T}} = \left[\left.\frac{\partial f}{\partial u_1}\right|_{U_k} \left.\frac{\partial f}{\partial u_2}\right|_{U_k} \cdots \left.\frac{\partial f}{\partial u_N}\right|_{U_k} \right]^{\mathrm{T}}$$

Suppose that the model prediction errors are the same for the k-th batch and the $(k+1)$-th batch; then the optimal control of the $(k+1)$-th batch can be represented as:

$$\min_{\triangle U^{k+1}} J = \| \hat{y}_k(t_f) + G^{\mathrm{T}} \triangle U^{k+1} + e_k - y_d \|_Q^2 + \| \triangle U^{k+1} \|_R^2 \tag{17}$$

where Q is the weighting for product quality control errors and R is the weighting for control action.

Let $\dfrac{\partial J}{\partial \triangle U^{k+1}} = 0$. Then the optimal control updating can be expressed as:

$$\triangle U^{k+1} = (GQG^{\mathrm{T}} + R)^{-1} GQ(y_d - \hat{y}_k(t_f) - e_k) \tag{18}$$

$$U^{k+1} = U^k + \triangle U^{k+1} \tag{19}$$

For a neural network model, the gradient of the model output with respect to the control policy U, G, can be calculated analytically. If the neural networks used are single hidden layer networks with the sigmoidal function as the activation function for hidden neurons and the linear function as the

activation function for the output layer neurons, then the element on the i-th row and j-th column of G, G_{ij}, can be computed as follows:

$$G_{ij} = \frac{\partial y_i}{\partial u_j} = \sum_{k=1}^{nh} \frac{\partial y_i}{\partial O}_k \frac{\partial O_k}{\partial u_j} = \sum_{k=1}^{nh} W_2^{k,i} O_k (1 - O_k) W_1^{j,k} \tag{20}$$

where nh is the number of hidden neurons, O_k is the output of the k-th hidden neuron, $W_2^{k,i}$ is the weight from k-th hidden neuron to the i-th output layer neuron, $W_1^{j,k}$ is the weight from the j-th input layer neuron to the k-th hidden neuron.

This method was applied to the batch polymerisation process. Here the desired product quality variables, Mn, Mw, and conversion, were selected as $yd = [2 \times 10^5 \quad 5 \times 10^5 \quad 1]^{\mathrm{T}}$, the weighting matrix Q was selected as $Q = [2 \times 10^{-5} \quad 8 \times 10^{-6} \quad 4]^{\mathrm{T}}$, the weighting matrix R was selected as $R = [0.1 \ 0.1 \ 0.1 \ 0.1]^{\mathrm{T}}$, and the reactor temperature was constrained to the range 320°K to 360°K.

Applying the optimal control policy calculated from the neural network model to the actual process (*i.e.* the mechanistic model based simulation), the resulting final product quality are as follows: $Mn(t_f) = 1.3898 \times 10^5$g/mol, $Mw(t_f) = 5.1655 \times 10^5$g/mol, and $X(t_f) = 0.8924$. It can be seen that significant differences exist between the desired product quality and actual product quality. This is due to the model-plant mismatches of the neural network model.

Fig. 11 gives the results of applying iterative learning control. It can be seen from Fig. 11 that the control performance was significantly improved in the 2nd batch and further improved in the 3rd batch. The control performance is still improving after the 3rd batch but not very significantly. This indicates that the iterative learning control is almost converged after 3 batches. The final product quality variables at the 10th batch are: $Mn(t_f) = 1.8735 \times 10^5$g/mol, $Mw(t_f) = 4.9134 \times 10^5$g/mol, and $X(t_f) = 0.8432$. This indicates that the iterative learning control can effectively overcome the detrimental effects of model-plant mismatches on the optimal control of batch processes. Fig. 12 gives the final product quality variables at the end of each batch.

In order to demonstrate the effectiveness of iterative learning control on overcoming the detrimental effects of unknown disturbances, the initial initiator mass was reduced from its normal value of 2.5g to 1.9g from batch 11 onward so as to represent the effect of reactive impurities. A practically very significant problem in batch polymerisation processes is the presence of reactive impurities[37~39]. Reactive impurities are mainly due to the recycle of unreacted monomers and usually present as traces of inhibitors and oxygen. It is shown in [37] that reactive impurities in polymerisation can rapidly consume free radicals and prevent the growth of polymer particles. The effect of reactive impurities can usually be simulated as a step decrease in the initial initiator mass.

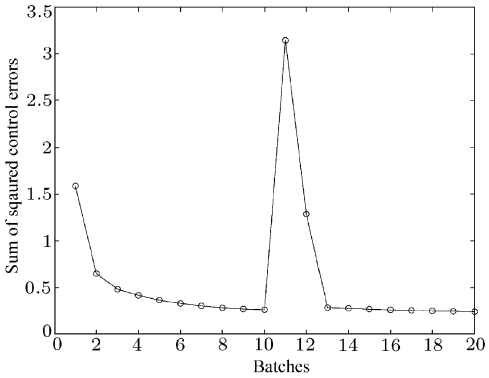

Fig. 11 Sum of squared control errors

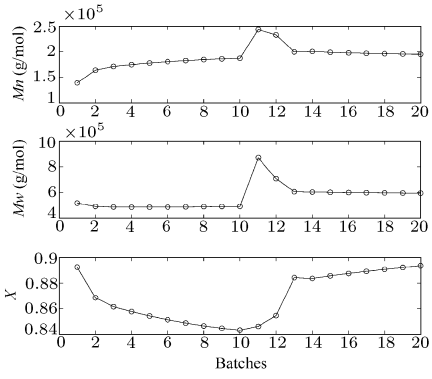

Fig. 12 Product quality variables

Due to the presence of reactive impurities from the 11th batch onward, the control policy obtained at the 10th batch is no longer optimal on the 11th batch. Fig. 11 shows that large control errors were obtained when the control policy obtained from the 10th batch was applied to the 11th batch. The final product quality variables at the end of the 11th batch are as follows: $Mn(t_f) = 2.4316 \times 10^5$g/mol, $Mw(t_f) = 8.7177 \times 10^5$g/mol, and $X(t_f) = 0.8462$, which are much worse than those from the 10th batch. The process operation was significantly improved in the 12th and the 13th batches through

iterative learning control. The final product quality variables at the end of the 13^{th} batch are as follows: $Mn(t_f) = 1.9959 \times 10^5$g/mol, $Mw(t_f) = 6.0527 \times 10^5$g/mol, and $X(t_f) = 0.8842$, which are very close to the desired product quality. This demonstrates that iterative learning control can effectively overcome the problems of unknown disturbances in batch processes.

Apart from neural network model based iterative learning control, the author and his co-workers also proposed techniques for iterative learning control of batch processes based on linearised models directly identified from process operation data[40], iterative learning control of batch processes based on incrementally updated linearised models identified from minimum amount of process operation data[41], and iterative optimisation of batch processes through iterative modification of neural network model predictions[42].

6 Conclusions

Neural network based techniques for the modelling and optimal control of batch processes are presented in this paper through applications to a simulated batch polymerisation reactor. It is shown that bootstrap aggregated neural networks have good generalisation capability and can provide model prediction confidence bounds. Aggregated neural networks can be effectively used in building models for batch processes from process operational data. Incorporating the neural network model prediction confidence bounds into the batch process optimisation objective function can significantly enhance the reliability of neural network model based optimal control. Utilising the repetitive nature of batch processes, a neural network model based iterative learning control strategy is presented in this paper and it is demonstrated that this control strategy can effectively overcome the problems due to model-plant mismatches and unknown disturbances.

References

1 Bonvin, D. Optimal operation of batch reactors: a personal view. *Journal of Process Control*, 1998, **8**: 355∼368

2 Park S, Ramirez W F. Optimal production of secreted protein in fed-batch reactors. *AIChE Journal*, 1988, **34**: 1550∼1558

3 Thomas I M, Kiparissides C. Computation of the near-optimal temperature and initiator policies for a batch polymerisation reactor. *Canadian Journal of Chemical Engineering*, 1984, **62**: 284∼291

4 Cybenko G. Approximation by superposition of a sigmoidal function. *Math. Control Signal Systems*, 1989, **2**: 303∼314

5 Girosi F, Poggio T. Networks and the best approximation property. *Biological Cybernetics*, 1990, **63**: 169∼179

6 Tian Y, Zhang J, Morris A J. Modelling and optimal control of a batch polymerisation reactor using a hybrid stacked recurrent neural network model. *Ind. Eng. Chem. Res.*, 2001, **40**(21): 4525∼4535

7 Tian Y, Zhang J, Morris A J. Optimal control of a fed-batch bioreactor based upon an augmented recurrent neural network models. *Neurocomputing*, 2002a, **48**: 919∼936

8 Tian Y, Zhang J, Morris A J. Optimal control of a batch emulsion copolymerisation reactor based on recurrent neural network models. *Chemical Engineering and Processing*, 2002b, **41**(6): 531∼538

9 Xiong Z, Zhang J. Modelling and optimal control of fed-batch processes using a novel control affine feedforward neural network. *Neurocomputing*, 2004, **61**: 317∼337

10 Su H T, McAvoy T J, Werbos P. Long-term prediction of chemical processes using recurrent neural networks: a parallel training approach. *Industrial & Engineering Chemistry Research*, 1992, **31**: 1338∼1352

11 Zhang J, Morris A J. A sequential learning approach for single hidden layer neural networks. *Neural Networks*, 1998, **11**(1): 65∼80

12 Zhang J, Morris A J, Martin E B. Long term prediction models based on mixed order locally recurrent neural networks. *Computers & Chemical Engineering*, 1998, **22**: 1051∼1063

13 Elman J L. Finding structures in time. *Cognitive Science*, 1990, **14**: 179∼211

14 Morris A J, Montague G A, Willis M J. Artificial neural networks: studies in process modelling and control, Chem. Eng. Res. Des., 1994, **72**: 3∼19

15 Bishop C. Improving the generalisation properties of radial basis function neural networks. *Neural Computation*, 1991, **3**(4): 579∼588

16 Chen S, Chng E S, Alkadhimi K. Regularised orthogonal least squares algorithm for constructing radial basis function networks. *International Journal of Control*, 1996, **64**: 829∼837

17 Sjoberg J, Zhang Q, Ljung L, Benveniste A, Delyon B, Glorennec P Y, Hjalmarsson H, Judisky A. Nonlinear black-box modelling in system identification: a unified overview. *Automatica*, 1995, **31**: 1691∼1724

18 Giles C L, Omlin C W. Pruning recurrent neural networks for improving generalisation performance. *IEEE Transactions on Neural Networks*, 1994, **5**: 848∼851
19 Reed R. Pruning algorithms – a survey. *IEEE Transactions on Neural Networks*, 1993, **4**: 740∼747
20 Cho S B, Kim J H. Multiple network fusion using fuzzy logic. *IEEE Transactions on Neural Networks*, 1995, **6**: 497∼501
21 Hashem S. Optimal linear combinations of neural networks. *Neural Networks*, 1997, **10**: 599∼614
22 Jacobs R A, Jordan M I, Nowlan S J, Hinton G E. Adaptive mixture of local experts. *Neural Computation*, 1991, **3**: 79∼87
23 Sridhar D V, Seagrave R C, Bartlett E B. Process modelling using stacked neural networks. *AIChE Journal*, 1996, **42**: 2529∼2539
24 Wolpert D H. Stacked generalization. *Neural Networks*, 1992, **5**: 241∼259
25 Zhang J, Morris A J, Martin E B, Kiparissides C. Inferential estimation of polymer quality using stacked neural networks. *Computers & Chemical Engineering*, 1997, **21**: 1025∼1030
26 Jordan M I, Jacobs R A. Hierarchical mixture of experts and the EM algorithm. *Neural Computation*, 1994, **6**: 181∼214
27 Breiman, L. Bagging Predictors. *Machine Learning*, 1996, **24**: 123∼140
28 Ahmed M, Zhang J. Improved inferential feedback control through combining multiple PCR models. In: Proceedings of The 2003 IEEE International Symposium on Intelligent Control, Houston, Texas, U.S.A., 2003, **5-8**: 78∼883
29 Efron B, Tibshirani R. *An Introduction to Bootstrap*. London: Chapman and Hall, 1993
30 Tibshirani R. A comparison of some error estimates for neural network models. *Neural Computation*, 1996, **8**: 152∼163
31 Achilias D, Kiparissides C. Development of a general mathematical framework for modelling diffusion-controlled free-radical polymerisation reactions. *Macromolecules*, 1992, **25**: 3739∼3750
32 Penlidis A, Ponnuswamy S R, Kiparissides C, O'Driscoll K F. Polymer reaction engineering: modelling considerations for control studies. *Chemical Engineering Journal*, 1992, **50**: 95∼107
33 Ruppen D, Benthack C, Bonvin D. Optimization of batch reactror operation under parametric uncertainty – computational aspects. *Journal of Process Control*, 1995, **5**: 235∼240
34 Terwiesch P, Ravemark D, Schenker B, Rippin D W T. Semi-batch process optimization under uncertainty: theory and experiments. *Computers & Chemical Engineering*, 1998, **22**: 201∼213
35 Zhang J. A reliable neural network model based optimal control strategy for a batch polymerisation reactor. *Ind. Eng. Chem. Res.*, 2004, **43**: 1030∼1038
36 Marquardt D. An algorithm for least squares estimation of nonlinear parameters. *SIAM J. Appl. Math.*, 1963, **11**: 431∼441
37 Penlidis A, MacGregor J F, Hamielec A E. Effect of impurities on emulsion polymerisation: case I Kinetics. *Journal of Applied Polymer Sciences*, 1988, **35**: 2023∼2038
38 Kiparissides C. Polymerisation reactor modelling: A review of recent developments and future directions. *Chemical Engineering Science*, 1996, **51**: 1637∼1659
39 Zhang J, Morris A J, Martin E B, Kiparissides C. Estimation of impurity and fouling in batch polymerisation reactors through the application of neural networks. *Computers and Chemical Engineering*, 1999, **23**, 301∼314
40 Xiong Z, Zhang J. Product quality trajectory tracking in batch processes using iterative learning control based on time-varying perturbation models. *Ind. Eng. Chem. Res.*, **42**: 6802∼6814
41 Xiong Z, Zhang J. Batch-to-batch optimal control of non-linear batch processes based on incrementally updated models. *IEE Proceedings – Control Theory and Applications*, 2004, **151**(2): 158∼165
42 Xiong Z, Zhang J. Batch-to-batch iterative optimisation control based on recurrent neural network models. *Journal of Process Control*, 2005, **15**: 11∼21
43 Ellis M F, Taylor, T W, Gonzalez V, Jensen K F. Estimation of the molecular weight distribution in batch polymerisation. *AIChE Journal*, 1988, **34**: 1341∼1353

Jie Zhang Lecturer in the School of Chemical Engineering & Advanced Materials, University of Newcastle, England. He received his bachelor degree in control engineering from Hebei University of Technology, Tianjin, P.R. China, in 1986 and his Ph.D. degree in control engineering from City University, London, in 1991. His research interests include: neural networks, neuro-fuzzy systems, fault detection and diagnosis, intelligent control systems, genetic algorithms, optimal control of batch processes, and multivariate statistical process control. He has published over 140 papers in international journals, books, and conferences. He served as a reviewer for many prestigious international journals including IEEE Transactions on Neural Networks, IEEE Transactions on Fuzzy Systems, Neural Networks, Automatica, Chemical Engineering Science, and IEE Proceedings. He is a senior member of IEEE. He is on the editorial board of the journal Neurocomputing published by Elsevier.