Temperature control of the batch polypropylene Reactor by ADRC

Yongqing Wang and Xuefeng Zhu

*(College of Automation Science & Engineering, South China University of Technology
Guangzhou , China)*

Abstract: A new control method called Active-Disturbance Rejection Controller (ADRC) is proposed for temperature control of a batch polypropylene reactor in this paper. This controller is mainly composed of three parts, i.e. “extended state observer” (ESO), input reference signal tracking-differentiator (TD) and a non-linear state error feedback (NLSEF) control law. The simulation results have shown that ADRC can obtain quite good performances with the process uncertain. As the control algorithm in DCS ADRC is developed and tested for a batch polypropylene reactor in a local Petro-Chemical plant. The experiment results have indicated that the controller can give much better dynamic responses than the other control algorithms conducted several years ago.

Key words: Polypropylene reactor, Temperature control, ADRC (Active-Disturbance Rejection Control)

1. INTRODUCTION

Batch polymerisation reactors are still extensively used to produce useful products, due to their production flexibility and similarity in principle to the laboratory reactors. To increase product quality and insure the reproducibility, it is necessary to improve the automation level of such processes. The dynamic characteristics of batch polymerisation reactors would considerably change with the progress of reaction because the reactors have strong non-linearity and the uncertainties caused by process raw materials, catalysts and so on. Therefore the control of the polypropylene rectors are very difficult in practice. In this case, it is important to design the Advanced Process Control system (APC) for the reactor control. The APC must be robust to the process non-linearity and uncertainty.
Several applications of adaptive controllers to the control of polymerisation process are reported (Embiricu.M,*et al*,1996). However, they need some assumptions for noise, disturbance and manipulated variables in order to estimate process model parameters. Recently a number of model predictive controllers which can treat constraints on manipulated variables and provide more economical operation have been reported. (Zoltan Nagy and Serban Agachi, 1997). But they can not consider the uncertainty of process parameters and the dynamics change caused by non-linearity.

In this paper, a new type of robust non-linear control strategy i.e. Active Disturbance Rejection Controller (ADRC) (Han 1998) is proposed to control the temperature of a batch polypropylene reactor.
The whole paper is organized as follows. The control problem is described in Section 2, section 3 is related to the design of the non-linear controller ADRC, including a simulation example. In section 4 the conducted experiments and results in the batch polypropylene reactor of Guangdong Petro-Chemical plant are presented..

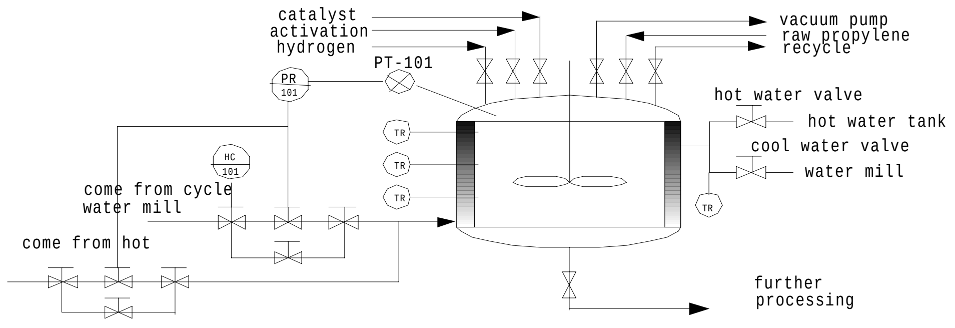


Fig.1. The sketch diagram of the batch polypropylene reactor

2. FORMULATION OF CONTROL PROBLEM

2.1 Process Description

The Fig.1 shows the sketch diagram of a batch polypropylene reactor that is under control. The batch stirred tank reactor is used to produce polypropylene. The operation of the whole process generally consists of the following steps:
As the initial step, the propylene, surfactants, initiators and monomer mixture are charged to the reactor. After this initial stage, the reaction mixture is heated to the desired temperature of polymerisation. At the third stage the polymerisation is going on, which generally will take about 3~4 hours. The last step is to cool down the reaction mixture to the temperature required for further processing,, then unload the all materials from the reactor.

2.2 Control Scheme

Since the whole process need heating and cooling respectively depending on what stage the process is in the split control scheme is applied.
When the controller output is between 0 and 50%, the controller is in cooling mode, the controller output is used to control the cooling water Sc valve. While the controller output is in the range of 50 and 100%, the controller is in heating mode, the controller is used to control the heating water valve Sh,.
When the heating water valve is open, the heating water is flowed directly into the jacket, the temperature of reaction mixture is then heated to nearly 40℃,at this point the reaction transits from endothermic to exothermic, correspondingly Sh is close and Sc is open, the temperature of the reactor is controlled to track the profile required by products quality via adjusting the flow of cooling water.

2.3.The process characteristics and control requirements

The polymerisation of propylene presents the following characteristics:

- ✧ The index of the product include: average molecular weight, molecular weight distribution (NWD), particle diameter, particle size distribution and porosity. A proper temperature would keep these index within the good range.
- ✧ The process is of batch type and the physical-chemical properties of the mixture are changing during the batch, thus the temperature control is fairly difficult.
- ✧ The disturbances often occur during the batch cycle. The process is of high nonlinear as far as the dynamic behaviour is concerned.

The control requirement imposed by the process engineer is that the maximum deviation of the temperature is within ±0.5℃ around the set point in order to ensure the good quality of the product. Therefore the previous feature of the process require advanced control method.

3. ACTIVE DISTURBANCE REJECTION CONTROLLER (ADRC)

3.1 The structure of ADRC

ADRC is developed by Professor Han in 1998. (Han 1998) In the following the scheme and control algorithm of ADRC is depicted.
In order to control a class of uncertain system:

$$y^{(n)} = f(y, \dot{y}, \cdots, y^{(n-1)}, t) + w(t) + u(t) \qquad (1)$$

where $f(y, \dot{y}, \cdots, y^{(n-1)}, t)$ represents uncertain function, $w(t)$ is an unknown disturbance , $u(t)$ is the control variable and y is the measurable state variable. The structure of non-linear active disturbance rejection controller is shown in Fig.2.

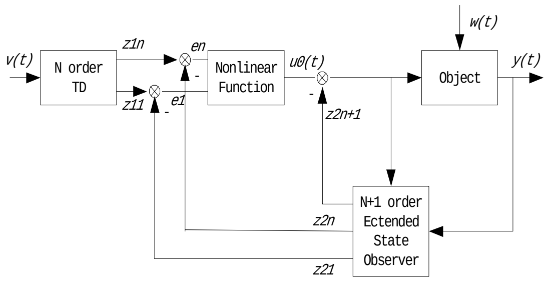


Fig.2 .The structure of ADRC

ADRC is composed of three parts: Tracking--Differentiator (TD), Extended State Observer (ESO) and Non-linear State Error Feedback (NLSEF) control law.
TD is a dynamic system where the response of the system can be designed for tracking the continuous and differential signal of the given input reference signal. Extended State Observer is used to estimate the state variables of the process and the estimation of the total disturbances. If the parameters and functions of the ADRC are properly chosen the controller is able to drive the state trajectory to the desired reference signal.

3.2 ADRC algorithm implementation

For a process of second order, the discrete algorithm of ADRC is as follows:

Tracking-Differentiator (TD)

The discrete form of TD (Han Jingqing and Yuan Lulin, 1999; Han Jingqing and Wang Wei, 1994) is

$$\begin{cases} x_1(k+1) = x_1(k) + Tx_2(k) \\ x_2(k+1) = x_2(k) \\ \qquad + Tfst(x_1(k), x_2(k), v(k), r, h) \end{cases} \qquad (2)$$

where T is sampling period, $v(k)$ is the input signal at time k, r determines the tracking rate, while h influences the filter effect when input signal is pollute by noise. The function of fst is defined as follows:

$$\delta = rh\,,\ \ \delta_0 = \delta h\,,\ \ y = x_1 - u + hx_2\,,$$

$$a_0 = \sqrt{\delta^2 + 8r\,|\,y\,|} \tag{3}$$

$$a = \begin{cases} x_2 + y/h, |\,y\,| \le \delta_0 \\ x_2 + 0.5(a_0 - \delta)sign(y), |\,y\,| > \delta_0 \end{cases} \tag{4}$$

$$fst = \begin{cases} -ra/\delta, |\,a\,| \le \delta \\ -rsign(a), |\,a\,| > \delta \end{cases} \tag{5}$$

If we properly select the parameter r, TD can obtain the continuous and differential signal of the input by tracking the input reference signal $v(t)$.

Extended State Observer (ESO)

Defining non-linear function as

$$fal(x, a, \delta) = \begin{cases} |\,x\,|^a\ sign(x), |\,x\,| \ge \delta \\ x/\delta^{1-a}, |\,x\,| < \delta \end{cases} \tag{6}$$

then the equation of ESO is

$$\begin{cases} e = z_1(k) - y(k) \\ z_1(k+1) = z_1(k) \\ \qquad + T[z_2(k) - \beta_{01} fal(e(k), a_1, \delta)] \\ z_2(k+1) = z_2(k) + \\ T[z_3(k) - \beta_{02} fal(e(k), a_2, \delta) + bu(k)] \\ z_3(k+1) = z_3(k) \\ -T\beta_{03} fal(e(k), a_2, \delta) \end{cases} \tag{7}$$

Appropriately select the parameters of ESO: $\{a_0, a_1, a_2, \delta, \beta_{01}, \beta_{02}, \beta_{03}, b\}$, z_1 and z_2 can track the system state variables y and $\dot{y}$, while z_3 can estimate $f(y, \dot{y}, w(t))$. (Han J, 1995)

Non-linear State Error Feedback (NLSEF) control law

The final control law consists of the following equations:

$$\begin{cases} e_1 = v_1(k) - z_1(k) \\ e_2 = v_2(k) - z_2(k) \\ u_0 = \beta_1 fal(e_1, \alpha_1, \delta_1) \\ \qquad + \beta_2 fal(e_2, \alpha_2, \delta_1) \\ u(k) = u_0 - z_3(k)/b \end{cases} \tag{8}$$

where e_1, e_2 is error and derivative of error between the outputs of TD and system output, b are modulation coefficient of the function $fal(e, \alpha, \delta)$. As a matter of fact, $f(y, \dot{y}, t) + w(t)$ is considered to be the extended state variable of uncertain system. If z_3 successfully converges to $f(y, \dot{y}, t) + w(t)$, it is possible to realize the state feedback and model, external disturbance compensation. If the parameters and function of the ADRC are suitably chosen, the system tracking,, regulation and stability can be guaranteed and the control variable $u(t)$ is able to drive the state trajectory to the desired reference signal.

3.3 Simulation example

Assumed a time variable system is as follows:

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = sign(\sin t) + u(t) \end{cases} \tag{9}$$

The reference signal is a square wave the amplitude of which is 5 and the frequency is 0.1 Hz. To control the output of the system for tracking the reference signal, ADRC is applied to the system as shown in Fig.2, The parameters of ADRC are tuned as follows:

$$\{r, h, T, a_0, a_1, a_2, \delta, \beta_{01}, \beta_{02}, \beta_{03}, \beta_1, \beta_2, \alpha_1, \alpha_2, d, b\}$$
$$= \{10, 0.01, 0.01, 0.0.5, 0.25, 100, 65, 80, 80, 300, 0.75, 1.25, 0.1, 1\}$$

The control performance was studied by simulation and the results are presented in Fig 3.a). From the figure, it can be observed that the algorithm can give good results for the non-linear time variable process.

If the process is changed as follows:

$$\begin{cases} \dot{x}_1(t) = x_2(t) \\ \dot{x}_2(t) = 15sat(\cos t) + u(t) \end{cases} \tag{10}$$

Keep the parameters and control scheme unchanged , Fig.3.b) illustrates the simulation result. Compare Fig.3.a) with Fig.3.b), it can be seen that the process response is almost the same, thus proving that ADRC is robust for the process change.

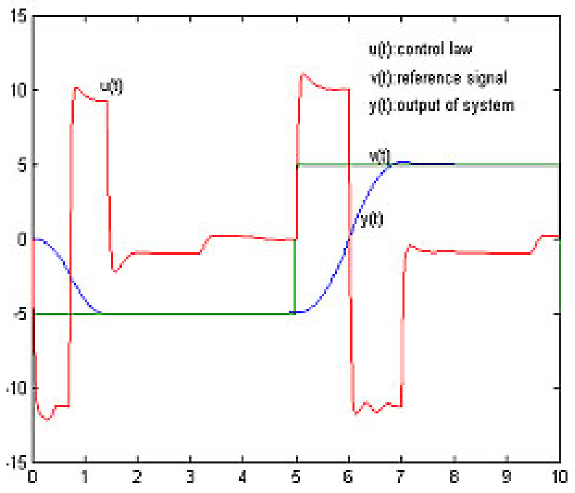


Fig. 3.a) Simulation results for system (9)

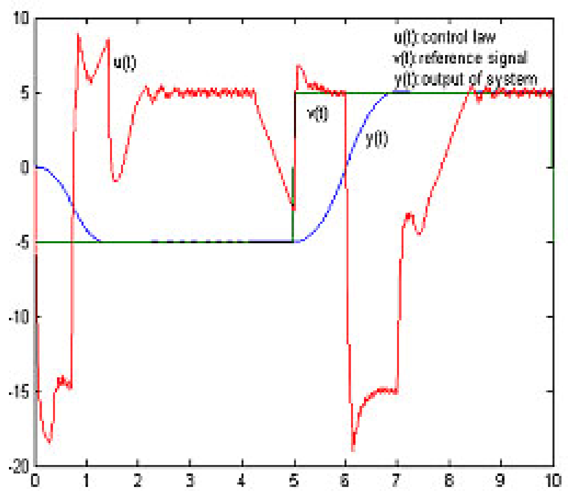


Fig. 3.b) Simulation results for system (10)

4. EXPERIMENTAL RESULTS

To verify the performance of the control based on ADRC, the tested reactor is controlled by a DCS system. ADRC control algorithm is programmed and inserted to the configuration software of DCS for the temperature control of the batch reactor. To ensure operating safety of the plant, an AUTO/MAN switch is programmed in the DCS configuration and monitoring software, When the switch is in AUTO, the plant is controlled by ADRC, while switch is to set to MAN, the plant is controlled by PID (Actually the reactor is controlled by the operators in heating stage).

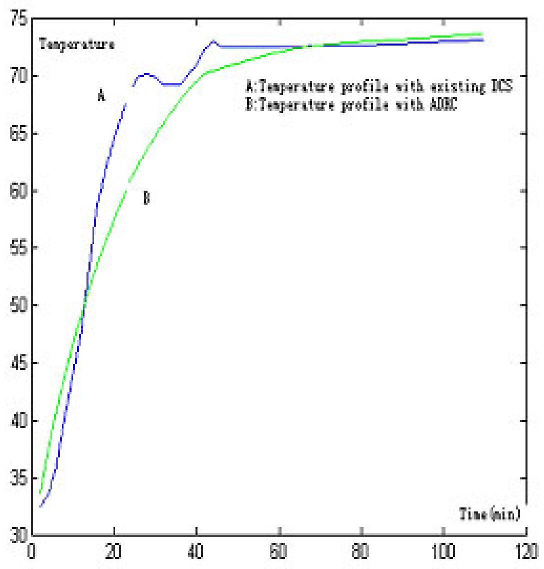


Fig.4 Responses of the temperature control in the batch reactor

In Fig. 4, the curve A shows a typical actual temperature response by PID in the rising stage of temperature. (Actually the process is manipulated by skill operators because conventional PID controller can not control the process well). Here it is shown that the temperature response is somewhat oscillated and has a little overshoot. One of the experimental result with ADRC as control algorithm is shown as Curve B. Comparing the two curves A and B it is easy to find the control performance by ADRC is obviously improved . The commissioning of the control test was lasted for nearly three month. ADRC performed fairly well in this period. Now the operating engineers and operators have completely accepted the control strategy.

5. CONCLUSION

In this paper a new robust non-linear controller ADRC is implemented for the temperature control of a batch polypropylene reactor. The Extended state observers are used to implement the state estimation and realize state feedback for the external disturbance compensation. The non-linear control law is to drive the state trajectory to the desired reference signal. ADRC has been programmed and tested in a chemical plant for controlling the reactor temperature. Both the simulation and experimental results have shown that the performance of ADRC is satisfactory. Comparing it with other control algorithms, the outstanding feature of ADRC is that it is robust to process variation and non-linearity. Besides that, it doesn’t require the accurate process model. That will provide a new and hopeful choice for controlling the difficult processes in industries.

REFERENCES

Embiricu M, et al (1996), A Survey of advanced Control of Polymerization Reactors, *polym. Eng.Sci*. **vol. 36**,pp433-438

Han Jingqing and Yuan Lulin (1999), The Discrete Form of Tracking-Differentiator, *J. Sys. Sci & Math. Scis*, **vol.19**, pp268-273, (in Chinese)

Han Jingqing and Wang Wei (1994), Nonlinear Tracking-Differentiator, *J. Sys. Sci & Math. Scis*, **vol.14**, pp.177-183, (in Chinese)

Han J (1998). Active Disturbance Rejection Controller and It’s Application, *Control and Decision*, **vol.13**, pp19-23, (in Chinese)

Han J (1995). The Extended State Observer of a Class of Uncertain Systems, *Control and Decision*, **vol.10**: pp85-88 (in Chinese)

Zoltan Nagy and Serban Agachi (1997), Model Predictive Control of a PVC batch reactor, *Computer & chem. Eng.*,**vol.21**,No.6,pp571-591

FERMENTATION BATCH PROCESS MONITORING BY STEP-BY-STEP ADAPTIVE MPCA

Ning He, Lei Xie, Shu-qing Wang, Jian-ming Zhang

National Key Laboratory of Industrial Control Technology,
Zhejiang University, Hangzhou 310027, People's Republic of China

Abstract: Multi-way principal component analysis (MPCA) has been successfully applied to the monitoring of batch and semi-batch processes in most chemical industry. A new approach is presented to overcome the method MPCA's need for estimating or filling in the unknown part of the process variable trajectory deviations from the current time until the end. The approach is based on the Multi-block PCA method and processes the data in a sequential and adaptive manner. The adaptive rate is easily controlled by a parameter that represents the similarity between current and past data. The method is evaluated on industrial fermentation process data and is compared to the traditional MPCA. The method may have significant benefit when monitoring multi-stage batch process where the latent vector structure can change at several points during the batch. Copyright © 2003 IFAC

Keyword: Statistical Process Control, Principal Component Analysis, Adaptive Algorithm

1. INTRODUCTION

Batch and semi-batch process play an important role in the chemical industry due to their low volume and high value products. In most of these processes product quality variables are only measured after the end of each batch, often hours later in a quality control laboratory. This makes it difficult to control the product quality or to monitor the progress of these batch processes. MacGregor (Kosanovich *et al*., 1994, Nomikos *et al* ,1994, Stefan Rännar, 1998), have presented very powerful process analysis, monitoring and diagnostic procedures which utilize these process variables trajectory data. These procedures, based on multi-way PCA (MPCA) method (Wold *et al*, 1982, 1994a), are now being widely adopted by the batch chemical industry.

The methods have proven to be very efficient for analysis historical data from past production and diagnosing operating problems. But when monitoring a new batch, the MPCA model, which assumes the complete history of the batch data, cannot be used directly. At any point during the batch, data on the deviations of the variables from its average trajectories for the remainder of the batch is not yet available. MacGregor and Nomikos (Nomikos *et al*, 1995) have proposed several approaches to handle this problem that have worked quite well in practice. However, it would be nice to have MPCA monitoring approaches that do not depend on having to fill in these missing data.

In this paper, we present a modification of the monitoring method that does not require estimates of the data for the uncompleted portion of the batch. The approach is based on a multi-block PCA algorithm that processes the data in a series manner. When monitoring future batches, one need only store the loading matrices for the local model at each point in time and the score vector from past. This step-by-step adaptive approach only requires fewer latent variable dimensions and appears to work as well as the existing methods. Application in monitoring an industrial fermentation process reveals that the proposed method gives more objective appraisal for new batch and may offer potential advantages in situation where the batch has several stages.

2. MULTI-WAY PRINCIPAL COMPONENT ANALYSIS

Multi-way principal component analysis (MPCA) (Nomikos *et al,* 1994) is an extension of conventional PCA to handle data in three-dimensional arrays. These data, which are collected from batch processes, are organized into an array $\boldsymbol{X}$ of three-dimension ($I*J*K$), where, I is the number of batches, J is the number of variables, and K is the number of time samples over the duration of the batch. As illustrated in Fig.1, the array X can be unfolded in such a way as to put each of its vertical slices ($I*J$) contain the observed variables for all batches at a given time interval side by side to the right. The result, a wide and short two-dimensional matrix has dimensions ($I*JK$). MPCA is equivalent to performing ordinary PCA on unfolded $\boldsymbol{X}$ and it explains the variation of variables about their mean trajectories.

MPCA decomposes X as follow,

$$\boldsymbol{X} = \sum_{r=1}^{R} \boldsymbol{t}_r \otimes \boldsymbol{p}_r + \boldsymbol{E} \,, \tag{1}$$

where R is the number of principal components used in the analysis, $\boldsymbol{t}$ is score vectors and $\boldsymbol{p}$ is loading matrices. $\boldsymbol{E}$ is residual matrix. It accomplishes this decomposition in accordance with the principles of PCA and separates the data in an optimal way into two parts.

This decomposition represents a new coordination system obtained by rotating the original variables and projecting the data into the reduced space defined by the first few principal components, where the data are described adequately and in a simpler and more meaningful way (Wold *et al*, 1978). By doing this, MPCA utilizes not just the magnitude of the deviation of each variable from its mean trajectory but also the correlations among them. The appropriate number of principal components may be determined by cross- validation (Jackson.,1991).

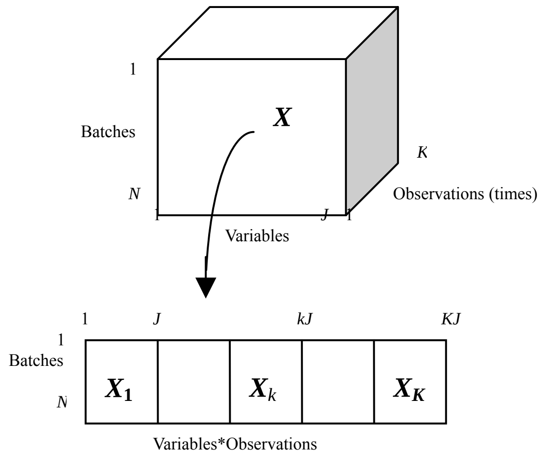


Fig.1. Overview of the MPCA method.

The MPCA algorithm derives directly from the nonlinear iterative partial least squares (NIPALS). NIPALS is a simple, fast and effective algorithm to extract the principal components in a sequential manner and is a variant of the power method for calculating eigenvectors of a matrix. The algorithm proceeds as follow. At first, scale $\boldsymbol{X}$ by subtracting from each column its mean and dividing by standard deviation, choose arbitrary a column of $\boldsymbol{X}$ as $\boldsymbol{t}$, arrange $\boldsymbol{E} = \boldsymbol{X}$, this vector is multiplied by score vector $\boldsymbol{t}$ to give the loading vector $\boldsymbol{P}$ and normalizes $\boldsymbol{P}$ to have length one.

$$\boldsymbol{P} = \boldsymbol{E}^{'}.\boldsymbol{t} \,, \tag{2}$$

where, $\boldsymbol{E}^{'}(\boldsymbol{j},\boldsymbol{i}.\boldsymbol{k}) = \boldsymbol{E}(\boldsymbol{i},\boldsymbol{j},\boldsymbol{k})$,

$$\boldsymbol{P}(\boldsymbol{k},\boldsymbol{j}) = \sum_{\boldsymbol{i}=1}^{\boldsymbol{I}} \boldsymbol{E}^{'}(\boldsymbol{j},\boldsymbol{i},\boldsymbol{k})\boldsymbol{t}(\boldsymbol{i}) \,,$$

$$\boldsymbol{P} = \boldsymbol{P} / \|\boldsymbol{P}\| \,, \tag{3}$$

$$\|\boldsymbol{P}\| = \sqrt{\sum_{\boldsymbol{k}=1}^{\boldsymbol{K}} \sum_{\boldsymbol{j}=1}^{\boldsymbol{J}} \boldsymbol{P}(\boldsymbol{k},\boldsymbol{j})^2} \,,$$

$$\boldsymbol{t} = \boldsymbol{E} \circ \boldsymbol{P} \,, \tag{4}$$

$$\boldsymbol{t}(\boldsymbol{i}) = \sum_{\boldsymbol{j}=1}^{\boldsymbol{J}} \sum_{\boldsymbol{k}=1}^{\boldsymbol{K}} \boldsymbol{E}(\boldsymbol{i},\boldsymbol{j},\boldsymbol{k})\boldsymbol{P}(\boldsymbol{k},\boldsymbol{j}) \,,$$

$$\boldsymbol{E} = \boldsymbol{E} - \boldsymbol{t} \otimes \boldsymbol{P} \,, \tag{5}$$

$$\boldsymbol{X} = \boldsymbol{t} \otimes \boldsymbol{P}, \boldsymbol{X}(\boldsymbol{i},\boldsymbol{j},\boldsymbol{k}) = \boldsymbol{t}(\boldsymbol{i})\boldsymbol{P}(\boldsymbol{j},\boldsymbol{k}) \,,$$

The new score vector t is calculated and the convergence of t is checked. If t has converged then to equation (5). Otherwise, one iterates equations (2)-(4).

Two approach for MPCA monitoring of process are used, one is the squared prediction error (SPE) (Jackson, 1991) of residual space as follows,

$$Q = \boldsymbol{E}^{'}\boldsymbol{E} \,, \tag{6}$$

The critical value for Q is,

$$Q_\alpha = \theta_1 [\frac{c_\alpha \sqrt{2\theta_2 h_0^2}}{\theta_1} + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} + 1]^{1/h_0} \,, \tag{7}$$

where,

$$\theta_1 = \sum_{i=R+1}^{J} \lambda_i \,,$$

$$\theta_2 = \sum_{i=R+1}^{J} \lambda_i^2 \,,$$

$$\theta_3 = \sum_{i=R+1}^{J} \lambda_i^3 \,,$$

$$h_0 = 1 - \frac{2\theta_1\theta_3}{3\theta_2^2} \,,$$

where, λ_i is the eigenvalue of $\boldsymbol{X}$'s covariance.

Another index is the Hotelling T^2 test. T^2 value can be expressed as,

$$T_k^{\ 2} = \boldsymbol{t}_{\boldsymbol{k}}^{'} \boldsymbol{S}^{-1} \boldsymbol{t}_{\boldsymbol{k}} \,, \tag{8}$$

The critical value for T^2 is,

$$T_{k,\alpha}^2 = \frac{R(I-1)}{I-R} F_\alpha (R, I-R) \,, \tag{9}$$

If an observation vector that produce a value ofT_k^2 greater than $T_{k,\alpha}^2$, the process will be out of control.

However, one problem arises when MPCA monitors a new batch, only the data up to the present time is available and nothing is known about the remainder of the batch. In order to calculate the score vectors for the present batch, the missing data will have to be filled with some assumed approaches. MacGregor and Nomikos (Nomikos *et al*, 1995) suggests several methods to solve this, usually, two methods are used. The first is put zeros in the vector for all remaining missing batch. It assumes that the process will continue as a normal batch from the present time until the end of the batch. The second approach fills in the future data with the current observed value that has been normalized. All these method have been seen to work in practice. However, it would be more satisfying if one would not have to make such assumptions when monitoring an on going batch. In the following sections, it is shown that by using a step-by-step adaptive MPCA, this can be accomplished.

3. STEP-BY-STEP ADAPTIVE MPCA

In the proposed Step-by-Step Adaptive MPCA algorithm the ***X***-block is divided into a number of blocks, which are the time slices from the three-dimensional batch data matrix and each block has *N* batches and *J* variables. The different between the step-by-step adaptive MPCA from ordinary PCA algorithm is that it only looks at one time slice each time rather than all of the blocks at once.

The step-by-step adaptive MPCA algorithm proceeds as follows. The initial step is to calculate one PCA model for the first time slice (*k*=1), giving the first block's score vector $\boldsymbol{T}_1$ and loading vector $\boldsymbol{P}_1$, then lead in a forgetting factor β_1 to decide *l* columns ahead of the score vector $\boldsymbol{T}_1$ should be retained to make a new score vector $\boldsymbol{T}_{1new}$ ($\boldsymbol{T}_{1new}$ captures greater than β_1 percentages of total variations among $\boldsymbol{T}_1$), the $\boldsymbol{T}_{1new}$ will join in the PCA model building of the next time slice. The value of *l* can be express as,

$$\sum_{i=1}^{l} \lambda_{i1} \geq \beta_1 \sum_{i=1}^{J} \lambda_{i1}, \tag{10}$$

The algorithm begins with the second time slice and continues for the rest of the duration (*k*= 2, … *K*).

The first step in building the PCA model for time *k* is to combine the previous score $\boldsymbol{T}_{(k-1)new}$ vector that summarizes the recent prior history and the present $\boldsymbol{X}_k$ matrix together, then apply the PCA to $\boldsymbol{D}_k$, $\boldsymbol{T}_k$ is relative score vectors and $\boldsymbol{P}_k$ is loading vectors respectively.

$$\boldsymbol{D}_k = [\boldsymbol{T}_{(k-1)new}, \boldsymbol{X}_k], \tag{11}$$

$$\boldsymbol{T}_k = \boldsymbol{D}_k \boldsymbol{P}_k, \tag{12}$$

The second step is, through (13) decided the first *l* score vectors of $\boldsymbol{T}_k$ compose $\boldsymbol{T}_{knew}$, that will take part in the next time slice.

$$\sum_{i=1}^{l} \lambda_{ik} \geq \beta_k \sum_{i=1}^{J} \lambda_{ik}, 0 \leq \beta_k \leq 1, \tag{13}$$

where λ_{ik} is the eigenvalue of the D_k's covariance matrix.

The third step: define $k = k+1$. If k less than *K* then return the first step. Otherwise, ends.

During this algorithm, β_k is the forgetting factor. It controls the number of the columns of the score vector $\boldsymbol{T}_k$. A higher value of β_k will put more weight on the previous history and the model will adapt slowly, while a low value of β_k will make the model adapt fast. Setting β_k to zero will be equation to calculating separate PCA modes since it uses no memory of previous history of the batches. The value of β_k depends on the type of process to be monitored.

The criteria of choosing the forgetting factor β_k is described as follows, first lead-in the similarity (Manabu kano *et al*, 2002) between time slice *k* and *k*+1 of data sets.

When two data sets are similar to each other, the coefficient of similarity must be near one; the corresponding forgetting factor β_k is also high. However, the coefficient of similarity should be near zero when data sets are quite different from each other, the value of β_k is also low.

When the calculation as mentioned above is finished, *K* PCA models can be obtained. The *I*th model relates to the *I*th sample time of the batch process. In the monitoring phase two statistics with complementary information, the squared prediction error (SPE) and the Hotelling's statistics (T^2)(Jackson, 1991) can be used.

4. APPLICATION

In this section, an industrial typical multi-stage streptomycin fermentation process will be investigated to evaluate the performance of Step-by-Step Adaptive MPCA. The available on-line measurements for the employed process are shown in table1 and all these measurements are obtained at regular sample intervals.

Table 1 Online measurements obtained from the streptomycin fermentation process

1	Temperature	5	Nitrogen(N) Concentration
2	Air flow	6	Sugar Concentration
3	PH	7	Product Concentration
4	Viscosity		

With both good final product concentration (26000~30000) and normal variable trajectory, 20 good batches are used to train the model for process monitoring. Both variables are centred about their average trajectory and scaled to have unit variance prior to the analysis.

A special batch (the initial value of *PH* of this batch is abnormally high, due to the operator's operation, the final product concentration is 24920) is employed for comparison between the proposed method and MacGregor's method (Nomikos *et al*, 1995). The Squared Predictive Error (SPE) (solid line) monitoring results are listed in following figures, where the dotted line represents the 95% SPE confidence level.

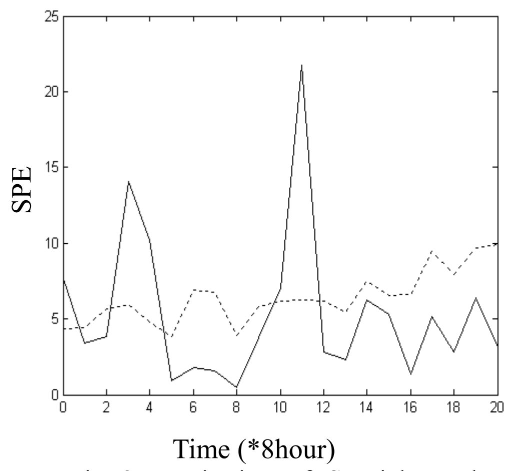


Fig 2 Monitoring of Special Batch Using Step-By-Step MPCA

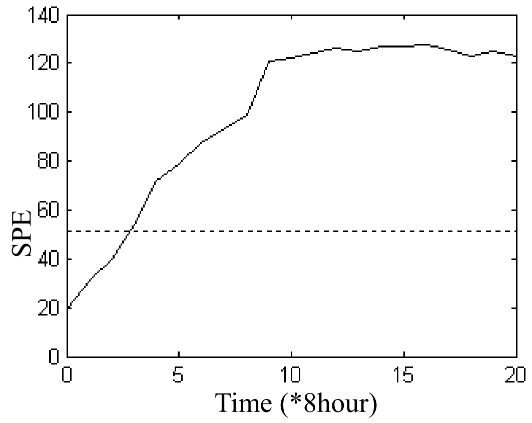


Fig 3 Monitoring of Special Batch Using MPCA (Method 1 to fill future data)

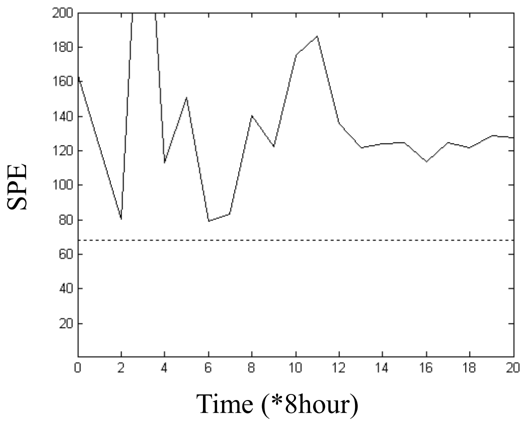


Fig 4 Monitoring of Special Batch Using MPCA (Method 2 to fill future data)

From figure 2, it's clear that Step-by-Step MPCA detects abnormal behaviours at the beginning of the bad batch. As we know, the initial condition of fermentation process has large effect on the final product quality, thus it's quite important to detect abnormal initial condition as soon as possible. Due to operators' efforts and compensations for the bad initial condition, this bad batch is drawn back to sub-normal operation trajectory. Compared with Step-by-Step MPCA, ignoring the multi-stage characteristics and taking the batch as whole, traditional MPCA method neglects the efforts of operators and evaluates this batch badly. Furthermore, using the first method to fill future data, MPCA is less sensitive to abnormal operation and the second method is too sensitive.

5. CONCLUSIONS

In this paper, a new Step-by-Step Adaptive MPCA algorithm has been presented for the purpose of monitoring batch processes. One advantage of this approach is that in the monitoring phase, there is no need to assume anything about the future deviations of the going batch from the normal trajectory. In traditional MPCA algorithm, one has to assume the operation of the future of the batch in order to full-fill the data vector from the current time period until the end of the batch. The step-by-step adaptive MPCA only works with the present and past data of the batch. Thus avoids the filling procedure. Another advantage is the model's ability of adapting to different stages of the batch process making it very suitable for monitoring multistage batch processes. In the industrial fermentation example involved in this paper, the step-by-step adaptive MPCA has represented its advantage. If one has a process with even more stages this method could be more advantageous since it will reduce the number of variables for monitoring and thereby simplifying the presentation of the monitoring results.

ACKNOWLEDGEMENTS

This project is supported by National High-Tech Program of China under grant No. 2001AA413110.

REFERENCES

Jackson J.E (1991), *A User's Guide to Principal Components*, Wiley Inter-science, New York,

Kosanovich K. A., Piovoso M. J (1994), Multi-way PCA applied to an industrial batch process, *In Am. Control conference,* **J.** 1294-1298

Nomikos P., MacGregor J. F (1994), Monitoring of batch processes using multi-way principal components analysis, *In Am. Inst. Chem. Eng.* **J. 40** 1361-1375

Nomikos P., MacGregor J. F (1995), Multivariate SPC charts for batch processes, *Techno-metrics* **37** 41-59

Stefan Rännar, MacGregor J. F (1998), Adaptive batch monitoring using hierarchical PCA, In *Chemo-metrics and Intelligent Laboratory System* **41** 73-81

Wold S. (1978), Cross-validation estimation of the number of components in factor and principal component analysis, *Techno-metrics* **20** 397-406

Wold S., Geladi P. (1982), Multi-way principal component analysis and LS analysis, *J. Chemo-metrics* 41-56

Wold S (1994), Exponentially weighted moving principal component analysis and projection to latent structures, *Chemo-metrics and Intelligent Laboratory System* **23** 149-161

IMPROVED OPERATION OF A BATCH POLYMERISATION REACTOR THROUGH BATCH-TO-BATCH ITERATIVE OPTIMISATION

Zhihua Xiong and Jie Zhang

Centre for Process Analytics and Control Technology
School of Chemical Engineering and Advanced Materials
University of Newcastle, Newcastle upon Tyne, NE1 7RU, U. K.
E-mail: zhihua.xiong@ncl.ac.uk, jie.zhang@ncl.ac.uk

Abstract: A batch-to-batch iterative product quality optimisation control strategy for a batch polymerisation reactor is proposed. Recurrent neural networks are used to model the dynamic behaviour of product quality variables. Model-plant mismatches and unknown disturbances are reflected in the model prediction errors. The repetitive nature of batch processes enables this information being discovered from previous batches and used to improve the current batch operation. Recurrent neural network predictions for the current batch are modified using prediction errors in previous batches. Because modified model errors are gradually reduced from batch to batch, the control trajectory gradually approaches to the optimal control policy and tracking errors also converge. The proposed scheme is illustrated on a simulated batch polymerisation reactor. *Copyright © 2003 IFAC*

Keywords: iterative learning, optimisation, recurrent neural networks, batch processes, polymerisation.

1. INTRODUCTION

Batch-to-batch optimisation of operating conditions for improving product quality and/or process efficiency has generated a challenging area of research in batch processes. Batch-to-batch optimisation exploits the repetitive nature of batch processes to determine the optimal operating policy. The general idea of batch-to-batch optimisation is to use results from previous batches to find iteratively the optimal control policies for subsequent batches, while performing the smallest number of sub-optimal runs (Srinivasan *et al*., 2001). Various strategies have been proposed for batch-to-batch optimisation in the literatures. Some strategies have been employed to compensate for modelling error (Dong *et al.*, 1996; Crowley *et al.*, 2001). Recently, iterative learning control (ILC) using optimisation has been introduced to directly update input trajectory. Campbell *et al.* (2002) presented a brief survey of linear model based run-to-run control algorithms for batch processes. Lee and co-workers in several related articles (Lee *et al.*, 1999; Lee *et al.*, 2000) proposed the Q-ILC approach with quadratic criterion for temperature control of batch processes. It combines ILC with model predictive control. A linear time-varying model was built to represent reactor temperature in relation to feed and a pre-specified trajectory of the temperature was tracked. It has been shown that effective tracking control performance can be achieved despite model errors and disturbances. This approach demonstrates that based on a linear model and pre-specified trajectory, temperature control of batch processes can be carried out by an ILC type approach.

However, this ILC type approach is difficult to be used directly for product quality control of batch processes. It is usually more difficult to set the reference trajectories for product qualities $\mathbf{Y}_d = (\boldsymbol{y}_d(t))$, $t\in(0,\ t_f)$, practically and reasonably than for temperature. Even if such a reference trajectory can be set, it is usually difficult to measure how well the reference trajectory is tracked since many product quality variables are difficult to be measured on-line. Although the reference sequences of product qualities during a whole batch may not be obtained, the desired values of product qualities $\boldsymbol{y}_d(t_f)$ at the end time of a batch are usually known. This makes it still possible to improve the product qualities from batch to batch. Furthermore, dynamics of product qualities cannot be represented accurately using a linear model since batch processes are operated in transient modes and their dynamics are intrinsically non-linear. The development of accurate mechanistic models of batch processes is usually costly and time-consuming. Alternatively, an empirical model, e.g. a recurrent neural network (RNN) model, can be built using process operation data to represent the non-linear dynamic characteristics of a batch process. In this study, RNN models are used to represent the non-linear relationship between the control trajectory and some product quality variables. The model predictions are iteratively modified by using model prediction errors in previous batches and optimisation is carried out based on the modified predictions.

The rest of this paper is structured as follows: Section 2 presents a batch-to-batch model-based iterative optimisation strategy. A simulated batch methyl methacrylate (MMA) polymerisation reactor is presented in Section 3. Section 4 gives simulation results of the proposed scheme on the MMA polymerisation reactor. Finally Section 5 draws some concluding remarks.

2. MODEL BASED BATCH-TO-BATCH ITERATIVE OPTIMISATION

2.1 Batch process modelling using recurrent neural networks

We consider a batch process where the run length (t_f) is fixed and divided into N equal intervals. Let us define the input and product quality sequences as

$$\mathbf{U}_k=[\boldsymbol{u}_k(0), \boldsymbol{u}_k(1),\ldots, \boldsymbol{u}_k(N\text{-}1)]^T \qquad (1)$$

$$\mathbf{Y}_k=[\boldsymbol{y}_k(1), \boldsymbol{y}_k(2),\ldots, \boldsymbol{y}_k(N)]^T \qquad (2)$$

where k is the batch index, $\boldsymbol{y}\in R^n$ are product quality variables, $\boldsymbol{u}\in R^m$ are the input (manipulted) variables for controlling the product quality.

In this study, RNN models are used to model the non-linear relationship between $\mathbf{U}_k$ and $\mathbf{Y}_k$. Given the initial conditions ($\boldsymbol{y}_0$, $\boldsymbol{u}_0$) and the input sequence $\mathbf{U}_k$, RNN models can predict recursively the output $\hat{\boldsymbol{y}}(t_f)$ at the end of a batch. Thus the predictions from RNN models are long range or multi-step-ahead predictions. The networks are trained using the Levenberg-Marquart optimisation algorithm to minimise its long-range prediction errors. Therefore RNN models can usually offer much better long-range predictions than feed forward neural networks (Tian *et al.*, 2001).

The RNN model predictions can have errors due to model-plant mismatches and unknown disturbances. To reduce these errors, the RNN model predictions can be corrected by adding filtered model errors of previous batch runs. Crowley *et al.* (2001) introduced a strategy where the model predictions are corrected by adding the filtered model prediction residuals obtained from only the immediate previous run and showed that this can reduce batch-to-batch variability. As the dynamics of product quality in batch processes are usually very non-linear and measurement noise always exists, information of all previous runs should be used in updating model predictions for the next run. In this study, the average model errors of all previous runs are used to modify RNN model predictions. The RNN model error $\hat{\boldsymbol{e}}_k(t)$ is defined as

$$\hat{\boldsymbol{e}}_k(t) = \boldsymbol{y}_k(t) - \hat{\boldsymbol{y}}_k(t) \qquad (3)$$

where $\boldsymbol{y}_k(t)$ and $\hat{\boldsymbol{y}}_k(t)$ are, respectively, the measured and predicted values of product quality at time t of the kth batch. The average model error $\bar{\hat{\boldsymbol{e}}}_k(t)$ of all previous runs is defined as

$$\bar{\hat{\boldsymbol{e}}}_k(t) = \frac{1}{k}\sum_{i=1}^{k}\hat{\boldsymbol{e}}_i(t) = \frac{1}{k}\sum_{i=1}^{k}(\boldsymbol{y}_i(t) - \hat{\boldsymbol{y}}_i(t)) \qquad (4)$$

By filtering this average model error, the modified prediction $\tilde{\boldsymbol{y}}_{k+1}(t)$ of a RNN model is defined as

$$\tilde{\boldsymbol{y}}_{k+1}(t) = \hat{\boldsymbol{y}}_{k+1}(t) + \alpha\,\bar{\hat{\mathbf{e}}}_k(t) \qquad (5)$$

where α is an adjustable filter parameter.

2.2 Batch-to-batch iterative optimisation control

Given that the whole reference sequence $\mathbf{Y}_d$ is available, both Amann *et al.* (1996) and Lee *et al.* (2000) have proposed a quadratic objective that penalises the input change instead of the input. The algorithm has an integral action with respect to the batch index k and achieves the minimum achievable error in the limit (Lee *et al.*, 2000). For product quality control of batch processes, only product qualities at the end of a batch, $\boldsymbol{y}_d(t_f)$, are available to set. Then only errors at the end of a batch, $\tilde{\mathbf{E}}^f_{k+1} = \boldsymbol{y}_d(t_f) - \tilde{\boldsymbol{y}}_{k+1}(t_f)$, are penalised in the objective function. Considering the constraints on the input trajectory, the batch-to-batch iterative optimisation problem for product quality control can be formulated as

$$\min_{\mathbf{U}_{k+1}} J = \|\tilde{\mathbf{E}}^f_{k+1}\|^2_Q + \|\Delta\mathbf{U}_{k+1}\|^2_R \qquad (6)$$

s. t.

$$\tilde{\mathbf{E}}^f_{k+1} = \boldsymbol{y}_d(t_f) - \tilde{\boldsymbol{y}}_{k+1}(t_f) \qquad (7)$$

$$\Delta\mathbf{U}_{k+1}=\mathbf{U}_{k+1} - \mathbf{U}_k \qquad (8)$$

$$\hat{\boldsymbol{y}}_{k+1}(t) = \boldsymbol{f}_{\boldsymbol{RNN}}[\hat{\boldsymbol{y}}_{k+1}(t\text{-}1), \ldots, u_{k+1}(t\text{-}1), \ldots] \qquad (9)$$

$$\tilde{\boldsymbol{y}}_{k+1}(t) = \hat{\boldsymbol{y}}_{k+1}(t) + \alpha\,\bar{\hat{\boldsymbol{e}}}_k(t) \qquad (10)$$

$$u_{min} \le \mathbf{U}_{k+1} \le u_{max} \qquad (11)$$

where $\tilde{\mathbf{E}}^f_{k+1}$ is the difference at the end of the (k+1)th batch between the desired product qualities and the modified RNN model predictions, $\boldsymbol{f}_{\boldsymbol{RNN}}[\cdot]$ represents the RNN model, u_{min} and u_{max} are low and high bounds of the input trajectory, $\boldsymbol{Q}$ and $\boldsymbol{R}$ are weighting matrices and they are selected of the following forms in this study: $\boldsymbol{Q}=\lambda_q\cdot\boldsymbol{I}_n$, and $\boldsymbol{R}=\lambda_r\cdot\boldsymbol{I}_N$.

A larger weight λ_r on the input change will lead to more conservative adjustments and slower convergence. The weight λ_q on the quality error term should be appropriately selected in relation to the weight λ_r so that the performance due to input changes will not be degraded while the product quality control is enforced. There are also other variants of the objective function. For example, the weight matrix $\boldsymbol{R}$ may be designed to be increaseing with batches reflecting the improved confidence of product quality prediction.

The modified prediction error $\varepsilon_{k+1}(t)$ is calculated as

$$\varepsilon_{k+1}(t) = \boldsymbol{y}_{k+1}(t) - \tilde{\boldsymbol{y}}_{k+1}(t) \qquad (12)$$

Considering Eq(5), Eq(12) can be rewritten as

$$\varepsilon_{k+1}(t) = \hat{\boldsymbol{e}}_{k+1}(t) - \alpha\,\bar{\hat{\boldsymbol{e}}}_k(t) \qquad (13)$$

Eq(4) can be reformed as

$$\bar{\hat{\boldsymbol{e}}}_k(t) = \frac{k-1}{k}\bar{\hat{\boldsymbol{e}}}_{k-1}(t) + \frac{1}{k}\hat{\boldsymbol{e}}_k(t) \qquad (14)$$

Then Eq(14) can be rewritten as

$$\varepsilon_{k+1}(t) = \frac{k-1}{k}\varepsilon_k(t) + [\hat{\boldsymbol{e}}_{k+1}(t) - \frac{k-1+\alpha}{k}\hat{\boldsymbol{e}}_k(t)] \qquad (15)$$

Beacasue $(k\text{-}1)/k < 1$, $\varepsilon_{k+1}(t)$ will converge with respect to the batch number k. Due to the reduced prediction errors of the RNN model, the control trajectory will gradually approach the optimal policy.

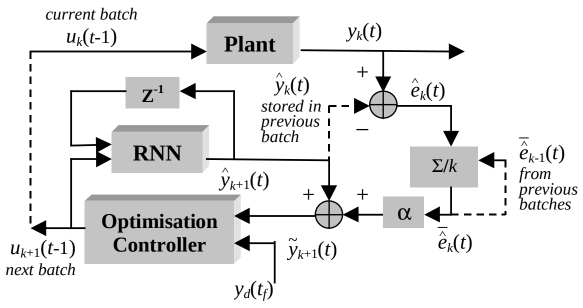


Fig. 1. Batch-to-batch iterative optimisation

As shown in Fig. 1, the batch-to-batch model-based iterative optimisation scheme is outlined as follows: At the current batch k, the input trajectory $\mathbf{U}_k$ is implemented and the outputs $\boldsymbol{y}_k(t)$ are obtained by on-line or off-line analysis of samples taken during the batch. The RNN model predictions for the next batch are modified by using prediction errors of all previous runs. Based on the modified predictions $\tilde{\boldsymbol{y}}_{k+1}(t)$, the quadratic optimisation problem specified by Eq(6) to Eq(11) is solved and a new control policy $\mathbf{U}_{k+1}$ for the next batch is calculated. This procedure is repeated from batch to batch. Because iterative optimisation is done in the interval between two adjacent batch runs, the outputs $\boldsymbol{y}_k(t)$, $t\in(1,N)$, could be obtained after the completion of each run by off-line analysis of samples taken during the batch.

3. A BATCH POLYMERISATION REACTOR

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 MMA with a water solvent and benzoyl peroxide initiator. 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. A detailed mathematical model covering reaction kinetics and heat mass balances has been developed for the bulk polymerisation of MMA. Based on this model, a rigorous simulation program was developed and validated on the pilot reactor. The simulation programme is used to test modelling and control strategies.

The optimisation control problem for this batch polymerisation reactor is to find the optimal temperature profile through minimising a performance index at a given final time. The performance index to be minimised is given below

$$\boldsymbol{PI}\,(T) = \|\boldsymbol{y}_d(t_f) - \boldsymbol{y}(t_f)\|^2 \qquad (16)$$

where $\boldsymbol{y} = [X, Mn, Mw]^T$, X is the monomer conversion, Mn is the dimensionless number-average molecular weight (MN) and Mn=MN/MN$_{ref}$, MN$_{ref}$ is set to 3.0×10^5 (g/mol), Mw is the dimensionless weight-average molecular weight (MW) and Mw = MW/MW$_{ref}$, MN$_{ref}$ is set to 9.0×10^5(g/mol), T is the dimensionless temperature profile and $T = (T_r - T_{min})/(T_{rmax}\text{-}T_{min})$ with T_r being the unscaled temperature, T_{min} is 310K and T_{rmax} is 360K, $\boldsymbol{y}_d(t_f)$ is the desird value at the batch end which is set to $[1.0, 1.0, 1.0]^T$, and t_f is the final batch time which is set to 120 minutes in this study. The polymer property constraint is on Mn:

$$0.95 \le Mn \le 1.05 \qquad (17)$$

Also the temperature profile is bounded by

$$0 \le T \le 1 \qquad (18)$$

4. SIMULATION RESULTS AND DISCUSSIONS

RNN models are developed to model X, Mn and Mw. In this study, 31 different sets of temperature profiles have been randomly chosen within a reasonable range to generate 31 runs of simulation data. As the batch duration is 120 minutes and the sampling time is 4 minutes, each set of data contains 30 samples. All data are scaled to dimensionless values. Normally distributed random noises with zero means were added to all the simulation data to represent the effects of measurement noises. The standard deviations of the noises are 0.012, 0.009 and 0.01 for the dimensionless X, Mn and Mw respectively. The entire data set was divided into 3 parts, 25 batches of data were used for training, 5 batches of data for validation, and the remaining 1 batch for testing.

In the recurrent neural networks, hidden neurons use the sigmoidal activation function whilst output layer neurones use linear activation function. The network weights were initialised as random numbers uniformly distributed over the range (-0.1, 0.1). The training algorithm is based on the Levenberg-Marquart algorithm. Network structures were determined through cross-validation. It was tested that introducing monomer conversion X as one of the inputs of the other two neural networks was necessary. Different networks were trained on the training data and the network with the least sum of squared errors (SSE) on the validation data was chosen as the best network. The generalisation performance was then assessed on the unseen testing data set. The best representations of the three RNN models selected through cross-validation are summarised as follows

$$\hat{X}(t) = f_1[\hat{X}(t\text{-}1), \hat{X}(t\text{-}2), T(t\text{-}1), T(t\text{-}2), T(t3)] \quad (19)$$

$$\hat{Mn}(t) = f_2[\hat{Mn}(t\text{-}1), \hat{Mn}(t\text{-}2), T(t\text{-}1), T(t\text{-}2), T(t\text{-}3), \hat{X}(t\text{-}1), \hat{X}(t\text{-}2)] \quad (20)$$

$$\hat{Mw}(t) = f_3[\hat{Mw}(t\text{-}1), \hat{Mw}(t\text{-}2), T(t\text{-}1), T(t\text{-}2), \hat{X}(t\text{-}1), \hat{X}(t\text{-}2)] \quad (21)$$

where $\hat{X}$, $\hat{Mn}$ and $\hat{Mw}$ are, respectively, the model predictions of X, Mn and Mw. The numbers of hidden neurons for the above three networks were determined through cross-validation as 8, 12 and 10 respectively. The SSE of the above models, Eq(19) to Eq(21), on the validation data set are 0.0978, 0.1935 and 0.2033 respectively. Fig. 2 shows the long-range predictions of these RNN models on the unseen testing data set. It is clear that the RNN models have captured the dynamic trends of the product quailities in the data quite well, though some prediction errors still exist.

To investigate the performance of the proposed control strategy, three cases were studied: Case 1 – optimisation based upon the mechanistic model; Case 2 – RNN model-based optimisation; and Case 3 – RNN model based batch-to-batch iterative optimisationp. In Case 1 and Case 2, the optimisation problem is specified by Eq(16) to Eq(18) based on the mechanistic model and the RNN models respectively, whereas in Case 3 it is specified by Eq(6) to Eq(11). All these non-linear optimisation problems were solved by the sequential quadratic programming method with the termination tolerance on objective function being set to 10^{-6}. Model errors in Case 2 are used as the initial condition for the model prediction modification in Case 3. Here the batch length was divided into N=10 equal intervals. The parameters in the optimisation problem in Case 3 were chosen as follows: α = 0.25, λ_q =3, and λ_r=10. Batch processes always exhibit batch-to-batch variations due to unknown disturbances such as reactive impurities and reactor fouling (Zhang *et al.*, 1999). In this study, the initial initiator weight was set to the nominal value 2.5*g* for the first 15 batches and then drops to 2.1*g* starting at the 16th batch to simulate the effect of reactive impurities. Consequently, the simulated reactive impurities act as a batchwise persisting disturbance.

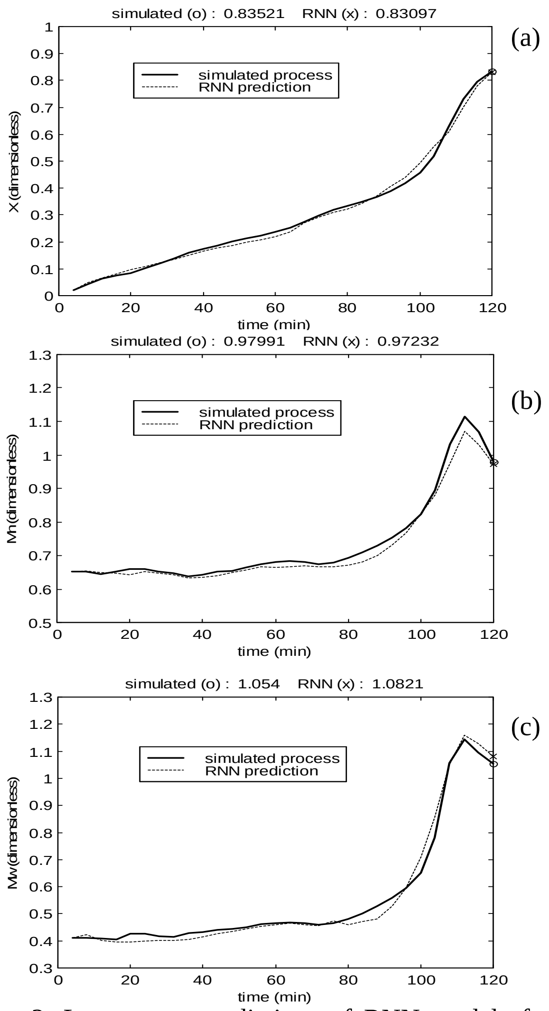


Fig. 2. Long-range predictions of RNN models for (a) X, (b) Mn, and (c) Mw

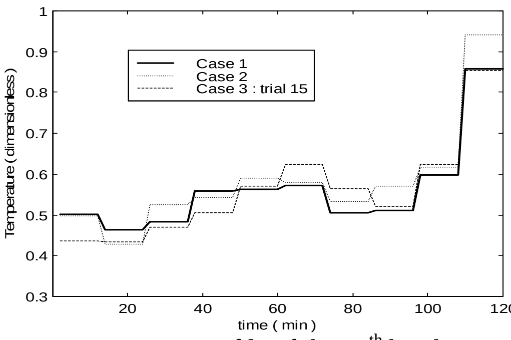


Fig. 3. Temperature profile of the 15th batch in Case 3 compared with those in Case 1 and Case 2

The results of the product qualities at the batch end and the performance indices (***PI***) in all three cases are shown in Table 1. It should be noticed that ***PI*** in Case 3 was obtained according to Eq(16) after the optimal temperature policy in Case 3 was applied to the simulated reactor. Due to RNN model-plant mismatches, the ***PI*** in Case 2, 0.0493, is 45.8% worse than that in Case 1. In Case 3, after 15 batches with iterative optimisation, the ***PI*** is decreased to 0.0382, only 13.0% higher (worse) than that in Case 1. Fig. 3 compares the temperature profile of the 15th batch in Case 3 with the results in Case 1 and Case 2.

Fig. 4 shows convergence of temperature profiles of the 1st, 3rd, 11th and 15th batches in Case 3. Under the proposed batch-to-batch iterative optimisation scheme, because model predictions were modified by the results of previous batches and model errors were gradually reduced, the reactor temperature profile also converge to the optimal one.

Table 1. The product qualities and ***PI*** in all three cases (without impurities)

	Case 1	Case 2	Case 3 (15th batch)
$X(t_f)$	0.8389	0.8321	0.8334
$Mn(t_f)$	0.9527	0.8902	0.9168
$Mw(t_f)$	0.9248	0.9046	0.9403
PI	0.0338	0.0493	0.0382

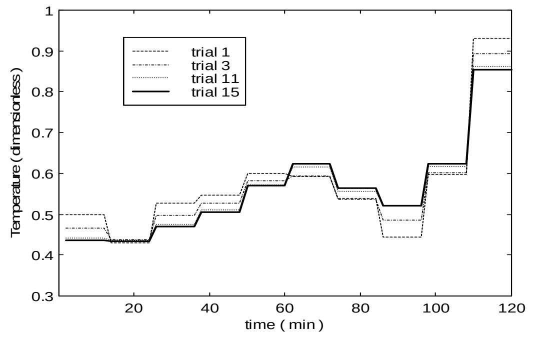


Fig. 4. Temperature profiles in Case 3

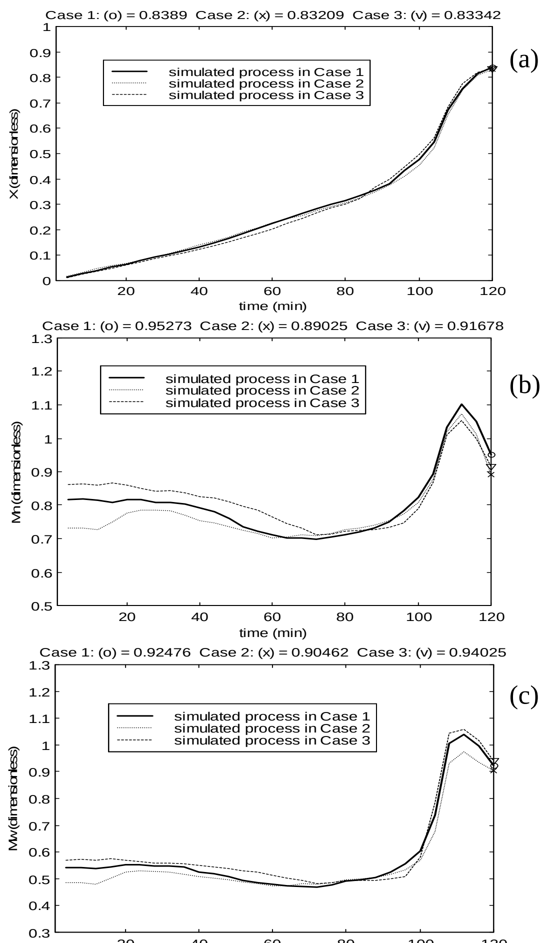


Fig. 5. Trajectories of product qualities on the simulated process: (a) *X*, (b) *Mn*, and (c) *Mw*

Fig. 5 shows the trajectories of *X*, *Mn* and *Mw* in three cases after the corresponding optimal temperature policies were applied to the simulated polymerisation reactor. It can be seen that with iterative optimisation control, the product qualities in Case 3 are improved compared to those in Case 2. Fig. 6 shows the convergence of tracking errors $\boldsymbol{e}_k^f = \boldsymbol{y}_d(t_f)-\boldsymbol{y}_k(t_f)$ of the product quality variables after the first 15 batches in Case 3. It can be seen that tracking errors have converged after about 11 batches.

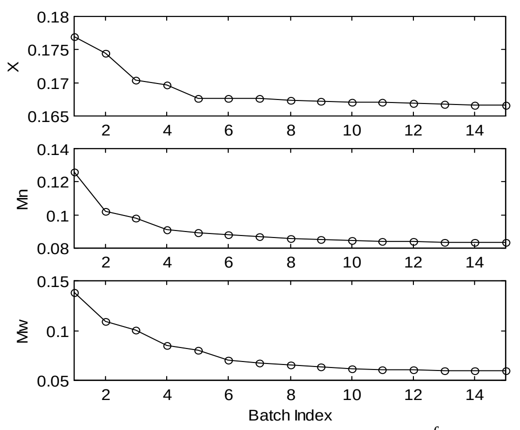


Fig. 6. Convergence of tracking errors $\boldsymbol{e}_k^f$ for the first 15 batches in Case 3

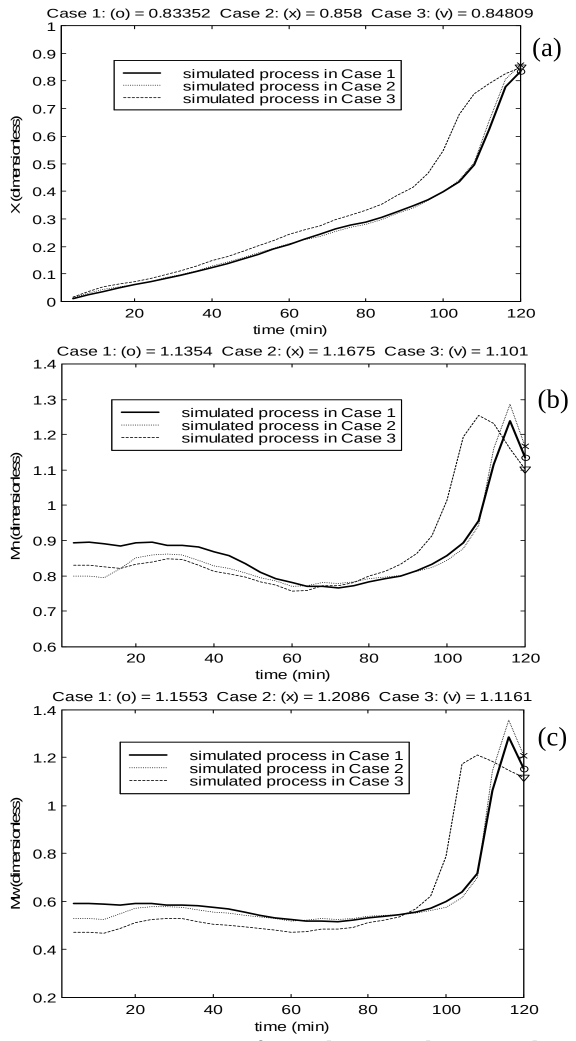


Fig. 7. Trajectories of product qualities under reactive impurities: (a) *X*, (b) *Mn*, and (c) *Mw*

If there are disturbances in a batch process, the optimal control profile calculated in the previous batch may not result in the expected product quality. When the initial initiator weight drops to 2.1*g* from its nominal value 2.5*g*, the ***PI*** in Case 1 and Case 2 become worse if the control policies calculated under the norminal condition are still employed. As shown in Table 2, the ***PI*** in Case 1 increases to 0.0702 and the ***PI*** in Case 2 increases to 0.0917. However, under the batch-to-batch iterative optimisation scheme, the ***PI*** in Case 3 can be brought down to 0.0467. Fig. 7 shows the trajectories of product quality variables under reactive impurities.

Due to the presence of reactive impurities, the temperature trajectory obtained in the 15th batch of Case 3 was no longer optimal and prediction errors of RNN increased. As shown in Fig. 8, the tracking errors e_k^f significantly increased in the 16th batch. This issue has been successfully addressed by the model-based iterative optimisation scheme. It can be seen that $\boldsymbol{e}_k^f$ has converged after about 10 batches. The results in Case 3 demonstrate that, although there are some model-plant mismatches and unknown disturbances, the performance index can be gradually improved under the iterative optimisation scheme.

Table 2. The product qualities and ***PI*** in all three cases (with unknown impurities)

	Case 1	Case 2	Case 3 (30th batch)
$X(t_f)$	0.8335	0.8580	0.8481
$Mn(t_f)$	1.1354	1.1675	1.1010
$Mw(t_f)$	1.1553	1.2086	1.1161
PI	0.0702	0.0917	0.0467

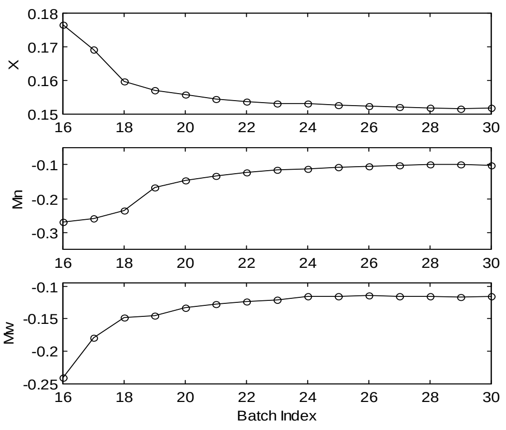


Fig. 8. Convergence of tracking errors $\boldsymbol{e}_k^f$ for the last 15 batches in Case 3

5. CONCLUSIONS

A model-based batch-to-batch iterative product quality optimisation control scheme for batch processes is proposed in this paper. Recurrent neural network models are built to represent the operation of a batch process and the model predictions are modified using prediction errors of previous batches. A quadratic objective function is introduced to the optimisation problem of product quality control. Using batch-to-batch iterative optimisation, it has been demonstrated that model errors are gradually reduced from batch to batch and the control policy converges to the optimal policy. The proposed scheme is illustrated on a simulated batch MMA polymerisation reactor.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the support of the UK EPSRC (Grant GR/N13319) and thank Prof. C. Kiparissides of the Aristotle University of Thessaloniki, Greece, for providing the polymerisation reactor model and the simulation programme.

REFERENCES

Amann, N., D. H. Owens and E. Rogers (1996). Iterative learning control for discrete-time system with exponential rate of convergence. *IEE Proce. D -Control Theory Applications*, **143**(2), 217-224.

Campbell, W. J., S. K. Firth, A. J. Toprac and T. F. Edgar (2002). A comparison of run-to-run control algorithms. *Proce. ACC*, Anchorage, AK, May 8-10, 2150-2155.

Crowley, T. J., C. A. Harrison and F. J. Doyle III (2001). Batch-to-batch optimization of PSD in emulsion polymerization using a hybrid model. *Proce. ACC*, Arlington, VA, June 25-27, 981-986.

Dong, D., T. J. McAvoy and E. Zafiriou (1996). Batch-to-batch optimization using neural network models. *Ind. Eng. Chem. Res.*, **35**, 2269-2276.

Lee, K. S., I. S. Chin, H. J. Lee and J. H. Lee (1999). Model predictive control technique combined with iterative learning control for batch processes, *AIChE J.*, **45**(10), 2175-2187.

Lee, J. H., K. S. Lee and W. C. Kim (2000). Model-based iterative learning control with a quadratic criterion for time-varying linear systems. *Automatica*, **36**, 641-657.

Srinivasan B., C. J. Primus, D. Bonvin and N. L. Ricker (2001). Run-to-run optimization via control of generalized constraints. *Control Engineering Practice*, **9**, 911-919.

Tian, Y., J. Zhang and A. J. Morris (2001). Modeling and optimal control of a batch polymerization reactor using a hybrid stacked recurrent neural network model. *Ind. Eng. Chem. Res.*, **40**, 4525-4535.

Zhang, J., A. J. Morris, E. B. Martin and C. Kiparissides (1999). Estimation of impurity and fouling in batch polymerisation reactors through the application of neural networks, *Computers. Chem. Eng.*, **23**, 301-314.

KAPPA NUMBER PREDICION BY HYBRID MODEL FOR BATCH PULP COOKING PROCESS

Yan Li, Jian Zhang, Xuefeng Zhu, Daoping Huang

College of Automation Science & Engineering,
South China University of Technology, Guangzhou, China

Abstract: In batch pulp cooking process the wood chips are converted into pulp by lignin dissolution in cooking acid. The percentage of non-dissolved lignin is often expressed by so called Kappa number. To obtain desired quality of the pulp, Kappa number of the pulp should be decreased to the desired value at the end of batch cycle. Since reliable on-line commercial sensors of Kappa number are still unavailable, developing the soft sensor for measuring Kappa number in batch pulp cooking process is of practical significance. In this paper, a kinetic hybrid model is developed to predict the Kappa number for the batch cooking process. The effectiveness of the proposed hybrid model can be illustrated by the predicted errors for a actual cooking process.

Keywords: Soft sensing, Hybrid mode, Radial basis function network, Pulp cooking process.

1. INTRODUCTION

Pulp and paper industry bears the stamp of exhaustive energy and raw material consumption. To achieve better yield at lower production costs, many researchers have been working on the measurement and control of Kappa number, an important quality index of pulp cooking. Although lots of research workers have been conducted in the field of Kappa sensor technology in recent years, on-line reliable Kappa number measurements in batch digesters is still very difficult. Therefore developing Kappa number model and the model-based control strategy of batch pulp cooking process is a challenge task for the pulp industry.

By analyzing the physical-chemical mechanism of the cooking process, our research team has developed a simplified model for predicting the Kappa number, in which the initial charge conditions are expressed and correlated with an initial effective alkali concentration (sampled at the time of H factor equal to a certain number). The model can achieve very good predictive result in laboratory condition but while the model is used in practical cooking process the performances of the model are not very satisfied. Because of lack of sufficient off-line data to provide comprehensive knowledge of the complicated industrial process, the model is only useful over a narrow range of operating conditions. For this reason, in this paper it is to try developing a hybrid Kappa number model, which will provide the predictive Kappa number for the last phase of the batch cook process by means of learning from the history data.

2. BACKGROUND

2.1 Soft Sensing Technology

Soft sensing technology is a measurement method which employing easily measured variables (auxiliary variables) and their relationship (soft sensing model) to acquire some variables (primary variables), which hard to measure directly. We can say that, soft sensing technology is a method of information utilizing and rule discovery, data classification and variable prediction. Data classification extents the data space by estimating the unknown data class experientially. Simultaneously, variable prediction extents the data temporal space by forecasting the variable development. During the process of soft sensing modeling, different kind of theory and method should be utilized comprehensively to dig out useful information.

Artificial neural network technology has been introduced in the control field because there are many systems whose rigid mathematical models are hard to acquire, such as highly non-linear chemical processes including those found in the pulp industry. In this case, ANN may be an effective tool to cope with these problems, especially for systems whose characteristics and uncertainties are difficult to identify using mathematical models. To model such systems, ANN can provide some promising solutions (Thompson and Kramer, 1994).

In this paper, an empirical predictive hybrid model is provided which based on neural network. It also takes advantage of Rough Set Theory and fuzzy theory to construct the model. First, certain rules and uncertain rules are acquired by analyzing the history data using Rough Set Theory. Then, Radical Based Neural Network is employed to realize the fuzzy model. The main steps are:

(1) Rules Extraction by attribute reduction strategy based on rough set theory.
(2) Using the rules as the node centres of the hidden layer to train the RBF.

2.2 Rule Extraction

Rough Set Theory

Rough Set Theory was introduced by Z. Pawlark, a polish mathematician, in 1982. It is a relatively new soft computing-tool to deal with vagueness and uncertainty (Pawlark, 1996). It has received much attention of the researchers around the world. Rough Set Theory has applied to many area successfully including pattern recognition, machine learning, decision support, process control and predictive modeling.

Rule Extraction

The concept of un-differentiate relationship is the basement of the RS. The other important concepts include upper approach, lower approach, boundary area and rough abstract function. The main steps are listed below using Rough Set methods to discover knowledge and decision rules by analyzing and simplifying the great amount of measure data (Wang, *et al.*, 1998).

(1) Disperse the continuous data interval into discrete intervals, using the code of the sub-area as the value of the continuous data.
(2) Acquire the discrete decision table and begin attributes reduction, using reduction strategy based on differentia matrix, which defines the times of the attribute appeared in differentia matrix as the attribute significance.
(3) Based on the simplified result, look for the upper approximate set and lower approximate. Then sum up the logic rules.

Review these rules and compare them with expert experience to acquire the final results. These rules will supervise the training of the RBFNN.

2.3 Radical Basis Function Network

In a sense, radical basis function has common ground with fuzzy system, or we can realize a certain kind of fuzzy system by means of Radical Basis Function Network. The number of the nodes in input layer is the number of reduction attribute vectors. The number of the nodes in hidden layer is decided by number of the rules. The transform function of hidden layer is Gauss function.

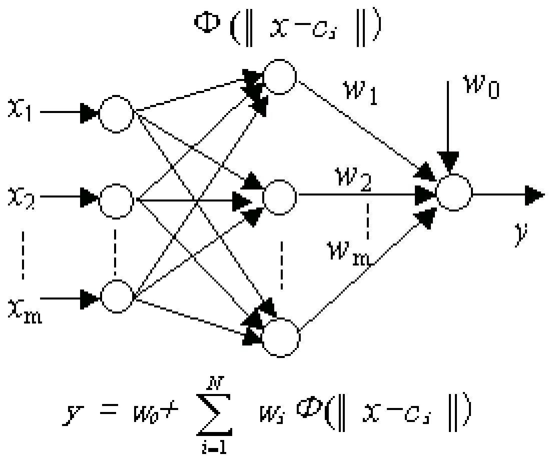


Fig.1. Structure of RBFNN

$$\Phi_i(x) = e^{\frac{\|x-c_i\|^2}{\sigma_i^2}} \qquad (1)$$

Contract to number i rule of a simple system:
IF x_1 IS A_{1i} and x_2 IS A_{2i} ... and x_n IS A_{ni} , THEN y_1 IS w_{i1} and y_2 IS w_{i2} ...and y_m IS w_{im}

The centres of Gauss Function are decided by the precondition vectors of the fuzzy rules. The weights of the output layer are corresponding with the posterior parameter of the fuzzy rule. That is RBFNN is of some comparability with fuzzy system. Comparing with the classical BP neural network, RBFNN is more apprehensible (Krzyzak and Linder, 1998).

3. HYBRID MODEL

Choosing initial effective alkali (sampled at time of H=200), sulfide degree, H factor and wood chip eligible rate as the input variables and Kappa number as the output the hybrid model can be developed.

3.1 find the centre value of hidden layer node by Rough Set Theory

Using 160 groups data from a factory cooking process as an example to illustrate the construction of the hybrid model. The first 120 groups data are chosen as learning data, last 40 groups of data are used to verify the effect.

construct binary decision table

To construct a decision table, the consecutive variables should be converted to be discrete firstly, so Equal Interval Division method and Equal Probabilities method have been applied, but results are not so ideal. These methods are difficult to determine the discrete grade. Too rough grade leads to appear large amount of inconsistent data. Consequently, more inconsistent part of the

constructed decision table will be produced. On the other hand, if the grade is too precise, rules can't be abstracted effectively from the decision table. To solve the problem, code the decision table, which is dispersed by equal interval division method, and the table is converted to an approximately binary attribute table, then conduct attribute reduction using differential matrix method (Wang, Y.Y., 2001). As the result, some neighbourhood intervals would join together. The steps in details are listed as following:
(1) Evaluate frequency distribution table of condition variables using statistic analysis software, as presented in table 1.

Table 1 Variable frequency distribution table

fre-quency	dividing point			
	effective alkali	sulfide degree	wood chip eligible	H factor
-10%	25.43	25.90	69.80	1826.20
-20%	26.97	26.80	73.00	1890.80
-30%	27.90	27.43	74.33	1954.60
-40%	28.68	27.64	75.68	2016.80
-50%	29.37	28.20	77.80	2048.00
-60%	29.92	28.60	79.38	2148.40
-70%	30.54	29.00	80.41	2234.20
-80%	31.16	29.50	81.36	2299.20
-90%	32.55	29.90	84.00	2480.60

(2) Utilize the frequency distribution table and maximum limit of variables Condition variables are divided into 10 intervals, then each interval is coded, for example, the coding result of effective alkali variable is shown in table 2.

Table 2 code of effective alkali interval

Inter-val	[22.00 25.44)	[25.44 26.97)	[26.97 27.90)	[27.90 28.68)	[28.68 29.38)
Code	0	1	2	3	4
Inter-val	[29.38 29.92)	[29.92 30.54)	[30.54 31.16)	[31.16 32.55)	[32.55 43.00)
Code	5	6	7	8	9

(3) Construct a binary decision table, for a decision system, any condition attribute among the system can be represented by 9 binary attributes $Z_q^0, \dots, Z_q^8$, whose value domain is $\{0,1\}$.(as shown in table 3)

For example, if the value of a sample effective alkali is 28.72, the interval code will be 4, then the attribute is $\{Z_e^0, Z_e^1, \dots, Z_e^8\} = \{0,0,0,0,1,1,1,1\}$ after conversion.
Decision attributes can be divided into 3 intervals by equal frequency. (as presented in table 4)

Table 3 binary table of attribute q

Value	Z_q^0	Z_q^1	...	Z_q^7	Z_q^8
0	0	0	...	0	0
1	0	0	...	0	1
...	...	...	...	...	...
8	0	1	...	1	1
9	1	1	...	1	1

Table 4 Discretization of Decision Attributes

interval	[22.00, 33.27]	[33.27, 36.90]	[36.90, 49.00]
code	0	1	2

Consequently, there are 36 condition attributes and 1 decision attribute in the constructed binary decision table (as seen in table 5), while the original table only has 4 condition attributes and 1 decision attribute., the value domain of condition attributes is $\{0,1\}$, while the value domain of decision attributes is $\{0,1,2\}$

Table 5 binary attribute decision table

Case no.	$Z_e^0 Z_e^1 Z_e^2$ $Z_e^3 Z_e^4 Z_e^5$ $Z_e^6 Z_e^7 Z_e^8$	...	$Z_H^0 Z_H^1 Z_H^2$ $Z_H^3 Z_H^4 Z_H^5$ $Z_H^6 Z_H^7 Z_H^8$	K_a
1	0 0 0 1 1 1 1 1 1	...	0 0 0 1 1 1 1 1 1	0
2	0 0 0 0 0 0 1 1 1	...	0 0 0 0 1 1 1 1 1	2
...	...	...	...	...
119	0 0 1 1 1 1 1 1 1	...	0 0 0 0 0 1 1 1 1	0
120	0 0 0 0 0 0 0 1 1	...	0 0 0 1 1 1 1 1 1	1

(4) Reduce the attributes of the constructed binary decision table using the decision attributes reduction technique presented in paper (Wang, *et al.*, 1998), reduplicate samples reduction is made in vertical and reduction based on differential matrix method is made on horizontal. As a result, two reduplicate samples are reduced and the reduced condition attributes are:
effective alkali $\{Z_e^1, Z_e^3\}$
sulfide degree $\{Z_s^1, Z_s^3, Z_s^5, Z_s^6, Z_s^8\}$
wood-chip eligible rate $\{Z_m^3, Z_m^4\}$
H factor $\{Z_H^2, Z_H^4, Z_H^5, Z_H^7, Z_H^8\}$
How to merge conditions variables is shown in table 6.

In the next section, we will train the network by using centre value of rule precondition interval as centre value of hidden layer nodes of RBF neural network, so the logic rules induction is not needed here. We can recode each interval based on the condition variables interval table above, the coding sequence can be 0,...,n-1, where n is number of intervals, then the reduction decision table is created.

Table 6 Discretzation of Decision Attributes

Variable	Inter-val Num-ber	Intervals
Effective alkali	8	[22.00,25.44], [25.44,26.97], [26.97,27.90], [27.90,28.68], [28.68,29.38], [29.38,30.54], [30.54,32.55], [32.55,43.00],
Sulfide degree	5	[23.00,26.80], [26.80,28.20], [28.20,29.00], [29.00,29.90], [29.90,33.00]
Wood chip eligible rate	8	[58.00,69.80], [69.80,73.00], [73.00,74.33], [74.33,75.68], [75.68,80.41], [80.41,81.36], [81.36,84.00], [84.00,89.00],
H factor	5	[1300.0,1954.6], [1954.6,2148.4] [2148.4,2299.2], [2299.2,2480.6], [2480.6,5400.0]
Kappa number	3	[22.00,33.27], [33.27,36.90], [36.90,49.00]

3.2 Determination of RBF neural network hidden layer nodes and learning samples

Although equal interval coding method is applied to discrete continues interval, the data still may be lost or distorted, which will produce inconsistent rule, which is that two samples have the same attribute value while the decision value is difference. There are 3 pairs of inconsistent rules in previous process. The appearance of inconsistent rules is a common problem when constructing model with data from industry fields, especially, when constructing model with data from some complex process. Such situation does often appear: a pair of input data is nearby in distance while the corresponding output is reverse by distance meaning, so the extension capability of the model is not so ideal. There are many reasons, including improper selection of raw pulp sample spot, miss-manipulate by operators and some baffling reason etc. Consequently, the error between evaluation value of model and measurement value (off-line) is great when the soft-sensor model is running.

To solve above problem, we should select proper discretazation method, separate interval reasonable and consult technical mechanism. The necessary rules and possible rules should be compared with expert experiences and then made proper adjustment. Applying distributed RBF network structure, we can select the number of decision classification as number of subnet, and then divide the original problem with m decision attributes into m sub problems. To set up soft-sensor model for paper pulp Kappa value, decision values are divided into 3 interval and 3 sub networks are constructed. The last result is weighted average of decision values, so can keep balance between inconsistent rules.

When training RBF sub network, the number of hidden layer's nodes is determined by number of samples in decision table, while the centre value of transform function is chosen based on centre of interval of condition attributes, for example, a RBF sub network with 39 hidden layer nodes, whose decision value is 0, the centre value corresponding to node i is normalization of centre values for interval of sample i. Variances of the Gauss function are all set as 1.

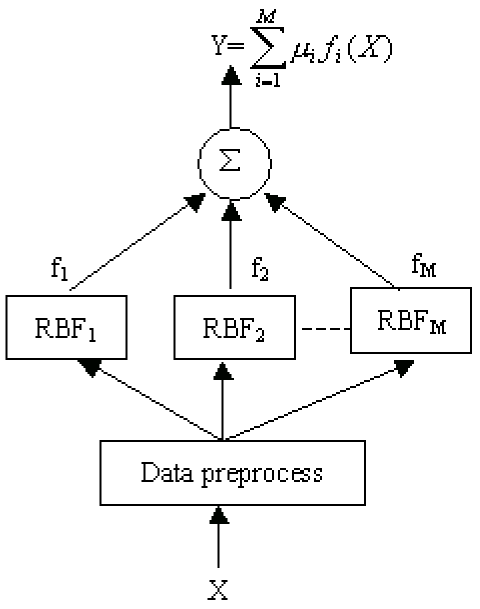


Fig.2. Distributed RBF network structure

(1) Normalization of data
Normalization of data is necessary for that measurement scale is difference between variables. Firstly we can easily get the maximum value domain by using statistical analysis because the technical variables are limited strictly in certain range. The transform formula is as following:
Provided that variety range is $[x_{\min}, x_{\max}]$, after normalizing, $x' = (x - x_{\min})/(x_{\max} - x_{\min})$.

To satisfy the requirement of precision, we can adjust the weight of output. For subnet work 1, where there are 38 hidden layer nodes and 41 training samples. The following table 7 illustrates how to train the three RBF subnet works.

Table 7 Training of the RBF sub-network

RBF Sub-network	Nodes of Hidden layer	Training samples	Training error
1	39	41	0.080
2	32	32	0.032
3	47	47	0.087

(2) Distributed model structure
we should keep to the following rules when select proper μ_i,

$$\alpha_i = \frac{\sum_{k=1}^{M_i} \frac{1}{d_k}}{M_i}, \mu_i = \frac{a_i}{\sum_{l=1}^{M} a_l} \qquad (2)$$

where M is number of sub-network (in the paper, M=3); M_i is number of training samples for sub-network i, d_k is square of Euclidean distance between testing samples and learning samples.

3.3 Verify the model

In previous work, we have developed a regressive equation for Kappa number of paper pulp based on kinetics mechanism of delignification (Luo and Liu, 2000).

$$K_a = A - B\ln[(H - H_b)(CEA_b)^n] \qquad (3)$$

where K_a is Kappa value, H is H factor
H_b, EA_b are H coefficient and effective alkali on mass delignification period separately.
C is function of sulfide degree, $C = \ln(S)$
A, B, R are coefficients to be determined.

Table 8 performance compare

Absolute Error	Mini-mum	Maxi-mum	mean	Standard deviation
RBFNN	0.08	9.20	2.332	2.1213
Mechanism-regression model	0.10	12.70	3.592	2.8041

Analyzing 40 groups of error tolerances absolute values, comparing them with predicted results of mechanism-regressive equation, it can be seen that the result of RBF neural network is better than former (table 8).

4. CONCLUSIONS

In this paper, a kinetic hybrid model is developed to predict the Kappa number on the lat phase of the batch cook process. The hybrid model is consisted of two modules: the lower essential module and the upper module. The essential part of the model is Radical Basis Function Neural Networks, which is composed by three sub networks. Considering some non-linear factors, such as the conflicts existing in the sample data, undetectable initial conditions, disturbances in cooking and so on, the upper part divides the whole secondary variables space into different fuzzy subspaces by applying expert knowledge and RS (Rough Set) data mining. In each space, the sub RBF network is trained to get better prediction. The final result is given by synthesize the outputs of the three sub network. Effectiveness of the proposed hybrid model is illustrated by the error between the predictive values and the data obtained from the lab. analysis in actual cooking process. It also indicates that the empirical model is effective for certain non-linear and complicated processes.

REFERENCES

Krzyzak, A. and Linder, T. (1998). Radical Basis Function Networks and Complexity Regulation in Function Learning. *IEEE Transactions on Neural Networks.* **Vol. 9(2)**, 247-256.

Luo, Q. and Liu, H.B.(2000). Modelling of the batch kraft pulping. Journal of South China University of Technology(Natural Science Edition). 28(1), 25.

Pawlak, Z. (1996). Rough Sets, Rough Relations and Rough Functions. *Fundamenta Informaticae.* 27: 103-108

Thompson, L.M., and Kramer, A.M. (1994). Modeling Chemical Process Using Prior Knowledge and Neural Networks. *AIChE Journal,* **Vol. 40(8)**, 1328-1339.

Wang, J. (1998). Data Enriching Based on Rough Set Theory. *Chinese J. Compuers.* **Vol. 21(5),** 393-400

Wang, Y.Y., and Shao H.H. (2001). Binary Decision System-based Rough Set Approach for Knowledge Acguisition. *Chinese J. Control and Decision.* **Vol. 16(3),** 374-377

Acknowledgement

The authors wish to acknowledge the support of this project by National 863 Project (2001AA413110) and National Science Foundation of China (60274033).

A MODULAR BATCH LABORATORY PROCESS

Rasmus Olsson and Karl-Erik Årzén

Department of Automatic Control
Lund Institute of Technology
Box 118, SE-221 00 Lund, Sweden

Abstract: In this paper a new batch laboratory process is described. The process is inexpensive, modular, and portable. Several processes can be connected together to form a multi-purpose batch pilot plant. The use of the process in process control education is described.

Keywords: Batch Control, Laboratory Processes, Control Education

1. INTRODUCTION

A good control engineering course should include hands-on experiments. A number of laboratory control processes are available commercially, e.g., inverted pendulums, helicopter processes, level-controlled water tanks, etc. These are commonly used in laboratory exercises in different basic and advanced control courses. However, very few processes exist that illustrate the typical problems associated with batch control.

In this paper a new laboratory batch tank process is presented. The process can be used stand-alone for illustrating the control problems associated with single-unit batch control. It is also possible to connect several processes together to form a multi-purpose batch cell. Used in this way the process can be used in teaching laboratories on recipe-based batch control and scheduling of batch processes.

The process consists of a water tank equipped with inlet and outlet pumps, a heating device, a cooling device, and an agitator. The sensors consist of a level sensor and a temperature sensor. The relatively large amount of actuators means that the number of discrete operations that can be performed is quite large (start/stop inlet pump, start/stop heating, etc). The actuators can also be controlled with analog signals. This means that the tank also can be used as a continuous jacketed tank reactor. The possibility to both cool and heat can be used to emulate the temperature behavior of different types of chemical reactions, e.g., the cooler can be used to simulate an endothermic reaction in the reactor. In this way the simulation of the reaction rate can be made temperature dependent, non-linear, or constant. The process can also be used for multi-variable control experiments, e.g., simultaneous control of level and temperature.

The process is made from standard components and therefore inexpensive. The process is also small, so small that it fits easily on the desk beside a standard PC used for control. The connection between the process and the PC is an ordinary RS-232 serial line. This also means that it is straightforward to control the process from an ordinary laptop PC at lectures and presentations.

[1] This work has been supported by the Center for Chemical Process Design and Control, CPDC, and the Swedish Foundation for Strategic Research, SSF. The development of JGrafchart is funded by the EC/GROWTH project CHEM aimed at developing advanced decision support systems for different operator-assisted tasks, e.g., process monitoring, data and event analysis, fault detection and isolation, and operations support, within the chemical process industries. The CHEM project is funded by the European Community under the Competitive and Sustainable Growth Programme (1998-2002) under contract G1RD-CT-2001-00466.

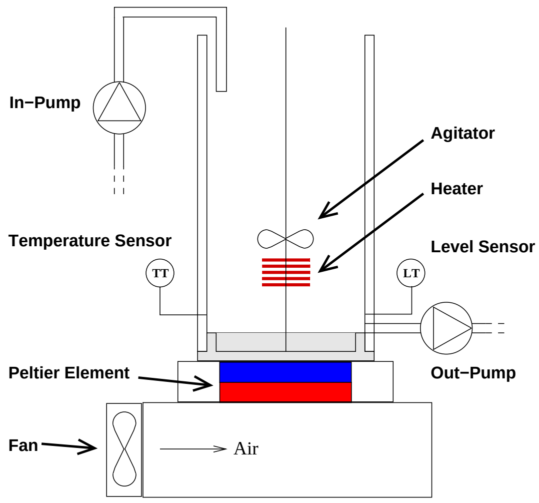


Fig. 1. Outline of the laboratory batch process.

1.1 *Outline of the Paper*

The process is described in detail in Section 2. Currently the process is used in the basic course in process control at the Department of Automatic Control, Lund Institute of Technology. Section 3 describes the laboratory exercise. A dynamic simulator for the batch tank process is also described. In the laboratory exercises the process control system is implemented in JGrafchart, a Java-based environment for graphical programming that combines the functionality of Grafcet/SFC from IEC 61131-3, with Statecharts and procedural and object-oriented programming concepts. A brief overview of JGrafchart is given in Section 4. By combining three tanks in series a multi-purpose batch cell can be emulated. This is described in Section 5.

2. PROCESS DESCRIPTION

The batch tank consists of a plexi-glass tube with a brass bottom. The volume of the reactor is approximately 0.5 liter. Connected to the reactor is an agitator, an in-pump, an out-pump, a level sensor, a temperature sensor, a heater, and a cooler, see Fig. 1. The cooler consists of a 40 W Peltier element cooled by a fan.

The outputs from the level and temperature sensors are in the range $0 - 10$ V, which corresponds to Empty-Full tank and $0 - 100^o$ C respectively. The resolution OF THE SENSORS can be up to 10 bits.

The pumps, heater, and cooler can either be controlled with digital, i.e. On/Off, signals or analog signals. The analog signal is transformed using an integrated microcontroller for pulse width modulation (PWM). The built-in microcontroller is an 8-bit ATmega8 from Atmel. The agitator only has a digital control signal, but to overcome the problem with friction the microcontroller is programmed to give a higher control signal right when the agitator is turned on. The heater is a 24 V, 150 W heating element. There are three light-emitting diodes (LED) on the front of the process: one red for showing if the heater is on, one blue for showing if the cooler is on, and one red/green LED, which is red if there is some error and green if the power is on and the process is OK.

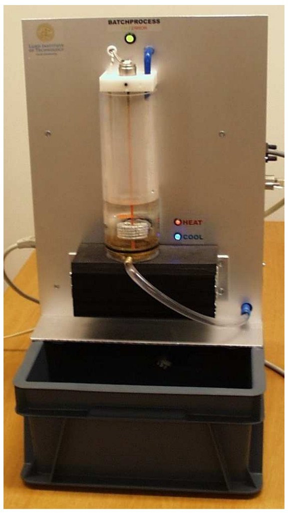


Fig. 2. The real process.

The microcontroller is used for safety interlocks, e.g., it is not possible to turn the heater on if the tank is empty, the in-pump will stop once the tank is full, and there is a protection against over heating. The microcontroller enables communication over an RS-232 serial line. This makes it possible to control the process with a laptop, e.g., at presentations and lectures, without using an IO card.

The water container below the process is an off the shelf plastic tank with the size: 15 cm high, 30 cm wide, and 40 cm deep. The container also becomes a protection for the front part of the process during transportation. The real batch process can be seen Fig 2.

The cost of building one process, without the development cost, is approximately US$1200, of which half is material and half is work cost.

3. A BATCH PROCESS LABORATORY EXERCISE

The process is used in a laboratory exercise in the course “Automatic Process Control”, a fourth-year course for chemical engineering students at the Lund Institute of Technology. The laboratory exercise introduces the students to discrete-event control of a batch reactor and it also requires the students to implement a discrete PID-controller.

Both the sequential control and the PID-controller are implemented in JGrafchart. The sequential control consists of the following steps:

I) Fill the reactor.
II) Heat the reactor, during simulation of an endothermic reaction, to a specified temperature using PID-control.
III) Empty the reactor.
IV) Clean the reactor.

The sequence should then be restarted and a new batch made. A solution in JGrafchart can look like in Fig. 4.

In the exercise the cooler is used to simulate an endothermic reaction in the reactor. This way the simulation of the rate of the reaction can be made temperature dependent, non-linear, or constant.

3.1 *Process Simulator*

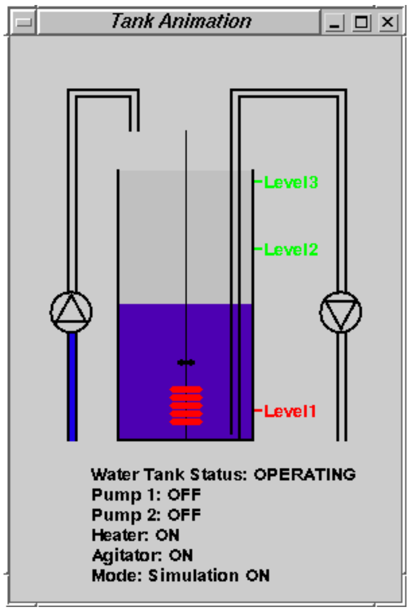
During the design of the hardware the temperature dynamics of the process was estimated to be a first order system with time constant $T \approx 1000$s (if the temperature step is not too large). The laboratory exercise would take a very long time if all the experiments should be made on the real process and therefore a simulation model has been developed. The simulation model can be used for development of controller schemes and parameter adjustments. The speed of the simulation can be changed.

To be able to plot signals in real time for the real process it was decided to construct a simulator server program that offers the following services:

a) A simulated batch process with the same interface as the real batch process.
b) Possibilities to plot temperature measurements, control and reference signals.
c) An animation of the batch process.
d) A socket interface so that clients can acquire all of the above services.

It was decided to implement the simulator server in Java. Two communication threads were implemented. One that sends measurement signals from the virtual tank to clients connected to the server, and one that receives commands from clients.

The simulated batch process was implemented as a subclass of an existing process simulation class package. The class package provides methods to simulate systems that are described by differential or difference equations in real time. The coefficients for the differential equations were identified from the real process. The class also provides methods to animate the process with the public Graphics2D-class package. A screen-shot from the simulation is shown in Fig. 3.



Water Tank Status: OPERATING
Pump 1: OFF
Pump 2: OFF
Heater: ON
Agitator: ON
Mode: Simulation ON

Fig. 3. Animation of the tank used both for the simulated and the real process.

A client, which connects to the server's socket can switch the simulation on and off. If the simulation is on the virtual process will react to the client's command signals and visualize the effect in the animation window. Moreover the signals will be plotted. If the simulation is turned off only the animation and plotting capabilities of the server are used to plot the signals from the real process.

4. JGRAFCHART

JGrafchart is the name of a Java-based version of Grafchart (Årzén, 2002), an object-oriented extension to Grafcet/Sequential Function Charts. JGrafchart consists of a graphical editor and a run-time system. In the graphical editor the user creates Grafchart sequential function charts by copying language elements from a palette using drag-and-drop. The language elements are placed on a workspace and connected together graphically. After compilation the function charts are executed by the runtime system within the JGrafchart machine. No native code generation takes place. A separate execution thread is used for each top-level workspace. The execution is periodic. In every cycle the inputs are read, a scan of the function charts is performed, and the outputs are written. JGrafchart is shown in Fig. 4. The drag-and-drop palette is shown to the left. JGrafchart is implemented in Java 1.4 and Swing. The graphical object editor class package JGo from Northwoods Software Corporation is used in the implementation of the graphical editor. The public domain parser generator JavaCC is used to generate parsing code for all textual expressions. JavaCC is also used to build abstract syntax trees. Storage to file is provided in the form of XML. The

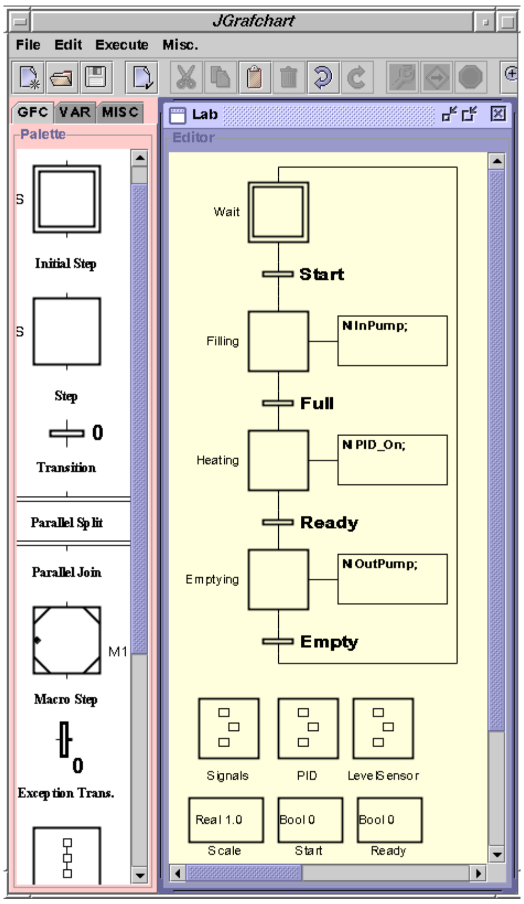


Fig. 4. Control sequence in JGrafchart

contents of each document can be stored as XML and later reloaded. It is also possible to store all top-level workspaces as a single XML file.

The main language elements are steps, transitions, macro steps, procedures, and procedure steps. A step represents a state where actions are performed. Four types of actions are supported: stored actions are executed when the step becomes active, exit actions are executed immediately before the step becomes deactivated, periodic actions are executed periodically while the step is active, and, finally, normal actions which are used to associate the truth value of a boolean variable with the activation status of the corresponding step. Transitions contains a boolean condition and/or event expression that decides when the transition should be fired. The grammar for transition expressions and step actions is defined formally using JavaCC.

JGrafchart supports four different data types: booleans, integers, reals, and strings. Using special workspace objects it is possible to create complex variables, e.g., structs. Workspace objects can also be used to encapsulate procedures, e.g., behavior. In this way it is possible to mimic simple object structures. JGrafchart can interact with the external environment in four different ways: using digital IO, using analog IO, using sockets, and using XML based communication. Using digital inputs and outputs JGrafchart can be used directly as a logical controller. Using analog input and output blocks JGrafchart can communicate with A-D and D-A converters. This functionality only works when executing on the Linux PCs in our laboratory. Each workspace can act as a client in a TCP socket connection connecting to some server, e.g., a simulator. Using socket input and output blocks it is possible to read and write variable values using a simple text-based protocol where each test message has the following structure: `variable identifier '|' value`. Finally, using XML input and output blocks, JGrafchart can communicate with xmlBlaster, an XML-based message-oriented middle-ware layered on top of XML-RPC or CORBA. The xmlBlaster server supports communication using the publish/subscribe method and using remote procedure calls.

The graphical editor provides a number of features, e.g., graphical selection of objects, move, delete, cut-copy-paste, undo-redo, change object size, group-ungroup, move-to-front, send-to-back. It is possible to change the grid size and color of a workspace. In addition to the Grafchart language elements the editor supports graphical objects, e.g., texts, rectangles, ellipses, icons, buttons, etc. Step actions can be associated with a button and executed when the button is pressed. It is also possible to dynamically change the attributes (position, size, color, and so on) of the graphical objects using step actions. In this way it is possible to implement GUIs in the form of animated process diagrams.

5. A LABORATORY BATCH CELL

The mixed discrete/continuous nature of batch production systems makes them a challenging control problem. In multi-purpose batch cells it is possible to produce multiple products at the same time using a finite set of equipment units. The units are organized in a network structure with a high degree of connectivity. The sequential information about the ordering of the operations, and the equipment unit types required for the different operations, is captured in a recipe. In order to execute an operation in an equipment unit, the batch control system must allocate the resource that the equipment unit constitutes. A control system for recipe-based batch production must, hence, include substantially more functions than what is needed in a control system for continuous production. An example of a batch pilot plant is the Procel plant at UPC in Barcelona, Spain. Procel has been used in several ways in the research of batch control, see e.g. (Cantón *et al.*, 1999; Ruiz *et al.*, 2001; Olsson, 2002).

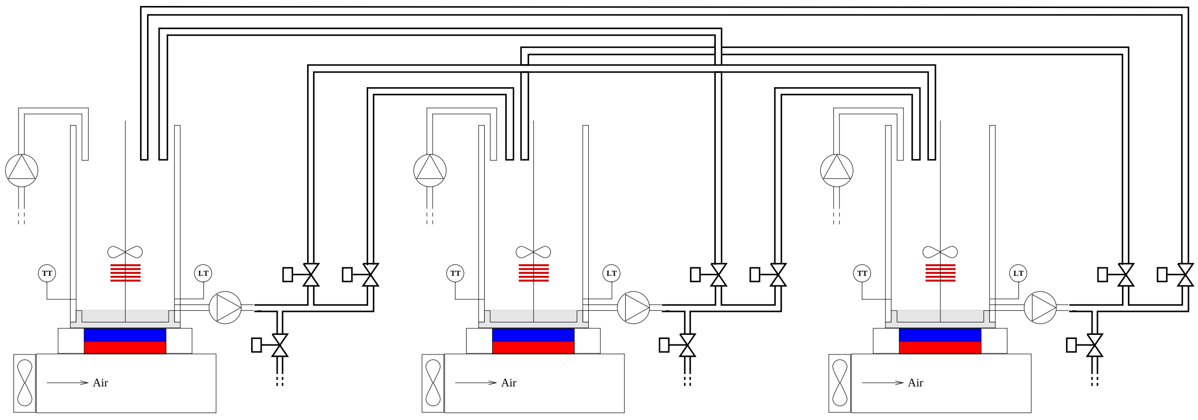


Fig. 5. Outline of the batch pilot plant consisting of three processes.

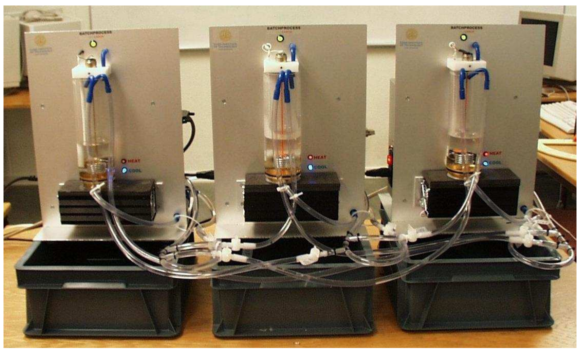


Fig. 6. Tank processes connected together to a small batch pilot plant.

Batch cells are by nature quite large (Procel is 4 m by 2 m by 1 m) and expensive, and not so well suited for teaching laboratory purposes. However, by connecting the developed single-tank processes together a small pilot batch cell plant can be created. As a test three processes have been connected together using plastic pipes and manual valves, see Fig. 5 and Fig. 6. The manual valves can easily be replaced with automatic magnetic valves if desired. The plant is highly flexible and connected in such a way that each of the tanks can transfer its content to any of the other tanks or empty it out to the water container below. The plant has the same functionality as the Procel pilot plant.

The control system is implemented in JGrafchart according to the S88 batch control standard (ISA, 1995; IEC, 1997). Each tank is modeled as an equipment unit in the control system. The equipment unit contains equipment operations or phases for performing the different control actions on the physical batch tank unit. The recipes are represented as JGrafchart procedures consisting of sequences of recipe operations or recipe phases. The recipe operations or phases call the corresponding equipment operations or phases using method calls. In JGrafchart it is also possible to include an animated process diagram that can be used as operator interface. The JGrafchart

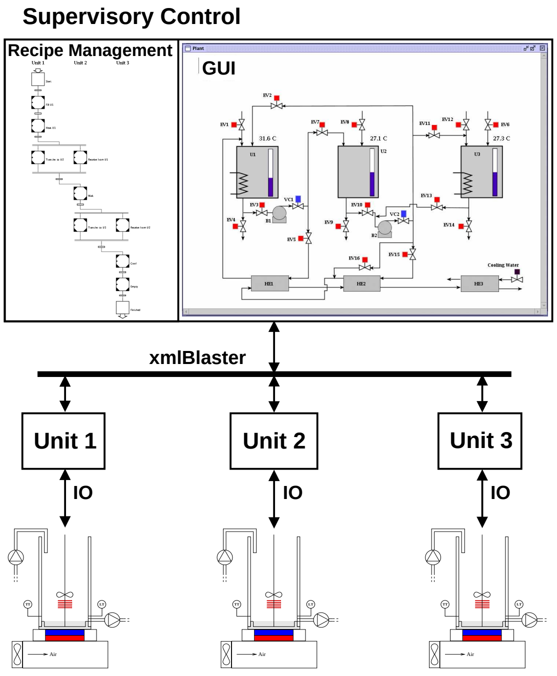


Fig. 7. Distributed control system according to the batch control standard S88.

solution can either be centralized executing on a single PC or distributed executing as multiple JGrafchart applications either within the same PC or on different PCs, e.g., one PC for each equipment unit and one supervisory PC for the recipe handling and the operator interface, see Fig. 7. The distributed solution relies on the possibility in JGrafchart to communicate using XML-messages.

To include the manual valves in the control system, dialog windows requesting the operator to manually open and close the valves have been added. Instead of opening the valve automatically it brings up the dialog and waits for the operator to confirm that the valves are in the correct position. A dialog window can be seen in Fig. 8. How JGrafchart can be used for operator support is described in (Årzén *et al.*, 2002). If automatic magnetic valves are added to the plant they can

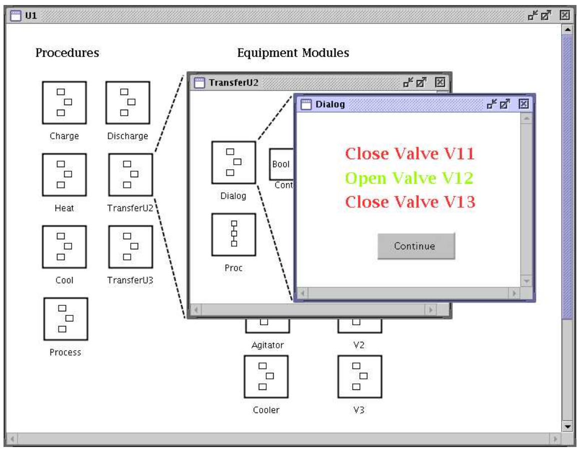


Fig. 8. Dialog window in JGrafchart for manual control of valves in a transfer operation

either be considered to be a part of the existing units or they can form a separate valve-battery unit. If the valves are grouped as a new unit the control system for the single tank process can be re-used without change. New operations need to be added for the transfer from one process to another but to a large content this could be part of the valve-battery unit. This way the plant could easily be extended to more units over time, only the control of the valve battery needs to be changed.

A recipe for the manufacture of a certain product in the plant may for example consist of the following sequential operations:

- Fill Unit 1.
- Heat Unit 1.
- Transfer to Unit 2.
- Wait.
- Transfer to Unit 3.
- Cool Unit 3.
- Empty Unit 3.

The recipe implemented as a procedure in JGrafchart is shown in Fig.9.

6. CONCLUSIONS

An inexpensive, portable, and flexible laboratory batch process has been developed. The process is used within the chemical engineering education at Lund Institute of Technology. The process can be used for teaching sequential, PID, multi-variate, and recipe-based control. It has also been used in the development of an exception handling strategy for batch control (Olsson, 2002).

A major advantage of the modularity and size of the process is that it can easily be disconnected and stored or transported. Once disconnected each process can be fitted into a bag and easily carried by one person.

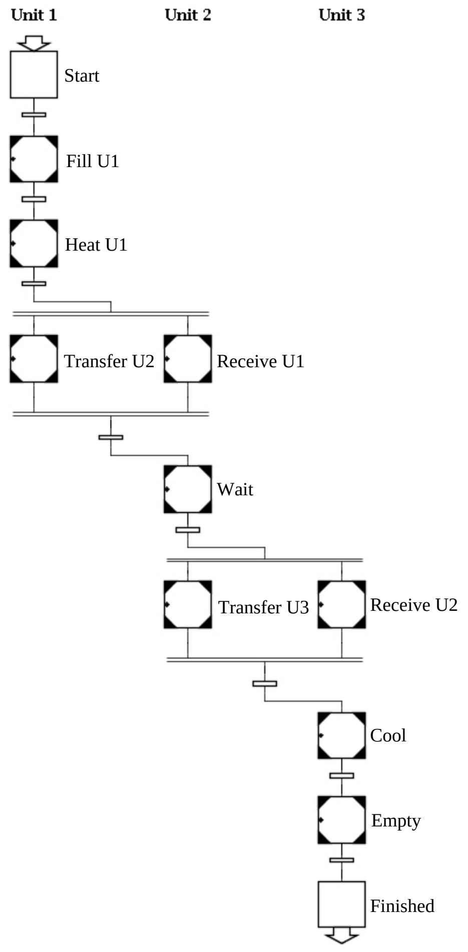


Fig. 9. A recipe implemented in JGrafchart

7. REFERENCES

Cantón, J., D. Ruiz, C. Benqlilou, J.M. Nougués and L. Puigjaner (1999). Integrated information system for monitoring, scheduling and control applied to batch chemical processes. In: *Proc. of the 7th IEEE Int. Conf. on Emerging Technologies and Factory Automation.*

IEC (1997). IEC 61512-1: Batch control, Part 1, Models and termonology. Technical report. International Electrotechnical Commission.

ISA (1995). ISA S88.01 batch control. Instrument Society of America.

Olsson, Rasmus (2002). Exception Handling in Recipe-Based Batch Control. Licentiate thesis ISRN LUTFD2/TFRT–3230–SE.

Ruiz, D., J. Cantón, J.M. Nougués, A. Espuña and L. Puigjaner (2001). On-line fault diagnosis system support for reactive scheduling in multipurpose batch chemical plants. *Computers and Chemical Engineering* **25**, 829–837.

Årzén, Karl-Erik (2002). JGrafchart. Home page: http://www.control.lth.se/∼grafchart/.

Årzén, K.-E., R. Olsson and J. Åkesson (2002). Grafchart for Procedural Operator Support Tasks. In: *Proceedings of the 15th IFAC World Congress, Barcelona, Spain.*