ONE-STEP-AHEAD PRODUCT PREDICTOR FOR PROFIT OPTIMIZATION OF PENICILLIN FERMENTATION

J. Q. Yuan[1,2] and P. A. Vanrolleghem[2]

[1] State Key Laboratory of Bioreactor Engineering , ECUST, 130 Meilong Road,
200237 Shanghai, PR China
[2] BIOMATH Department of University Gent, Coupure Links 653, B-9000 Gent, Belgium

Abstract: In antibiotics industry, the titre in bioreactors is the most important process variable both for process supervision and scheduling. It is therefore of great significance to develop a software sensor to predict the product formation. In this contribution, a pseudo dynamic product predictor based on artificial neural network is designed. The input process variables of the predictor include substrates, precursor, nutrients and oxygen consumption, carbon dioxide evolution and also the product formation. These process variables are usually available in practice. Only the accumulated values of these variables rather than the instant ones are used. A feedforward neural network is chosen for one-step-ahead prediction of total product titre at the next step. The input vector of the neural network is the time series of process variables over a predetermined time period. The database for network training is composed of a series of such vectors taken from historical charges which are processed as well as updated by a moving-data-window's technique. The software sensor is tested by data of three industrial charges and ten model "generated" batches. MATLAB is used for training and testing of the neural network. The predictor is expected to be applied in optimal fermentation time scheduling for antibiotics production. Improvements and further development of the software sensor for real time applications are discussed.

Keywords: Software sensor, product prediction, neural networks, biotechnology, process optimization.

1. INTRODUCTION

Application of neural networks to modeling and control of bioprocesses has been investigated for a long time. Many encouraging results have been obtained by applying ANNs to bioprocess state estimation, modeling, pattern recognition and control (Karim and Rivera, 1992; Psichogios and Ungar, 1992; Raju and Cooney, 1992; Schubert *et al.*, 1994; Thibault *et al.*, 1990). Some industrial application oriented investigations have shown a promising prospect of neural networks. Montague and Morris (1994) applied neural network models in biomass prediction and fault diagnosis for the penicillin production operated by SmithKline Beecham (Irvine, UK). Linko *et al.* (1995) successfully applied a dynamic neural network to predict product formation and substrate consumption for commercial lysine production. In this contribution, a neural network configuration similar to those reported in the literature is used. Our purpose, however, is to develop a product predictor that could meet the requirement of real time applications by using only routinely available data. It is expected to be applied to predict the profit function, a central part in on-line profit optimization.

It is a common experience that in antibiotics factories, the titre in a bioreactor is the most interesting process variable. This leads to the necessity of designing a software sensor for product formation. Since the product concentration is frequently sampled and analyzed (though with time delay), the software sensor can be "calibrated" by routine analysis data supplied during industrial production. This makes it to be easily adopted by end-users. Besides the product concentration, other variables routinely available such as OUR, CER, substrate consumption, precursor consumption etc. should also be treated as input variables. Biomass concentration is a very important variable in any fermentation process, but it is usually not routinely analyzed in antibiotics production so that it can not be included from an industrial application point of view. In many investigations in the literature, instant values of the concerned variables are used for the purpose of state estimation. The instant values may be corrupted by measurement noise, especially the measurements like OUR, CER and product concentration. Therefore, in our product predictor, integrated or accumulated values of these variables are chosen as input. This does not mean, however, any loss of dynamic information. Since the input vectors of the neural network are taken from the transients of the accumulated variables, they contain therefore both stationary and dynamic process information over the specified operation period. The neural network is said dynamic since it accepts this dynamic information as input.

2. DATA PRETREATMENT AND THE NEURAL NETWORK TOPOLOGY

Once a charge is started, the following variables are closely correlated with its dynamic behaviour.

(1) Initial medium composition, off-line measurable
(2) Volume of medium, on-line measurable
(3) Feeding rate of all substrates and nutrients, on-line measurable
(4) Intermediate withdrawal of the medium, on-line or off-line measurable
(5) OUR and CER, on-line measurable
(6) Temperature, dissolved oxygen and pH, on-line measurable
(7) Antifoam agent and alkali consumption, on-line or off-line measurable
(8) Other off-line analysed variables

In a well equipped fermentation system, the measurements of these variables are usually available. For neural network training, these data are pretreated in such a way that at any time, as soon as the latest sampling data of product concentration is available, one knows how much product and carbon dioxide have been produced and how much oxygen, glucose, precursor and other nutrients has been consumed. Fig. 1 shows a typical time course of some of the pretreated input variables. The data originate from an industrial scale charge. For reasons of confidentiality, the real scales of the figure (as well as the other figures in this text) have been removed. The measurement noises of the original records were relatively high. However, after integration, they change much more smoothly. Other variables which correlate with product formation but measured at a higher frequency, such as temperature, pH and PO_2, are processed using a moving-averaging technique to get step-wise changing profiles.

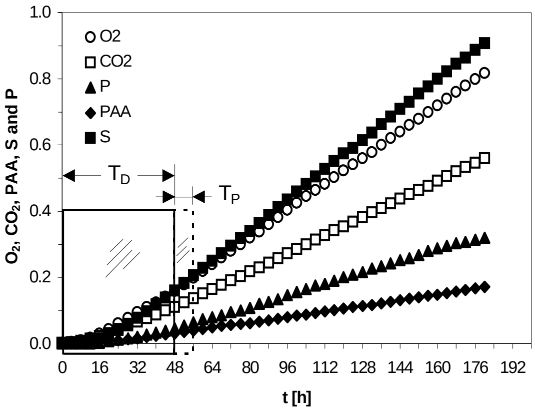

Fig. 1 Part of input variables of an industrial charge (Charge 1) used in the software sensor testing for penicillin titre prediction

Assuming the sampling time interval fixed, all profiles of accumulated and averaged variables of a historical charge will be cut into a series of dynamic pieces via moving data windows' technique so as to obtain the input and output data vector for the neural network. Fig. 1 shows the first data window (solid frame) and output window (or prediction widow, dotted frame). The width of these two windows is denoted by T_D and T_P, respectively. There are some principles to determine the width of the windows. A larger T_D involves more dynamic process information, but it may increase the dimension of the input data vectors since, in order to keep the discretisation accurate, the discretising time interval can not be increased accordingly. If T_D is too small, then the network may be too sensitive to measurement noises. As for T_P, generally speaking, it should not exceed T_D.

The input vector $X(T_k)$ is represented by Eq. (1):

$$X(T_k)=[\ T_k\ x(T_k)\ x(T_k-1\tau)\ x(T_k-2\tau)\ \cdots\ x(T_k-m\tau)\ ]^T \qquad (1)$$

where,

$$x(T_k)=[\ O_2(T_k)\ CO_2(T_k)\ P(T_k)\ PAA(T_k)\ S(T_k)\ Nit(T_k)\ Temp(T_k)\ pO_2(T_k)\ pH(T_k)\ ALK(T_k)\ OIL(T_k)\ \cdots\ ]^T \qquad (2)$$

In Eq. (1) and (2), k represents the k*th* data window;

T_k the fermentation time at the right border of the k*th* data window; and $X(T_k)$ the k*th* input vector. m is the number of dating back step and τ the time interval for discretisation of the input variables. Hence, $m\times\tau$ determines the time span which both the data window and $X(T_k)$ covers. The other symbols in $x(T_k)$ are accumulated or averaged values of the related process variables at time T_k. Their meanings and units are given in the Nomenclature.

For one-step-ahead prediction, the output data pair, $Y(T_k)$, is given by Eq. (3):

$$Y(T_k)=P(T_k+T_P) \qquad (3)$$

For demonstration, T_P takes the value of 1τ. For a historical charge with a cultivation period of Tf, the number of input-output data-pairs, N, is readily calculated by:

$$N = \mathrm{int}(\frac{Tf - T_D - T_P}{T_M}) \qquad (4)$$

where, T_M is the moving step length of the data windows.

Clearly, the number of input nodes of the neural network in the software sensor is determined by the dimension of $X(T_k)$. Determination of τ, m and selection of variables in $x(T_k)$ is critical for a successful design of the software sensor. In antibiotics industry, the sampling interval is typically 4 hours. If τ takes the same value as the sampling interval, m is chosen as 8 and the input variables are chosen as the first ten elements in $x(T_k)$, then the input vector will cover a time span of 8×4=32 hours and the number of input nodes should be $1+(8+1)\times10=91$. Furthermore, if a two layer feedforward neural network is used and five neurons in the hidden layer are chosen, then there are $91\times5=455$ weights to be determined, a rather large number.

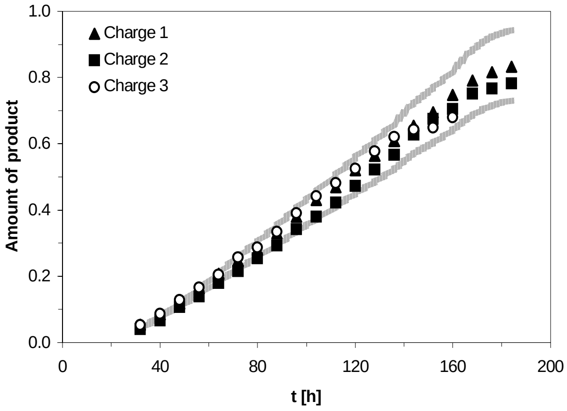

Fig. 2 Time course of product formation during 3 industrial scale charges (symbols) and the product change region of 10 model-generated batches (area between shadow lines).

There are several ways to simplify the problem. First, a larger τ, e.g. 8 hours, may be used. In this case, significant loss of dynamic information may be avoided by keeping shorter moving step (4h for antibiotics fermentation) of the data window so that the database still contains almost the same dynamic feature of the process. The advantage of doing this is the drastic reduction of the number of the input nodes and synchronizing with shift time (usually 8h) in industrial practice. Second, only the most important variables should be involved in the input data vector. Some relatively constant variables such as pH and T could possibly be excluded. And at last, trial and error can be used to reduce the number of neurons in the hidden layer(s). In our case, for example, even 3 neurons in the hidden layer may give a relatively satisfactory result.

3. TESTING RESULTS

The data for the software sensor testing consist of two parts. On the one hand, there are three industrial scale charges and, second, ten model-generated batches. The symbols in Fig. 2 show the time course of the product formation during Charge 1-3, while the area between the two shadow lines indicates the region of product changes of the model-generated batches. The model is a combined cell age and kinetic model which has been verified by industrial scale experiments (Yuan *et al.*, 1997b). The model parameters were identified with the data of an industrial batch (Charge 1). The model-generated batches were obtained by changing initial biomass concentration, feeding rate, and some critical model parameters such as the maximum product formation rate, yield coefficient and maintenance coefficient. To a more or less extent, these simulated data reflect the process fluctuations observed in real production. At the moment, usage of the simulated batches for neural network training is necessary due to the limited number of real data sets the authors had access to. In future applications, about 10 to 40 real charges are available for neural network training.

By choosing m and τ as 3 and 8h, respectively, the input vector contains discrete input variables over a period of 24 hours and the output is the amount of product 8h ahead. The input variables include O_2, CO_2, P and S. Precursor and nitrogen consumption as well as other variables could not be included at present because the major part of the input data are generated by the model system that does not involve these state variables. Therefore, the input data vector $X(T_k)$ contains only 17 elements:

$$X(k)=[\ T_k\ O_2(T_k)\ CO_2(T_k)\ P(T_k)\ S(T_k) \cdots\cdots O_2(T_k\text{-}24)\ CO_2(T_k\text{-}24)\ P(T_k\text{-}24)\ S(T_k\text{-}24)\ ]^T \qquad (5)$$

Denoting θ_1, θ_2, θ_3 and θ_{sim} as the input-output data set for Charge 1, 2, 3 and simulated batches, respectively, one obtains three combinations for training and testing of the neural network:

Case 1: (θ_2, θ_3, θ_{sim}) for training, θ_1 for testing
Case 2: (θ_1, θ_3, θ_{sim}) for training, θ_2 for testing
Case 3: (θ_1, θ_2, θ_{sim}) for training, θ_3 for testing

The neural network used here is a two layer tansig/purelin network, i.e., a tan-sigmoid transfer function for the hidden layer and a linear transfer function for the output layer, one of the standard structures in the MATLAB Neural Network Toolbox (Demuth and Beale, 1994). The output layer contains one neuron since only the product formation at the next step will be predicted.

Fig. 3 shows the one-step-ahead prediction results of a 17-3-1 network. The product predictions for Charge 1 and 2 are quite close to their measurements. However, for Charge 3 the prediction error is high after 140h. The reason for that may lie on the incomplete training database (θ_1, θ_2, θ_{sim}). Indeed, most of the data used for training are model simulations and the dynamic information of charges like Charge 3, which reached the saturation phase earlier than usual, were not included. On the other hand, considering that only four input variables are used, the results in Fig. 3 are rather promising. In real applications, two more input variables (precursor and nitrogen) and much more training batches will be available which may enhance the robustness and accuracy of the predictor.

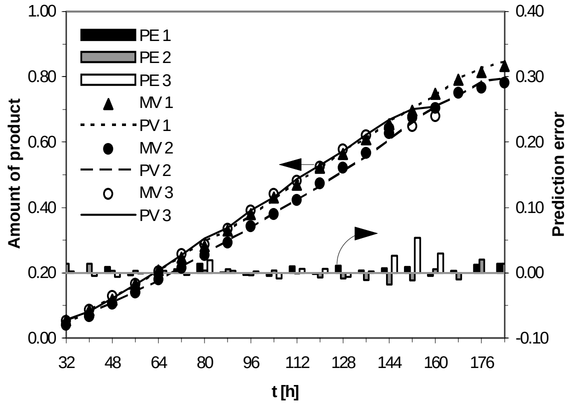

Fig. 3 Comparison between measurements (symbols) of product and the predictions (lines) by an one-step-ahead predictor based on a 17-3-1 neural network. PE, MV and PV means the prediction error, measured value and predic-tion value. 1~3 corresponding to Charge 1~3.

Fig. 4 shows the influence of the number of neurons in the hidden layer on the training and prediction accuracy. Here only Case 2 and 3 are focused upon (Case 1 is not very interesting because most of the training data, the model generated batches, are based on the parameters of Charge 1). Since the initial weights and biases are generated randomly which results in a difference between the final training results, the values plotted in Fig. 4 take the average of ten training-prediction exercises. One may find the general tendency of decreasing training error (solid symbols) as the number of hidden neurons increases. However, this does not apply for the prediction error (empty symbols). On the contrary, the prediction error corresponding to 15 hidden neurons is the highest in both cases which indicates the tendency of overfitting. Hence, 3 hidden neurons seem to be acceptable by balancing computation efforts and errors (of training and prediction).

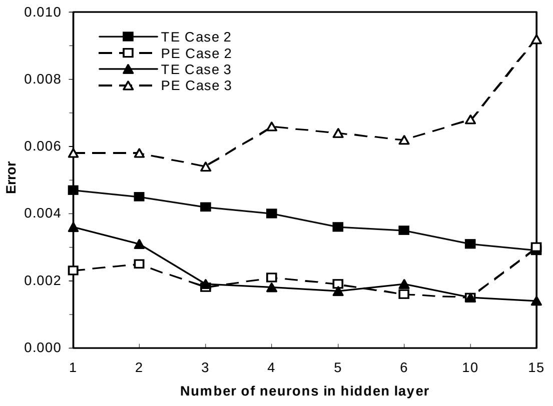

Fig. 4 Training error (solid symbols) and prediction error (empty symbols) as function of the number of hidden neurons for Case 2 and 3. TE: Training error; PE: Prediction error.

The Levenberg-Marquart optimization approach is chosen for backpropagation training. By using a 486DX computer with 32M extended memory and under the predefined sum-squared error goal of 10^{-3}, training usually finished within several minutes if the number of hidden neurons ≤ 6. In the case of 15 hidden neurons, training time is as much as 50 minutes. Therefore, if the number of neurons in the hidden layer can be limited to around 5-6 in industrial case, the training speed would be fast enough to meet the requirements of on-line application.

4. DISCUSSION AND CONCLUSION

The one-step-ahead product predictor described here focused on the prediction of total product. The predictor is a basic element of the profit optimization system described by Yuan *et al.* (1997a) because it enables the on-line prediction of the revenue. Revenue estimation is the centrepart in process behavior classification (from the economic point of view) and subsequent on-line optimal scheduling for a multi-fermenter plant. Besides, the software sensor may give an earlier alarm signal for possible abnormal batches such as contamination or other stoppages if the predicted product formation is significantly lower than average values.

The predictor itself will be “calibrated” with routine off-line analysis data of the product concentration. Calibration means that the training data set is regularly updated and the training process repeated as soon as the complete data set of the latest charge is available. The use of accumulated variables is a distinguished feature of the software sensor. It is advantageous for eliminating the influence of measurement noises but by no means at the cost of dynamic information contained in instant data. The accumulated input variables are broken down into a series of dynamic pieces via the moving-data-window technique to form the input-output data-pairs for the neural network. The shortest moving step of data windows is equal to the sampling interval.

The precondition for the software sensor to work is a reliable mass balance both in liquid and gas phases. Selection of the input process variables, τ, m and the neural network’s topology plays also an important role to the performance of the software sensor. For penicillin production, it seems that six input variables are necessary: O_2, CO_2, P, S, PAA and N, although in this paper only the first four variables were used for lack of enough data. For other bioprocesses, the number of input variables may probably be limited at the same level after careful investigation on the process kinetics and working conditions. Note that variables like biomass in mycelia cultivations which are usually not routinely available will be excluded from the input process variables. τ and m are two process dynamics dependent factors. For most industrial bioprocesses, τ may change between 1 and 8h and m between 2 and 5.

The number of neurons in the hidden layer should be as few as possible because, in the case of industrial applications, 30 to 40 input nodes may exist and the training database may include one thousand input-output data-pairs. A simpler structure will surely result in a shorter training time. Too many neurons in the hidden layer may not be always necessary or even cause overfitting (Fig. 4). Since the backpropagation training algorithms are sensitive to the number of hidden neurons (Demuth and Beale, 1994), trial and error should be used to determine how many hidden neurons are required. Empirically, the number of unknown weighting factors and biases in the network should be much less than that of input-output data-pairs in the training database. It is important to note that the sampling interval of the original data should be kept constant. Otherwise more additional input nodes are needed since the sampling time at each discrete point must be fed into the neural network as well, rather than only one input node for sampling time T_k as shown in Eq. (5).

In order to obtain a satisfactory prediction accuracy, the training database must be a representative one. That means, on the one hand, it should include as much as possible situations occurring in industrial cultivations except contaminations and other extraordinary charges (the software sensor works most effectively under normal fluctuations of production). On the other hand, the training database should consist of recent historical charges and it should be updated as soon as the latest charge is finished. The time span covered by the database should be reasonably short. This may eliminate the influence of some gradually changing factors, such as climate and deterioration of equipment. Further-more, it could happen that some fermenter, for its structural reasons, has evidently different behavior as the average level of other fermenters. In this case an extra database which consists of charges carried out only by this fermenter should be built up and a special predictor becomes necessary. Some principles to deal with such kind of special case have been described elsewhere (Yuan et al, 1997a).

In this paper, only one-step-ahead prediction is tested. For antibiotics production, the prediction time span corresponds to ca. 8 hours. Actually, such a prediction, if it is reliable, can be already a most valuable support for process supervision and profit optimization. Theoretically, a multi-step predictor may be designed in the same way. However, care should be taken because the neural network basically only performs a nonlinear regression and extrapolation. The software sensor makes prediction based on the context among the major process variables so that it contains some most important information on the process kinetics. Therefore, after successful network training, a certain robustness can be expected.

During previous discussion, database come from historical charges is assumed. Actually, the data collected so far in the charge of present interest can be also incorporated into the training database. This may make the predictor even more accurate and more robust, since these data contain individual characteristics of the present charge which may not have occurred in the historical charges. Such individual characteristics are usually caused by inherent quality fluctuations of precultures, composition changes of substrates and other ill-known disturbances during the earlier phase of cultivation. In this context, the basic structure of the product predictor described above has been recently improved and a rolling learning-prediction procedure is proposed. In this procedure, the training database is extended as the process progresses and the learning-prediction is iteratively repeated every time the database is updated with the analysis results of the latest sample. By using the rolling learning-prediction procedure, testing results revealed that a multi-step prediction up to 24h ahead can be realized for penicillin production. Furthermore, a prediction accuracy of higher than 5% has been obtained for ten industrial charges (submitted paper by Yuan and Vanrolleghem, 1997).

ACKNOWLEDGEMENT

The financial support of Alexander von Humboldt Foundation/Germany and Aplitek NV, Deinze /Belgium is greatly acknowledged. The authors are grateful to professor K.-H. Bellgardt from the Institute of Chemical Engineering of University Hannover/Germany for his valuable advises and discussions. Part of this work was supported by a grant (G.0286.95) of the Fund for Scientific Research (F.W.O), Belgium.

REFERENCES

Demuth, H. and M. Beale (1994). *Neural Network Toolbox User's Guide*. The Math Works, Inc.

Linko, S., Y.-H. Zhu and P. Linko (1995). Neural networks in lysine fermentation. In: *Computer Applications in Biotechnolg: A postprint volume from the 6th Intl. Conf. on Computer Application in Biotechnology, Garmisch-Partenkirchen, Germany, 14-17 May 1995* (A. Munack and K. Schuegerl, Eds.), pp 336-339. Pergamon Press, London.

Montague, G and J. Morris (1994). Neural-network contributions in biotechnology. *Trends in Biotechnology*, **12**, 312-324.

Psichogios, D. C. and L. H. Ungar (1992). A hybrid neural network-first principles approach to process modeling. *AIChE Journal*, **38**, 1499-1511.

Raju,G. K. and C. L. Cooney (1992). Using neural networks for the interpretation of bioprocess data. in "*Modeling and Control of Biotechnical Processes 1992*" *IFAC Symposia Series, 1992, Number 10*, (M. N. Karim and G. Stephanopoulos, Eds.), pp. 425-428. Pergamon Press, London.

Schubert, J., R. Simutis, M. Dors, I. Havlik and A. Luebert (1994). Bioprocess optimization and control: application of hybrid modeling. *J. Biotechnology*, **35**, 51-68.

Thibault, J. V. C. Van Breusegem and A. Cheruy (1990). On-line prediction of fermentation variables using neural networks. *Biotechnol. Bioeng.*, **43**, 1041-1048.

Karim, M. N. and S. L. Rivera (1992). Artificial neural networks in bioprocess state estimation. *Adv. Biochem. Eng./Biotechnol*. **46**, 1-31.

Yuan, J. Q., S. R. Guo, K. Schuegerl and K.-H. Bellgardt (1997a). Profit optimisation for mycelia fed-batch fermentation. *J. Biotechnol.*, **54**, 175-193.

Yuan, J. Q., S. R. Guo and K.-H. Bellgardt (1997b). A cell age model for penicillin cultivation. In: *Application in Modelling and Simulation: Proceedings of 15th IMACS World Conference on Scientific Computation, Modelling and Applied Mathematics, Berlin, Germany, 24-29 September 1997, Vol. 6* (A. Sydow, Ed.), pp. 177-182. Wissenschaft & Technik Verlag, Berlin.

NOMENCLATURE

CER	carbon dioxide evolution rate, kmole h^{-1} m^{-3}
OUR	oxygen uptake rate, kmole h^{-1} m^{-3}
CO_2	total carbon dioxide production, kmole
m	dating back steps for discretising input variables
N	number of input-output data-pairs for a historical charge
Nit	total nitrogen source consumption, kg
O_2	total oxygen consumption, kmole
OIL	total vegetable oil consumption, kg
P	total product formation, kg
PAA	total precursor consumption, kg
pH	average pH value
pO_2	average dissolved oxygen in the medium, % of saturation level
S	total titratable sugar consumption, kg
Temp	average temperature of the medium, ^oC
T	fermentation time, h
T_D	width of input data windows, h
Tf	fermentation period of a charge, h
T_M	moving step length of data windows, h
T_P	width of output or prediction data windows, h
$X(T_k)$	neural network's input data vector obtained by discretising the input variables within k*th* input data window
$Y(T_k)$	output of the neural network. In the case of one-step-ahead prediction, it is equal to the amount of product formation at time T_k+T_P, kg
θ	input-output data set
τ	step length for discretising input variables, h