On-line Identification of Multivariable Processes Using EKF Learning-based Adaptive Neural Networks

Karim Salahshoor
Department of Automation and Instrumentation
Petroleum University of Technology
Tehran, Iran
salahshoor@put.ac.ir

Amin Sabet Kamalabady
Department of Automation and Instrumentation
Petroleum University of Technology
Tehran, Iran
a.sabet_kamalabady@yahoo.com

***Abstract*—This paper presents online identification of multivariable processes with time-varying and nonlinear behaviors using two adaptive learning approaches for radial basis function (RBF) neural networks. These approaches are called as growing and pruning algorithm for radial basis function (GAP-RBF) and minimal recourse allocation network (MRAN). The extended kalman filter (EKF) is proposed as learning algorithm to adapt the parameters of multi-input, multi-output (MIMO) RBF neural network in both GAP-RBF and MRAN approaches. Some desired modifications on the growing and pruning criteria in the original GAP-RBF have been proposed to make it more adequate in online identification. The performances of the algorithms are evaluated on a highly nonlinear and time-varying CSTR benchmark problem for comparison purposes. Simulation results show the better performance of the modified GAP-RBF (MGAP-RBF) neural network with respect to the original GAP-RBF and MRAN algorithms.**

***Keywords*— multivariable process, on-line multivariable identification, MRAN, GAP-RBF, EKF.**

I. INTRODUCTION

Modeling of multivariable nonlinear time-varying processes of increasing complexity is an important need in process industries. Because, obtaining accurate mathematical models for actual processes results in obtaining accurate control strategies for them. However, these accurate mathematical models are difficult to be achieved. Therefore, empirical models which are constructed based on experimental data offer a better practical approach. On the other hand, off-line black-box models cannot be accurate representations for the actual nonlinear time-varying processes. This problem can be solved by using online model identification approaches which have been considerably considered in recent years. Neural networks (NNs) are excellent tools for modelling of nonlinear time-varying processes due to their functional approximation capabilities and availability of effective learning algorithms.

Radial basis function (RBF) neural network is a well established type of neural network to be used for online identification approaches because of its simple structure, ease of implementation and faster training process.

Online learning of RBF network (RBFN) is an important subject. Recourse allocation network (RAN), proposed by Platt [1], is a well-known sequential learning algorithm for RBF network (RBFN) which is suitable for online modeling of non-stationary processes. In RAN approach, the network begins with one hidden neuron when the first observation data reaches. Then, it grows by creation of new hidden neurons based on the novelty of the new data and the NN parameters are adapted using least mean square (LMS) method. Using a more efficient learning method for NN parameters adaptation, can lead to a more parsimonious network. Niranjan and Kadirkamanathan [2] used an extended kalman filter (EKF) method for NN parameters adaptation. The minimal RAN (MRAN) [3,4] is based on the RAN-EKF in which the pruning strategy is incorporated to remove inactive hidden nodes and hence decrease the NN dimension.

Recently, a sequential learning algorithm called as growing and pruning for radial basis function (GAP-RBF) neural networks have been proposed by Huang el al. [5]. In GAP-RBF learning algorithm, the hidden neuron with nearest center to the observation data is pruned or added according to its significance otherwise, its parameters are updated. Significance of a hidden neuron is defined as a neuron's statistical contribution over all the inputs seen so far to the overall performance of the network. Therefore, GAP-RBF algorithm has a low computational complexity, because only the nearest neuron is checked for significance or parameter adaptation.

The NN parameters adaptation can be performed by different methods such as recursive orthogonal least square (ROLS), recursive least square (RLS), LMS and EKF which have been used in previous works. In this paper, the extended kalman filter (EKF) is being used as an effective method for NN parameters adaptation. The main focusing, in this work, is on the GAP-RBF and MRAN approaches for online identification of a highly nonlinear and time-varying multivariable CSTR benchmark process. The main advantage of these network architectures is that their dimensionality is not predetermined but grows incrementally along with the complexity of the model. Therefore, the resulting NNs find automatically the complexity of the problem. A modification of GAP-RBF, called as MGAP-RBF, has been proposed in this paper. Through this modification, the neuron growing and pruning criteria have been explored to generate a more smooth, compact and robust NN model.

978-1-4244-1674-5/08 /$25.00 ©2008 IEEE

The paper is organized as follows. Section II describes the development of MRAN, GAP-RBF and modified GAP-RBF neural networks for multivariable identification. The EKF learning algorithm is presented in Section III. The performance of different adaptive MRAN, GAP-RBF and MGAP-RBF neural networks being adapted by the EKF learning algorithm have been evaluated on a multivariable nonlinear and time-varying CSTR benchmark problem in Section IV. Section V summarizes the resulting conclusions.

II. MRAN AND GAP-RBF NEURAL NETWORKS

A. MRAN Neural Network

The MRAN is a suitable algorithm for online training of Gaussian RBF network to model an unknown system dynamic. For a multivariable system, the MRAN NN output is calculated as follow:

$$\hat{y}_{ni} = f_{NNi}(x_n, \theta_i) = \sum_{k=1}^{K} \alpha_{ki}\, \Phi_k(\mathbf{x_n}) \qquad (1)$$

$$(\hat{y}_{ni} \in \hat{\mathbf{y}}_{\boldsymbol{n}} \;;\; i = 1, \dots, m)$$

Where $\hat{y}_{ni}$ denotes the ith network output corresponding to the ith system output, $\Phi_k(.)$ is a continuous nonlinear Gaussian function corresponding to the response of the kth hidden neuron. $\mathbf{x_n} \in R^l$ is the input regression data vector (in this paper, bold letters indicate the vectors), θ_i comprises the set of parameters to be identified by the learning algorithm and α_{ki} indicates the connecting weights to the ith output neuron. $\Phi_k(.)$ is defined by:

$$\Phi_k(\mathbf{x_n}) = \exp\left(-\frac{\left\|\mathbf{x_n} - \boldsymbol{\mu}_{\mathbf{k}}\right\|^2}{{\sigma_k}^2}\right) \qquad (2)$$

Where $\boldsymbol{\mu}_{\mathbf{k}} \in R^l$ is called the center of the RBF neuron and σ_k indicates the spread of the neuron. ||.|| represents the Euclidean norm.

During the online learning, as the new input and output pairs $(\mathbf{x_n}, \mathbf{y_n})$ are sequentially collected, the following algorithm is performed to construct the MRAN model:

1) Allocation of a new hidden unit: The new hidden neuron is added if the following three conditions are satisfied:

a) Outputs estimation error must be bigger than a threshold (e_{min}), i.e.,

$$\|\mathbf{e_n}\| = \|\mathbf{y_n} - \hat{\boldsymbol{y}}_{\boldsymbol{n}}\| > e_{\min} \qquad (3)$$

$$(e_{ni} \in \mathbf{e_n}, y_{ni} \in \mathbf{y_n}, \hat{y}_{ni} \in \hat{\mathbf{y}}_{\boldsymbol{n}} \;;\; i = 1, \dots, m)$$

b) The nearest center distance must be bigger than a threshold ϵ_n:

$$d_n = \begin{cases} \min\limits_k \|\boldsymbol{x_n} - \boldsymbol{t_k}\| & (k \neq 0) \\ \epsilon_{max} & (k = 0) \end{cases} > \epsilon_n \qquad (4)$$

$$\epsilon_n = max\{\epsilon_{max}\gamma^n, \epsilon_{min}\} \qquad (0 < \gamma < 1) \qquad (5)$$

c) The windowed RMSE error must be bigger than a threshold e'_{min}:

$$e_{rms} = \sqrt{\frac{\sum_{i=n-(M-1)}^{n} \|\mathbf{e_i}\|^2}{M}} > e'_{min} \qquad (6)$$

When all the three criteria are satisfied, a new neuron K+1 is added to the network with the following parameters:

$$\begin{cases} \boldsymbol{\alpha}_{\mathbf{K+1}} = \mathbf{e_n}, \\ \boldsymbol{\mu}_{\mathbf{K+1}} = \mathbf{x_n}, \\ \sigma_{K+1} = \kappa\left\|\mathbf{x_n} - \boldsymbol{\mu}_{\mathbf{nr}}\right\| \end{cases} \qquad (7)$$

Where, κ is an overlap factor which determines the overlap of the hidden units responses with the neighbour hidden units in the input space.

2) Updating the network parameters: When one of the three criteria declared in (3), (4) and (5) is not satisfied, the parameter vector $\theta(k)$ containing the tuning parameters of the MRAN network is updated online using the EKF learning algorithm in this work.

3) Pruning Strategy: A pruning strategy is implemented to avoid an excessive increase of the MRAN size. That is why the resulting algorithm is called as the minimum RAN or MRAN. The algorithm prunes the redundant hidden units that have little contributions to the network outputs for M_p consecutive observations in the sliding data window:

$$o_k(\mathbf{x_n}) = \boldsymbol{\alpha}_{\mathbf{k}} \Phi_k(\mathbf{x_n}) \qquad (8)$$

in which the largest absolute values are found as follows:

$$o_{max}(\mathbf{x_n}) = \left[\max_k |o_{k1}|, \dots, \max_k |o_{kj}|\right]^T \qquad (9)$$

If the *k*th hidden unit's normalized output value of all the outputs, i.e.,

$$r_{kj} = \left|{o_{kj}}/{o_{maxj}}\right| \qquad (j = 1, \dots, J) \qquad (10)$$

is less than a threshold for M_p consecutive observations, the *k*th hidden unit is regarded as a redundant unit and pruned.

B. GAP-RBF Neural Network

GAP-RBF NN uses the same GRBF networks model employed in the MRAN. Similarly, the learning process of this network begins with no hidden neurons. When the new sampled input-output pairs $(\mathbf{x_n}, \mathbf{y_n})$, $(\mathbf{x_n} \in R^l, \mathbf{y_n} \in R^m)$ are available, the following steps are performed:

1) Allocate a new hidden neuron if all the following conditions are fulfilled:

$$\begin{cases} \left\|\mathbf{x_n} - \boldsymbol{\mu}_{\mathbf{nr}}\right\| > \epsilon_n, \\ \|\mathbf{e_n}\| = ||\mathbf{y_n} - f(\mathbf{x_n})|| > e_{min}, \\ E_{sig}(K+1) > e_{min} \end{cases} \qquad (11)$$

$$(y_{ni} \in \mathbf{y_n},\; x_{nj} \in \mathbf{x_n} \;;\; i = 1, \dots, m \;,\; j = 1, \dots, n)$$

Where, $\boldsymbol{\mu}_{\mathbf{nr}}$ is the center which is nearest to $\mathbf{x_n}$ and nr denotes the index number of the nearest neuron. The definitions for

distance ϵ_n and threshold e_{min} are the same as definitions given in the MRAN. $E_{sig}(K)$ indicates the significance of the kth hidden neuron defined on the basis of its average contribution output on the ith network output over all input samples received so far [5] which can be estimated as bellow:

$$E_{sig}(K) = \left| \frac{(1.8\sigma_{nr})^l \boldsymbol{\alpha}_{\mathbf{nr}}}{S(X)} \right| \tag{12}$$

Where, $S(X)$ denotes the size of the input space and l is the number of inputs.

It has been proven [6] that the second condition in (11) is always contained in the third Condition and it's not required. So, when the first and third criteria in (11) are satisfied, a new neuron K+1 is added to the network with the following tuning parameters:

$$\begin{cases} \boldsymbol{\alpha}_{\mathbf{k+1}} = \mathbf{e}_{\mathbf{n}}, \\ \boldsymbol{\mu}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{n}}, \\ \sigma_{k+1} = k\left\|\mathbf{x}_{\mathbf{n}} - \boldsymbol{\mu}_{\mathbf{nr}}\right\| \end{cases} \tag{13}$$

2) Adjust the network parameters $\boldsymbol{\alpha}_{\mathbf{nr}}$, $\mu_{\mathbf{nr}}$, σ_{nr} for the nearest neuron using the EKF learning algorithm if step 1 wasn't performed.

3) Prune the nearest hidden neuron if $E_{sig}(K) < e_{min}$ and do the necessary changes in the EKF learning algorithm.

C. The Modified GAP-RBF (MGAP-RBF) Neural Network

In order to have smoothly output response and avoid oscillation, the mechanism of adding and pruning should be allowed to change smoothly. The rate of adding or pruning neurons can be controlled with threshold values of e_n and ϵ_n, respectively. Selection of these values depends on the complexity of the system, input data for identification and the required accuracy for the model. The most important and effective factor is the persistent exciting (PE) of the input data. If the input data have enough degree of PE, smooth and accurate output can be obtained with suitable adjusting of the threshold values. But, if the inputs do not be PE, which may happen for instance in the case of closed-loop identification, then the threshold values cannot help and hence modification on adding and pruning rules will be necessary.

In on-line identification, it should be better to change the adding strategy in such a way that, the rate of adding neurons is increased in the beginning of the identification process, and as the identification process continues on, the rate of adding neurons is decreased. In the original algorithm [5] and its applications [5, 6, 7], neurons are added hardly in the start of the modeling process. This causes large errors in the start of the algorithm and hence the learning EKF techniques can not estimate the relevant parameters. As the rate of neuron creation can be controlled with ϵ_n, an exponential function is proposed to be used in order to increase ϵ_n as follows:

$$\epsilon_n = \epsilon_{min} + (\epsilon_{max} - \epsilon_{min})\left(1 - e^{-\frac{n}{\tau}}\right) \tag{14}$$

where τ is the parameter that can be used to control the rate of increasing ϵ_n.

Another problem is oscillating in the number of created neurons that can cause big errors in the identification results. This oscillation will be occurred in on-line identification, when the numbers of neurons that are created are small and the inputs data have small degree of PE, too. This effect can be reduced with changing the pruning criteria by adding a new pruning factor (β) as follows:

$$E_{sig}(K) \leq \beta e_{min} \tag{15}$$

Where, 0<β≤1.

III. EKF LEARNING ALGORITHM

The EKF is considered as one of the most effective methods for both nonlinear *state* and *parameter estimation*. The EKF gives an approximation of the optimal estimate. The non-linearities of the system's dynamics are approximated by a linearized model around the last state estimate. For this approximation to be valid, this linearization should be a good approximation of the non-linear model in the entire uncertainty domain associated with the state estimate. Consider the non-linear dynamics:

$$\begin{aligned} x_{k+1} &= f_k(x_k, u) + w_k \\ y_k &= h_k(x_k) + v_k \end{aligned} \tag{16}$$

One iteration of the EKF algorithm consists of the following consecutive steps:

1) Consider the last filtered state estimate $\hat{x}(k|k)$,

2) Linearize the system dynamics, $x_{k+1} = f_k(x_k) + w_k$ around $\hat{x}(k|k)$,

3) Apply the prediction step of the Kalman filter to the linearized system dynamics just obtained, yielding: $\hat{x}(k+1|k)$ and $P(k+1|k)$,

4) Linearize the observation dynamics, $y_k = h_k(x_k) + v_k$ around $\hat{x}(k+1|k)$,

5) Apply the filtering or update cycle of the Kalman filter to the linearized observation dynamics, yielding: $\hat{x}(k+1|k+1)$ and $P(k+1|k+1)$.

Let $F(k)$ and $H(k)$ be the Jacobian matrices of $f(.)$ and $h(.)$, denoted by

$$\begin{aligned} F(k) &= \nabla f_k|_{\hat{x}(k|k)} \\ H(k+1) &= \nabla h|_{\hat{x}(k+1|k)} \end{aligned} \tag{17}$$

The EKF algorithm can alternatively be stated in terms of the following two distinct cycles:

Predict Cycle

$$\begin{aligned} \hat{x}(k+1|k) &= f_k\big(\hat{x}(k|k)\big) \\ P(k+1|k) &= F(k)P(k|k)F^T(k) + Q(k) \end{aligned} \tag{18}$$

Filtered Cycle

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K(k+1)\big[y_{k+1} - h_{k+1}\big(\hat{x}(k+1|k)\big)\big] \tag{19}$$

$$K(k+1) = P(k+1|k)H^T(k+1)[H(k+1)P(k+1|k)H(k+1)^T + R(k+1)]^{-1} \tag{20}$$

$$P(k+1|k+1) = [I - K(k+1)H(k+1)]P(k+1|k) \tag{21}$$

As the EKF is not an optimal filter, the matrices $P(k|k)$ and $P(k+1|k)$ do not represent the true covariance of the state estimates. Moreover, as the matrices $F(k)$ and $H(k)$ depend on previous state estimates and therefore on measurements, the filter gain $K(k)$ and the matrices $P(k|k)$ and $P(k+1|k)$ can not be computed off-line as occurs in the Kalman Filter. Contrary to the Kalman filter, the EKF may diverge, if the consecutive linearizations are not a good approximation of the non-linear model in the entire associated uncertainty domain.

IV. Simulation Case Study

A series of simulation tests will be conducted in this section to explain the capability of the proposed on-line identification approaches for a highly nonlinear and time-varying CSTR benchmark problem.

A. CSTR Process Description

The process is a non-isothermal CSTR with an irreversible reaction (A $\longrightarrow$ B) which consists of two nonlinear ordinary differential equations, given by [8]:

$$\frac{dC_A}{dt} = \frac{q}{V}\left(C_{Af} - C_A\right) - K_0 C_A \exp\left(-\frac{E}{RT}\right)\Phi_c(t) \tag{22}$$

$$\frac{dT}{dt} = \frac{q}{V}\left(T_f - T\right) + \frac{(-\Delta H)K_0 C_A}{\rho C_p}\exp\left(-\frac{E}{RT}\right)\Phi_c(t) + \frac{\rho_c C_{pc}}{\rho C_p V} q_c \left[1 - \exp\left(-\frac{hA}{q_c \rho C_{pc}}\Phi_h(t)\right)\right] \times (T_{cf} - T) \tag{23}$$

Where,

$\Phi_h(t)$	Fouling coefficient
$\Phi_c(t)$	Deactivation coefficient
C_A	Effluent concentration, the controlled variable
T	Output temperature, the controlled variable
q_c	Coolant flow rate, the manipulated variable
q	Feed flow rate, disturbance (for SISO)
C_{Af}	Feed concentration
T_f	Feed temperature
T_{cf}	Coolant inlet temperature

The remaining model parameters together with the operating conditions are presented in Table I.

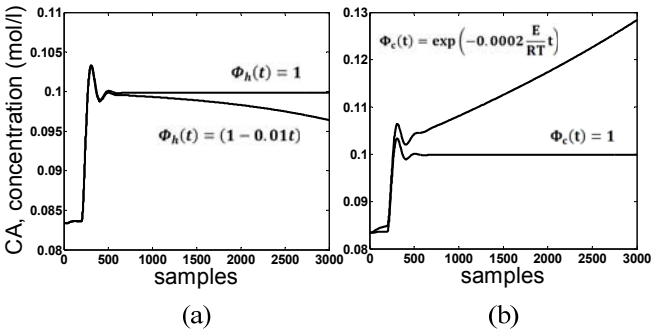
TABLE I. Nominal CSTR Operating Condition

$q = 100\ l.min^{-1}$	$E/R = 9.95 \times 10^3\ K$
$C_{Af} = 1\ moll^{-1}$	$-\Delta H = 2 \times 10^5\ cal.mol^{-1}$
$T_f = 350\ K$	$\rho, \rho_c = 1000\ gl^{-1}$
$T_{cf} = 350\ K$	$C_p, C_{pc} = 1\ calg^{-1}K^{-1}$
$V = 100\ l$	$q_c = 103.41\ l.min^{-1}$
$h_A = 7 \times 10^5\ cal.min^{-1}K^{-1}$	$T = 440.2\ K$
$k_0 = 7.2 \times 10\ min^{-1}$	$C_A = 8.36 \times 10^{-2}\ moll^{-1}$

The CSTR nonlinear dynamic model exhibits time-varying characteristics due to the fouling and catalyst deactivation phenomena, formulated as follows:

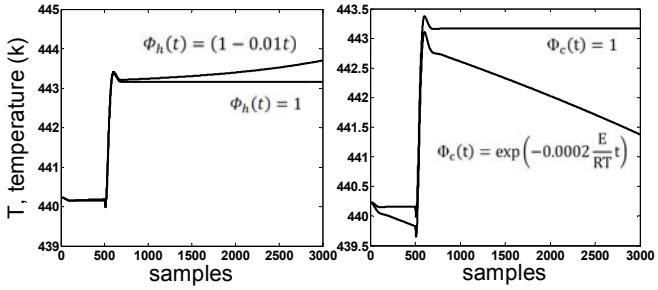
$$\Phi_h(t) = (1 - 0.01t) \tag{24}$$

$$\Phi_c(t) = \exp\left(-0.0002 * \frac{E.R}{T}t\right) \tag{25}$$



(a) (b)

Figure 1. (a) fouling effect and (b) deactivation effect for 5% step change in coolant flow rate



(a) (b)

Figure 2. (a) fouling effect and (b) deactivation effect for 5% step change in feed flow rate

B. Identification of the CSTR with Time-varying Parameters

In this section, the MIMO CSTR, described in previous section, is identified in an on-line manner.

In order to make the picture of the performed simulation experiments more real, two distinct white noises with variance of 0.01 have been added to the original dynamics in (22) and (23) as the process noises. Since the effluent concentration (CA) and its temperature (T) have been assumed as the process outputs, white noises with variance of 0.001 and 1 have also been added to output concentration and output temperature, respectively, as the measurement noises.

The identification experiments have been performed using white noises with variance of 1 superimposed on the multi-levels coolant flow rate q_c and feed flow rate q as the manipulated variables in order to drive the CSTR process at different operating conditions starting with the initial nominal condition specified by Table I. So, the process is excited by two different inputs q_c, q as shown in Figs.3a and 3b. The obtained process output responses have been illustrated in Figs.3c and 3d. Fig.3 (e)-(j) shows the resulting performances of the different algorithms for identification of MIMO CSTR.

The free parameters have been fixed in all experiments as follows in order to be able to compare and analyze the experiment results at the same conditions:

$\epsilon_{\max} = 0.1, \epsilon_{\min} = 0.02, \kappa = 3, \gamma = 0.98, e_{\min} = 0.01,$ $\tau = 1000$ and $\beta = 0.6$ and the free parameters for MRAN are chosen as $e'_{min} = 0.01, M = 50$.

With the following NN input regressor vector:

$x_n(t) = [u_1(t)\ u_1(t-1)\ u_2(t)\ u_2(t-1)\ y_1(t-1)\ y_2(t-1)]$

Where, $u_1 = q_c, u_2 = q, y_1 = C_A, y_2 = T$.

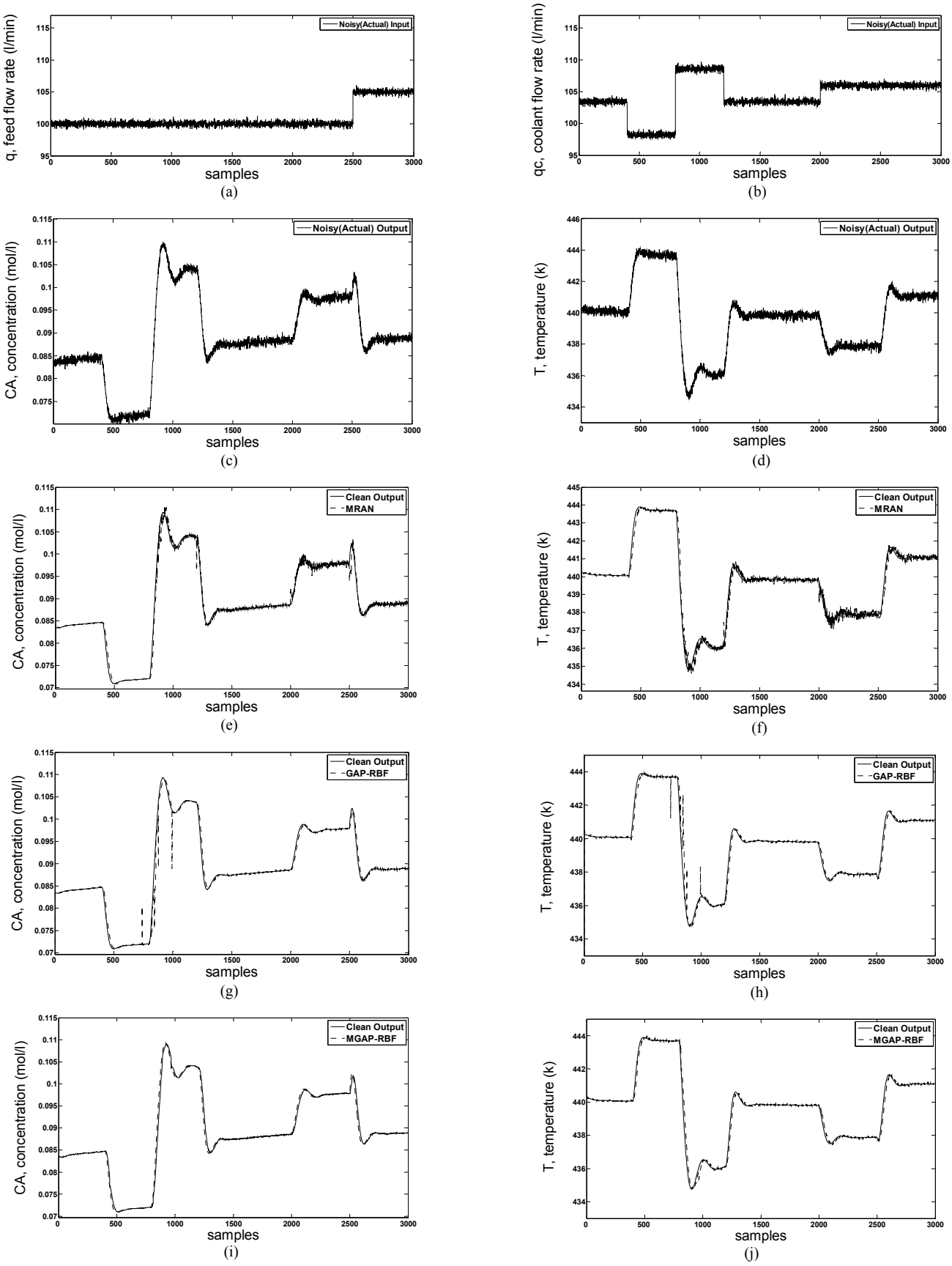


Figure 3 (a)-(d) Real process inputs and outputs (e)-(j) each identification algorithm performance

TABLE II. MEAN SQUARE ERROR (MSE) OF MODELING FOR EACH ALGORITHM

NN	MSE(CA)	MSE(T)
MRAN	8.952e-006	0.4947
GAP	1.182e-006	0.3361
MGAP	9.276e-007	0.2749

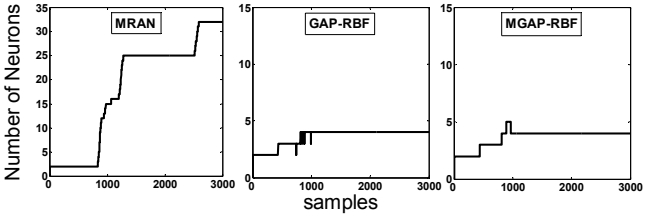


Figure 4 Neuron number comparison of different algorithms

According to the obtained results, it can be derived that the MGAP-RBF neural network has good performance in both accuracy and simplicity of the identified network. As shown in Fig.4, by using the pruning factor in MGAP-RBF neural network, the oscillations in the created neurons decreases in comparison with the original GAP-RBF neural network. This leads to elimination of spikes in output responses, caused by oscillation in neuron evolution, which can be checked by comparing Fig.3i with 3g or Fig.3j with 3h. On the other hand, using the modified growing criterion prevents more ascending of neuron numbers which is inevitable because of using the pruning factor.

V. CONCLUSIONS

Two different types of adaptive neural networks, i.e. GAP-RBF and MRAN, with EKF learning algorithm have been utilized for on-line identification of multi-input multi-output non-linear systems. The original versions of the two utilized neural networks have been employed in sequential identification modes using on-line data. In this paper, the original GAP-RBF network structure has been modified in order to enhance its applications for on-line MIMO identification purposes. The basic contributions in this respect can be summarized as follows:

1) Modification in neuron creation criterion.

2) Modification in neuron pruning criterion.

The proposed on-line identification techniques have been implemented on a CSTR process as a multi-input multi -output benchmark problem with high non-linearity and time varying features due to a time-varying profile in the fouling and deactivation coefficients.

The analysis of the resulting outcomes indicates the following observations:

1) The main flaw of the MRAN-based identification algorithms is the dependency of its learning accuracy and generalization performance on the growing-pruning window size which is normally chosen on an exhaustive simulation trial and error basis.

2) The MRAN-based identification algorithms require higher computational complexity due to the fact that all the neuron parameters should be updated during each learning phase when no neuron is added or omitted.

3) The main benefit of the GAP-RBF-based identification algorithms is that any desired accuracy requirement can directly be introduced in the algorithm by defining a significance quality measure for the quality of each neuron.

4) The GAP-RBF-based identification algorithms have a lower degree of computational complexity in comparison with the MRAN-based ones due to the fact that only the nearest neuron parameters should be updated during each learning phase. However, their tuning parameters are higher and more sensitive to the threshold initialization values.

5) The proposed MGAP-RBF neural network gives the best performance. This achievement can mainly be expressed in terms of the resulting accuracy and the identified network-model structure simplicity.

REFERENCES

[1] Platt J., "*A resource allocating network for function interpolation*". Neural Computat **3**(2);213–225.(1991).

[2] Kadirkamanathan V, Niranjan M "*A function estimation approach to sequential learning with neural Network".* Neural Computat **5**(6):954–975.(1993).

[3] Yingwei L., Sundararajan N., Saratchandran P., "*A sequential learning scheme for function approximation by using minimal radial basis function neural networks*". Neural Computat **9**(2):461–478.(1997).

[4] Yingwei L., Sundararajan N., Saratchandran P., "*Performance evaluation of a sequential minimal radial basis function (RBF) neural network learning algorithm*". IEEE Trans Neural Netw **9**(2):308–318. (1998).

[5] Huang GB., Saratchandran P., Sundararajan N., "*An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) Networks"*. IEEE Trans Syst Man Cybern–B **34**(6):2284–2292. (2004).

[6] Huang GB., Saratchandran P., Sundararajan N.,*"A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation"*. IEEE Trans Neural Netw 16(1):57-67.(2005).

[7] Wang Y., Huang G.-B., Saratchandran P., and Sundararajan N., "*Time Series Study of GGAP-RBF Network: Predictions of Nasdaq Stock and Nitrate Contamination of Drinking Water".* Proceedings of International Joint Conference on Neural Networks, Montreal, Canada, July 31 - August 4, (2005).

[8] Nikravesh M., *"Dynamic neural network control"*, Ph.D. Dissertation, University of South Carolina, Columbia, SC, (1994).