Development of Adaptive p-Step RBF Network Model with Recursive Orthogonal Least Squares Training

Lei Gu[#], D. K. Siong Tok* and Ding-Li Yu*[$]

School of Electronic Information, Changchun Architecture & Civil Engineering College, Changchun, China.

* Process Control Group, Liverpool John Moores University, Liverpool, U.K.

[$] Corresponding author: D.Yu@ljmu.ac.uk

Abstract. An adaptive p-step prediction model for nonlinear dynamic processes is developed in this paper, and implemented with a radial basis function (RBF) network. The model can predict output for multi-step ahead with no need for the unknown future process output. Therefore, the long-range prediction accuracy is significantly enhanced, and consequently is especially useful as the internal model in a model predictive control framework. An improved network structure adaptation is also developed with the recursive orthogonal Least Squares (ROLS) algorithm. The developed model is on-line updated to adapt both its structure and parameters, so that a compact model structure and consequently a less computing cost are achieved with the developed adaptation algorithm applied. Two nonlinear dynamic systems are employed to evaluate the long-range prediction performance and minimum model structure, and compared with an existing PSC model and a non-adaptive RBF model. The simulation results confirm the effectiveness of the developed model and superior over the existing models.

Keywords: RBF network, p-step model prediction, ROLS training algorithm, RBF structure adaptation, model predictive control.

1 Introduction

Radial basis function (RBF) network models have been studied intensively in modelling nonlinear dynamic systems due to its learning abilities and simple architecture [1]. The attractive feature that, the RBF network can approximate a smooth nonlinear mapping to any specified accuracy provided that the network includes enough number of hidden layer nodes, was first studied by Broom and Lowe [2]. A RBF network that based on a nonlinear autoregressive with exogenous input model (NARX) was renowned for its promising abilities in modelling nonlinear dynamic systems [3]. However, one of the major drawbacks of NARX RBF network models is the lack of efficiency in long range output prediction due to its accumulated errors at each prediction step [4]. Bhartiya and Whiteley [5] developed a factorable *p*-step control (PSC) model that could make long prediction. The simulation results showed that the output prediction of their model performed better than the cascaded 1-step-ahead prediction. However, a major drawback of this model is that a huge network structure is required, which greatly increases the model complexity. A compact RBF network structure is imperative to avoid numerical ill-conditioning and for good generalization [6].

Structure adaptation of a RBF model plays a major role in achieving a compact network while maintaining the model performance. Moreover, when the model is used in a model-based control scheme, model adaptation can model the time varying dynamics or post-fault dynamics change of the system, to achieve robust or fault tolerant control. There are several RBF network adaptation algorithms reported in the literature, such as that in [7-10]. A common feature of these adaptive networks is that the number and location of their hidden neurons are flexible and are adapted according to the dynamics of system to be modelled.

To reduce the size of a network while maintaining its modelling accuracy, orthogonal decomposition has been employed in building adaptive structure of RBF networks. The batch orthogonal Least Squares algorithm was firstly employed to train neural networks by Chen and Billings [11] and the work was later extended to train RBF networks in [6, 12]. Yu *et al.* [13] applied recursive orthogonal Least Squares (ROLS) algorithm to train the RBF network parameters in on-line mode and achieved a convincing performance. Gomm and Yu [14] introduced a forward and a backward centre selection algorithm using ROLS algorithm to achieve compact network structures while maintaining the

prediction performance. However, the centres were only selected from a pre-specified candidate centre set in [14], which limited the model's prediction capability. To broaden the options of centres, Yu and Yu [15] proposed an adaptive structure algorithm for RBF networks using ROLS algorithm in which the new data were added as new centres based on the desired modelling performance. The results in [15] demonstrated that a compact network structure was achieved while maintaining the desired performance. The drawback is the network performance degraded for a number of sample periods after the migration of system's operating point. To address this problem, Tok *et al.* [16] developed a new learning strategy to improve the tracking ability and the results showed that a better recovery speed was achieved. However, the tracking ability is obtained at the high expense of computational cost.

The objective of this work is twofold. Firstly, a modified PSC model implemented with RBF network (therefore is called PS-RBF model) is adapted with both structure and parameters using the ROLS algorithm. The advantage of the proposed PS-RBF model is that it retains the important feature of reducing the model's dependency on the future unknown system outputs to make p-step ahead predictions, while reducing the network model complexity and maintaining the prediction accuracy. The superior feature of the developed PS-RBF model over the PSC model [5] is that when the developed model is used in the model predictive control, the control performance will be more accurate. Moreover, due to the adaptation of the developed model, the performance degradation caused by dynamics change due to the faults or time varying components of the process will be greatly improved. Thus, the effectiveness of the PS-RBF network model is investigated, and its performance is compared with two network models of the same type: the PSC model and the multistep-ahead RBF network model (MSA-RBF). The simulation results show that not only does the developed model outperform the other two network models, it also produces a more compact model structure compared with the PSC model in [5].

The second objective is to ease the computational load in the adaptation algorithm in [16] by re-structuring the adaptation procedure. To achieve this, the amount of calculation in computing the decomposition of matrix R is reduced. The efficiency of improved adaptation algorithm is verified by comparing it with existing algorithm in [16].

In particular, an adaptive PS-RBF network is proposed. One of the advantages is that the adaptive PS-RBF network provides a smaller network structure than non-adaptive network while maintaining the network performance. The performance of adaptive PS-RBF network is verified by modelling a nonlinear dynamical system.

The outline of this paper is as follows. Section 2 describes the development of PS-RBF network. Section 3 introduces the ROLS training algorithm of PS-RBF network; it also presents the performance of PS-RBF model and the comparison with other network models. Section 4 describes the optimized PS-RBF network model structure adaptation. Section 5 shows the experimental results of adaptive PS-RBF network. The conclusion is drawn in Section 6.

2 p-step RBF network

It was reported in [5] that a PSC model was developed and implemented with RBF network. The PSC model is capable of predicting the system output over a prediction horizon without requiring future unknown system outputs. However, the downside of the PSC model is that it demands a huge network structure to achieve satisfactory performance. This makes the model impossible to be applied to fast systems due to the big computing load. Furthermore, a limitation is imposed to the selected input order of the network model, $N_u \geq 2$ in [5]. To address these problems, an improved model: a p-step ahead prediction model, which is also implemented with a RBF network, is developed in this paper. The derivation of PS-RBF model is as follows.

In the application of RBF networks [15, 17], a continuous-time nonlinear dynamic system is typically represented by a NARX model in (1).

$$y_k = f\left[y_{k-1}, \ldots, y_{k-Ny}, u_{k-1}, \ldots, u_{k-Nu}\right] + e_k \tag{1}$$

where $u \in \Re^m$ and $y \in \Re^n$ are the input and output vectors of the system, respectively; N_y and N_u are output and input orders, and e is the error. $f[*]$ is a vector valued nonlinear function. A RBF network can be trained as a one-step-ahead (OSA) predictor or a multistep-ahead (MSA) predictor as depicted in Fig. 1.

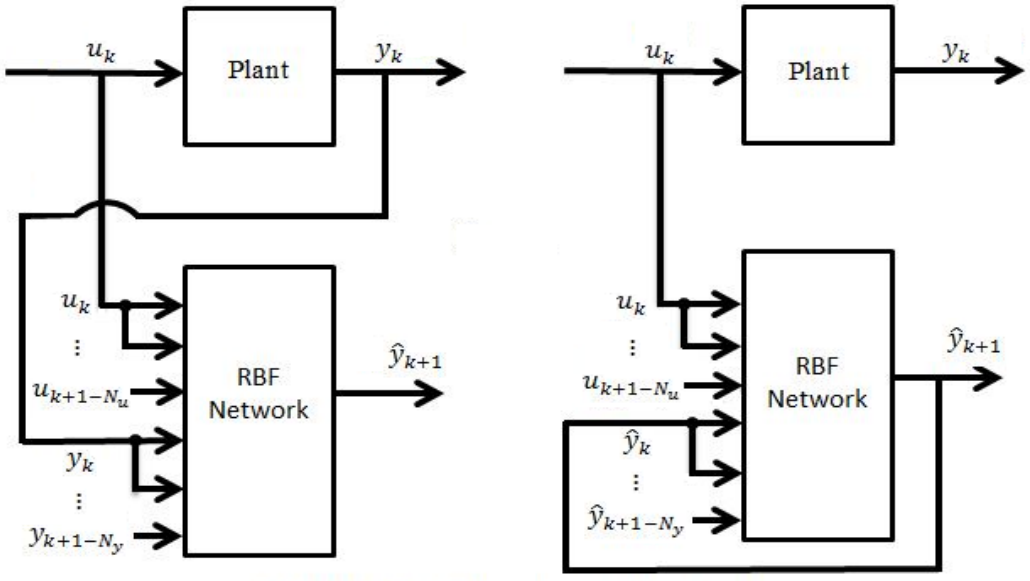

Fig.1 Block diagrams of OSA and MSA predictors

From Fig.1, it is understood that an OSA predictor is trained using the system inputs $u_k, \dots, u_{k+1-N_u}$ and outputs $y_k, \dots, y_{k+1-N_y}$ at sample time k to make a one-step-ahead prediction of $\hat{y}_{k+1}$. The predictions $\hat{y}_{k+1}, \dots, \hat{y}_{k+p}$ by the OSA predictor over a prediction horizon H_p are described as

$$\begin{aligned}
\hat{y}_{k+1} &= f\left[y_k, \dots, y_{k+1-N_y}, u_k, \dots, u_{k+1-N_u}\right] \\
\hat{y}_{k+2} &= f\left[y_{k+1}, \dots, y_{k+2-N_y}, u_{k+1}, \dots, u_{k+2-N_u}\right] \qquad (2) \\
&\vdots \\
\hat{y}_{k+H_p} &= f\left[y_{k+H_p-1}, \dots, y_{k+H_p-N_y}, u_{k+H_p-1}, \dots, u_{k+H_p-N_u}\right]
\end{aligned}$$

In contrast, a different prediction mode of the MSA predictor from the OSA lies in that the predicted outputs $\hat{y}_{k+1}, \dots, \hat{y}_{k+H_p-1}$ are used instead of system output in the prediction of future samples, $\hat{y}_{k+2}, \dots, \hat{y}_{k+H_p}$ across a prediction horizon H_p, as described below

$$\begin{aligned}
\hat{y}_{k+1} &= f\left[y_k, \dots, y_{k+1-N_y}, u_k, \dots, u_{k+1-N_u}\right] \\
\hat{y}_{k+2} &= f\left[\hat{y}_{k+1}, y_k, \dots, y_{k+2-N_y}, u_{k+1}, \dots, u_{k+2-N_u}\right] \qquad (3)
\end{aligned}$$

$$\vdots$$

$$\hat{y}_{k+H_p} = f\left[\hat{y}_{k+H_p-1}, \dots, \hat{y}_{k+H_p-N_y}, u_{k+H_p-1}, \dots, u_{k+H_p-N_u}\right].$$

From (1) and (2), it can be understood that the OSA predictor is able to provides more accurate prediction than the MSA as it is trained using the original system outputs and inputs. Whilst in the MSA the predicted outputs are used as the input for further prediction. As can be seen in (2) that the prediction errors are involved in the predicted output and are accumulated at each sample time when the predicted outputs are iteratively used for future predictions. This greatly reduces the prediction accuracy. Although the MSA is not as accurate as the OSA for one step ahead prediction, its ability in predicting for multi-step-ahead is essential in the applications of model predictive control [17, 18].

In order to improve the performance of MSA predictor, the p-step prediction (PS) model is proposed in this research. The PS model is designed to make predictions over a prediction horizon H_p, which avoids using the future process outputs. In other words, it combines the characteristics of OSA and MSA predictors. The derivation of PS model starts with an NARX model. An example is used to demonstrate the concept of PS model. Consider an example with the input order $N_u = 2$ and the output order $N_y = 2$ to make predictions across a prediction horizon $H_p = 3$. Using this example, the NARX model in (1) can be expressed as

$$\hat{y}_k = F[y_{k-1}, y_{k-2}, u_{k-1}, u_{k-2}] \tag{4}$$

The outputs $y_{k-1}, \dots, y_{k-2}$ can be described in prediction forms of

$$y_{k-1} = F[y_{k-2}, y_{k-3}, u_{k-2}, u_{k-3}]$$

$$y_{k-2} = F[y_{k-3}, y_{k-4}, u_{k-3}, u_{k-4}] \tag{5}$$

Now, substituting (5) into equation (4) yields,

$$\hat{y}_k = F[F[y_{k-2}, y_{k-3}, u_{k-2}, u_{k-3}], F[y_{k-3}, y_{k-4}, u_{k-3}, u_{k-4}], u_{k-1}, u_{k-2}]$$

$$\hat{y}_k = F[F[F[y_{k-3}, y_{k-4}, u_{k-3}, u_{k-4}], y_{k-3}, u_{k-2}, u_{k-3}], \tag{6}$$

$$F[y_{k-3}, y_{k-4}, u_{k-3}, u_{k-4}], u_{k-1}, u_{k-2}]$$

Using a function G to represent the composite function F in (6),

$$\hat{y}_{k/k-3} = G[y_{k-3}, y_{k-4}, u_{k-1}, u_{k-2}, u_{k-3}, u_{k-4}] \tag{7}$$

Using (7) for predictions over $H_p = 3$, they are as follows,

$$\hat{y}_{k+1/k-2} = G[y_{k-2}, y_{k-3}, u_k, u_{k-1}, u_{k-2}, u_{k-3}]$$

$$\hat{y}_{k+2/k-1} = G[y_{k-1}, y_{k-2}, u_{k+1}, u_k, u_{k-1}, u_{k-2}]$$

$$\hat{y}_{k+3/k} = G[y_k, y_{k-1}, u_{k+2}, u_{k+1}, u_k, u_{k-1}] \tag{8}$$

From (7), it can be written in a general form,

$$\hat{y}_{k/k-p} = G\left[y_{k-p}, \dots, y_{k-p+1-N_y}, u_{k-1}, .., u_{k-p+1-N_u}\right] \tag{9}$$

or alternatively,

$$\hat{y}_{k+p/k} = G\left[y_k, \dots, y_{k+1-N_y}, u_{k-1+p}, .., u_{k+1-N_u}\right]. \tag{10}$$

From (10), it can be observed that $\hat{y}_{k+p/k}$ is predicted according to system outputs up to k^{th} sample, which are all available at the current sample period *k*. This avoids using $y_{k+1}, \dots, y_{k+p-1}$, which are not available yet and have to be approximated with the predicted values. Therefore, the dependency of the prediction on the predicted outputs over the prediction horizon H_p is avoided, and it improves the prediction accuracy.

3 p-Step RBF network training

3.1 Training with ROLS algorithm

The recursive orthogonal Least Squares algorithm is used to train the developed PS-RBF network. A RBF network is a three layer network with input layer, hidden layer and output layer. There are hidden neurons in the hidden layer and there is a centre in each hidden neuron. The network input vector x_k using the PS model in (9) is

$$x_k = \left[y_{k-p} \ \dots \ y_{k-p+1-N_y} \ \ u_{k-1} \ \dots \ u_{k-p+1-N_u}\right] \tag{11}$$

where u and y are system input and output, respectively. A Gaussian function is used as the activation function then the hidden layer output $\phi_i(k)$ is given as

$$\phi_i(k) = \exp\left(-\frac{\|x_k - c_i\|^2}{\sigma_i^2}\right), i = 1, \dots, n_h \tag{12}$$

where n_h is the number of hidden neurons and $c_i \in \Re^{n \times n_h}$ is the ith centre. σ_i represents the width of the Gaussian function at the ith centre. The network output is

$$\hat{y}(k) = W^T(k)\phi(k) \tag{13}$$

where $W_k \in \Re^{n_h \times p}$ is the weighting matrix connecting hidden layer nodes and network outputs.

Considering (11) as a set of collected input-output training data for N samples,

$$Y = \hat{Y} + E = \Phi W + E \tag{14}$$

where $Y \in \Re^{N \times p}$ is the desired output matrix and $\hat{Y} \in \Re^{N \times p}$ is the network output matrix. $\Phi \in \Re^{N \times n_h}$ is the hidden layer output matrix and $E \in \Re^{N \times p}$ is the prediction error matrix. The matrix formations are as follows

$$\begin{aligned} Y^T &= [y(1), \dots, y(N)], \qquad \hat{Y}^T = [\hat{y}(1), \dots, \hat{y}(N)], \\ \Phi^T &= [\phi(1), \dots, \phi(N)], \qquad E^T = [e(1), \dots, e(N)], \end{aligned} \tag{15}$$

The procedure to train the weights of the network using ROLS algorithm is reviewed as follows [13]. The following cost function is formed based on (14) and is used to solve for weight W,

$$J(W) = \|E\|_F = \|Y - \Phi W\|_F \tag{16}$$

where $\|*\|_F$ is the F-norm of a matrix defined as $\|A\|_F^2 = trace(A^T A)$. With ROLS training algorithm, the cost function at each sample time k becomes

$$J(k) = \|E(k)\|_F = \left\| \begin{bmatrix} Y(k-1) \\ y^T(k) \end{bmatrix} - \begin{bmatrix} \Phi(k-1) \\ \phi^T(k) \end{bmatrix} W(k) \right\|_F. \tag{17}$$

Applying the orthogonal transformation, (17) becomes

$$\Phi(k-1) = Q(k-1) \begin{bmatrix} R(k-1) \\ 0 \end{bmatrix}, \qquad Q^T(k-1)Y(k-1) = \begin{bmatrix} \hat{Y}(k-1) \\ \tilde{Y}(k-1) \end{bmatrix}$$

$$J(k) = \left\| \begin{bmatrix} \hat{Y}(k-1) \\ y^T(k) \\ \tilde{Y}(k-1) \end{bmatrix} - \begin{bmatrix} R(k-1) \\ \phi^T(k) \\ 0 \end{bmatrix} W(k) \right\|_F \tag{18}$$

leaving $\left\|\tilde{Y}(k-1)\right\|_F$ as the modelling residual.

With the arrival of new sample data, the update to network structure is as follows,

$$\begin{bmatrix} R(k-1) \\ \dots \\ \phi^T(k) \end{bmatrix} = Q(k) \begin{bmatrix} R(k) \\ \dots \\ 0 \end{bmatrix} \tag{19}$$

$$\begin{bmatrix} \hat{Y}(k) \\ \dots \\ \tilde{y}^T(k) \end{bmatrix} = Q^T(k) \begin{bmatrix} \hat{Y}(k-1) \\ \dots \\ y^T(k) \end{bmatrix} \tag{20}$$

The eventual cost function is

$$J(k) = \left\| \begin{bmatrix} \hat{Y}(k) - R(k)W(k) \\ \tilde{y}^T(k) \\ \tilde{Y}(k-1) \end{bmatrix} \right\|_F \tag{21}$$

and then, the optimal weight $W(k)$ can be computed using

$$R(k)W(k) = \hat{Y}(k). \tag{22}$$

The modelling residual is

$$\left\|\tilde{Y}(k)\right\|_F^2 = \left\| \begin{bmatrix} \tilde{y}(k) \\ \tilde{Y}(k-1) \end{bmatrix} \right\|_F^2 = \left\|\tilde{y}^T(k)\right\|_F^2 + \left\|\tilde{Y}(k-1)\right\|_F^2. \tag{23}$$

The procedure of ROLS training algorithm is summarized as the following: Set the initial value, $R(0) = \alpha I$ where α is a small positive value, and $\hat{Y}(0)$ with $\left\|\tilde{Y}(0)\right\|_F^2 = 0$. Then, at iteration k, new data $y^T(k)$ arrives, calculate $\phi(k)$. It is followed by computing $R(k)$ and $\hat{Y}(k)$ using (19) and (20), respectively. Finally, solve the weight $W(k)$ using (22).

3.2 p-Step RBF network modelling

Two simulation examples including a set of data from a dryer and another set of data from a chaotic Mackey-Glass time series are used to evaluate the performance of the proposed PS RBF model. In addition, the proposed network model is compared with two

existing RBF models, one is the MSA predictor [17] and the other is the PSC model [5]. The MSA model is selected due to its popularity for being employed as long range predictors. The PSC network model was developed based on PS model, which implies that the performance comparison between them is imperative. In order to further evaluate the ability of proposed network in long range prediction, the proposed PS-RBF model and the PSC model are simulated using two different prediction horizons $H_p = 5$ and $H_p = 15$ respectively.

For a fair comparison, all the three network models are trained with the ROLS algorithm and the parameters in all networks are carefully tuned. The K-means clustering algorithm and P-nearest neighbour algorithm are used to compute the position of centres and radius of the Gaussian functions, respectively. The mean absolute error (MAE) is used to measure the prediction errors,

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|y_i - \hat{y}_i| \tag{24}$$

where N is the number of data samples. The MAE is used here rather than the mean square error (MSE) is because in the former the error is the same level with the output while in the latter the error is in the same level with the squared error. The latter is easy to cause confusion. The whole set of obtained input-output data samples are linearly scaled to [0 1] to minimize the error caused by the big difference between ranges of different variables.

$$u_s = \frac{u - \min(u)}{\max(u) - \min(u)} \qquad y_s = \frac{y - \min(y)}{\max(y) - \min(y)} \tag{25}$$

where u and y are input and output in raw data and u_s and y_s are the scaled data; $\min(u)$ and $\min(y)$ are the minimum values of input and output, respectively. The scaled output predictions are then scaled back after the model is used.

3.2.1 Modelling the dryer

The first simulation example is a set of dryer data that is available in MATLAB. According to MATLAB, the dryer data is collected from a single input single output (SISO) real laboratory-scale 'hairdryer'. The obtained 1000 input-output dryer data

samples are halved into two sets – first 500 data samples are used as training data and the remainders are used as validation data.

The simulation is arranged in the following way. The proposed PS-RBF model is compared with the two other models, the MA model and the PSC model. The MA model uses the process input and output as that input the ARX model, so it is a typical model used for dynamic system modelling. The MA model is selected here for comparison of modelling ability. The PSC model is a model developed for long range prediction and used here for comparison of long range prediction performance. In comparison with the MA model only 5 step ahead prediction is chosen while 20 and 60 centres are chosen. But in comparison with the PSC model not only different prediction horizons are chosen and different centres are also chosen. The criteria and modelling performances of all the three network models are listed in Table 1.

Table 1 Performance comparison of different RBF models for dyer data

RBF Network	MA		PSC		Proposed PS-RBF	
Prediction Horizon	$H_p = 5$		$H_p = 5$	$H_p = 15$	$H_p = 5$	$H_p = 15$
Number of centres	20	60	60	205	20	20
Training data MAE	0.11713	0.1162	0.11641	0.12764	0.10127	0.07708
Validation data MAE	0.11867	0.1178	0.10419	0.14878	0.11593	0.10421

It can be concluded from the results presented in Table 1 that the proposed PS-RBF model uses much fewer centres and has smaller prediction errors compared with the PSC model. On the other hand, the proposed PS-RBF model also outperforms the MSA model in terms of smaller prediction error when they used the same size network. By using two different prediction horizons, $H_p = 5$ and $H_p = 15$, it indicates another benefit of the proposed PS-RBF model, which is that the prediction error is reduced with the longer prediction horizon, while the PSC model is with a bigger prediction error even with a

much larger size of network. From this point of view, the prediction performance of the proposed network model is significantly better than the other two models especially when the models are used for the model predictive control. And, it has good long range prediction ability as shown in Fig. 2 and Fig. 3.

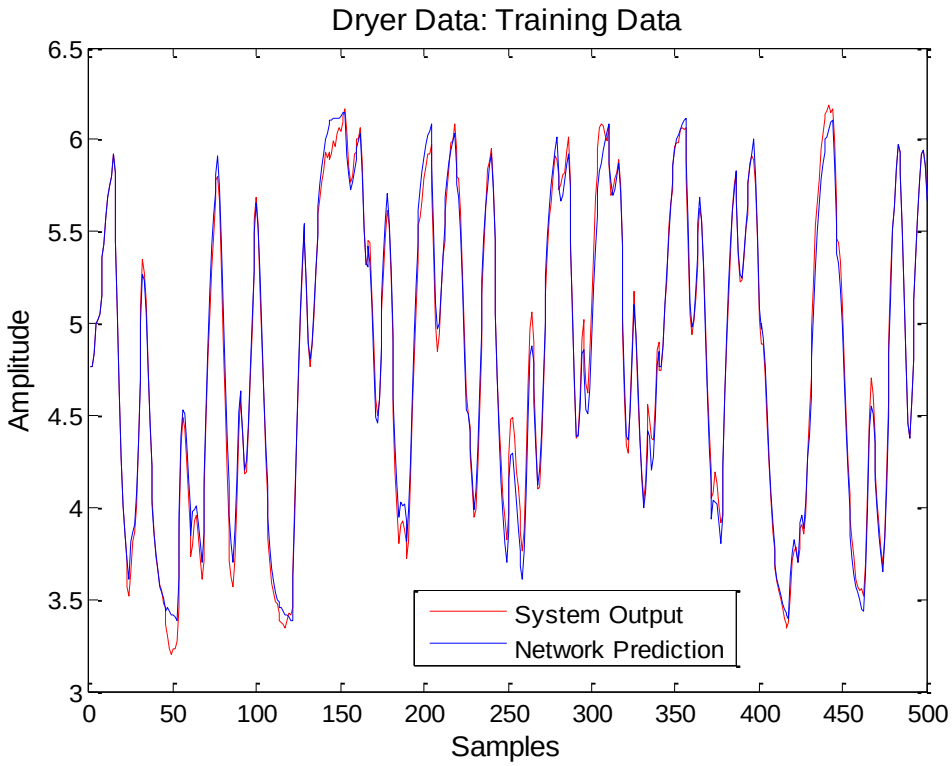

Fig.2 Performance of PS-RBF network ($H_p = 15$) in training data of dryer

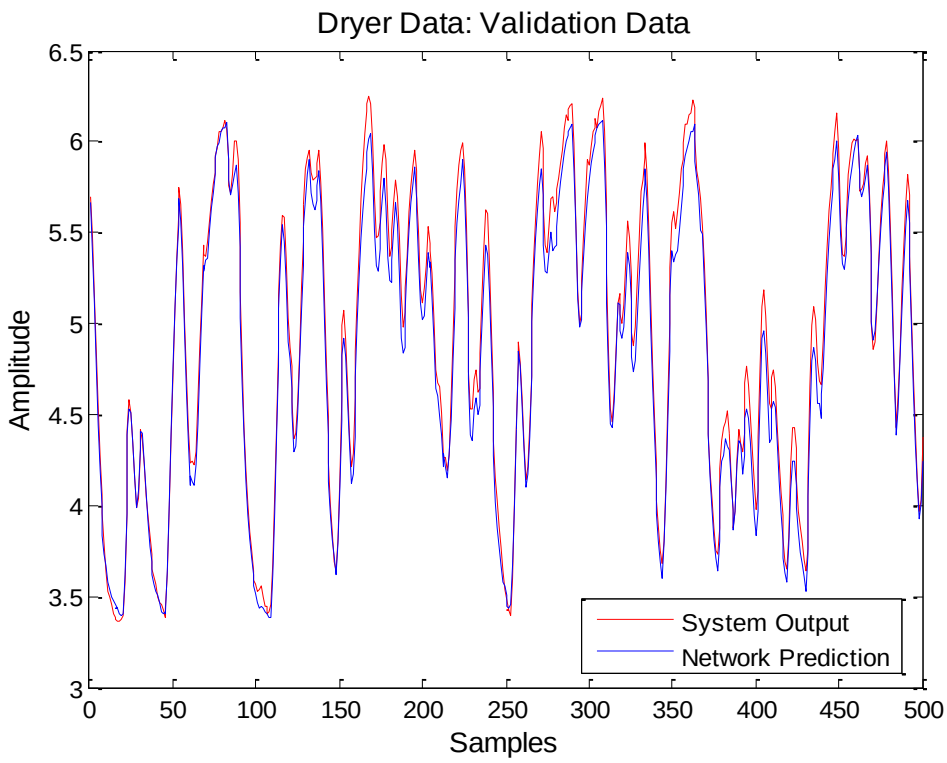

Fig.3 Performance of PS-RBF network ($H_p = 15$) in validation data of dryer

3.2.2 Modelling Chaotic Mackey-Glass time series

The chaotic Mackey-Glass time series is a nonlinear model that is frequently employed as a system identification example. The discrete time series is described as [19]

$$x(t+1) = (1-a)x(t) + \frac{bx(t-\tau)}{1+x^{10}(t-\tau)} \tag{26}$$

where the parameters $a = 0.1, b = 0.2$ and the initial condition $x(0) = 1.2, \tau = 17$.

The criteria and prediction performance of all the three network models are recorded in Table 2.

Table 2 Prediction results of the three models for Mackey-Glass time series

RBF Network	MA	PSC		Proposed PS-RBF	
Prediction Horizon	-	$H_p = 5$	$H_p = 15$	$H_p = 5$	$H_p = 15$

Number of centres	25	24	62	20	45
Training data MAE	0.063434	0.010178	0.008264	0.0093808	0.0080148
Validation data MAE	0.065759	0.010379	0.0086541	0.0089941	0.0073537

It is evident from the results in Table 2 that both the proposed PS-RBF models with $H_p = 5$ and $H_p = 15$ significantly outperform the MSA model in both training and validation data. The second point is that the performance of the proposed PS-RBF model uses fewer centres than the PSC model, while the prediction error is still smaller than the latter model. From Table 2, it has again shown the feature of the proposed network model that the prediction error is reduced for longer prediction horizon, with the cost of larger size of the network. In this example, it indicates that the proposed network is again the best choice of all the three models. The prediction performances of the proposed PS-RBF model for $H_p = 15$ are displayed in in Fig.4 and Fig.5 for the training data and validation data, respectively.

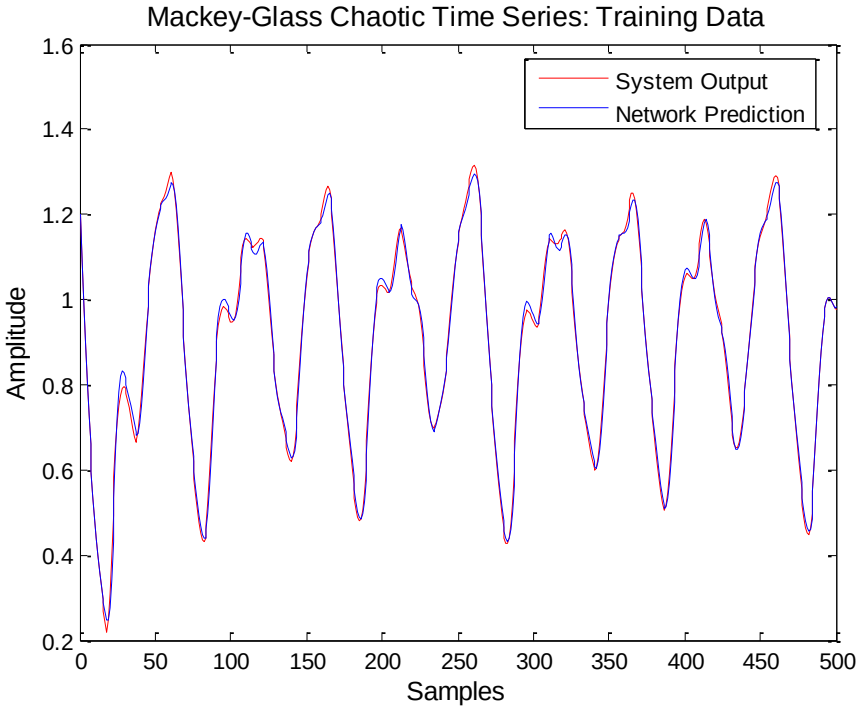

Fig.4 Performance of PS-RBF model ($H_p = 15$) in training data of Mackey-Glass time-series

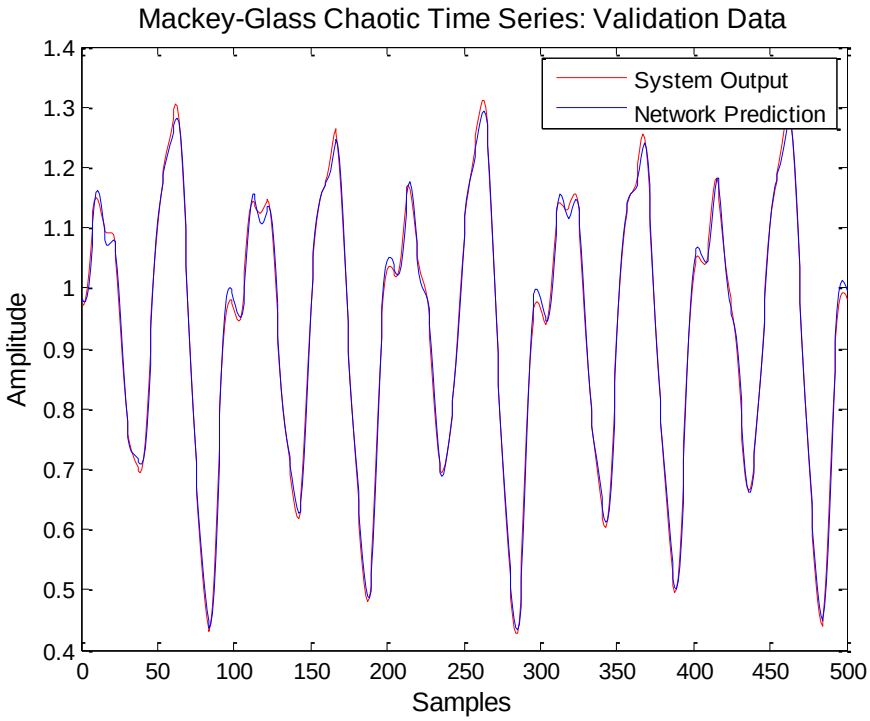

Fig.5 Performance of PS-RBF model ($H_p = 15$) in validation data of Mackey-Glass time series

4. p-step RBF model structure adaptation

The purpose of introducing a structure adaptation into PS-RBF model is to enhance its model compactness. The network structure adaptation algorithm developed in [16] is improved and used in this work. The procedure in the adaptation algorithm in [16] is re-structured to minimize the resource usage.

The ROLS training algorithm is particularly useful in developing an adaptive network structure because it allows the assessment of contributions of each centre in the network structure. However, one of the drawbacks is the R matrix is needed to re-triangularize after it is augmented. Thus, the objective of improvement is to reduce the amount in computing the decomposition of augmented R matrix especially in evaluating the contributions of centres and modelling residual. The concept of structure adaptation in [16] is to manipulate the contributions of centre and modelling residual to perform a series of actions such as adding, pruning and grouping centres to achieve a parsimonious

network structure. In this work, in order to improve the computational efficiency, the procedure of adaptation is amended and compared with the existing algorithm [16] in Table 3.

Table 3 Comparison of procedures in adaptation algorithms

Step	Adaptation Algorithm in [16]	Improved Adaptation Algorithm
1	Adapt new data, compute the contribution of each centre	Adapt new data and compute the contribution of each centre
2	Add a new centre	Prune a centre
3	Compute and assesses the modelling residual, prune a centre	Add a new centre
4	Group the centre	Group the centre

The proposed adaptation algorithm at each sample period is as follows: The first step is to adapt the new data into the network structure and evaluate the computed contribution of each centre. The next step is to prune a centre which has the least contribution. Then, the learning strategy developed in [16] is employed to add a new centre. The last step is to group the centre and the active centres are used to form the final network.

For the existing method in [16], the procedure is to compute the contribution of each centre and add a new centre and then, compute and asses the modelling residual caused by removing each centre to prune a centre. In contrast, the improved adaptation algorithm is to prune a centre and add a new centre based on the computed centres' contributions at step 1, which avoids the computing of the modelling residual caused by removing each centre. The advantage of this improved algorithm is, at each sample time, it reduces one cycle of computing the decomposition of augmented R matrix at step 3, which reduces the total computing load. The action of adding, pruning and grouping of centres is briefly explained in following sections.

4.1 Prune & add centres

At sample time k, the new data $x(k)$ is adapted into R matrix using (19) and (20). The R matrix is updated with the information of new data and the contribution of each centre is evaluated using

$$\left\|\hat{Y}\right\|_F = \sum_{i=1}^{n_h} \left\|\hat{y}_i \hat{y}_i^T\right\|_F \tag{27}$$

where $\hat{y}_i^T$ is the ith row of $\hat{Y}$. This shows that the ith centre has its individual contribution to $\left\|\hat{Y}\right\|_F$ of $\left\|\hat{y}_i \hat{y}_i^T\right\|_F$.

Then, the centre with the least contribution is pruned. The following step is, based on the computed contributions of each centre, add a new centre. The location of newly added centre C_{new} is decided [16] using

$$C_{new} = \varepsilon C_{mc} + (1 - \varepsilon) x(k) \tag{28}$$

where ε is a parameter that affects the location of new centre and is chosen between 0 and 1. A bigger ε will move the new centre to the new operating region, while a smaller ε tends to allow the new centre stay at current location. After a new centre is added, the R matrix and $\hat{Y}$ matrix are retrained.

4.2 Grouping centres

The grouping centre algorithm developed in [16] is employed here. The selected centres are stored in a centre bank, where they have been divided into two groups: active centre group and redundant centre group. In a sample period, after a centre is pruned or added, the added centre is classified into one of the group, according to the modelling residual caused by the centre. At the current sample time k, selected active centres are used to form a network model for prediction purpose. Redundant centres are preserved for use in the later sample periods. The grouping centre algorithm commences by quarantining the redundant centres and when the grouping algorithm halts, the remainders are active centres. Akaike's final prediction error (FPE) is used as the stopping criterion,

$$FPE = \frac{1 + \beta(n_w/N_c)}{1 - \beta(n_w/N_c)} V, \quad V = \left\|\tilde{Y}(N)\right\|_F^2 / N_c \tag{29}$$

where V is the loss function, n_w is the number of weights. β is a weighting factor and is also used to decide the number of active centres due to that the sample data N_c in (29) is a fixed parameter. The number of active centres c_a can be decided using [16],

$$c_a = \frac{N_c}{\beta}. \tag{30}$$

The procedure of centre grouping algorithm is summarized as:

Step 1) After the centre bank is updated (adding and pruning of centres), initialize V and FPE.

Step 2) Calculate the loss function V_j when each centre is grouped in turn using (29).

Step 3) Set i = arg min (V_j) and compute the FPE for the smallest loss function, FPE_i using (23). If $FPE_i < FPE$, group the centre i as redundant centre and go to step 4). If $FPE_i > FPE$, go to step 5).

Step 4) After that, set $\hat{R} = R_i$, $\hat{Y} = \hat{Y}_i$, $V = V_i$, FPE = FPE$_i$ and $n_h = n_h - 1$. Go to step 2).

Step 5) Stop the grouping procedure. The remaining centres are active centres and the optimal weight W_j can be computed using (22).

4.3 p-step RBF network adaptation procedure

At each sample time k, the procedure involves the action of adding, pruning and grouping of centres. The overall procedure of the proposed structure and parameter adaptation algorithm for the PS-RBF network model is as follows.

Step 1) Form a centre bank with a pre-specified number of centres using K-means clustering algorithm. Then, use a set of sample data to initialize R matrix and W matrix.

Step 2) At each sample time k, adapts the new data into R matrix using (19) and then, evaluate the contribution of each centre using (27). Prune the centre with the least contribution from the centre bank.

Step 3) Add a new centre into the centre bank using (28). Then, retrain the R matrix and $\hat{Y}$ matrix.

Step 4) The final step is to group the centres using the procedure in Section 4.2 and form a final network model using active centres.

Step 5) Sample time $k = k + 1$, go to step 2.

5 Simulations with network structure adaptation

In this section, simulations have been done with the two nonlinear dynamic processes used in Section 3 to compare the computational cost and prediction performance of non-adaptive and adaptive PS-RBF models. Firstly, the computation times of the existing adaptation algorithm and of the proposed algorithm are compared. Here the computation time is used to indicate the computing load of the algorithm. Secondly, the performance and model compactness of non-adaptive and adaptive PS-RBF networks are also compared using the two systems.

5.1 Modelling the dryer

This simulation example provides two comparisons. The first comparison is for the computational cost between improved and existing adaptation algorithms. And, the second comparison is for the performances of output prediction between the adaptive and non-adaptive PS-RBF networks. MATLAB R2009a on an Intel Core i3 laptop with Windows 7 system is used to carry out the simulation. In Section 3.1.1, it indicates that the PS-RBF network with $H_p = 15$ has the best overall performance. Thus, it is employed in this example.

The adaptive PS-RBF network is chosen to have $N_y = 1$, $N_u = 1$ and delay time $d = 2$. A centre bank with 20 centres is initially formed. In this simulation, ε is selected as 0.1. N_c and β are chosen as 50 and 4, respectively. The computation time and modelling performances are recorded in Table 4. It demonstrates that the computation time of the improved adaptation algorithm is relatively less than the existing adaptation algorithm in [16] while retaining its modelling performance.

At the same time, the performance of the non-adaptive PS-RBF network ($H_p = 15$) in Table 1 is used for comparison purpose. In order to compare the performances of

adaptive and non-adaptive networks, the identical 500 validation data in Section 3.1.1 is used for this simulation. From Table 1, the MAE values of PS-RBF network (with 20 centres) with $H_p = 15$ in validation data is 0.10421. The compared results show that the proposed adaptive PS-RBF network, as shown in Fig. 6 and Table 4, outperforms the non-adaptive PS-RBF network using a more compact network structure. In Table 4, the CPU time shown is for all the 500 samples calculated with the Matlab codes. If C code is used the time used would be greatly reduced to probably 1/10 of original time or less. Thus, the benefit of adopting the structure adaptation algorithm is that it enables PS-RBF network to employ a compact network structure without degrading the modelling performance.

Table 4 Performance comparison between different adaption algorithms for dryer data

Adaptation Algorithm	Number of centres	CPU time (s)	Validation data MAE
Algorithm in [16]	13	40.774462	0.061487
Improved algorithm	13	36.978412	0.061793

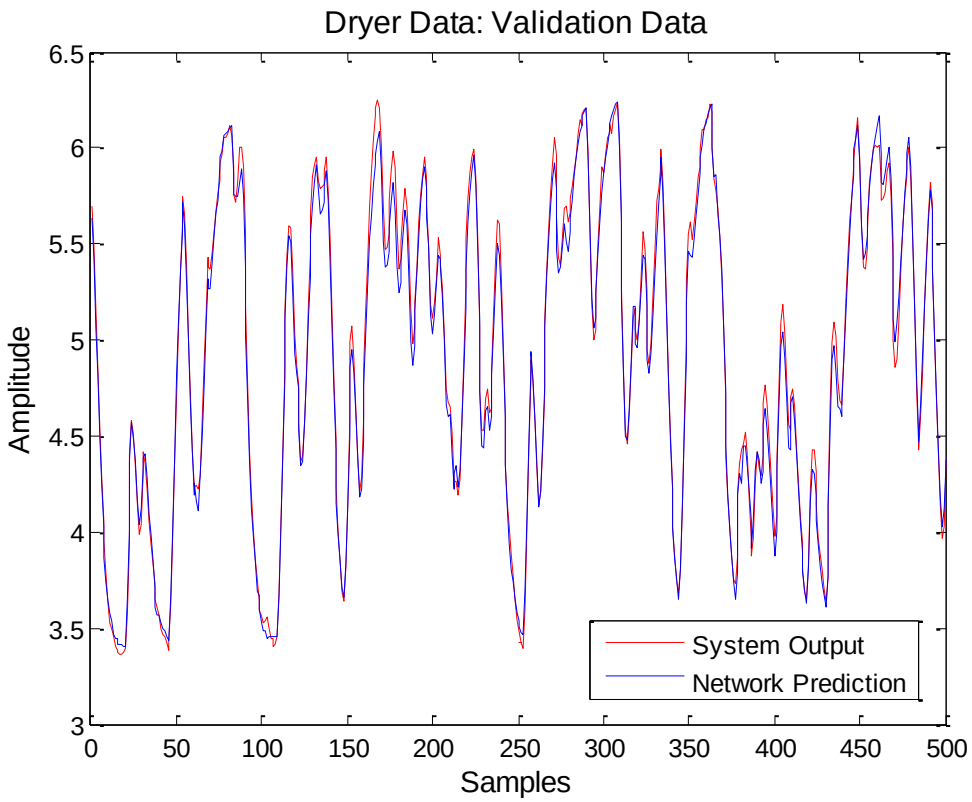

Fig. 6 Performance of adaptive PS-RBF model ($H_p = 15$) in validation data of dryer

5.2 A numerical example

Consider a nonlinear dynamic system to be modelled which is described in [20, 21],

$$y(k+1) = \frac{y(k)y(k+1)(y(k)+2.5)}{1+y(k)^2+y(k-1)^2} + u(k) \qquad (24)$$

where u and y are the system input and output, respectively. The initial conditions are $y(1) = 0$ and $y(2) = 0$. The 500 inputs data are generated by a unit value addition to random values superimposed by a constant value 0.2. The prediction horizon $H_p = 15$ is selected. The model output and input orders of adaptive PS-RBF network model are $N_y = 1$ and $N_u = 1$, respectively. Other parameters are $\varepsilon = 0.2$, $N_c = 50$ and $\beta = 3$. An initial centre bank with 40 centres is formed. The performances of both non-adaptive and proposed adaptive PS-RBF network models are compared. The parameters and modelling performances of both networks are listed in Table 5. The performance of proposed adaptive PS-RBF network model is displayed in Fig. 7. With the proposed adaptive

structure, the adaptive PS-RBF model manages to achieve similar performances with a more compact network structure as shown in Table 5.

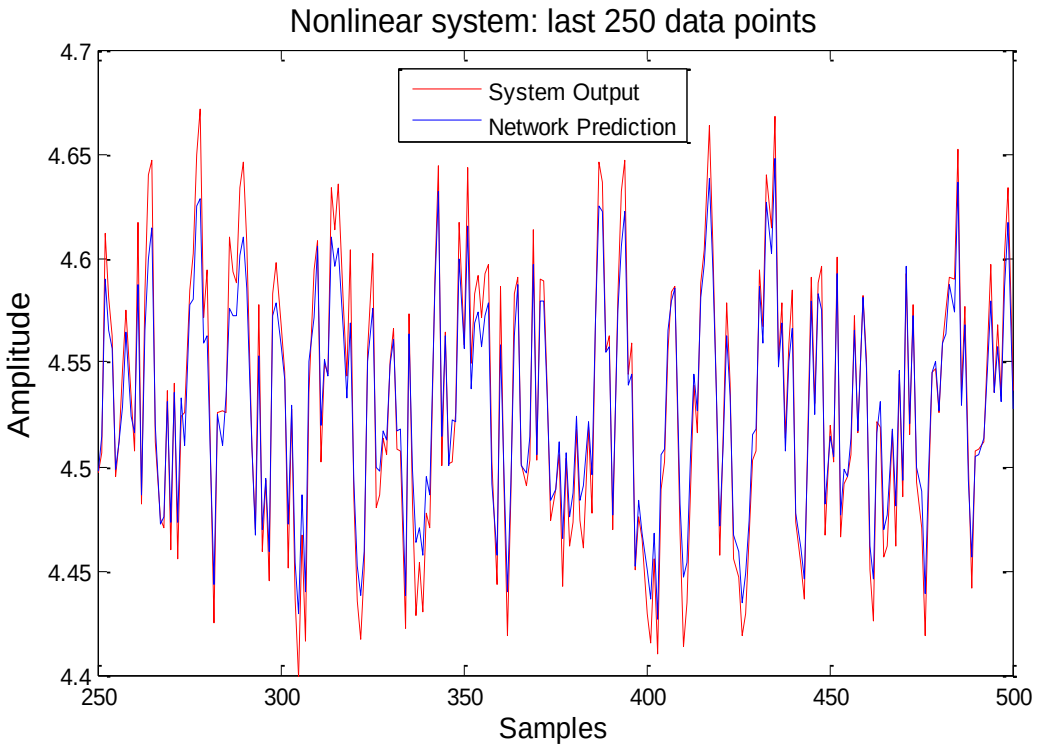

Fig.7 Performance of proposed adaptive PS-RBF model

Table 5 Performance of non-adaptive and adaptive PS RBF networks

Adaptive PS RBF Network	Non-adaptive	Adaptive
Number of centres	60	17
MAE	0.010105	0.010236

6 Conclusions

In this paper, a p-step ahead prediction model is developed for nonlinear dynamic systems and implemented with RBF network. The proposed model is adapted for both structure and parameters with the ROLS algorithm. The model is especially useful to be used as the internal model in the model predictive control framework for nonlinear dynamic systems. Compared with the existing PSC model for modelling two nonlinear dynamic systems, it is demonstrated by simulations that the developed model

outperforms the PSC network model in terms of output prediction performance and model compactness. These two features make the developed model more suitable to be used in the model predictive control.

Furthermore, the structure adaptation using the ROLS algorithm is improved by restructuring the algorithm for decomposition of the augmented *R* matrix, so that the computing time is significantly reduced. Then, the algorithm is used to adapt both structure and parameters of the developed model. This further enhances the effectiveness of the developed model with less computing time and more compact model. Simulation results of the two application examples confirmed these improvements.

References

1. Hao, Y., et al., *Advantages of radial basis function networks for dynamic system design.* Industrial Electronics, IEEE Trans. on, 2011. **58**(12): p. 5438-5450.
2. Broomhead, D.S. and D. Lowe, *Multivariable functional interpolation and adaptive networks.* Complex Systems, 1988. **2**: p. 321-355.
3. Diaconescu, E., *The use of NARX neural networks to predict chaotic time series* WSEAS Trans. on Computer Research, 2008. **3**(3): p. 182-191.
4. Su, T.H. and T.J. McAvoy, *Artificial neural network for nonlinear process identification and control*. Nonlinear Process Control, ed. M.A. Henson and D.E. Seborg. 1997, Englewood Cliffs, NJ: Prentice Hall. 371-428.
5. Bhartiya, S. and J.R. Whiteley, *Factorized approach to nonlinear MPC using a radial basis function model.* AIChE Journal, 2001. **47**(2): p. 358-368.
6. Chen, S., P.M. Grant, and C.F.N. Cowan, *Orthogonal least-square algorithm for training multioutput radial basis function networks.* Radar and Signal Processing, IEE Proceedings F, 1992. **139**(6): p. 378-384.
7. Platt, J., *A resource-allocating network for function interpolation.* Neural Computing, 1991. **3**: p. 213-225.
8. Karayiannis, N.B. and G.W. Mi, *Growing radial basis neural networks: merging supervised and unsupervised learning with network growth techniques.* IEEE Trans. on Neural Netwoks, 1997. **8**(6): p. 1492-1506.
9. Todorovic, B. and M. Stankovic, *Sequential growing and pruning of radial basis function network*, in *Neural Networks, 2001. Proceedings. IJCNN '01. International Joint Conference on*. 2001, IEEE. p. 1954-1959.

10. Han, H.-G., Q.-l. Chen, and J.-F. Qiao, *An efficient self-organizing RBF neural network for water quality prediction.* Neural Networks, 2011. **24**(7): p. 717-725.

11. Chen, S., C.F.N. Cowan, and P.M. Grant, *Orthogonal least squares learning algorithm for radial basis function networks.* IEEE Trans. on Neural Netwoks, 1991. **2**(2): p. 302-309.

12. Chang, E., H.H. Yang, and S. Bos, *Orthogonal least-squares learning algorithm with local adaptation process for the radial basis function networks.* IEEE Signal Processing Letters, 1996. **3**(8): p. 253-255.

13. Yu, D.L., J.B. Gomm, and D. Williams, *A recursive orthogonal least squares algorithm for training RBF networks.* Neural Processing Letters, 1997. **5**(3): p. 167-176.

14. Gomm, J.B. and D.L. Yu, *Selecting radial basis function network centres with recursive orthogonal least squares training.* IEEE Trans. on Neural Netwoks, 2000. **11**(2): p. 306-314.

15. Yu, D.L. and D.W. Yu, *A new structure adaptation algorithm for RBF networks and its application.* Neural Computing and Applications, 2007. **16**(1): p. 91-100.

16. Tok, D.K.S., et al., *Adaptive structure radial basis function network model for processes with operating region migration.* Neurocomputing, 2015. **155**: p. 186-193.

17. Wang, S.W., et al., *Adaptive neural network model based predictive control for air–fuel ratio of SI engines.* Engineering Applications of Artificial Intelligence, 2006. **19**(2): p. 189-200.

18. Yu, D.L., et al. *Adaptive RBF model for model-based control*. in *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on*. 2004.

19. Qiao, J.-F. and H.-G. Han, *Identification and modeling of nonlinear dynamical systems using a novel self-organizing RBF-based approach.* Automatica, 2012. **48**(8): p. 1729-1734.

20. Han, H.-G., X.-L. Wu, and J.-F. Qiao, *Real-Time Model Predictive Control Using a Self-Organizing Neural Network.* IEEE Trans. on Neural Netwoks and Learning Systems, 2013. **24**(9): p. 1425-1436.

21. Lu, C.H. and C.C. Tsai, *Adaptive predictive control with recurrent neural network for industrial process: An application to temperature control of a variable -frequency oil-cooling machine.* IEEE Trans. Ind. Electron, 2008. **55**(3): p. 1366-1375.