amcs

ADAPTIVE CONTROL SCHEME BASED ON THE LEAST SQUARES SUPPORT VECTOR MACHINE NETWORK

TAREK A. MAHMOUD

Industrial Electronics and Control Engineering Department
Faculty of Electronic Engineering, Menoufia University, Menouf 32952, El-Menoufiya, Egypt
e-mail: `tarek_momeen@yahoo.com`

Recently, a new type of neural networks called Least Squares Support Vector Machines (LS-SVMs) has been receiving increasing attention in nonlinear system identification and control due to its generalization performance. This paper develops a stable adaptive control scheme using the LS-SVM network. The developed control scheme includes two parts: the identification part that uses a modified structure of LS-SVM neural networks called the multi-resolution wavelet least squares support vector machine network (MRWLS-SVM) as a predictor model, and the controller part that is developed to track a reference trajectory. By means of the Lyapunov stability criterion, stability analysis for the tracking errors is performed. Finally, simulation studies are performed to demonstrate the capability of the developed approach in controlling a pH process.

Keywords: least squares support vector machine, multi-resolution wavelet least squares support vector machine neural network, nonlinear system modeling and control, pH control.

1. Introduction

In the last decades, neural networks have proved to be a powerful scheme in control strategies (Robert and Jean-Jacques, 1992; Parisini and Zoppoli, 1994; Ge and Wang, 2004; Ge *et al*., 2008). Despite all the these successes, they suffer from many local minima solutions and the issue of the selection of an optimum structure. At this point, considerable progress has been achieved by means of a new class of neural networks called Support Vector Machines (SVMs), introduced by Vapnik (1998).

SVMs are constructed based on statistical learning and structural risk minimization, which can give attention to both the expectation risk and the generalization performance. The structure and parameter learning of SVM networks are characterized by solving a convex optimization problem. The notion of network complexity (e.g., the number of hidden units) is primarily related to solving this convex optimization problem. Fundamentally, this network uses a kernel to map the data in the input space to a high dimensional feature space in which the problem becomes linearly separable (Vapnik, 1998; Suykens and Vandewalle, 1999; Suykens *et al*., 2002). There are many kinds of kernel that can be used, such as the Radial Basis Function (RBF) and polynomial kernels. Motivated by the

effective characteristics of the wavelet for each of the non-stationary signal approximation and classifications, some studies using the wavelet kernel for the SVM have been presented (Zhang *et al.*, 2004; 2005).

In fact, the solution of the convex optimization problem of SVM networks is a computational intensive and memory consuming method, especially if a large training set of high-dimensional data is used. This is a serious drawback. Least Squares Support Vector Machine (LS-SVM) networks are developed as an improvement of SVM networks (Saunders *et al*., 1998; Suykens, 2001). In this scheme, the objective function of the optimization problem includes an additional sum squared error term. Hence, it reduces the computation time of the convex optimization problem. Motivated by this characteristic, many efforts have been conducted using the LS-SVM network for modeling and control purposes (Suykens, 2001; Suykens *et al*., 2001; Li *et al*., 2006; Li-Juan *et al*., 2007; Zhang and Wang, 2008; Wang and Yuan, 2008).

In the work of Mahmoud (2010), a modified structure of LS-SVM networks using the wavelet kernel named the Multi-Resolution Wavelet Least Squares Support Vector Machine (MRWLS-SVM) network is developed. The network emerges between the multi-resolution property of the wavelet transform and the LS-SVM learning

algorithm. It develops the strategy of forming a nonlinear system to be modeled with a set of Wavelet Least Squares Support Vector Machine (WLS-SVM) networks. Each sub-WLS-SVM network consists of a wavelets kernel with a specified resolution. The outputs of these sub-WLS-SVM networks are weighted and aggregated via a set of weights to produce the final output. One of the main features of the MRWLS-SVM network is a dynamic network that is inherited from the MRWLS-SVM network output fed back to the input via a set of tapped delays.

This study develops a stable adaptive control scheme based on the MRWLS-SVM network. The MRWLS-SVM has been employed as a model predictor in the control scheme. The parameters of the controller are trained using the gradient descent method. The convergence of this controller is proved using Lyapunov stability theory.

The rest of the paper is organized as follows. Section 2 discusses the basic concept of the LS-SVM network. Section 3 reviews the basic structure and the learning phases of the MRWLS-SVM network. Section 4 presents the proposed control scheme. Section 5 tests the soundness of the proposed control scheme via controlling a pH neutralization process. Section 6 concludes the proposed scheme.

2. LS-SVM neural network

The basic concept of the LS-SVM network can be discussed as follows: Given a training data $(x_i; y_i)_{i=1}^N$, where N is the number of training data pairs, $x_i \in \mathbb{R}^m$ is the input vector and $y_i \in \mathbb{R}$ is the output. According to SVM theory, the input space $\mathbb{R}^m$ is mapped into a higher dimension feature space using a nonlinear function $\Phi(x_i)$. In the feature space, the following function is used to estimate the unknown function:

$$y_i(x_i) = w^T \Phi(x_i) + b, \quad (1)$$

where w and b are the estimated parameters. The optimization problem of the LS-SVM network is defined as

$$\min_{w,b,e} J(w,e) = \frac{1}{2} w^T w + \frac{\gamma}{2} \sum_{i=1}^N e_i^2, \quad (2)$$

subject to the equality constraints

$$y_i = w^T \Phi(x_i) + b + e_i, \quad (3)$$

where e_i is the error between the actual and the estimated outputs of the i-th datum. The Lagrange function to solve this optimization problem is defined as

$$L(w, b, e, \alpha)$$
$$= J(w,e) - \sum_{i=1}^N \alpha_i (w^T \Phi(x_i) + b + e_i - y_i), \quad (4)$$

where $\alpha = (\alpha_i)_{i=1}^N$ is the set of Lagrange multipliers. Hence, the set optimality conditions are given by

$$\frac{\partial L}{\partial w} = 0 \rightarrow w = \sum_{i=1}^N \alpha_i \Phi(x_i), \quad (5)$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^N \alpha_i = 0, \quad (6)$$

$$\frac{\partial L}{\partial e_i} = 0 \rightarrow \alpha_i = \gamma e_i, \quad (7)$$

$$\frac{\partial L}{\partial \alpha_i} = 0 \rightarrow w^T \Phi(x_i) + b + e_i - y_i = 0,$$
$$\forall i = 1, 2, \ldots, N. \quad (8)$$

Defining

$$Y = [y_1, y_2, \ldots, y_N]^T,$$
$$1 = [1, 1, \ldots, 1]^T,$$
$$\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_N]^T,$$

Ω is a square matrix in which the element located in the i-th row is $\Omega_{il} = \Phi(x_i)\Phi(x_l) = K(x_i, x_l), \forall i, l = 1, 2, \ldots, N$. Eliminating w, e, the following set of linear equations is obtained:

$$\begin{bmatrix} 0 & 1^T \\ 1 & \Omega + \gamma^{-1}I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix}. \quad (9)$$

Choosing $\gamma > 0$ ensures that the matrix

$$\begin{bmatrix} 0 & 1^T \\ 1 & \Omega + \gamma^{-1}I \end{bmatrix}$$

is invertible. Then the analytical solution for b and α is given by

$$\begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 & 1^T \\ 1 & \Omega + \gamma^{-1}I \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ Y \end{bmatrix}. \quad (10)$$

Substituting the obtained b and α into Eqns. (5–7) yields

$$y(x) = \sum_{i=1}^N \alpha_i K(x, x_i) + b, \quad (11)$$

where $K(x, x_i)$ is the kernel function, which can be any symmetric function satisfying the Mercer condition (Vapnik, 1998). Several choices for the kernel $K(x, x_i)$ are possible:

- linear kernel: $K(x, x_i) = (x_i^T x)$;

- polynomial kernel of degree r:

$$K(x, x_i) = (x_i^T x + 1)^r;$$

- RBF kernel: $K(x, x_i) = \exp(- \|x - x_i\|_2^2 / \delta^2)$.

Owing to the effective characteristic of wavelets for non-stationary signal approximation and classification, the wavelet kernel has been investigated (Zhang *et al.*, 2004; 2005; Huicheng and Taiyi, 2008). It is given by

$$K(x, x_i) = \prod_{j=1}^{m} h\Big(\frac{x^j - x_i^j}{a}\Big), \quad (12)$$

where $h(x)$ is a mother wavelet and a denotes the dilation (i.e., resolution) parameter.

3. MRWLS-SVM neural network

Mahmoud (2010) developed a modified structure of LS-SVM networks named the multi-resolution wavelet LS-SVM network. The architecture of the this network comprises a set of sub-LS-SVM networks based on the wavelet kernel named the WLS-SVM network with a specified resolution. The outputs of these sub-WLS-SVM networks are weighted and aggregated via a set of adaptable weights to produce the final output of the proposed network. This section reviews in brief the structure of the MRWLS-SVM network.

With no loss of generality, suppose that the proposed network has m inputs and one output. The structure of the network in the sense of the layer description can be described as follows

1. *Layer 1* is an input layer. This layer accepts the input variables and directly transmits the accepted inputs to the next layer. The input vector x of the network can be given by

$$\begin{aligned} x &= [x_1, x_2, \ldots, x_m]^T \\ &= [u(t), u(t-1), \ldots, u(t-s), \\ &\quad y_N(t-r), \ldots, y_N(t-1)]^T, \end{aligned}$$

where u is the external input variable of the system to be modeled, $y_N(t-i)$, $i = 1, 2, \ldots, r$ are the past values of the network output, s is the number of the past values of the input variable, and r is the number of the past values of the network output.

2. *Layer 2* is a set of sub-WLS-SVM networks. The output of each WLS-SVM network is given by

$$y_w^l(x) = \sum_{i=1}^{c} \alpha_{il} K_l(x, x_i) + b_l, \quad l = 1, \ldots, d, \quad (13)$$

where α_{il} is the weight of the i-th Lagrange multiplier associated with the support vectors x_i for the l-th WLS-SVM network, b_l is the bias term for the l-th WLS-SVM network, and d is the number of the sub-WLS-SVM networks. $K_l(x; x_i)$ is the wavelet kernel for the l-th WLS-SVM network and can be

calculated by the tensor product of m single wavelet basis functions that is generated by dilating and translating the mother wavelet, i.e.,

$$K_l(x, x_i) = \prod_{j=1}^{m} h\Big(\frac{x^j - x_i^j}{a_l}\Big), \quad i = 1, \ldots, c, \quad (14)$$

where $h(x)$ is a mother wavelet, a denotes the dilation (i.e., resolution) parameter, and c is the number of the support vectors.

3. *Layer 3* is an output layer. At this layer, the outputs of the sub-WLS-SVM networks are weighted and aggregated to find the final output of the network. Therefore, the output of the proposed network can be given as follows:

$$y_N(x) = \sum_{l=1}^{d} w_l y_w^l(x), \quad (15)$$

where w_l is the l-th network weight and $y_w^l(x)$ is the output of the l-th WLS-SVM network defined in (13), which has the same structure of the LS-SVM network using the wavelet kernel. By substituting (13) in (15), the final output of the MRWLS-SVM network can be reformulated as

$$y_N(x) = \sum_{l=1}^{d} w_l \Big(\sum_{i=1}^{c} \alpha_{il} K_l(x, x_i) + b_l \Big). \quad (16)$$

The learning of the network consists of two phases: structure learning and parameter learning. The structure learning phase is the corner stone to develop an optimal structure of the MRWLS-SVM network. It should determine the following seeds:

- the number of the support vectors c for each WLS-SVM network,

- the Lagrange multiplier weights α_{il} associated with each support vectors x_i and the bias term b_l for each l-th WLS-SVM network.

A hybrid technique including the fuzzy C-mean clustering algorithm (Bezdek, 1981) and the LS-SVM algorithm was developed to learn the structure of the MRWLS-SVM network. Thus, the fuzzy c-mean clustering method was provided to construct the structure of each WLS-SVM network. It partitions the training data into a group of clusters and then uses these clusters to train each WLS-SVM network. This means that each WLS-SVM network incorporates only clusters in its structure instead of incorporating all training data. More details of this algorithm can be found in the work of Mahmoud (2010).

After setting up the structure of each sub-WLS-SVM network, an approach using the Back-Propagation (BP)

algorithm with an adaptive learning rate is developed to train the contribution weights w_l of the sub-WLS-SVM networks. The main objective of this training phase is to minimize the following cost function:

$$E = \frac{1}{2}(y_r(t) - y_N(t))^2 = \frac{1}{2}e(t)^2, \qquad (17)$$

where $y_r(t)$ is the current desired output of the network, $y_N(t)$ is the current network output and $e(t)$ is the error in the learning phase. The gradient of the error with respect to the network weight vector $W = [w_1, w_2, \ldots, w_d]^T$ is given by

$$\frac{\partial E}{\partial W(t)} = -e(t)\frac{\partial y_N(t)}{\partial W(t)}. \qquad (18)$$

Hence, the update rule of the weight W can be given by

$$\begin{aligned} W(t+1) &= W(t) + \eta\Big(-\frac{\partial E}{\partial W(t)}\Big) \\ &= W(t) + \eta e(t)\frac{\partial y_N(t)}{\partial W(t)}, \end{aligned} \qquad (19)$$

where η is the learning rate for the weights. The stability of the MRWLS-SVM network has been investigated with the aid of Lyapunov theory according to the following theorem.

Theorem 1. *Let η be the learning rate for the weights of the MRWLS-SVM network and $D_{\max}$ be defined as $D_{\max} = \max_t \|Y_w(t)\|^2$, where Y_w is the vector containing the outputs of the sub-WLS-SVM networks, and $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^d$. Then the convergence of the proposed network is guaranteed if η is chosen as*

$$0 < \eta < \frac{2}{D_{\max}}. \qquad (20)$$

The details of the proof of the above theorem can be found in the work of Mahmoud (2010).

Theorem 1 can make the update rule of the weight W with an adaptable learning rule as follows:

$$W(t+1) = W(t) + \eta(t)e(t)\frac{\partial y_N(t)}{\partial W(t)}, \qquad (21)$$

where $\eta(t)$ can be adopted at any instant t as

$$\eta(t) = \eta(0)\frac{2}{D_{\max}(t)}, \qquad (22)$$

with $\eta(0)$ being a positive constant number less than 1 and $D_{\max}(t)$ given as

$$D_{\max}(t) = \max(D_{\max}(t-1), \|Y_w(t)\|^2), \qquad (23)$$

Passing reference to the nonlinear modeling results and the dynamic features of the MRWLS-SVM described by Mahmoud (2010), the following sections develop an adaptive control scheme based on this network.

4. Proposed adaptive control scheme based on the MRWLS-SVM network

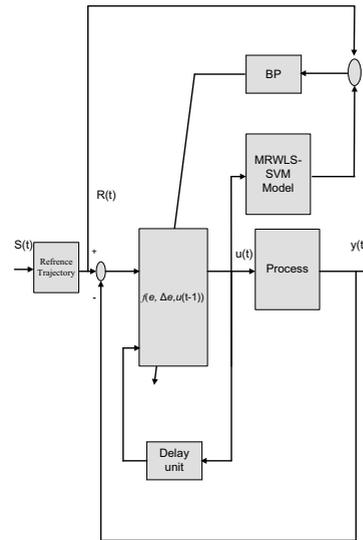This section describes the overall structure and the stability analysis of the proposed control scheme.

Fig. 1. Structure of the adaptive control scheme.

4.1. Structure of the proposed adaptive control scheme.
The overall structure of the MRWLS-SVM network based adaptive control scheme is depicted in Fig. 1. It includes three main parts: the actual plant, the process model based on the MRWLS-SVM network and the controller. In this scheme, the MRWLS-SVM network is introduced as a tool to predict the future outputs of the plant. That is given by the nonlinear approximation g:

$$y_N(t+j) = g(\theta, u(t), y_N(t+j-1)), \qquad (24)$$

where $y_N(t+j)$ is the prediction output of the plant at the prediction horizon j using the MRWLS-SVM network, θ denotes the network parameters, and $u(t)$ is the controller output at instant t that can be assumed as

$$u(t) = f(e_c(t), \Delta e_c(t), u(t-1)), \qquad (25)$$

where $f(\cdot)$ is a linear or a nonlinear function, $e_c(t) = R(t) - y(t)$ is the controller error at the sample instant t, $R(t)$ is the reference trajectory, $\Delta e_c(t)$ is the rate of change in the controller error, and $u(t-1)$ is the controller output at the sample instant $t-1$. For simplicity, a linear form of the controller function is assumed as

$$u(t) = u(t-1) + b_1 e_c(t) + b_2 \Delta e_c(t), \qquad (26)$$

where $b_{1,2}$ are the controller gains.

The aim of the adaptive scheme is that the future outputs on the horizon considered should follow a predetermined reference trajectory. Accordingly, the cost function considered is given by

$$J_p = 0.5(E_m^T(t+1)E_m(t+1)), \qquad (27)$$

where $E_m(t+1)$ is the prediction error defined by

$$E_m(t+1) = [e_p(t+1), e_p(t+2), \ldots, e_p(t+N_p)]^T, \qquad (28)$$

with

$$e_p(t+j) = R(t+j) - y_N(t+j) \qquad (29)$$

being the prediction error at instant j, $j = 1, 2, \ldots, Np$, R the desired trajectory, y_N the output of the MRWLS-SVM network predictor, and N_p the maximum output horizon.

In order to minimize J_p, the controller gain vector $B = [b_1, b_2]^T$ is recursively adapted using the following update scheme:

$$B(t+1) = B(t) + \eta_c\left(-\frac{\partial J_p}{\partial B(t)}\right), \qquad (30)$$

where η_c is the learning rate for the controller gains. Differentiating the cost function J_p with respect to the controller gain vector can be implemented using the back-propagation algorithm as a gradient descent algorithm via the MRWLS-SVM network predictor. Those are obtained by

$$\frac{\partial J_p}{\partial B(t)} = -\left[\frac{\partial Y_N(t+1)}{\partial B(t)}\right]E_m(t+1)$$
$$= X_c\left[\frac{\partial Y_N(t+1)}{\partial u(t)}\right]E_m(t+1), \qquad (31)$$

where $Y_N(t+1) = [y_N(t+1), y_N(t+2), \ldots, y_N(t+N_p)]^T$ is the vector of the future values of the network predictor output and $X_c = [e_c(t), \Delta e_c(t)]^T$ is the input vector of the controller. The computing procedure for $[\partial Y_N(t+1)/\partial u(t)]$ is presented in Appendix.

4.2. Stability analysis for the proposed control scheme.
To investigate the stability of the controller the following discrete Lyapunov function is considered:

$$V_m(t) = \frac{1}{2}(E_m^T(t+1)E_m(t+1)), \qquad (32)$$

where $E_m(t+1)$ is the prediction error defined in (28). The change in the Lyapunov function is obtained by

$$\Delta V_m(t) = \frac{1}{2}(E_m^T(t+2)E_m(t+2) - E_m^T(t+1)E_m(t+1)). \qquad (33)$$

The prediction error difference can be adopted as (Ku *et al.*, 1995; Yoo *et al.*, 2005)

$$\Delta E_m(t+1) \approx \left[\frac{\partial E_m(t+1)}{\partial B(t)}\right]\Delta B(t)$$
$$\approx \left[\frac{\partial Y_N(t+1)}{\partial u(t)}\right]X_c^T \Delta B(t), \qquad (34)$$

where $\Delta B(t)$ is the change in the controller gain. Using (30) and (31), $\Delta B(t)$ can be obtained by

$$\Delta B(t) = -\eta_c X_c \left[\frac{\partial Y_N(t+1)}{\partial u(t)}\right]^T E_m(t+1). \qquad (35)$$

Then $\Delta E_m(t+1)$ in (34) can be rewritten as

$$\Delta E_m(t+1) = -\eta_c \|X_c\|^2 GE_m(t+1), \qquad (36)$$

where $G_{N_p \times N_p}$ is a square matrix given by

$$G = \left[\frac{\partial Y_N(t+1)}{\partial u(t)}\right]\left[\frac{\partial Y_N(t+1)}{\partial u(t)}\right]^T. \qquad (37)$$

Lemma 1. *Let a positive matrix $M_{N \times N}$ be defined as $M = \beta I - A$ where $A_{N \times N}$ is a positive symmetric matrix and β is a real number, and I is the identity matrix. Then the eigenvalues of M can be represented as follows:*

$$\mathrm{eig}(M) = \mathrm{eig}(\beta I) - \mathrm{eig}(A),$$

where

$$\mathrm{eig}(M) = [\lambda_{M,1}, \lambda_{M,2}, \ldots, \lambda_{M,N}]^T,$$
$$\mathrm{eig}(A) = [\lambda_{A,1}, \lambda_{A,2}, \ldots, \lambda_{A,N}]^T,$$
$$\mathrm{eig}(\beta I) = [\beta, \beta, \ldots, \beta]^T.$$

Theorem 2. *Let η_c be the learning rate for the controller gains. Then the stability of the controller is guaranteed if η_c is chosen to satisfy*

$$0 < \eta_c < \frac{2}{g_{\max}\lambda_{\max}}, \qquad (38)$$

where $\lambda_{\max}$ is the maximum eigenvalue of the matrix G defined in (37), and $g_{\max} = \max_t \|X_c(t)\|^2$.

Proof. Using (36), the change in the Lyapunov function defined in (33) can be expressed as

$$\Delta V_m(t) = \Delta E_m^T(t+1)(E_m(t+1) + \frac{1}{2}\Delta E_m(t+1))$$
$$= -E_m^T(t+1)\eta_c \|X_c(t)\|^2 G^T$$
$$= E_m(t+1) - \frac{1}{2}\|X_c(t)\|^2 GE_m(t+1)$$
$$= -E_m^T(t+1)\Gamma PE_m(t+1), \qquad (39)$$

where

$$\Gamma = \eta_c \left\| X_c(t) \right\|^2 G^T,$$
$$P = I - \frac{1}{2}\eta_c \left\| X_c(t) \right\|^2 G.$$

∎

If Γ and P are positive definite matrices, then $\Delta V_m(t) < 0$. This means that Lyapunov stability is guaranteed. To get $\Gamma > 0$ and $P > 0$, all eigenvalues in each of them must be positive. The condition $\Gamma > 0$ is satisfied if $\eta_c > 0$, and

$$P = I - \frac{1}{2}\eta_c \left\| X_c(t) \right\|^2 G = \frac{2}{\eta_c \left\| X_c(t) \right\|^2} I - G > 0.$$

Here, according to Lemma 1, $\eta_c < 2/g_{\max}\lambda_{\max}$ is obtained such that $\lambda_{\max}$ is the maximum eigenvalue of the matrix G defined in (37), and $g_{\max} = \max_t \left\| X_c(t) \right\|^2$. Therefore, the stability of the controller is guaranteed if η_c is chosen as

$$0 < \eta_c < \frac{2}{g_{\max}\lambda_{\max}}.$$

According to Theorem 2, the adaptation of the controller parameters defined in (30) can be expressed as

$$B(t+1) = B(t) + \eta_c(t)\left(-\frac{\partial J_p}{\partial B(t)}\right), \qquad (40)$$

where $\eta_c(t)$ is determined at any instant t as follows:

$$\eta_c(t) = \eta_c(0)\frac{2}{g_{\max}(t)\lambda_{\max}(t)}, \qquad (41)$$

with $\eta_c(0)$ being a small positive number < 1,

$$g_{\max}(t) = \max(g_{\max}(t-1), \left\| X_c(t) \right\|^2),$$
$$\lambda_{\max}(t) = \max(\lambda_{\max}(t-1), \mathrm{eig}(G)). \qquad (42)$$

5. Simulation results

In this section, the proposed adaptive control algorithm is investigated via controlling the pH neutralization process. In fact, the problem of controlling the pH process can be included in a variety of practical areas such as waste water treatment, biotechnology processing, and chemical processing (Henson and Seborg, 1994; Wang *et al.*, 2002; Faanes and Skogestad, 2004). The difficulty in controlling the pH process arises mainly from its heavy nonlinearity and uncertainty. Therefore, the control of the pH process is used here as a good example to investigate the proposed adaptive control scheme.

5.1. pH neutralization process description. The pH neutralization process is a dynamic system with strong nonlinearity. In this process, acetic acid (weak acid) is neutralized by a strong base NaOH in water. The physical model of a pH process in a Continuously Stirred Tank Reactor (CSTR) consists of two parts; a linear dynamical part followed by a nonlinear static part (Nie *et al.*, 1996). The dynamic part is given by

$$V\frac{\partial w_a}{\partial t} = F_a c_a - (F_a + F_b)w_a,$$
$$V\frac{\partial w_b}{\partial t} = F_b c_b - (F_a + F_b)w_b, \qquad (43)$$

F_a and F_b denote the inlet flow rate of the acid and base [cm³/min], respectively, C_a and C_b are the inlet concentration of acid and base [mol/L], V is the volume of the content in the reactor (cm³), and w_a and w_b are the concentration of acid and base after the neutralization process [mol/L]. Simultaneously, w_a and w_b are the inputs of the static part. That is given by

$$w_b + 10^{-\mathrm{pH}} - 10^{\mathrm{pH}-14} - \frac{w_a}{10^{\mathrm{pKa}-\mathrm{pH}}} = 0, \qquad (44)$$

where $k_a = 1.76 \times 10^{-5}$ is the dissociation constant of the acetic acid and $\mathrm{pKa} = -\log(Ka)$. The nominal parameters of this physical model are given in Table 1. During the simulation, the acid inlet flow rate F_a is fixed at its nominal value given in Table 1. This means that the process is regarded as a single variable system with the base inlet flow rate F_b and the pH value being the input and the output, respectively.

The first step for the proposed control scheme is to design a predictor model of the pH neutralization process using the MRWLS-SVM network. Hence, the following subsection will describe the MRWLS-SVM model of the pH process.

5.2. MRWLS-SVM model of the pH neutralization process. While building an MRWLS-SVM model for the pH process, the base inlet flow rate F_b was set as the sum of its nominal value given in Table 1 plus a uniformly distributed signal ranging in the interval $[-51.5, 51.5]$, as depicted in Fig. 2, to generate a 1000 training data set $\{x(t), \mathrm{pH}(t)\}_{t=1}^{1000}$. This training data set is partitioned into 20 clusters to construct the structure of sub-WLS-SVM networks for the MRWLS-SVM network, and 5 WLS-SVM networks are constructed. This means that 100 wavelet kernels were used in the MRWLS-SVM network structure for the pH neutralization process. During the structure learning phase, the input vector is assumed as $x(t) = [F_b(t-1), \mathrm{pH}(t-1]^T$, and then it is shifted to $x(t) = [F_b(t-1), y_N(t-1]^T$, where $y_N(t)$ is the network output during the learning parameter phase to achieve the dynamic properties of the constructed model for the pH neutralization process.

Table 1. Nominal parameters for the pH neutralization process.

Parameters	Nominal values
F_a	81 cm^3/min
$F_b(0)$	515 cm^3/min
c_a	0.32 mol/L
c_b	0.05 mol/L
V	1000 cm^3
$w_a(0)$	0.0435 mol/L
$w_b(0)$	0.0432 mol/L
sample period	0.5 min
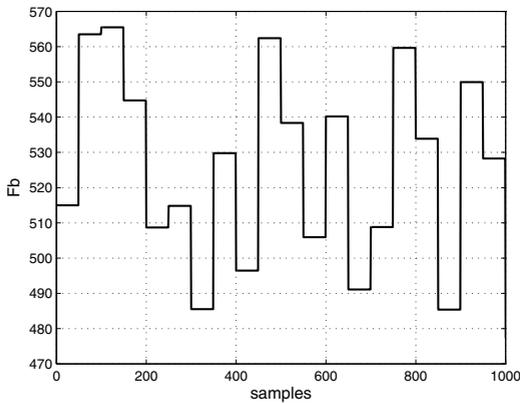
Fig. 2. Base inlet flow rate training signal.

Fig. 3. Base inlet flow rate signal during the test phase.

Fig. 4. pH value and the output of the MRWLS-SVM model during the test phase.

After the MRWLS-SVM network model of the pH process is constructed, it is tested using a 1000 test data set $\{x(t), \mathrm{pH}(t)\}_{t=1}^{1000}$ generated by the following test function:

$$F_b(t) = F_b(0) + 51.5\sin(2\pi t/100). \tag{45}$$

Figure 3 depicts the base inlet flow rate F_b used in the test phase of the MRWLS-SVM model for the pH process. The model output and the pH response during the test phase are shown in Fig. 4. This figure shows the fast convergence of the MRWLS-SVM network and the efficiency of the network to effectively handle the strong nonlinearity of the pH process. However, it is clear that the output errors are high for minimum but not for maximum points. This can be improved by increasing the number of training epochs as shown in Fig. 5.

For comparison, an RBF kernel based LS-SVM network was designed to model the pH neutralization process. In this comparison, the set of input-output data is generated using the base flow rate given by

$$F_b(t) = \begin{cases} F_b(0) + 51.5\sin(2\pi t/25) & \text{if } t \leq 150, \\ F_b(0) + 25.75\sin(2\pi t/25) \\ \quad + 25.75\sin(2\pi t/10) & \text{if } t > 150. \end{cases} \tag{46}$$
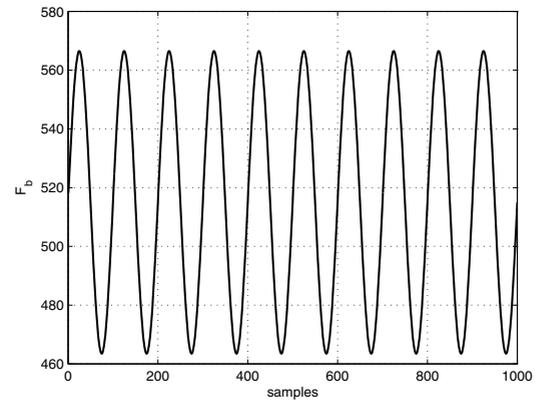
Figures 6 and 7 show the results of the MRWLS-SVM and RBF kernel based LS-SVM models, respectively. It is shown that the modeling accuracy of the MRWLS-SVM network for the pH process is sound compared with that of the RBF kernel based LS-SVM network.

In addition, using the input-output training data set constructed using (46), the performance of the MRWLS-SVM model for the pH process and other algorithms such as the fuzzy neural networks algorithms developed by Nie *et al.* (1996) was compared in terms of the Mean Square
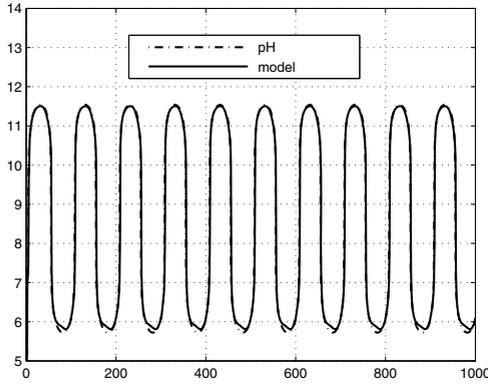
Fig. 5. pH value and the output of the MRWLS-SVM model during the test phase with more training epochs.
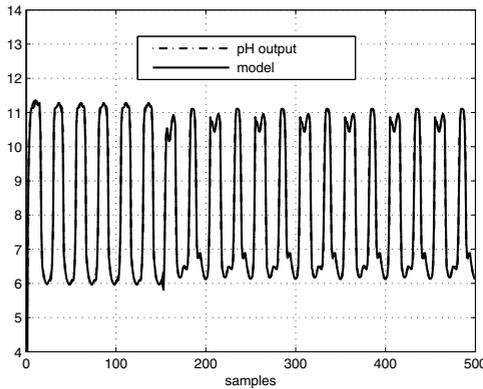

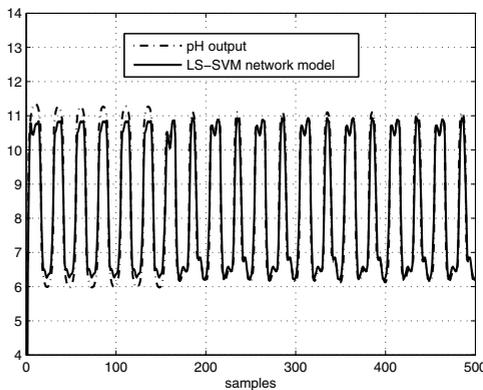Fig. 6. pH value and the output of the MRWLS-SVM model due to the base flow rate defined in (46).

Fig. 7. pH value and the output of the RBF kernel based LS-SVM network model due to the base flow rate defined in (46).

Error (MSE) given as

$$\text{MSE} = \frac{1}{N} \sum_{t=1}^{N} (y_m - y_{\text{pH}})^2, \qquad (47)$$

where y_m and y_{pH} are the model and pH process outputs,

respectively, and N is the number of training data elements. Table 2 shows the obtained MSEs for the MRWLS-SVM network and the other algorithms. The obtained results show that the efficiency of the MRWLS-SVM model of the pH process is higher compared with those of the LS-SVM algorithm and unsupervised self-organized counterpropagation (Nie *et al.*, 1996). At the same time, both the MRWLS-SVM and the supervised self-organized counterpropagation network have almost the same performance.

5.3. Proposed adaptive control scheme of the pH neutralization process.
During control application, the following parameters are considered:

(i) The desired reference trajectory $R(t)$ was given by the step response of a first order system with gain = 1, and a time constant is 5 samples.

(ii) The acid inlet flow rate F_a is fixed at its nominal value given in Table 1.

(iii) The maximum output horizon is set to $Np = 2$.

(iv) The controller gains are set to small values (0.25).

Three control simulation tasks are performed to investigate the performance of the developed adaptive control scheme. These are as follows:

- Test the controller performance with constant setpoint, i.e., the input set point $(S(t))$ is set to 8.

- Test the controller performance with the set-point changes, i.e., the set-pint changes are set to

$$S(t) = \begin{cases} 8 & \text{if} & t < 50, \\ 9 & \text{if} & 50 < t < 100, \\ 10 & \text{if} & 100 < t < 150, \\ 9.5 & \text{if} & 150 < t < 200, \\ 8 & \text{if} & t > 200. \end{cases} \qquad (48)$$

- Test the controller performance with the pH neuterilization process disturbed by noise. In this task, the output of the system is corrupted by a random noise with unity variance and zero mean and the set point is set as in the first task.

For the first simulation task, the changes in the base flow rate F_b and the response of the pH value during the control action are shown in Figs. 8 and 9, respectively. Moreover, the adaptation of the controller gains is depicted in Fig. 10. The obtained response of the proposed control scheme provides good tracking and very reasonable control moves. This results from the effectiveness of the adaptive control strategy and the modeling efficiency of the MRWLS-SVM network.

When the second control simulation task (i.e., the set-point changes) is performed, the response for both the base flow rate F_b and the pH value are obtained as in Figs. 11

Table 2. Comparison between the MRWLS-SVM model of the pH neutralization process and other algorithms.

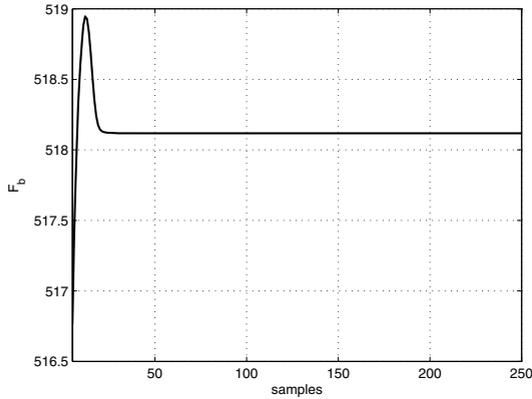| Algorithm	MSE
MRWLS-SVM network	0.042
RBF kernel based LS-SVM network	0.2078
Unsupervised self-organized counter-propagation network (Nie *et al.*, 1996)	0.230
Supervised self-organized counter-propagation network (Nie *et al.*, 1996)	0.012

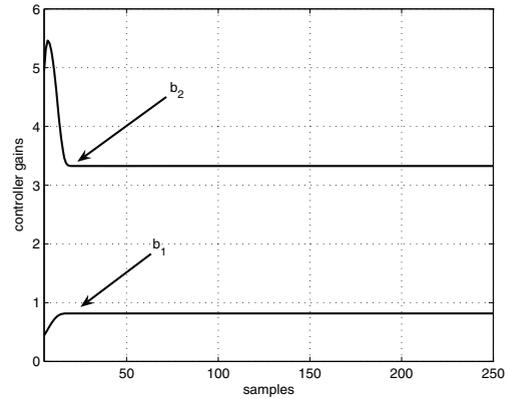Fig. 8. Change of the base flow rate during the constant set-point task.

Fig. 10. Adaptation of the controller gains during the constant set-point task.
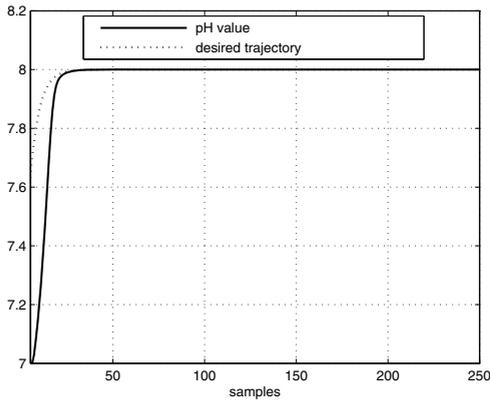
Fig. 9. Response of the pH process using the proposed control scheme during the constant set-point task.

and 12, respectively. The adaptation of the controller gains for this task is depicted in Fig. 13. It can be observed that the controller algorithm can track the set-point rapidly, stably and accurately with a very small steady state error. However, the obtained control action is not smooth, especially when the set-point decreases from 9.5 to 8. This can be controlled by carrying out the simulation task with other small initial controller gains, e.g., set to 0.05. Figures 14 and 15 depict the achieved results with these initial gain parameters. It is observed that the controller action is smoother and the output response can still track the set-point rapidly, stably and accurately with a very small stea-

dy state error.

During the third control simulation task (i.e., the pH process is disturbed by noise), the response of the system output (i.e., the pH value) and the base flow rate F_b are depicted in Figs. 16 and 17, respectively. The obtained result shows that the proposed controller can still track the set-point with very small steady state error and acceptable smooth control action even when the system is disturbed by noise.

Finally, for comparison, the three control tasks mentioned above were carried out with the same parameters but the proposed control scheme was constructed using the RBF kernel based LS-SVM network. The two schemes were compared in terms of the Root Mean Squared Error (RMSE) defined as

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{t=1}^{N} (S(t) - \text{pH}(t))^2}, \qquad (49)$$

where $S(t)$ and $\text{pH}(t)$ are the desired and pH process outputs, respectively.

The obtained errors are listed in Table 3. Here it can be observed that the proposed control scheme based on the MRWLS-SVM network can yield control performance superior to the control scheme based on the LS-SVM using the RBF kernel. This is inherited from the generalization and the modeling accuracy of the MRWLS-SVM model for the pH process.

Table 3. Comparison between the proposed control algorithm based on both the MRWLS-SVM and LS-SVM networks.

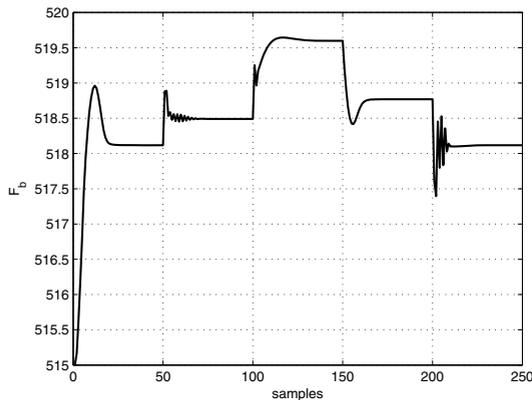| Algorithm	Task 1	Task 2	Task 3
Proposed control scheme based on the MRWLS-SVM network	0.1339	0.1433	0.1308
Proposed control scheme based on the LS-SVM network	0.1933	0.2172	0.1910

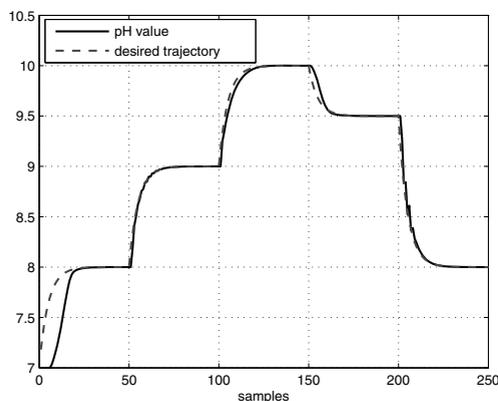Fig. 11. Change of the base flow rate with the set-point changes.

Fig. 12. Response of the pH process using the proposed control scheme with the set-point changes.
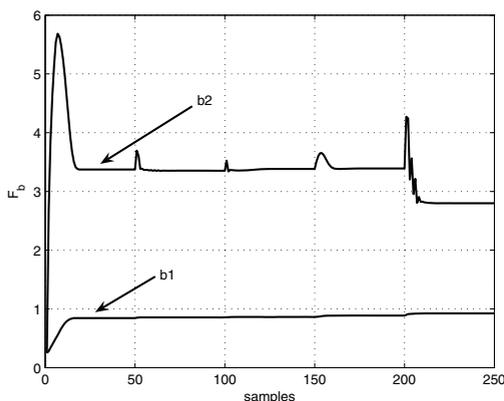

Fig. 13. Adaptation of the controller gains with the set-point changes.

6. Conclusion

Based on the generalization performance of the LS-SVM network, an adaptive control scheme for nonlinear systems was developed. Basically, the proposed scheme used a modified structure of LS-SVM, named the MRWLS-SVM network, as a model for the controlled process. Since the MRWLS-SVM network is a type of dynamic neural networks, it can be used as the model predictor in the proposed control scheme. In the sense of the prediction error, the control scheme uses the gradient descent with an adaptable learning rate to adapt the controller parameters. The adaptation of the learning rule is deduced using the Lyapunov stability approach. The control simulation results via controlling the pH neutralization process showed that the stable adaptive control scheme based on the MRWLS-SVM network can be employed effectively to control this strongly nonlinear system. Two issues have to be discussed in the future based on this study. The first one is the control performance in the presence of variations in system parameter time. The second is testing how the control algorithm can control MIMO systems.

References

Bezdek, J. (1981). *Pattern Recognition with Fuzzy Objective Function-Algorithms*, Plenum Press, New York, NY.

Faanes, A. and Skogestad, S. (2004). pH-neutralization: Integrated process and control design, *Computers and Chemical Engineering* **28**(8): 1475—1487.

Ge, S.S. and Wang, C. (2004). Adaptive neural control of uncertain MIMO nonlinear systems, *IEEE Transactions on Neural Networks* **15**(3): 674—692.

Ge, S.S., Yang, C. and Lee, T.H. (2008). Adaptive predictive control using neural network for a class of pure-feedback systems in discrete time, *IEEE Transactions on Neural Networks* **19**(9): 1599—1614.

Henson, M.A. and Seborg, D.E. (1994). Adaptive nonlinear control of a pH neutralization process, *IEEE Transactions on Control System Technology* **2**(3): 169–182.

Huicheng, W.L.L. and Taiyi, Z. (2008). An improved algorithm on least squares support vector machines, *Information Technology Journal* **7**(2): 370–373.

Ku, C.-C., and Lee, K.Y. (1995). Diagonal recurrent neural networks for dynamic systems control, *IEEE Transactions on Neural Networks* **6**(1): 144–156.

Li-Juan, L., Hong-Y, S. and Jian, C. (2007). Generalized predictive control with online least squares support machines, *Acta Automatica Sinica* **33**(11): pp. 1182–1188.
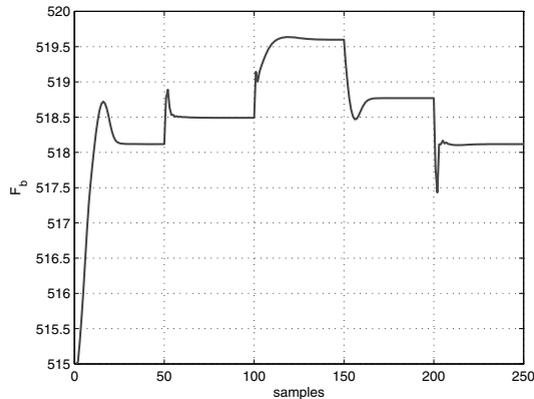
Fig. 14. Change of the base flow rate with the set-point changes (the initial controller gains are set to 0.05).
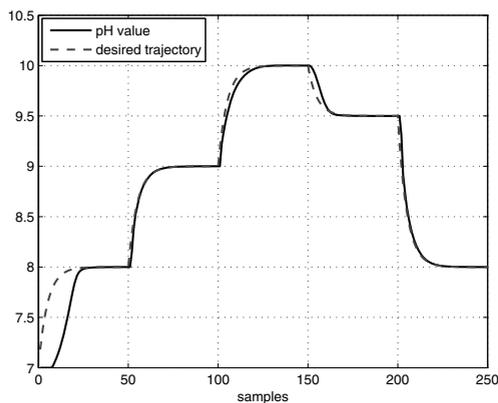

Fig. 15. Response of the pH process using the proposed control scheme with the set-point changes (the initial controller gains are set to 0.05).
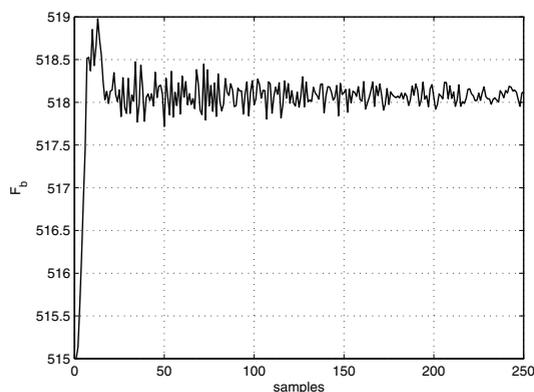

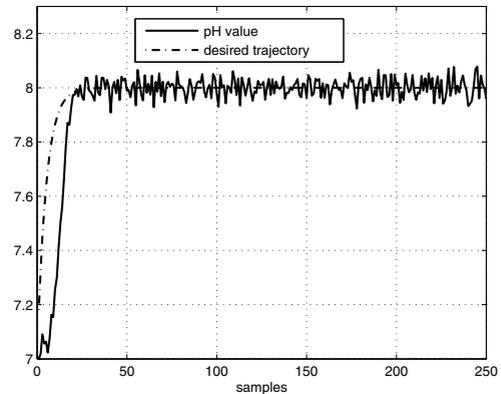Fig. 16. Change of the base flow rate with the pH process disturbed by noise.

Fig. 17. Response of the pH process using the proposed control scheme with the pH process disturbed by noise.

deling, *Proceedings of the 15th International Conference in Automation and Robotics MMAR2010, Międzyzdroje, Poland*, pp. 405–410.

Nie, J., Loh, A. and Hang, C. (1996). Modeling pH neutralization processes using fuzzy-neural approaches, *Fuzzy Sets and Systems* **78**(1): 5–22.

Parisini, T. and Zoppoli, R. (1994). Neural networks for feedback feedforward nonlinear control systems, *IEEE Transactions on Neural Networks* **5**(3): 437–449.

Robert, M.S. and Jean-Jacques, E.S. (1992). Gaussian networks for direct adaptive control, *IEEE Transactions on Neural Networks* **3**(6): 837–863.

Saunders, C., Gammerman, A. and Vovk, V. (1998). Ridge regression learning algorithm in dual variables, *Proceedings of the 15th International Conference on Machine Learning ICML-98, Madison, WI, USA*, pp. 515–521.

Suykens, J.A.K. (2001). Nonlinear modeling and support vector machines, *IEEE Instrumentation and Measurement Technology Conference, Budapest, Hungary*, pp. 287–294.

Suykens, J.A.K., Van Gestel, T., De Brabanter, J., De Moor, B. and Vandewalle, J. (2002). *Least Squares Support Vector Machines*, World Scientific, Singapore.

Suykens, J.A.K. and Vandewalle, J. (1999). Least squares support vector machine classifiers, *Neural Processing Letters* **9**(3): 293–300.

Suykens, J.A.K., Vandewalle, J. and Moor, B.D. (2001). Optimal control by least squares support vector machines, *Neural Networks* **14**: 23–35.

Vapnik, V. (1998). *Statistical Learning Theory*, John Wiley, New York, NY.

Wang, Y.-N. and Yuan, X.-F. (2008). SVM approximate-based internal model control strategy, *Acta Automatica Sinica* **34**(2): 696–702.

Wang, Y., Rong, G. and Wang, S. (2002). Hybrid fuzzy modeling of chemical processes, *Fuzzy Sets and Systems* **130** (2–1): 265–275.

Li, X., Yi Cao, G. and Jian Zhu, X. (2006). Modeling and control of PEMFC based on least squares support vector machines, *Energy Conversion and Management* **47**(7–8): 1032–1050.

Mahmoud, T.A. (2010). Multi resolution wavelet least squares support vector machine network for nonlinear system mo-

Yoo, S.J., Park, J.B., and Choi, Y.H. (2005). Stable predictive control of chaotic systems using self-recurrent wavelet neural networks, *International Journal of Control, Automation, and Systems* **3**(1): 43–55.

Zhang, L., Zhou, W. and Jiao, L. (2004). Wavelet support vector machine, *IEEE Transactions on Systems Man, and Cybernetics* **34**(1): 34–39.

Zhang, R. and Wang, S. (2008). Support vector machine based predictive functional control design for output temperature of coking furnace, *Journal of Process Control* **18**(5): 439–448.

Zhang, X.-G., Gao, D., Zhang, X.-G. and Ren, S.S. (2005). Robust wavelet support vector machine for regression estimation, *International Journal of Information Technology* **11**(9): 35–45.

Tarek A. Mahmoud received the M.Sc. and Ph.D. degrees in automatic control engineering from the Faculty of Electronic Engineering, Menoufia University, Menouf, Egypt, in 2001 and 2007, respectively. He is currently a lecturer at the Industrial Electronics and Control Engineering Department of the same university. His main research interest is in the development of intelligent control methodologies (neural networks, fuzzy systems, evolutionary computations).

Appendix A

Observe that

$$
\frac{\partial Y_N(t+1)}{\partial u(t)} = \left[\frac{\partial y_N(t+1)}{\partial u(t)}, \ldots, \frac{\partial y_N(t+N_p)}{\partial u(t)} \right]^T,
$$
(50)

where

$$
\frac{\partial y_N(t+k)}{\partial u(t)}, \quad k = 1, 2, \ldots, N_P
$$
(51)

can be obtained as follows: At the k-th prediction step, the input of the MRWLS-SVM network predictor is given by

$$
\begin{aligned}
x(t+k) \\
= \big[u(t), \ldots, u(t-s), y_N(t+k-1), \\
\ldots, y_N(t+k-1) \big]^T,
\end{aligned}
$$
(52)

where s is the number of the past values of the input variable, r is the number of the past values of the network output, and t is the sampling instant.

According to (15), the output of the MRWLS-SVM network predictor can be obtained by

$$
y_N(t+k) = W^T Y_w(x(t+k)),
$$
(53)

where Y_w and W are the vectors that contain the outputs of the sub WLS-SVM networks and their associated weights, respectively. Those are

$$
\begin{aligned}
Y_w(x(t+k)) &= \left[y_w^1(x(t+k)), \ldots, y_w^d(x(t+k)) \right]^T \\
W &= [w_1, w_2, \ldots, w_d]^T.
\end{aligned}
$$
(54)

Accordingly,

$$
\frac{\partial y_N(t+k)}{\partial u(t)} = W^T \left[\frac{\partial Y_w(x(t+k))}{\partial u(t)} \right],
$$
(55)

where

$$
\begin{aligned}
&\frac{\partial Y_w(x(t+k))}{\partial u(t)} \\
&= \left[\frac{\partial y_w^1(x(t+k))}{\partial u(t)}, \ldots, \frac{\partial y_w^d(x(t+k))}{\partial u(t)} \right]^T \\
&= \left[\sum_{i=1}^{c} \alpha_{i1} \frac{\partial K_1(x(t+k), \breve{x}_i)}{\partial u(t)}, \ldots, \right. \\
&\quad \left. \sum_{i=1}^{c} \alpha_{id} \frac{\partial K_d(x(t+k), \breve{x}_i)}{\partial u(t)} \right]^T,
\end{aligned}
$$
(56)

with K_l being the wavelet kernel function of the l-th WLS-SVM network, x_j the j-th element of the input vector $x(t+k)$, and $\breve{x}_{ij}$ the j-th element of the i-th support vector. Define

$$
\varsigma_{ij}^l = \frac{x_j - \breve{x}_{ij}}{a_l},
$$
(57)

where a_l is the dilation parameter of the l-th WLS-SVM network.

Accordingly, the output of each kernel node for the l-th WLS-SVM network is given by

$$
K_l(x, \breve{x}_i) = \prod_{j=1}^{m} h(\varsigma_{ij}^l),
$$
(58)

where m is the dimension of the input vector of the MRWLS-SVM network. Then

$$
\frac{\partial K_l(x(t+k), \breve{x}_i)}{\partial u(t)} = \frac{1}{a_l} \frac{\partial h(\varsigma_{1j}^l)}{\partial \varsigma_{1j}^l} \prod_{j=2}^{m} h(\varsigma_{ij}^l).
$$
(59)