Design and analysis of neural/fuzzy variable structural PID control systems

C.-L. Chen
F.-Y. Chang

Indexing terms: Neural/fuzzy systems, Variable structural controller, PID, pH control

Abstract: The paper describes the design method of a neural/fuzzy variable structural proportional-integral-derivative (neural/fuzzy VSPID) control system. The neural/fuzzy VSPID controller has a structure similar to that of the conventional PID. In this controller, the PD mode is used in the case of large errors to speed up response, whereas the PI mode is applied for small error conditions to eliminate the steady-state offset. A sigmoidal-like neuron is employed as a preassigned algorithm of the law of structural change. Meanwhile, the controller parameters would be changed according to local conditions. Bounded neural networks or bounded fuzzy logic systems are used for constructing the nonlinear relationship between the PID controller parameters and local operating control conditions. Flexible changes of controller modes and resilient controller parameters of the neural/fuzzy VSPID during the transient could thereby solve the typical conflict in nature between steady-state error and dynamic responsiveness. A neutralisation process is used to demonstrate the applicability of such a controller for controlling highly nonlinear processes.

1 Introduction

The three-mode proportional-integral-derivative (PID) controller is widely used in chemical plants due to ease of use and robustness in the face of plant uncertainties. Nevertheless, the linear PID algorithm might be difficult to deal with processes with complex dynamics, such as those with large dead time, inverse response and highly nonlinear characteristics. To date, many sophisticated algorithms have been used to help the PID controller work under such difficult conditions. The various nonlinear PID controllers in which the simple controller structure has been reserved and superior performance has been achieved by allowing controller parameters to vary with local control conditions, such as three-piece PIDs [1] and nonlinear PI(D)s [2, 3], seem to be acceptable. Meanwhile, neural/fuzzy inferences based on self-tuning schemes of PI controllers have also been proposed to improve the control performance (for example, see [4, 5]) by using the neural/fuzzy capabilities to store the domain expert knowledge and to infer control decisions. On the other hand, improving the limited performance of PI controllers, such as the conflict in nature between static accuracy (steady-state error) and dynamic responsiveness (speed of response), the variable structural PID (VSPID), i.e. using the PD action to accelerate the speed of the response and using the PI mode to eliminate the steady-state offset, could be used to overcome the difficulties. The VSPID controller has a structure which is flexibly changed by a preassigned algorithm of the law of structural change. Its superior performance, with the aid of neural/fuzzy systems, additionally achieved by allowing controller parameters to vary with local control conditions, is proposed in this paper.

Radial basis function networks (RBFNs) and back-propagation neural networks (BPNNs) have yielded useful results in many practical areas such as pattern recognition [6], system identification [7, 8] and control [9], due primarily to their simple structures for realisation and well established training algorithms. Many fuzzy paradigms, meanwhile, have been studied in recent years by viewing a fuzzy logic system (FLS) as a functionally equivalent RBFN or BPNN [10, 11]. As indicated in [11], the most important advantage of such an FLS spanned by fuzzy basis functions is the provision of a natural framework for combining numerical values and linguistic symbols in a uniform way. From a mathematical point of view, the input–output expressions of those mappings are identical in spite of the distinct inference procedure. Capability discrimination between neural and fuzzy systems is thus diminished for proofs of universal neural/fuzzy approximators [11, 12]. Using neural networks or fuzzy systems to approximate a given plant or to control a process now depends on whether rich available data are at hand or whether the 'If-Then' control heuristics could be established by human experts familiar with system dynamics under consideration. The RBFN, the BPNN and the FLS are used interchangeably in this paper since they could provide equivalent functionality.

The design methodology of a neural/fuzzy variable structural PID controller (neural/fuzzy VSPID) for nonlinear processes is proposed in this paper. The neural/fuzzy systems discussed above are exploited to provide the nonlinear VSPID parameters according to local control conditions. A simple sigmoidal-like neuron is employed as a preassigned algorithm of the law of structural change which is directed by the current value of the error signal. High-quality control could be

IEE Proceedings online no. 19960261

Paper first received 9th May 1995 and in revised form 20th December 1995
The authors are with the Department of Chemical Engineering, National Taiwan University, Taipei 10617, Taiwan, Republic of China

assured if parameters of the neural/fuzzy VSPID are suitably determined and the law of structural change could be deduced properly. The PI mode with antireset windup is designed for the prevention of excessive overshoot caused by direct implementation of the integral action. The amount of maximal saturation of the integral action is studied by using the squeezing technique approach. The stability analysis of the PI/PD part is also discussed.

2 Topology of the L_∞ neural networks and fuzzy logic systems

2.1 The L_∞ neural networks (L_∞ NN)

A great number of multilayered feedforward neural networks (MLNNs) have been widely discussed in recent literature [13, 14]. An MLNN includes an input layer, an output layer and a number of hidden layers. Each input layer is composed of input nodes. Each hidden layer (and output layer) consists of processing units (the so-called hidden nodes or basis functions), and each unit is the composite of an activation function (AF) and a transfer function (TF).

The input signals to each processing unit are first transformed into an activation level via the activation function. The activation level is further mapped into a crisp output by the transfer function. The aim of those mappings is to store the input–output relationship in parameters of the AF/TF via learning. With various selections of activation functions and transfer functions, a variety of MLNNs could be synthesised.

Without loss of generality, the activation function $\nu(\mathbf{x};\mathbf{p})$ could be generally expressed in quadratic form:

$$\nu(\mathbf{x};\mathbf{p}) \triangleq (\mathbf{x}-\mathbf{c})^T\Lambda(\mathbf{x}-\mathbf{c}) + \lambda^T\mathbf{x} + b \qquad (1)$$

where ^{T} denotes the vector/matrix transpose, $\mathbf{x} \triangleq [x_1, ..., x_I]^T \in R^I$ is the input vector, $\mathbf{c} \triangleq [c_1, ..., c_I]^T \in R^I$ is called the centre, Λ is an $I \times I$ semi-positive-definite matrix; $\lambda \triangleq [\lambda_1, ..., \lambda_I]^T \in R^I$ and $\mathbf{p}$ = overall parameters of an AF. Some popularly used AFs are special conditions of such a quadratic form. For example [13–15],

$$\nu(\mathbf{x};\mathbf{p}) = \begin{cases} \sum_{i=1}^{I} \lambda_i x_i + b & \text{if } \Lambda = 0 \text{ (linear)} \\ \sum_{i=1}^{I} \left(\frac{x_i - c_i}{s_i}\right)^2 + b & \text{if } \lambda = 0,\ \Lambda = \mathrm{diag}\left[\frac{1}{s_i^2}\right] \text{ (ellipsoidal)} \\ \frac{\sum_{i=1}^{I}(x_i - c_i)^2}{s^2} & \text{if } \lambda = 0,\ b = 0,\ \Lambda = \mathrm{diag}\left[\frac{1}{s^2}\right] \text{ (spherical)} \end{cases} \qquad (2)$$

As to transfer function $\phi(\cdot)$, all TFs mentioned in the literature are analytical almost everywhere [16–18]. For instance,

$$\phi(\nu) = \begin{cases} \nu & \text{(linear)} \\ \frac{1}{1+\exp(-\nu)} & \text{(sigmoidal)} \\ \exp(\jmath\nu),\ \jmath \triangleq \sqrt{-1} & \text{(Fourier)} \\ \exp(-\nu) & \text{(inverse exponential)} \end{cases} \qquad (3)$$

where $\exp(j\nu)$ might include $\cos(\nu)$, $\sin(\nu)$ and their linear combinations. Combining any of the AFs and any of the various possible TFs, a processing unit with various characteristics could be synthesised. The typical processing unit of a BPNN, for instance, is the combination of a linear AF and a sigmoidal TF, and the typical processing unit of an RBFN is the combination of a spherical AF and any analytical TF.

Each layer of a multi-input/single-output (MISO) MLNN is composed of various processing units with a suitable choice of AFs and TFs. Without loss of generality, the three-layered MISO MLNN with I inputs and J hidden processing units would be used in the following, and both the AF and the TF in the output layer are assumed to be linear throughout this work. Each processing unit combining an AF and a TF in the hidden layer is called the basis function $\phi_j \triangleq \phi(\nu(\mathbf{x};\mathbf{p}_j))$, where $\mathbf{p}_j$ denotes overall parameters of the jth processing unit in the hidden layer. Such a neural network is mathematically equivalent to a finite-dimensional function space spanned by neural basis functions:

$$\hat{f}(\mathbf{x};\mathbf{P}) \triangleq \sum_{j=1}^{J} w_j \phi(\nu(\mathbf{x};\mathbf{p}_j)) \qquad (4)$$

where $\mathbf{P}$ denotes overall network parameters.

The bounded input-bounded output (L_∞) RBFNs and BPNNs, i.e. L_∞ NNs, would be used in this paper. Without loss of generality, the basis function ϕ_j for an L_∞ NN is assumed to be normalised for all j, i.e. $\|\phi_j\|_\infty = 1$ for all j. The relationship between the norm of the weight vector and the range space of an L_∞ NN could be derived from the Minkowski inequality and functional analysis. Let $\{\phi_1, \phi_2, ..., \phi_J\}$ be a basis for an L_∞ NN. Then for every choice of weights $w_1, w_2, ..., w_J$ one has

$$a\sum_{j=1}^{J}|w_j| \le \|\hat{f}(\mathbf{x};\mathbf{P})\|_\infty \le \sum_{j=1}^{J}|w_j| \qquad (5)$$

where $0 < a \le 1$. The inequality, in fact, could be viewed as the definition of an L_∞ NN.

2.2 The fuzzy logic systems (FLS)

A typical FLS is composed of four principal components [11]: (1) a fuzzifier; (2) a rule base; (3) an inference engine; (4) a defuzzifier. The fuzzifier deals with mapping scaled input variables and transforming the mapping into appropriate linguistic values. The rule base comprises the well established knowledge of the application domain. For an inference engine, it emulates human decision-making logic. Such operations are performed by employing fuzzy implication and fuzzy logic inference. As the final stage of the FLS, the defuzzifier generates a single crisp output from the inferred fuzzy action.

Without loss of generality, an FLS with the singleton fuzzifier, Gaussian membership functions, the product inference rule, the centre average defuzzifier and the equal area of membership functions for all output linguistic terms would be used in this paper. The output of such an FLS with J inference rules is [11]

$$\hat{f}(\mathbf{x};\mathbf{P}) = \frac{\sum_{j=1}^{J} w_j \left[\prod_{i=1}^{I} \exp\left(-\left(\frac{x_i - c_{ji}}{s_{ji}}\right)^2\right)\right]}{\sum_{j=1}^{J} \left[\prod_{i=1}^{I} \exp\left(-\left(\frac{x_i - c_{ji}}{s_{ji}}\right)^2\right)\right]}$$

$$= \frac{\sum_{j=1}^{J} w_j \phi(\nu(\mathbf{x}; \mathbf{p}_j))}{\sum_{j=1}^{J} \phi(\nu(\mathbf{x}; \mathbf{p}_j))} \quad (6)$$

where w_j is the corresponding crisp output of $\phi(\nu(\mathbf{x}; \mathbf{p}_j))$ with unit membership function value. **P** is the set of all adjustable parameters in the FLS. The FLS could be viewed as a modified Gaussian potential function network (modified GPFN [13]). Each rule in the FLS is functionally equivalent to one hidden node of a modified GPFN. The w_js here refer to the weights of the FLS in accordance with neural network jargon. The discussed FLSs are notably input–output bounded, and have the relationships with output weights as demonstrated in eqn. 5.

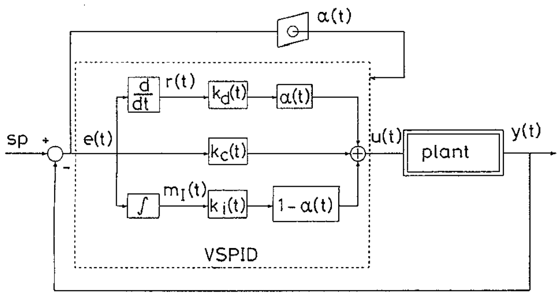

Fig. 1 *VSPID control system*

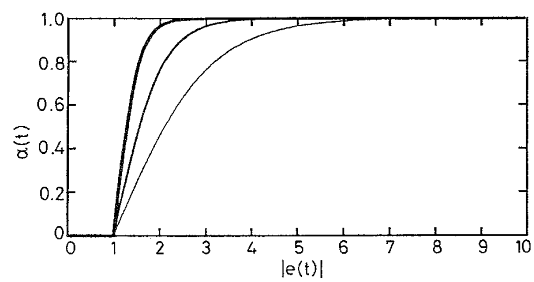

Fig. 2 *Characteristics of $\alpha(t)$ with $\varepsilon = 1.0$ and various η values*

$\eta = 0.5$
$\eta = 1.0$
$\eta = 2.0$

3 Neural/fuzzy variable structural PID controller (neural/fuzzy VSPID)

3.1 The variable structural PID controller (VSPID)

The system has notably no steady-state error in response to a step input disturbance provided that the control law employs the integral mode. Undesirable overshoot, increasing sharply as a function of the gain, will occur for sufficiently large integral action. Sharply increasing the controller gain could significantly accelerate the system response in the absence of the integral mode, but a steady-state offset would be displayed in such a case; thus, reaching a compromise is vital. A nonlinear variable structural PID controller could be used for such a correcting effort, as is depicted in Fig. 1:

$$\begin{aligned} u(t) &= \bar{u} + k_c(t)e(t) + \alpha k_d(t)r(t) + (1-\alpha)k_i(t)m_I(t) \\ &= \bar{u} + \alpha[k_c(t)e(t) + k_d(t)r(t)] \\ &\quad + (1-\alpha)[k_c(t)e(t) + k_i(t)m_I(t)] \end{aligned} \quad (7)$$

where $\bar{u}$ is a constant, $r(t) \triangleq de(t)/dt$, $m_I(t) \triangleq \int_0^t e(\tau)d\tau$ and $\alpha \in [0, 1]$. The controller would turn out to be either a PD or a PI controller if α is either 1 or 0. Instead of a drastic change of α value, α could be reasonably defined as

$$\alpha(t) = \tanh(\eta\beta(t)) \quad (8)$$

where

$$\beta(t) = \begin{cases} |e(t)| - \epsilon & \text{if } |e(t)| \geq \epsilon > 0 \\ 0 & \text{if } |e(t)| \leq \epsilon \end{cases} \quad (9)$$

α is an increasing function of $|e(t)|$, and converges to either 1 or 0 if $|e(t)|$ approaches infinity or $|e(t)|$ enters the tube $0 \leq |e(t)| \leq \varepsilon$ such as shown in Fig. 2, i.e.

$$\alpha(t) = \begin{cases} 1 & \text{if } |e(t)| \gg \epsilon \\ 0 & \text{if } |e(t)| \leq \epsilon \end{cases} \quad (10)$$

The η value in eqn. 8 determines how quickly α changes between zero and one. For reasonable η values, the VSPID would behave from the PD controller in the case of large error to the PID case and the PI case, i.e.

$$u(t) = \bar{u} + \begin{cases} k_c(t)e(t) + k_d(t)r(t) & \text{for } |e(t)| \gg \epsilon \\ k_c(t)e(t) + \alpha k_d(t)r(t) & \\ \quad +(1-\alpha)k_i(t)m_I(t) & \text{for } |e(t)| \approx \epsilon \\ k_c(t)e(t) + k_i(t)m_I(t) & \text{for } |e(t)| \leq \epsilon \end{cases} \quad (11)$$

An infinite η value would lead the VSPID to be either a PD or a PI controller according to the magnitude of $|e(t)|$. The time at which the controller structure change occurs is thereby determined by a flexible program which is directed by the current value of the error signal.

In order to put into operation the proposed VSPID controller in the practical process, stability analysis in the PD control mode and training procedures of the proposed controller parameters are discussed in the discrete time domain. The parameters of the proposed VSPID controller could be updated for every control interval T. A zero-order holder is used to keep a constant controller output during each interval. The notations therein are defined as follows:

$$\begin{aligned} e(n) &= \{sp(t) - y(t)\}|_{t=nT} \\ r(n) &= \frac{e(n) - e(n-1)}{T} \\ m_I(n) &= T\sum_{k=1}^{n} e(k) \\ u(n) &= \bar{u} + k_c(n)e(n) + \alpha(n)r(n) + (1-\alpha(n))m_I(n) \end{aligned} \quad (12)$$

$sp(t)$ and $y(t)$ are reference and process output; $e(n)$, $r(n)$, $m_I(n)$ and $u(n)$ are error, rate of change in error, integral of error and controller output, respectively, at the nth sampling point.

3.2 L_∞ NN/FLS based variable structural PID controllers (L_∞ NN_VSPID, FL_VSPID)

The neural/fuzzy VSPID has a structure similar to that of the conventional nonlinear VSPID controller, but its parameters are changed according to local conditions (see Fig. 3). Here, L_∞ NNs or the previously mentioned fuzzy systems are used for constructing a nonlinear relationship between controller parameters and local control conditions. Parameters of the L_∞ NN_VSPID or the FL_VSPID are defined as

$$\begin{aligned} k_c(t) &= \hat{f}(\mathbf{x};\mathbf{P}_1) \\ k_i(t) &= \hat{f}(\mathbf{x};\mathbf{P}_2) \\ k_d(t) &= \hat{f}(\mathbf{x};\mathbf{P}_3) \end{aligned} \tag{13}$$

where

$$\hat{f}(\mathbf{x};\mathbf{P}_m) = \begin{cases} \sum_{i\in\mathbf{I}_m}\sum_{j\in\mathbf{J}_m} w_{ij}\phi(\nu(\mathbf{x}(t);\mathbf{p}_{ij})) & \text{for the } L_\infty \text{ NN\_VSPID} \\ \dfrac{\sum_{i\in\mathbf{I}_m}\sum_{j\in\mathbf{J}_m} w_{ij}\phi(\nu(\mathbf{x}(t);\mathbf{p}_{ij}))}{\sum_{i\in\mathbf{I}_m}\sum_{j\in\mathbf{J}_m} \phi(\nu(\mathbf{x}(t);\mathbf{p}_{ij}))} & \text{for the FL\_VSPID, } m = 1,2,3 \end{cases} \tag{14}$$

Hence, $k_c(t)$, $k_i(t)$ and $k_d(t)$ are outputs of three three-layered L_∞ NNs or three FLSs with two inputs $\mathbf{x}(t) = [e(t), r(t)]^T$. $\mathbf{p}_{ij}$ denotes overall parameters in the ijth hidden processing unit or the ijth rule.

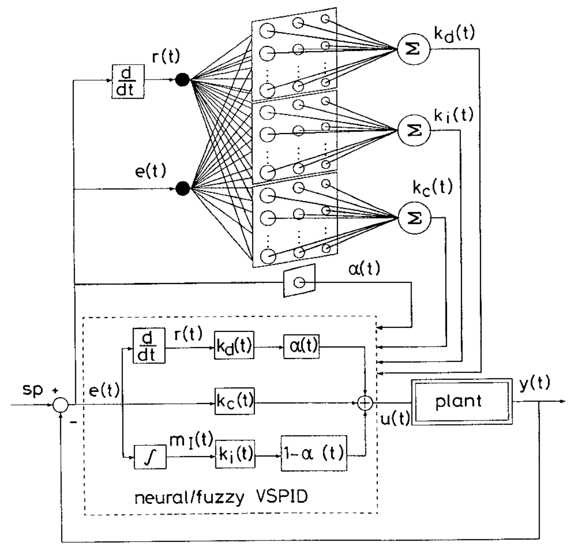

Fig.3 *Neural/fuzzy VSPID control system*

For the L_∞ NN_VSPID/FL_VSPID, $\nu(\mathbf{x}(t); \mathbf{p}_{ij})$ is the activation level of the ijth hidden processing unit

$$\nu(\mathbf{x}(t);\mathbf{p}_{ij}) = \begin{cases} \left(\dfrac{e(t)-c_{e_i}}{s_{e_i}}\right)^2 + \left(\dfrac{r(t)-c_{r_j}}{s_{r_j}}\right)^2 & \text{for the } L_\infty \text{ RBFN/FLS} \\ \lambda_{ij}^e e(t) + \lambda_{ij}^r r(t) + b_{ij} & \text{for the BPNN} \end{cases} \tag{15}$$

ϕ is the analytical function for the L_∞ RBFN, and ϕ is the sigmoidal function for the BPNN. As for the FLS, ϕ is the discussed fuzzy basis function. $\mathbf{p}_{ij} = [c_{ei}, c_{rj}, s_{ei}, s_{rj}]^T$ for the L_∞ RBFN/FLS, and $\mathbf{p}_{ij} = [\lambda_{ij}^e, \lambda_{ij}^r, b_{ij}]^T$ for the BPNN. λ_{ij}^e or λ_{ij}^r refers to the weight in the hidden layer that connects the ijth hidden node to the input node of $e(t)$ or $r(t)$. Subscript e_i denotes the ith node in the e direction, and subscript r_j denotes the jth node in the r direction for $i \in \mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3$ and $j \in \mathbf{J}_1, \mathbf{J}_2, \mathbf{J}_3$, where $\mathbf{I}_1 \triangleq \{1, ..., l_1\}$, $\mathbf{J}_1 \triangleq \{1, ..., l_2\}$, $\mathbf{I}_2 \triangleq \{1, ..., l_3\}$, $\mathbf{J}_2 \triangleq \{1, ..., l_4\}$, $\mathbf{I}_3 \triangleq \{1, ..., l_5\}$ and $\mathbf{J}_3 \triangleq \{1, ..., l_6\}$. The l_is refer to the number of AFs in either the e (i = 1, 3, 5) direction or the r (i = 2, 4, 6) direction. Tuning parameters of the specific L_∞ NNs/FLSs in $k_c(t)$, $k_i(t)$ and $k_d(t)$ might encompass centres $[c_{ei}, c_{rj}]^T$, widths of receptive field $[s_{ei}, s_{rj}]^T$, connective weights in the hidden layer $[\lambda_{ij}^e]$ and $[\lambda_{ij}^r]$, bias terms $[b_{ij}]$ and the output connective weights $[w_{ij}]$ for $i \in \mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3$ and $j \in \mathbf{J}_1, \mathbf{J}_2, \mathbf{J}_3$.

Outputs of controller parameters and output connective weights of associated NNs/FLSs have the following relationships:

$$\begin{aligned} a_1 \sum_{i\in\mathbf{I}_1}\sum_{j\in\mathbf{J}_1} |w_{ij}| &\le \|k_c(t)\|_\infty \le \sum_{i\in\mathbf{I}_1}\sum_{j\in\mathbf{J}_1} |w_{ij}| \\ a_2 \sum_{i\in\mathbf{I}_2}\sum_{j\in\mathbf{J}_2} |w_{ij}| &\le \|k_i(t)\|_\infty \le \sum_{i\in\mathbf{I}_2}\sum_{j\in\mathbf{J}_2} |w_{ij}| \\ a_3 \sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3} |w_{ij}| &\le \|k_d(t)\|_\infty \le \sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3} |w_{ij}| \end{aligned} \tag{16}$$

where $0 < a_1, a_2, a_3 \le 1$. Those relationships imply that parameters of the L_∞ NN_VSPID/FL_VSPID controller are bounded.

An L_∞ RBFN_VSPID or a FL_VSPID is functionally equivalent to a linear VSPID if all receptive field widths approach infinity in neural/fuzzy basis functions:

Property 1 (L_∞ RBFN_VSPID and FL_VSPID are functionally equivalent to linear VSPID): An L_∞ RBFN_VSPID or an FL_VSPID controller is functionally equivalent to a linear VSPID controller if $s_{ei}, s_{rj} \to \infty$ $\forall$ $i \in \mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3$, and $\forall$ $j \in \mathbf{J}_1, \mathbf{J}_2, \mathbf{J}_3$.

Proof: Each basis function in an L_∞ RBFN or an FLS becomes a constant function, providing that all receptive field widths in each basis function approach infinity. Hence, an L_∞ RBFN_VSPID or an FL_VSPID is functionally equivalent to a linear VSPID controller if $s_{ei}, s_{rj} \to \infty$ $\forall$ $i \in \mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3$ and $\forall$ $j \in \mathbf{J}_1, \mathbf{J}_2, \mathbf{J}_3$.

Property 1 gives the way to set L_∞ RBFN_VSPID/FL_VSPID parameters initially by letting the receptive field widths in the neural/fuzzy basis function be large enough so that the initial performance of the L_∞ RBFN_VSPID/FL_VSPID could be like the performance provided by the linear VSPID.

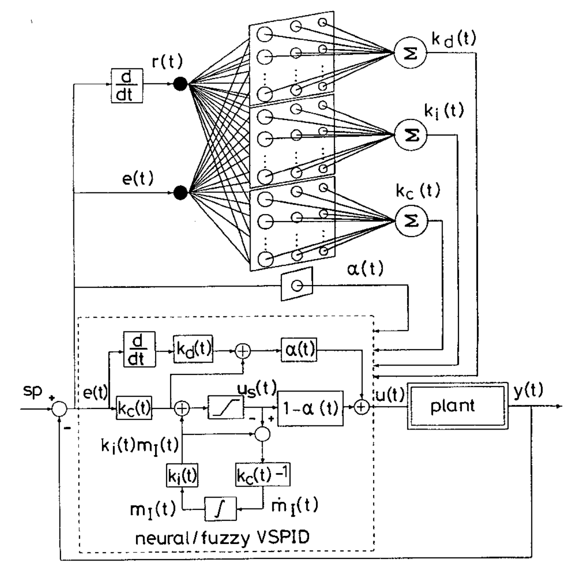

Fig.4 *Modified neural/fuzzy VSPID control system*

4 Nonlinear PI controller with antireset windup

Integral or reset action is usually taken to eliminate the steady-state error in feedback controllers. One of the penalties that must be paid for this convenience is 'reset windup' or excessive overshoot caused by the direct implementation of the integral action. Installing a limiter on the controller output to keep it from going beyond the operating range of the actuator is a way to prevent reset windup. To do this, let us first break down the nonlinear PI control rule into its parts (i.e. see Fig. 4).

Suppose a saturating actuator with a positive upper bound M^u and a lower bound M^l is installed. Then the output of the saturating actuator is

$$u_s(t) = \begin{cases} M^u & \text{for } m(t) \geq M^u \\ m(t) & \text{for } M^l \leq m(t) \leq M^u \\ M^l & \text{for } m(t) \leq M^l \end{cases} \tag{17}$$

where

$$m(t) = \overline{u} + k_c(t)e(t) + k_i(t)m_I(t) \tag{18}$$

and $M^u \geq \overline{u} \geq M^l \geq 0$. From the definition of $m_I(t)$ we get

$$\dot{m}_I(t) = e(t) \tag{19}$$

Since $k_c(t) > 0$ and $k_i(t) > 0$ in nature for all t, the reset-feedback implementation of the time-varying PI control law is thus given by

$$\dot{m}_I(t) = \frac{1}{k_c(t)}(m(t) - k_i(t)m_I(t) - \overline{u}) \tag{20}$$

Solving the above time-varying first-order ordinary differential equation we have

$$m_I(t) = \int_0^t \frac{m(s) - \overline{u}}{k_c(s)} \exp\left(-\int_s^t \frac{k_i(\tau)}{k_c(\tau)} d\tau\right) ds \tag{21}$$

Describing the exact trajectory of $m_I(t)$ needs full information of $k_c(t)$, $k_i(t)$ and $m(t)$. Estimating the dynamics of $k_c(t)$ and $k_i(t)$ is difficult if $k_c(t)$ and $k_i(t)$ are not prescribed. The possible extreme output at some instant and steady-state behaviour of $m_I(t)$ is investigated in the following.

There exist values of k_i^l, k_i^u, k_c^l and k_c^u for the proposed neural/fuzzy PI controller such that

$$0 < k_i^l \leq k_i(t) \leq k_i^u$$

and (22)

$$0 < k_c^l \leq k_c(t) \leq k_c^u$$

Then we have

$$0 < \frac{k_i^l}{k_c^u} \leq \frac{k_i(t)}{k_c(t)} \leq \frac{k_i^u}{k_c^l} \qquad \forall\, t \tag{23}$$

For all t and $s \leq t$, it thus implies that

$$0 < \int_s^t \frac{k_i^l}{k_c^u} d\tau \leq \int_s^t \frac{k_i(\tau)}{k_c(\tau)} d\tau \leq \int_s^t \frac{k_i^u}{k_c^l} d\tau \tag{24}$$

Hence

$$0 < \exp\left(-\int_s^t \frac{k_i^u}{k_c^l} d\tau\right) \leq \exp\left(-\int_s^t \frac{k_i(\tau)}{k_c(\tau)} d\tau\right) \leq \exp\left(-\int_s^t \frac{k_i^l}{k_c^u} d\tau\right) \tag{25}$$

Then

$$0 < \frac{\exp\left(-\int_s^t \frac{k_i^u}{k_c^l} d\tau\right)}{k_c^u} \leq \frac{\exp\left(-\int_s^t \frac{k_i(\tau)}{k_c(\tau)} d\tau\right)}{k_c(t)} \leq \frac{\exp\left(-\int_s^t \frac{k_i^l}{k_c^u} d\tau\right)}{k_c^l} \tag{26}$$

Since $-(\overline{u} - M^l) \leq m(t) - \overline{u} \leq M^u - \overline{u}$ for all t, thereby

$$\frac{-(\overline{u} - M^l)}{k_c^u} \exp\left(-\int_s^t \frac{k_i^u}{k_c^l} d\tau\right) \leq \frac{m(t) - \overline{u}}{k_c(t)} \exp\left(-\int_s^t \frac{k_i(\tau)}{k_c(\tau)} d\tau\right) \leq \frac{M^u - \overline{u}}{k_c^l} \exp\left(-\int_s^t \frac{k_i^l}{k_c^u} d\tau\right) \tag{27}$$

Then

$$\int_0^t \frac{-(\overline{u} - M^l)}{k_c^u} \exp\left(-\int_s^t \frac{k_i^u}{k_c^l} d\tau\right) ds \leq m_I(t) \leq \int_0^t \frac{M^u - \overline{u}}{k_c^l} \exp\left(-\int_s^t \frac{k_i^l}{k_c^u} d\tau\right) ds \tag{28}$$

Thus

$$\frac{-(\overline{u} - M^l)k_c^l}{k_i^u k_c^u}\left(1 - \exp\left(-\frac{k_i^u}{k_c^l}t\right)\right) \leq m_I(t) \leq \frac{(M^u - \overline{u})k_c^u}{k_i^l k_c^l}\left(1 - \exp\left(-\frac{k_i^l}{k_c^u}t\right)\right) \tag{29}$$

Therefore

$$(\overline{u} - M^l)\frac{k_i^l k_c^l}{k_i^u k_c^u}\left(1 - \exp\left(-\frac{k_i^u}{k_c^l}t\right)\right) \tag{30}$$

$$\leq k_i(t)m_I(t) \leq (M^u - \overline{u})\frac{k_i^u k_c^u}{k_i^l k_c^l}\left(1 - \exp\left(-\frac{k_i^l}{k_c^u}t\right)\right)$$

Then

$$\frac{-(\overline{u} - M^l)}{k}\left(1 - \exp\left(-\frac{k_i^u}{k_c^l}t\right)\right) \leq k_i(t)m_I(t) \leq (M^u - \overline{u})k\left(1 - \exp\left(-\frac{k_i^l}{k_c^u}t\right)\right) \tag{31}$$

where $k \triangleq k_i^u k_c^u / k_i^l k_c^l$ and $k > 1$. The result is consistent with the initial condition $k_i(0)m_I(0) = 0$. The derivation thus implies

$$-\frac{\overline{u} - M^l}{k} \leq k_i(t)m_I(t) \leq (M^u - \overline{u})k \tag{32}$$

To this end, Property 2 is thus derived.

Property 2 (Integral action of the neural/fuzzy VSPID controller is bounded): Since there exist constants k_i^l, k_i^u, k_c^l and k_c^u for neural/fuzzy VSPID controller parameters such that $0 < k_i^l \leq k_i(t) \leq k_i^u$ and $0 < k_c^l \leq k_c(t) \leq k_c^u$ for all t, with the proposed antireset windup implementation the integral action of the neural/fuzzy VSPID controller is bounded.

The possible maximal output of $k_i(t)m_I(t)$ is therefore equal to $k(M^u - \overline{u})$, and the possible minimal output is $-(\overline{u} - M^l)/k$. The rate at which the controller output, saturated at M^u, is reset is governed by the gain k, which determines how quickly the integral is reset, i.e. for large k it takes much time to reset the integral action and vice versa. Likewise, the rate at which the controller output, saturated at M^l, is reset is decided by $1/k$, which determines how quickly the integrator is reset, i.e. less time is spent to reset the integral action if $1/k$ is close to one and vice versa. Thus there is no effect on normal operation when the actuator does not saturate. Whilst saturating occurs, the feedback signal will try to drive the integrator to a value such that the controller output is exactly at the saturation limit. Preventing the integrator from winding up is therefore clear. There would be no saturation effect if k were equal to one. The result is consistent with the fact derived from linear control theory.

At steady state the error signal must be zero, since $e(t) = m_I(t) = 0$. No steady-state offset is therefore guaranteed for this modified PI control law. Suppose $k_c(t) = k_c^\infty$ and $k_i(t) = k_i^\infty$ at $e = r = 0$. It implies that

$$M^l \leq m(\infty) = k_i^\infty m_I(\infty) + \overline{u} \leq M^u \qquad (33)$$

The integral signal is confined to the range of the saturating actuator at steady state.

With the antireset windup, the nonlinear PI controller is always input–output stable, i.e. for finite energy input, the PI controller with antireset windup would yield finite energy output. A wide enough operating range of the controller is therefore required to put the controller into operation.

Suppose the nonlinear PI controller is operated at the normal level. Then for a given process, linear or nonlinear, the proposed nonlinear PI control system has the same local stability (asymptotically stable or unstable) at the equilibrium point, sp, as the linear PI ($k_c = k_c^\infty$, $k_i = k_i^\infty$) control system does. This is because the linearisation of the proposed nonlinear PI controller around the equilibrium point results in the linear PI controller. The two controllers thereby behave similarly in the region around the equilibrium point. According to Lyapunov's indirect method [19] (Lyapunov's first method on stability), the neural/fuzzy PI mode is asymptotically stable (unstable) at the equilibrium point if and only if the linear PI control system ($k_c = k_c^\infty$, $k_i = k_i^\infty$) is asymptotically stable (unstable).

5 Stability analysis of the neural/fuzzy PD control law

The closed-loop system is L_p stable if both subsystems, the process and the controller, are L_p stable in themselves and if the 'loop gain' is less than one for $1 \leq p \leq \infty$ [19]. Given any process in a complete function space, one could thus find a controller in the complete space such that the composite of the process and the controller converges at a fixed point in space, providing the gain of the composite is less than one. Suppose the controlled process is L_∞ stable. The proposed neural/fuzzy PD control law would be one of the choices since the PD controller is L_∞ stable, an assertion which is discussed in the following.

Consider the neural/fuzzy PD controller output at the nth sampling point:

$$u(n) = \overline{u} + k_c(n)e(n) + k_d(n)r(n) \qquad (34)$$

Then for $n \geq 1$ we have

$$\begin{aligned} |u(n)| &\leq |\overline{u}| + |k_c(n)||e(n)| + |k_d(n)||r(n)| \\ &\leq |\overline{u}| + \left(|k_c(n)| + \frac{|k_d(n)|}{T}\right)|e(n)| \\ &\quad + \frac{|k_d(n)|}{T}|e(n-1)| \end{aligned} \qquad (35)$$

Therefore for $n \geq 1$

$$\begin{aligned} &\sup|u(n)| \qquad (36) \\ &\leq \left(\sum_{i\in\mathbf{I}_1}\sum_{j\in\mathbf{J}_1}|w_{ij}| + \frac{\sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3}|w_{ij}|}{T}\right)\sup|e(n)| \\ &\quad + \frac{\sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3}|w_{ij}|}{T}\sup|e(n-1)| + |\overline{u}| \\ &\leq \left(\sum_{i\in\mathbf{I}_1}\sum_{j\in\mathbf{J}_1}|w_{ij}| + \frac{2\sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3}|w_{ij}|}{T}\right)\sup|e(n)| + |\overline{u}| \end{aligned}$$

By using the definition of the L_∞ gain of an L_∞ mapping $f(x(t))$ defined in [19],

$$\gamma_\infty(f) = \inf\{\gamma_\infty : \sup|f| \leq \gamma_\infty \sup|x| + b_\infty\} \qquad (37)$$

where γ_∞ and b_∞ are finite constants, the derivation leads to Property 3.

Property 3 (Gain of the neural/fuzzy PD controller): The gain of the PD controller $\gamma_\infty(u)$ is less than or equal to

$$\sum_{i\in\mathbf{I}_1}\sum_{j\in\mathbf{J}_1}|w_{ij}| + \frac{2\sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3}|w_{ij}|}{T}$$

Since there exists a finite upper bound of the proposed neural/fuzzy PD controller gain, an application of the small gain theorem yields the following stabilisation criterion for the proposed neural/fuzzy PD control system.

Property 4 (Sufficient criterion for the stable neural/fuzzy PD control system): The proposed neural/fuzzy PD control system is stable if

$$\sum_{i\in\mathbf{I}_1}\sum_{j\in\mathbf{J}_1}|w_{ij}| + \frac{2\sum_{i\in\mathbf{I}_3}\sum_{j\in\mathbf{J}_3}|w_{ij}|}{T} < \frac{1}{\gamma_\infty(y)}$$

where $\gamma_\infty(y)$ is the process gain.

Centres and receptive field widths in the neural/fuzzy basis functions are irrelevant to the stabilisation criterion. The most important factors related to the stabilisation criterion are output connective weights of the neural/fuzzy systems. According to Property 4, the number of parametric space dimensions related to the stability region of the proposed PD control system is reduced to $l_1 \times l_2 + l_5 \times l_6$, as compared to the dimensions ($l_1 \times l_2 + 2l_1 + 2l_2 + l_5 \times l_6 + 2l_5 + 2l_6$) in the original neural/fuzzy parametric space (PD part). Stabilising the PD mode control could be done by appropriately adjusting weights in k_c and k_d satisfying the stabilisation criterion.

6 Training procedures

The aim of training a neural/fuzzy VSPID controller is to minimise the following performance measure, i.e. to solve a finite horizon optimisation problem

$$J_e = \frac{T}{2}\sum_{p=1}^{P}\sum_{n=1}^{N_p}\{[sp(n) - y(n)]^2\}_p \qquad (38)$$

Here, $\{sp(\cdot)\}_p \in \mathbf{S}$, $\mathbf{S}$ is a set of P individual reference inputs to the process over the possible operating region; $\{y(\cdot)\}_p$ are outputs of the real plant in response to the given pth testing input excitation; N_p is the total number of control instants within the interval of each training time under consideration. Different choice of the input set for training could lead to somewhat different controller parameters. One can directly use any existing optimisation methods, such as the steepest gradient method or the conjugated gradient method, in search of the optimal parametrisation of the neural/fuzzy VSPID controller.

Suppose the controlled process has a time delay d. Let z_i be one of the elements in parametric space of the proposed neural/fuzzy VSPID controller. The associated gradient is

$$\frac{\partial J_e}{\partial z_i} = -T \sum_{p=1}^{P} \sum_{n=1}^{N_p} \left\{ \left[(sp(n+d) - y(n+d)) \frac{\partial y(n+d)}{\partial u(n)} \right] \frac{\partial u(n)}{\partial z_i} \right\}_p \tag{39}$$

In the equation, $\partial y(n + d)/\partial u(n)$ required to update parameter z_i in the parametric space of the neural/fuzzy VSPID controller during optimisation. Therefore, a first-principle mathematical model [20] or a well trained plant emulator [21] is required to estimate these partial derivatives of the plant response to neural/fuzzy VSPID controller parameters at current operating points. However, especially where more complex plants are concerned, such a precise plant emulator is often not available. When the information is not provided, one can take the sign of the partial derivatives of the plant, i.e. sign $\partial y(n + d)/\partial u(n)$, to give approximate information [22]. Such an approximation might, however, slow down the speed of convergence in finding optimal solutions.

Several training passes are required to search for convergent parameters (η^*, ε^*, c_{ei}^*, c_{rj}^*, s_{ei}^*, s_{rj}^* and w_{ij}^*, for $i \in \mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3$, $j \in \mathbf{J}_1, \mathbf{J}_2, \mathbf{J}_3$) of the optimal neural/fuzzy VSPID controller (u^*). Notice that the current VSPID controller parameters are kept constant through the whole control pass in the course of this searching. Given the specific input pattern $\mathbf{x}(t) = [e(t), r(t)]^T$, the control response of the optimal neural/fuzzy VSPID controller is

$$u^*(n) = \overline{u} + k_c^*(n)e(n) + \alpha^*(n)k_d^*(n)r(n) + (1 - \alpha^*(n))k_i^*(n)m_I(n) \tag{40}$$

7 Illustrations

7.1 pH control of a neutralisation process

The applicability of the neural/fuzzy VSPID controller is demonstrated by controlling the pH value of a neutralisation process. The control near the neutrality point (pH = 6 ~ 8) is notably difficult since a slight change of operating condition could make a drastic difference in pH value.

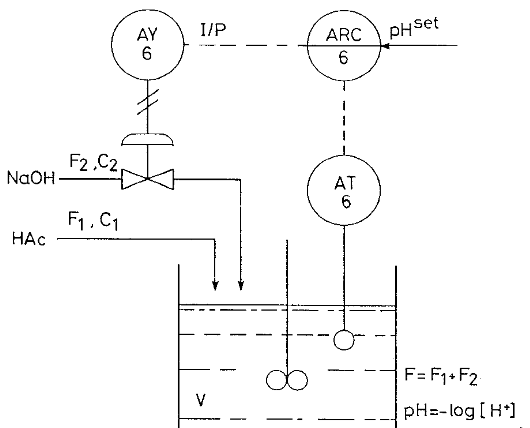

Fig.5 *pH control system*

In this example, the strong base NaOH is used in controlling the acidity of a wild process stream with weak acid HA. A perfectly effective liquid level controller is used to keep the reactor volume constant (see Fig. 5). The balanced equations are [23]:

Acid balance:

$$V\frac{d\xi}{dt} = F_1C_1 - (F_1 + F_2)\xi \tag{41}$$

Sodium balance:

$$V\frac{d\zeta}{dt} = F_2C_2 - (F_1 + F_2)\zeta \tag{42}$$

Charge balance:

$$[H^+]^3 + [H^+]^2\{\zeta + K_a\} + [H^+]\{(\zeta - \xi)K_a - K_w\} - K_aK_w = 0 \tag{43}$$

Sensory signal lag:

$$pH(t) = -\log_{10}([H^+](t - 0.4)) \tag{44}$$

In these equations, $\xi \triangleq [HA] + [A^-]$, $\zeta \triangleq [Na^+]$, $K_a = [H^+][A^-]/[HA]$ (the acid equilibrium constant) and $K_w = [H^+][OH^-]$ (the water equilibrium constant). The charge balance equation results from the fact that $\zeta + [H^+] = [OH^-] + [A^-]$. The operating range of the base inlet stream F_2 is 0 ~ 2L/min. Numerical values of the physical variables involved here are listed in Table 1.

Table 1: Nominal values of parameters in the neutralisation process

K_a	= 10^{-3}mol/L	K_w	= 10^{-14}mol^2/L^2
C_1	= 0.10998mol/L	C_2	= 1mol/L
V	= 50L	F_1	= 10L/min
pH	= 4	F_2	= 0.9988L/min
ξ	= 0.1mol/L	ζ	= 0.09081mol/L

In demonstrating the applicability of the proposed control method, the set-point of pH is changed from 4 (initial steady state) to 7 and two input loads, acid concentration from 0.10998 to 0.1105mol/L and acid flow rate from 10 to 10.5L/min are then introduced at 60 min and 100min, respectively. The control interval T is 0.2min.

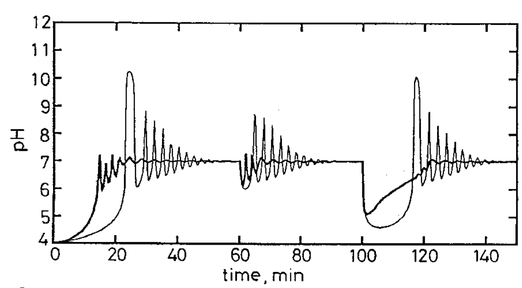

Fig.6 *Comparison of servo and regulatory control performance for the neutralisation process when using optimised linear PI and optimised L_∞ NN_PI: pH against time*
linear PI ——
L_∞ NN_PI ——

Two 5 × 5 (i.e. $l_i = 5$, $i = 1, ..., 4$) modified GPFNs are used to construct a nonlinear relationship between local control conditions and L_∞ NN_PI controller parameters. Optimisation is achieved for the linear PI and L_∞ NN_PI by using the steepest-descent technique with the criterion of minimum integral of square error. Figs. 6 and 7 show servo and regulatory control results derived from using both the optimised linear PI controller and the optimised L_∞ NN_PI. The k_c and k_i trajectories of the optimised L_∞ NN_PI are illustrated in Fig. 8.

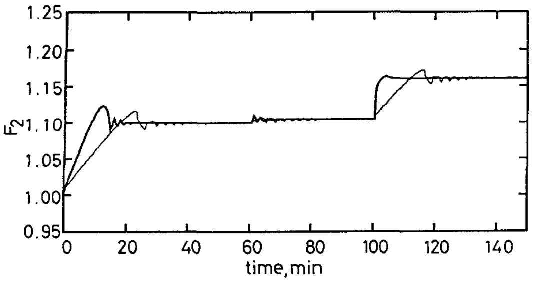

Fig. 7 *Comparison of servo and regulatory control performance for the neutralisation process when using optimised linear PI and optimised L_∞ NN_PI: F_2 against time*
linear PI ——
L_∞ NN_PI ——

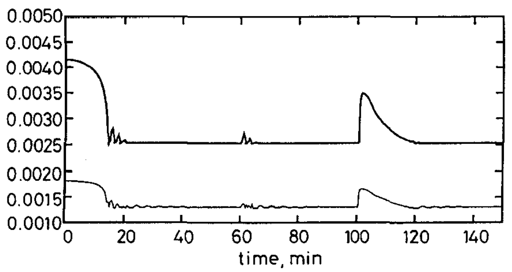

Fig. 8 *k_c and k_i trajectories of optimised L_∞ NN_PI*
k_c of L_∞ NN_PI ——
k_i of L_∞ NN_PI ——

Three 5 × 5 (i.e. $l_i = 5$, $i = 1, ..., 6$) modified GPFNs are used to construct a nonlinear relationship between local control conditions and L_∞ NN_VSPID controller parameters. Fig. 9 shows servo and regulatory control results derived from using both the optimised linear PID controller and the optimised L_∞ NN_VSPID. The k_c, k_i and k_d trajectories of the optimised L_∞ NN_VSPID are illustrated in Fig. 10. The variation of the α value for the optimised L_∞ NN_VSPID is illustrated in Fig. 11. The best obtainable servo and regulatory control performances for the optimised linear PI, the linear PID, the L_∞ NN_PI and the L_∞ NN_VSPID, are shown in Table 2.

Undesirable overshoots and large transients, as observed from the control performance derived from conventional controllers, can be satisfactorily eliminated by the neural/fuzzy VSPID controller. The improvement in servo and regulatory control performance derived from the preassigned algorithm of the law of the controller structure change and the resilient relationship between controller parameters and local control conditions is desirable and excellent. Satisfactory servo and regulatory control performance could also be derived from FL_VSPID controllers. This example has illustrated the potential value of using the proposed L_∞ NN_VSPID controller in highly nonlinear chemical processes.

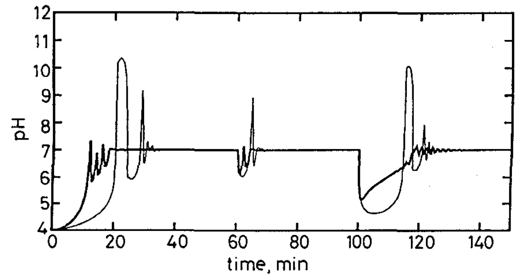

Fig. 9 *Comparison of servo and regulatory control performance for the neutralisation process when using optimised linear PID and optimised L_∞ NN_VSPID*
linear PID ——
L_∞ NN_VSPID ——

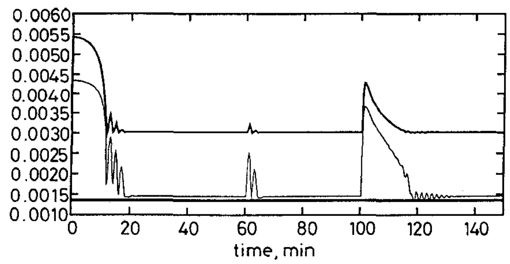

Fig. 10 *k_c, K_i and k_d trajectories of optimised L_∞ NN_VSPID*
k_c of L_∞ NN_VSPID ——
k_i of L_∞ NN_VSPID ——
k_d of L_∞ NN_VSPID ▬▬

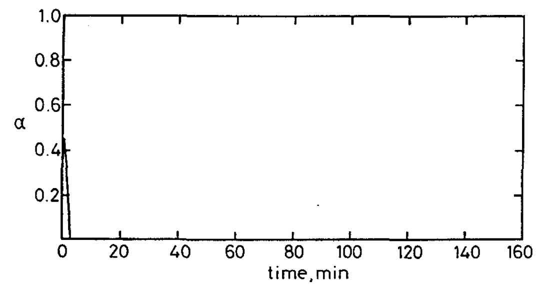

Fig. 11 *Variation of α value for optimised L_∞ NN_VSPID*

8 Conclusion

This paper has described the design method of a neural/fuzzy variable structural proportional-integral-derivative (neural/fuzzy VSPID) control system. The PD mode is emphasised in the case of a large error occurring so as to speed up response, and the PI mode is applied to small error conditions to eliminate the steady-state offset. A sigmoidal-like neuron has been employed as a preassigned algorithm of the law of the structural change. Meanwhile, the VSPID controller

Table 2: Comparison of servo and regulatory control performance for the neutralisation process when using the linear PI, the linear PID, the L_∞ NN_PI and the L_∞ NN_VSPID

Controller (optimised)	k_c	k_i	k_d	η	ε	J_e
linear PI	2.673×10^{-3}	1.938×10^{-3}	—	—	—	294.31
linear PID	2.436×10^{-3}	2.265×10^{-3}	5.399×10^{-4}	—	—	270.82
L_∞ NN_PI	trajectories of k_c and k_i : see Fig. 8		—	—	—	129.45
L_∞ NN_VSPID	trajectories of k_c, k_i, k_d and α : see Figs. 10, 11			10	2.951	106.21

parameters are changed according to local conditions. L_∞ neural networks or bounded-input/bounded-output fuzzy logic systems have been used for constructing a nonlinear relationship between the PID controller parameters and local operating control conditions. The PI mode with antireset windup has been designed for prevention of excessive overshoot caused by direct implementation of integral action. The stability analysis of the PI/PD part has also been discussed. A neutralisation process has been used to demonstrate the applicability of such a controller for controlling highly nonlinear processes.

9 References

1 SHINSKYEY, F.G.: 'Process control systems: application, design and tuning', (McGraw–Hill, New York, 1988, 3rd edn.)

2 CLARK, F.C.: 'PID algorithms and their computer implementation', *Trans. Inst. Meas. Control*, 1984, **6**, pp. 305–316

3 JUTAN, A.: 'A nonlinear PI(D) controller', *Can. J. Chem. Eng.*, 1989, **67**, pp. 485–493

4 TAN, S., LIN, YU., WANG, P., and HE, S.: 'Objective-centered formulation of an adaptive fuzzy control scheme', *Int. J. Uncertainty, Fuzziness and Knowledge-Based Syst.*, 1994, **2**, (3), pp. 321–331

5 CHEN, W.C., and CHEN, C.L.: 'Nonlinear PI controller design: a neural network approach', *J. Chin. I. Chem. E.*, 1995, **26**, (2), pp. 67–79

6 WANG, H., BROWN, M., and HARRIS, C.J.: 'Neural network modelling of unknown nonlinear systems subject to immeasurable disturbances', *IEE Proc., Control Theory Appl.*, 1994, **141**, (4), pp. 216–222

7 LIGHTBODY, G., and IRWIN, G.W.: 'Direct neural model reference adaptive control', *IEE Proc., Control Theory Appl.*, 1995, **142**, (1), pp. 31–43

8 RAOL, J.R.: 'Neural network based parameter estimation of unstable aerospace dynamic systems', *IEE Proc., Control Theory Appl.*, 1994, **141**, (6), pp. 385–388

9 YANG, Y.Y., and LINKENS, D.A.: 'Adaptive neural-network-based approach for the control of continuously stirred tank reactor', *IEE Proc., Control Theory Appl.*, 1994, **141**, (5), pp. 341–349

10 HORIKAWA, S., FURUHASHI, T., and UCHIKAWA, Y.: 'On fuzzy modelling using fuzzy neural networks with backpropagation algorithm', *IEEE Trans. Neural Netw.*, 1992, **3**, pp. 801–806

11 WANG, L.X.: 'Adaptive fuzzy systems and control' (Prentice–Hall International Inc., 1994)

12 PARK, J., and SANDBERG, I.W.: 'Approximation and radial-basis-function networks', 'Neural Comput.' , **5**, pp. 305–316.

13 MOODY, J., and DARKEN, C.: 'Fast learning in networks of locally-tuned processing units', *Neural Comput.*, 1989, **1**, pp. 281–294

14 RUMELHART, D.E., HINTON, G.E., and WILLIAMS, R.J.: 'Learning internal representations by error propagation', *in* RUMELHART, D.E., and McCLELLAND, J.L. (Eds.): 'Parallel distributed processing: explorations in the microstructures of cognition. Vol. 1: foundations' (MIT Press, Cambridge, MA, 1986), pp. 318–362

15 KAVURI, S.N., and VENKATASUBRAMANIAN, V.: 'Using fuzzy clustering with ellipsoidal units in neural networks for robust fault classification', *Comput. Chem. Eng.*, 1993, **17**, (8), pp. 765–784

16 CHEN, S., COWAN, C.F.N., and GRANT, P.M.: 'Orthogonal least squares learning algorithm for radial basis function networks', *IEEE Trans. Neural Netw.*, 1991, **2**, pp. 302–309

17 CHEN, C.L., CHEN, W.C., and CHANG, F.Y.: 'Hybrid learning algorithm for Gaussian potential function networks', *IEE Proc. D*, 1993, **140**, (6), pp. 442–448

18 POWELL, M.J.D.: 'Radius basis function for multivariate interpolation: a review', Technical report DAMPT 1985/NA12, Cambridge University, 1985

19 VIDYASAGAR, M.: 'Nonlinear system analysis' (Prentice–Hall International Inc., 1993, 2nd edn.)

20 PSALTIS, D., SIDERIS, A., and YAMAMURA, A.A.: 'A multilayered neural network controller', *IEEE Control Syst. Mag.*, April 1988, pp. 17–21

21 NGUYEN, D.H., and WIDROW, B.: 'Neural networks for self-learning control system', *IEEE Control Syst. Mag.*, April 1990, pp. 18–23

22 CHEN, F.C.: 'Back-propagation for nonlinear self-tuning adaptive control', *IEEE Control Syst. Mag.*, April 1990, pp. 44–48

23 MCAVOY, T.J., HSU, E., and LOWENTHAL, S.: 'Dynamics of pH in controlled stirred tank reactor', *Ind. Eng. Chem. Process Des. Develop.*, 1972, **11**, pp. 68–70