A comparative study of neuro fuzzy and recurrent neuro fuzzy model-based controllers for real-time industrial processes

B. Subathraa, S. Seshadrib and T.K. Radhakrishnanc

aICE Department, Kalasalingam University, Madurai, Tamil Nadu, India; bInternational Research Centre, Kalasalingam University, Madurai, Tamil Nadu, India; cNational Institute of Technology, Tiruchirappalli, Tamil Nadu, India

(Received 1 September 2014; accepted 21 May 2015)

Nonlinearities in system dynamics and the multivariable nature of processes offer a stiff challenge in designing predictive controllers that improve process performance in industries. This investigation presents a recurrent neuro fuzzy network (RNFN) model for a nonlinear multivariable system in process industries and a methodology to design model-predictive controllers (MPCs) using the proposed model. The RNFN model combines the learning features of artificial neural networks with human cognition capabilities of fuzzy systems. Therefore, RNFN leads to a modelling framework that has the ability not only to learn the model parameters, but also makes decision on operating region of the nonlinear model depending on the input–output data. Furthermore, the recurrent structure and the introduction of a memory unit between the fuzzy inference and fuzzification layer enhance the prediction capability due to the use of past input–output data, making the model more suitable for designing predictive controllers. Next, the MPC design methodology that exploits the advantages of the RNFN model to optimize the control moves is presented. The proposed MPC uses the gradient descent algorithm to minimize the control moves as against the traditional state-space approaches that require complex computations and solvers. Therefore, implementing the proposed MPC in embedded hardware becomes easier. The proposed modelling framework and the MPC design methodology are illustrated using experiments on a laboratory-scale quadruple tank. Our experiments show that the proposed RNFN-based MPC performs better than the neuro fuzzy network-based MPC for both servo and regulatory responses.

Keywords: modelling; intelligent techniques; hybrid soft computing; recurrent neuro fuzzy network; MPC; gradient descent algorithm

1. Introduction

Competition stemming from globalization and increasing operating cost necessitates new technologies that can optimize in process performance without significant investments industries. In this backdrop, a model predictive controller (MPC) that use process models, estimation on disturbance and an optimization routine to improve the process performance has emerged as a promising solution. However, model complexity due to system nonlinearities and the multivariable nature of the process, time-varying disturbances and the need for solvers and complex optimization routines offer stiff challenges in adapting MPCs within process industries. On the other hand, optimization has become a necessity to derive technological and market leadership. Therefore, a new modelling framework that can handle model complexities such as non-linearity and the multivariable nature of the process, and controller design methodologies using such models are required in process industries. In particular, models capturing the interactions among the manipulated and controlled variables are required for multivariable systems. Our objective in this investigation is to propose a new modelling framework for interacting multivariable and nonlinear systems that inherently estimates possible future disturbances, and then uses them for designing an MPC that optimizes process performance while simultaneously reducing the fluctuations in control moves.

Modelling and designing controllers for multivariable and nonlinear systems have attracted significant research. Zhang and Morris (1999) classified process models into two categories: first principle and empirical. Furthermore, the authors point out that the first principle models are computationally cumbersome and time consuming to develop, especially for complex systems. To overcome this difficulty, process models based on input–output data have been developed.

Empirical models using neural networks have shown better accuracy and simplicity over other methods such as system identification, and have been used successfully for modelling complex processes by many researchers (Bhat & McAvoy, 1990; Bulsari, 1995; Morris, Montague, & Willis, 1994). However, neural network models
are difficult to interpret and lack robustness when applied to unseen data. Model accuracy can be improved by using process knowledge along with process input–output data. For instance, the process knowledge can be used to derive local models for a given set of operating points using fuzzy approach as in Yager and Filev (1994). In the fuzzy modelling approach developed by Takagi and Sugeno (1985), the model input space is portioned into several fuzzy regions. A local linear model is used within each region and a global model is obtained using centre of gravity defuzzification. The modelling approach illustrated that process knowledge can be used to significantly reduce model complexity. A similar approach has been used to construct Nonlinear Auto Regressive Moving Average with eXogenous inputs models by Johansen and Foss (1993). The fuzzy modelling approach used process knowledge, but the potential of input–output data was not completely exploited in fuzzy models. Consequently, researchers combined fuzzy and neural approaches to derive process models that exploited the learning capability of neural networks and decision-making of fuzzy systems. This led to the development of adaptive-network-based fuzzy inference system (ANFIS) architecture to represent fuzzy models (Jang, 1992; Jang & Sun, 1995; Jang, Sun, & Mizutani, 1997); two types of feed-forward neuro fuzzy networks were proposed by the authors for nonlinear process modelling (Zhang & Morris, 1995b). The use of dynamic neural network for long range prediction models has been studied in globally recurrent neural networks (RNN, e.g. Su, McAvoy, & Werbos, 1992; Werbos, 1990), Elman networks (Elman, 1990; Scott & Ray, 1993), dynamic feed-forward network with filters (Montague, Tham, Willis, & Morris, 1992; Morris et al., 1994) and locally recurrent networks (Frasconi, Gori, & Soda, 1992; Tsoi & Back, 1994; Zhang & Morris, 1995a). Zhang and Morris (1995a) presented a sequential orthogonal training strategy which allows for hidden neurons to be added gradually, avoiding an unnecessarily large network structure. However, Su et al. (1992) proved that these models are not suitable for long-term predictions. In Zhang and Morris (1999), a recurrent neuro fuzzy network (RNFN) was proposed that allowed construction of a global non-linear multi-step ahead prediction model from the fuzzy conjunction of a number of local dynamic models for the pH neutralization process. This approach used both process data and knowledge to build multi-step ahead models. However, the role of the MPC design has not been explored, which is required for improving process performance. The RNFN model and the structure obtained in Zhang and Morris (2000) have been used for long-term prediction of outputs in level control of a conical tank. Juang and Chen (2003) proposed a six-layer, TSK-type, RNFN structure-based controller for a thermal process. Lia and Cheng (2007) proposed a six-layer RNF system, in which recurrence is introduced in the membership layer (self) and the results are illustrated for a simulated system. In the six-layer structure proposed, the network consists of two external inputs and a single output. Hence, only one output can be obtained at the end of training of the network and it is therefore not suitable for modelling multivariable processes. Review of the literature reveals that the use of RNFN for modelling real-time multivariable processes with interactions and design of predictive controllers has not been explored (Subathra & Radhakrishnan, 2011b). Motivated by this research gap, this investigation aims to model a nonlinear multivariable process with interactions using RNFN and then propose an MPC design methodology for the model.

To reach the objectives, this investigation first presents a new seven-layer RNFN structure for modelling and identification of a real-time nonlinear multivariable process, and then uses it to design the predictive controller. Recurrent structure is obtained by introducing delay units between the fuzzy inference and fuzzy membership layer. Therefore, weights in the inference layer determine the local operating regions, while the weights in the output of the membership layer represent the singleton values in the consequent part of the rules. Neural back propagation (BP) algorithm in fuzzy inference layer is used to tune the antecedent part of the rule and this can be used to control the input variable. In Juang and Chen (2003), firing strength of the fuzzy rules has been varied based on internal parameters and internal inputs and the role of varying the consequent part of the fuzzy rule has not been explored. In this study, the firing strength is varied by changing the consequent part of the fuzzy rule in addition to the conventional RBFN structure. This small modification leads to significant improvements in model accuracy. Furthermore, the long-range prediction capability of the nonlinear multivariable system is enhanced due to the recurrent structure and this can be observed from our experimental studies on a quadruple process. The model thus obtained is used to design the MPC for the nonlinear multivariable system. Multi-step ahead prediction based on the RNFN model and the set point calculations are used to predict the future control moves that optimize the process performance. To design the MPC using the RNFN model, gradient descent algorithm is used for optimizing the control moves. This eliminates the need for dedicated solvers and facilitates implementation in dedicated embedded hardware.

The main contributions of this investigation are: (i) a new seven-layer RNFN structure for modelling and identification of a nonlinear multivariable process (ii) an MPC design methodology employing the RNFN model and (iii) illustration of the model accuracy and controller performance using experiments on a laboratory-scale quadruple tank prototype. The paper is organized in seven sections. Section II presents the problem and discusses the model and working of the quadruple process. The RNFN model is presented in Section 3, and the empirical model for the
2. Problem description
The problem is to propose a new modelling framework and MPC design methodology for multivariable and nonlinear processes with interactions among manipulated variables. The idea of the modelling technique is to use input–output data and knowledge on process operating points to obtain a process model that can be used for designing MPC. The proposed predictive controller uses the model and computes the future control moves that optimize the performance.

2.1. Quadruple tank process
The input–output data and process knowledge from a laboratory-scale quadruple process first studied by Johansson and Nunes (1998) are used. The authors have derived the mathematical model of the process and have studied the working of the process in detail. Furthermore, the study established that linearized dynamics of the system has a multivariable zero that can move along the real-axis by changing the valve position. Because of right half plane zero performance limitations are present in the process. The tank process has received significant attention recently in both modelling and control due to its dynamic characteristics (Doyle, Gatzke, Vadigepalli, & Meadows, 1999). By modifying the valve position and location of zero the dynamic characteristics of the process can be modified and exhibits non-minimum phase behaviour. The quadruple tank system can be used to demonstrate not only the conventional controllers, but also sophisticated controllers such as MPC using linearized models.

In the quadruple process, two pumps are used to transfer water from a storage tank to the process tanks. The two tanks at the upper level drain into the two tanks at the bottom level as shown in Figure 1. The liquid levels in all the four tanks are measured and bottom two tanks controlled. The piping system is such that each pump affects the liquid levels of both the tanks. A part of the flow from
one pump is fed to one of the lower level tanks (where the level is monitored). A photograph of the experimental prototype is shown in Figure 1. The amount of interaction between inputs and outputs can be adjusted by varying the bypass valves of the process. External flow disturbances can be introduced into the upper level tanks. The process dynamics changes with respect to the ratio of the valve openings. The nomenclature and the nominal operating conditions used in this study are given in Table 1. The mass balance equation for the quadruple process is a nonlinear multivariable equation (Johansson & Nunes, 1998):

\[
\begin{align*}
\frac{dh_1}{dt} &= -\frac{a_1}{A_1} \sqrt{2gh_1} + \frac{a_3}{A_1} \sqrt{2gh_3} + \gamma_1 k_1 \nu_1, \\
\frac{dh_2}{dt} &= -\frac{a_2}{A_2} \sqrt{2gh_2} + \frac{a_4}{A_2} \sqrt{2gh_4} + \gamma_2 k_2 \nu_2, \\
\frac{dh_3}{dt} &= -\frac{a_3}{A_3} \sqrt{2gh_3} + \frac{(1 - \gamma_2) k_2}{A_3} \nu_2, \\
\frac{dh_4}{dt} &= -\frac{a_4}{A_4} \sqrt{2gh_4} + \frac{(1 - \gamma_1) k_1}{A_4} \nu_1.
\end{align*}
\]

Johansson and Nunes (1998) analysed the performance of the quadruple tank process and reported that the process yields an inverse response during \((\gamma_1 + \gamma_2) < 1\). Prior to the control analysis the manipulated inputs are scaled by 25%. This scaling corresponds to actual input levels in the range of 25–75%. Because of the nonlinearities introduced by the water head in the piping and limited pump capacities, the pump cannot operate satisfactorily below the 25% level. Hence, the pump settings are constrained to remain within 25–75% in our analysis.

### Table 1. Nominal Operating conditions and Parameter values.

| Symbol   | State/parameters | Value                  |
|----------|-----------------|------------------------|
| \(h^0\)  | Nominal levels  | [11.65; 9.6; 8.6; 5.6] cm |
| \(v^0\)  | Nominal pump settings | [60; 60] |
| \(a_i\)  | Area of the drain in Tank \(i\) | [0.236; 0.1985; 0.159; 0.127] cm² |
| \(A_i\)  | Areas of the tanks | 65.755 cm² |
| \(\gamma_1\) | Ratio of flow in Tank 1 to flow in Tank 4 | 0.453 |
| \(\gamma_2\) | Ratio of flow in Tank 2 to flow in Tank 3 | 0.307 |
| \(k_j\)  | Pump proportionality constants | [5.5143; 4.9728] cm³/ (V-sec) |
| \(G\)    | Gravitation constant | 980 cm/sec² |

3. **Hybrid RNFN model**

3.1. **Neuro fuzzy network**

Neuro fuzzy technique is widely used in modelling of complex processes (Brown & Harris, 1994) and the hybridization combines the unique advantages of the fuzzy and neural networks. The model can be developed in such a way as to mimic the process exactly with the help of empirical data and linguistic fuzzy rules can be used to represent the model. The advantages of hybrid modelling techniques are:

(a) Modelling and identification of a nonlinear process can be obtained by using both input–output data and knowledge about the process.

(b) The resulting models are simple.

Knowledge on process operations is used to define a suitable model structure; this model is then synthesized during the modelling process such that it can successfully represent and reproduce the available empirical data used for training. This adaptation step is often called learning. The main objective of neuro fuzzy modelling is to construct a model that accurately predicts the value(s) of the output variable(s) even when new values of the input variables are presented. The major drawback of this modelling technique is that it cannot be used for dynamic processes and yields a good one-step ahead prediction model because of its feed-forward structure (Brdys & Kulawski, 1999). While developing a long-range predictive controller, the history of input and output data is needed for dynamic processes. For this purpose, a feedback loop is introduced in this structure.

3.2. **Recurrent neuro fuzzy modelling**

Most of the industrial processes are complex and dynamic in nature. Therefore, modelling such processes requires the history of input–output data. The feedback structure offers a memory unit to store the past data, which enhances the long-range prediction capability of the models required for designing predictive controllers (Zhang, 2003). In literature, five- and six-layer RNFN structures have been proposed for modelling and control of the SISO process (Lia & Cheng, 2007). The role of RNFN to model a nonlinear process has been studied in Zhang and Morris (1999), where the authors proposed a five-layer Jordan network to model the pH neutralization (SISO) process. In the six-layer RNFN structure proposed in Juang Huang, and Duh (2006) to control the temperature, an Elman-type recurrent structure has been used. The role of RNFN in modelling a nonlinear and multivariable process has not been investigated.

3.3. **Proposed seven layer with RNFN structure**

In this investigation, a Takagi–Sugeno (TS)-type fuzzy inference system is implemented with an rRNN.

3.3.1. **Basic structure of RNFN**

The proposed approach uses the Elman-type recurrent network structure in which delay units act as memory...
elements that are located in the feedback path between the fuzzy inference layer and the fuzzy membership layer so that consequent parts of the fuzzy rules are tuned considering the dynamics of the process. The RNFN approaches in literature modify the fuzzy rule and firing strengths are fixed using the rule layer output and external input (for, e.g. Zhang & Morris, 1999; Juang et al., 2006). This investigation varies the firing strength by changing the consequent part of the fuzzy rule or in other words the internal parameters in addition to the conventional RBFN structure. Thus, proper tuning of fuzzy rules can be obtained by modifying the firing strength. The comprehensive rule base for this proposed structure is

RULE \( i \) : IF \( x_1(k) \) is \( A_{i1} \) \ldots \( x_n(k) \) is \( A_{in} \) AND \( h_i(k) \) is \( G \) THEN \( \hat{y}_i(k+1) = a_0 + a_1 x_1(k) + a_2 x_2(k) + a_3 x_3(k) + e(k) \) AND \( h_i(k+1) \) is \( W_i \), \( (2) \)

where the fuzzified inputs are \( x_1(k), \ldots, x_n(k) \), the fuzzy sets are ‘\( A \)’ and ‘\( G \)’, consequent parameters are ‘\( W \)’ and ‘\( a \)’ for the inference output ‘\( h \)’ and ‘\( y \)’, respectively, number of external input variables is denoted as ‘\( n \)’, error \( e(k) = y_{exp}(k) - \hat{y}(k) \), where \( y_{exp}(k) \) is the experimental output and \( \hat{y}(k) \) is the predicted output. The TS-type fuzzy method is used to model the process and the linear combination of the input variables \( x \) and internal variables \( h \) plus a constant. Process variables are defined with fuzzy operating regions using Gaussian membership functions and are the inputs to the fuzzification (input) layer. In the feedback path, sigmoidal membership functions are used for fuzzification to the rule layer as shown in Figure 2.

The rules and operating regions of the fuzzy controller are framed based on the knowledge about the process. The fuzzy operating regions of the process are defined with neurons and they are trained with neural network training algorithms. The fuzzified values of the process variables and delayed feedback from the fuzzy inference layer are the inputs to the rule layer which determine the firing strength. Output of the fuzzy rule layer is the result of AND operation applied to inputs and its membership function. Each process variable is represented as a fuzzy set.
with several fuzzy operating regions (Subathra, Raja Rao, & Radhakrishnan, 2009; Subathra & Radhakrishnan, 2010, 2011a).

3.3.2. Description of RNFN

(i) Input Layer (1)

In this structure, inputs to the fuzzification layer are process variables and they are used to define fuzzy operating regions. Each process variable can be defined as fuzzy sets in the fuzzification layer. Every fuzzy set can be represented by neurons; fuzzy set output gives the membership function. Two different types of neurons are employed; the nodes in the first layer represent an input variable. This layer does not perform any computation and the values are transmitted directly to the next layer.

(ii) Membership Layer (2)

A process variable can be represented in terms of the fuzzy membership function. Each node in the second layer corresponds to a linguistic variable and is represented using the fuzzy membership function. Two fuzzy membership functions are used in the proposed structure. For external input $x_j$, Gaussian membership function is used,

$$O_{ij}^{(2)} = \exp \left\{ - \frac{(x_j^{(2)}(k) - m_{ij})^2}{\sigma^2_{ij}} \right\} \quad \text{and} \quad x_j^{(2)}(k) = O_{ij}^{(1)}(k),$$

where $m_{ij}$ and $\sigma_{ij}$ are the centre and the width of the Gaussian membership function of the $i$th term of the $j$th input variable $x_j$. All the weights in membership layer are set to unity. The sigmoid membership function is used for internal feedback variables $h_i$,

$$O_{ij}^{(2)} = \frac{1}{1 + \exp[-x_j^{(2)}(k)]} \quad \text{and} \quad x_j^{(2)}(k) = O_{ij}^{(5)}(k),$$

where $O_{ij}^{(5)} = h_i(k)$ and $O_{ij}^{(2)} = p_i(k)$.

Membership value is calculated in each node in order to specify the degree to which an input variable belongs to a linguistic label. Other types of membership functions like trapezoidal and triangular could be used.

(iii) Rule Layer (3)

The output of each node in this layer is determined by fuzzy AND operation.

$$O_{ij}^{(3)}(k) = \prod_{j=1}^{n+r} O_{ij}^{(2)}(k) = \prod_{j=1}^{r} \frac{1}{1 + \exp[-O_{ij}^{(5)}(k)]} \star \exp \left\{ - \frac{\sum_{j=1}^{n} (O_{ij}^{(1)}(k) - m_{ij})^2}{\sigma^2_{ij}} \right\},$$

where ‘$n$’ is the number of external inputs and $r$ is the number of rules. The link weights are all set to unity. In this layer, the product operation is utilized to determine the firing strength of each rule.

(iv) TS Fuzzy Inference Layer (4)

The nodes in this layer perform a linear summation. Linear function in each RNFN rule consequent is replaced by a constant value. The mathematical function of each node $i$ is

$$O_{ij}^{(4)}(k) = \sum_{j=1}^{3} a_{4i-4+j}(k)x_j(k) + a_{4i}p_i(k-1) + err(k),$$

(v) Internal Feedback Layer (5)

The outputs of inference layer are fed back to the fuzzification layer, thus acting as internal feedback. The output of the nodes of the previous layer is normalized in this layer by the following operation.

$$h_i(k) = \frac{O_{ij}^{(4)}(k)}{\sum_{i=1}^{r} O_{ij}^{(4)}(k)}.$$ (7)

The link weights represent the singleton values in the consequent part of the internal rules. The simple weighted sum is calculated in each node. As shown in Figure 1, the delayed value of $h_i(k)$ is fed back to the fuzzification layer and acts as an input variable to the precondition part of a rule. Each rule has a corresponding internal variable $p_i(k)$ and is used to decide the current rule.

(vi) Output Layer (6)

The node in this layer computes the output linguistic variable $y$ of the RNFN. The output node along with links will be proceeding like the defuzzifier. The mathematical function is

$$y(k) = O_{ij}^{(6)} = \sum_{j=1}^{r} \frac{O_{ij}^{(3)} O_{ij}^{(4)}}{\sum_{j=1}^{r} O_{ij}^{(3)}},$$

(vii) Normalization Layer (7)

The output from the layer (6) has to be formatted such that it is suitable for real-world application and is called normalization. The final output of the network has to be suitable for the real-world process; this node yields the normalized output of the RNFN model.

$$\hat{y}(k) = \frac{1}{1 + \exp[-y(k)].}$$ (9)

This new seven-layer RNFN can be used to develop a model for any nonlinear chemical process. The recurrent
structure enhances the prediction capability. In this study a quadruple tank model has been developed by using the proposed RNFN structure. The NFN differs from RNFN by the internal feedback layer.

3.3.3. **BP learning**

The network structure assumes the initial values of the parameters. The problem is to adjust these parameters in such a way that it acquires the required knowledge and produces solutions to the required classification problems. The types of classification problems of general interest are too complex to solve a priori by analytical techniques. Hence, it is necessary to develop an adaptive training algorithm that is driven by example data. If there are adequate features, number of processing elements and sufficient representative training data samples, the parameters will slowly adjust correctly through training.

They adjust in such a way as to end up with a set of network parameters that will give a satisfactory classification performance for other inputs that the network has not seen during training. This optimization can be achieved most effectively by adjusting the parameters to minimize the mean square error (MSE) of the network outputs compared with desired responses. This can be very time consuming if it is necessary to compute the MSE of all the training pairs before the parameters can be incrementally adjusted once. The main idea behind BP -of-error learning is to adjust the parameters a little each time as a new random training input–output vector pair is presented to the network. This is done repeatedly until a satisfactory convergence occurs.

- **Step-by-step BP algorithm**

Initially, parameters of the network are chosen randomly, and then a BP algorithm computing the parameters and necessary modifications was implemented in the network. The algorithm consists of the following few steps:

- Feed-forward computation
- BP to the output layer
- BP to the hidden layer
- Parameter updates

The algorithm is repeated until the error value is sufficiently small.

3.3.4. **NFN and RNFN model estimation**

In the model estimation phase, the model parameters are estimated by minimizing the square of sum errors. The optimization problem is solved using parameter estimation, and new parameter values are found for the new model. The reason behind parameter learning is to tune the free parameters of the constructed network in an optimal manner. There are a number of training methods, such as the BP method (Morris et al., 1994), the conjugate gradient method (Frasconi et al., 1992), Levenberg–Marquardt optimization (Tsoi & Back, 1994) or methods based on genetic algorithms (Zhang & Morris, 1995b), available in literature. Among them, BP algorithm opts for training the neural network. BP algorithm is a common method of training networks. As the algorithm name implies the errors (and therefore the learning) propagate backwards from the output nodes to the inner nodes. BP usually allows quick convergence on satisfactory local minima for error in the kind of networks to which it is suited. To train the coefficients of the fuzzy inference layer, the BP algorithm Equation (10) is used.

\[ a_i(k + 1) = a_i(k) + \Delta a_i(k) = a_i(k) + \lambda \left( -\frac{\partial (e^2(k))}{\partial a_i(k)} \right), \]

where ‘\( a \)’ denotes the tuning parameter, \( e(k) \) is the error difference between the measured output and the target that is the desired output and learning rate is denoted by ‘\( \lambda \)’. Several values of ‘\( \lambda \)’ are considered and the one resulting in the smallest error on the testing data is adopted.

In the proposed model, output of the inference layer is fed back to the consequent part of the newly generated rule via delay units to the fuzzification layer; hence all the rules have memory elements separately to memorize the firing strength of past instants. One important point to note here is that parameter learning occurs concurrently with structure learning. In neural computation, this stage is usually called training of network. The internal parameters of the recurrent neuro fuzzy structure are trained using this BP algorithm. Often, the aim of the estimation is to minimize the sum of the error squares between the model output and the desired output. The cost function ‘\( e \)’ for the multi output case is defined as

\[ e(k) = \frac{1}{p} \sum_{k=1}^{p} (y_{sp}(k) - \hat{y}(k))^2, \]

where \( \hat{y}(k) \) is the model output, \( y_{sp}(k) \) is the desired output and ‘\( p \)’ denotes the number of output nodes. The weighting vector of the NFN/RNFN model is adjusted such that the error defined in Equation (11) is less than the desired threshold value following much iteration. The BP algorithm is used to train the network parameters recurrently.

3.3.5. **Model validation**

Validation ensures that the model meets its intended requirements in terms of the methods employed. The newly developed model must be validated. Normalized error (NE) is calculated.

\[ \text{Normalized error} = \frac{1}{N} \sum_{k=1}^{N} \frac{|y_{exp}(k) - \hat{y}(k)|}{y_{exp}(k)} \times 100, \]
Figure 3. Observed $h_1$.

Figure 4. Excitation input $u_1$.

Figure 5. Observed $h_2$. 
where $y_{\text{exp}}$ is the experimental (process) output, $\hat{y}$ is the NFN/RNFN model output and $N$ is the total data used for validation. The purpose of model validation is to make the model useful in the sense that the model addresses the problem exactly, yields accurate results about the system being modelled.

### 3.3.6. Multistep prediction of outputs

Identical inputs are given to the experimental process and the outputs from the NFN and RNFN models are stored. The outputs predicted from the RNFN model and obtained from the experiments are compared. The percent prediction error (PPE) is calculated (Henson & Seborg, 1997; Subathra et al., 2009).

$$PPE = \frac{\sum_{k=1}^{N} (y(k) - \hat{y}(k))^2}{\sum_{k=1}^{N} (y(k) - \bar{y})^2}, \quad (13)$$

where $\bar{y}$ is the mean value of the sequence $y(k)$. In this study, the model used is recurrent in nature; it can be used for multistep prediction of process outputs with great accuracy.

![Figure 6. Excitation input $u_2$.](image1)

![Figure 7. Comparison of outputs from experiments and the RNFN model.](image2)
### Table 2. Comparison of NE values for RNFN and NFN.

| Variable | RNFN model | NFN model |
|----------|------------|-----------|
|          | Estimation | Validation | Estimation | Validation |
| $h_1$    | 3.7756     | 1.9867    | 4.0246     | 7.3489     |
| $h_2$    | 2.6228     | 1.6468    | 4.9823     | 6.0386     |

4. Modelling of a quadruple tank using a recurrent network

4.1. Neuro fuzzy network modelling

Intelligent techniques also have some limitations such as poor decision-making capability of NN and lack of learning algorithms of FL. Such kind of problems can be overcome using hybrid soft computing techniques like neuro fuzzy networks (NFNs). Identification of the model is performed in a similar fashion to that performed for a two-tank heating system.

4.1.1. Model estimation and validation using the NFN model

Random excitation signals are given to both the manipulated variables (current inputs to drive 1 and drive 2) for both experimental and NFN model. The experimental and NFN model outputs are recorded. The data set is divided into training (estimation) and test (validation) sets. Among the 1000 data set, 700 are used for training and the remaining 300 are used for testing. In order to examine the fitness of the model most of the tests require a set of data that is not used in training. It is called validation set. It is necessary that the test set must satisfy the same demands as the training set regarding representation of the entire operating range. The NE is calculated using Equation (12) (Subathra et al., 2009; Subathra & Radhakrishnan, 2010). The estimation and validation results of NFN are shown in Figures 3–6 along with excitation inputs.

4.2. RNFN modelling

The same sets of random inputs in the manipulated variables used in the NFN modelling are given to the RNFN model. The estimation and validation plots are shown in Figure 7. The RNFN model output is compared with experimental outputs and the NE is calculated. From Table 2, it can be concluded that the recurrent model is endowed with lesser error. Among the two models studied in this investigation, the RNFN model gives good model accuracy.

Figure 8. Comparison of one-step ahead RNFN and NFN prediction models.
Table 3. Comparison of PPE for RNFN and NFN models.

| Model | \(h_1\) | \(h_2\) |
|-------|--------|--------|
|       | One step | Five step | One step | Five step |
| RNFN  | 0.03 | 0.14 | 0.02 | 0.07 |
| NFN   | 0.10 | 0.22 | 0.12 | 0.12 |

4.3. Prediction using the RNFN model

Long-range prediction models are obtained to design the multistep ahead predictive controllers. The heights of tanks 1 and 2 are predicted for one and five steps ahead using the NFN and the proposed RNFN model and the plots are shown in Figures 8 and 9. The model outputs are predicted for a random input sequence given to the process.

In this study, for the process variables \(h_1\) and \(h_2\), one-step and five-step prediction models are developed. The percentage prediction error is calculated using Equation (13) and the results are tabulated in Table 3. From the results given in Table 3, it can be seen that the proposed RNFN model can be used to develop multistep predictive controller outputs with good accuracy.

5. MPC design

The MPC approach uses a receding horizon principle and handles disturbances, constraints and model complexities inherently in its design. Therefore, it is more suited for process industries that are trying to optimize process performance in the presence of disturbances and constraints. The MPC design requires prediction of output using an approximate process model. Traditional MPCs use step response, impulse response or state-space models to predict the future changes based on current estimate of the disturbance. However, long range predictions with empirical models and a controller design that uses an empirical model are challenging.

A block diagram of an MPC that uses the proposed RNFN model is shown in Figure 10. The recurrent structure makes it possible to perform multi-step ahead predictions using the prior knowledge of the process operating conditions and process input–output data. Furthermore, model complexities such as non-linearity, inverse response and non-minimum phase behaviour can also be handled in the controller design inherently. This model is used for multi-step ahead prediction of the process outputs with more accuracy (Subathra & Radhakrishnan, 2010, 2011a). The prediction methodology used is illustrated in Figure 11.

The prediction obtained from the RNFN model is used as input to the control calculations where the set-points
Figure 10. Block diagram of model predictive control.

Figure 11. Comparison of servo response of RNFN-MPC with NFN-MPC.

are generated by the set-point calculations block depending on the operating mode. The predicted output and set-point changes are the input to the optimization routine (control calculations) block. Typically, the optimization objectives include maximizing a profit function or production rate or minimizing a cost function. The optimum values of set-points are changed frequently owing to varying process conditions. The MPC accounts for this and the set-point
trajectory is calculated each time when the control computations are performed. The control calculations are based on current measurements and predictions of future values of the outputs. The objective of the MPC control calculations is to determine a sequence of control moves so that the predicted response moves to the set-point in an optimal manner. The actual output is $y$, predicted output is $\hat{y}$ and the manipulated input is $u$. An optimization problem is formed that minimizes the magnitude of the future control moves. Therefore, the objective function is given by

$$J(k) = \sum_{p=N_1}^{N_2} \|y_{sp}(k+p|k) - \hat{y}(k+p|k)\|^2 + \lambda \sum_{p=0}^{N_2} \|\Delta u(k+p|k)\|^2,$$  

where $y_{sp}$ is the set-point value, $\hat{y}$ is the predicted output and $\lambda$ is a scalar. The $\lambda \geq 0$ defines in fact a ratio of the weight attributed to damping of the input moves versus the (unity) weight attributed to a reduction of the control errors. In this study, $N_1 = 1$; $N_2 = 5$; $N_u = 3$ and $\lambda = 1$. The control inputs that optimize the performance of the multivariable nonlinear system is the minimizer of the unconstrained optimization problem:

$$J(k) = \min_u \sum_{p=N_1}^{N_2} \|y_{sp}(k+p|k) - \hat{y}(k+p|k)\|^2 + \lambda \sum_{p=0}^{N_2} \|\Delta u(k+p|k)\|^2.$$  

The objective function in Equation (14) is a non-linear optimization problem and solving it using traditional optimization methods, for example, sequential quadratic programming, is difficult due to the absence of a mathematical model. Furthermore, implementing MPCs based on state-space models requires complex computations to be performed on embedded hardware and this makes the design complex. To overcome these computation difficulties and to facilitate implementation on dedicated hardware, this investigation uses the gradient descent method to find the optimal control input of the objective function in Equation (15).

### 6. Results and discussions

#### 6.1. RNFN model validation

In order to validate the RNFN and NFN models, manipulated variables (current inputs to drive 1 and drive 2) are varied randomly to both the process and the models. The output thus obtained are recorded. The data set is divided into training (estimation) and test (validation) sets. Among the 1000 data, 700 are used for training and the remaining 300 data for testing. The experimental outputs and RNFN model outputs and the excitation inputs are shown in Figures 6–9. The fitness function of the model is examined using a set of data that is not used when training. That data set is known as validation data. It is very important that the test set must satisfy the same demands as the training set regarding representation of the entire operating range. The results of both estimation and validation for both NFN and RNFN models are compared with the experimental results as shown in Figures 6 and 8. The NE is calculated using Equation (12). It can be concluded from Table 2 that the recurrent model is endowed with lesser error compared with the feed-forward model. Among the two models proposed in this investigation, it is found that RNFN gives good model accuracy than the NFN models.

#### 6.2. Comparison of NFN and RNFN model-based controllers

##### 6.2.1. Servo response

The RNFN-based model predictive control is designed and implemented in the quadruple tank process. Figure 11 gives the comparison between NFN and RNFN, with various reference trajectories. The step input changes with a positive magnitude of 4 cm at 1600 and the negative step change is given at 3400 s for $h_1$. The positive step changes with a magnitude of 4 cm at 1600 and the negative step change is given at 3400 s for $h_2$. The MPC will decide the future control action based on the error and set-point. The set-points for $h_1$ are 8-12-8 cm and for $h_2$ are 12-16-12 cm. Performance measure (MISE) of the developed controller is given in Table 4. From the comparative results, it can be seen that the time taken by the NFN and RNFN model-based controller to reach the new step change set-point is lesser.

##### 6.2.2. Regulatory performance

The process is subjected to an external disturbance after the level has reached a steady state.

A dozer pump, which has a high precision ejection, is used to inject an external step disturbance, which can be calibrated in terms of quantity of the nominal diaphragm pump discharge. The change in the level in tank 1 and tank 2 and also the response of the controllers are stored.
The disturbance in tank 3 and tank 4 is about 10% of the nominal flow.

The process is subject to an external disturbance after the level reaches steady state ($h_1$ is 12 cm and $h_2$ is 16 cm) and for the respective input, the changes in the process variables and also the response of the controller and its action to reject the disturbances are obtained and are shown in Figure 12. From MISE (Mean Integral Square Error) values shown in Table 4, one can infer that the developed RNFN-MPC has a better servo tracking performance than the NFN-MPC.

7. Conclusions

This investigation presented a new seven-layer RNFN structure for modelling a multivariable nonlinear process and an MPC design methodology using the proposed model. The modelling approach and controller design methodology were illustrated on a quadruple process.

In this study, the RNFN structure is used to model the process. Based on the open loop input–output data sets fuzzy technique is used to divide the entire operating regions and the NN is used for learning and training. The proposed model used the RNFN that combined the learning feature of ANN with the cognitive features of fuzzy systems. This led to a modelling framework that can be used to develop empirical models and it can use both input–output data and prior process knowledge. The introduction of memory elements between the fuzzy inference and fuzzification layer makes it possible to perform multi-step ahead predictions required for designing an MPC. In the RNFN model, the firing strength has been varied by changing the consequent part of the fuzzy rule in addition to the conventional RNFN structure, resulting in significant improvements in model accuracy. The obtained model was used to design an MPC and the use of empirical models necessitated new methods to solve the optimization problem. Gradient descent algorithm was used to obtain the control inputs that minimize the objective function or optimize the plant performance. As a result, the proposed MPC solution method does not require solvers for solving the quadratic optimization problem and therefore is more suitable for implementation in embedded platforms. Experimental studies on the quadruple process indicate that the proposed RNFN-based MPC performs better than the NFN-based MPC for both servo and regulatory performance. The effectiveness of the designed RNFN-MPC for the control of the quadruple tank process demonstrates that it can be used effectively for the control of multivariable interacting nonlinear processes. Implementation of the proposed MPC in dedicated hardware, studying the role of RNFN models in industries and studying the possibility of hardware-in-loop (HIL) simulation are the future prospects of this investigation.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Bhat, N. V., & McAvoy, T. J. (1990). Use of neural nets for dynamical modeling and control of chemical process systems. *Computers & Chemical Engineering, 14*, 573–583.

Bryds, M. A., & Kulawski, G. J. (1999). Dynamic neural controllers for induction motor. *IEEE Transaction of Neural Networks, 10*, 340–355.
Brown, M., & Harris, C. J. (1994). Neuro-fuzzy adaptive modeling and control. Englewood Cliffs, NJ: Prentice-Hall.

Bulsari, A. B. (Ed.). (1995). Computer-aided chemical engineering, Vol. 6, neural networks for chemical engineers. Amsterdam: Elsevier.

Doyle, F. J., Gatzke, E. P., Vadigepalli, R., & Meadows, E. S. (1999). Experiences with an experimental project in a graduate control course. Chemical Engineering Education, 4(33), 270–275.

Elman, J. L. (1990). Finding structures in time. Cognitive Science, 14, 179–211.

Frasconi, P., Gori, M., & Soda, G. (1992). Local feedback multilayered networks. Neural Computing, 4, 120–130.

Henson, M. A., & Seborg, D. E. (1997). Nonlinear process control. Upper Saddle River, NJ: Prentice-Hall PTR.

Jang, J. S. R. (1992). Self-learning fuzzy controllers based on temporal back propagation. IEEE Transaction on Neural Networks, 3, 714–723.

Jang, J. S. R., & Sun, C. T. (1995). Neuro-fuzzy modeling and control. Proceedings of the IEEE, 83, 378–406.

Jang, J. S. R., Sun, C. T., & Mizutani, E. (1997). Neuro-fuzzy and soft computing: A computational approach to learning and machine intelligence. Englewood Cliffs, NJ: Prentice-Hall.

Johansen, T. A., & Foss, B. A. (1993). Constructing NARMAX models using ARMAX models. International Journal of Control, 58(5), 1125–1153.

Johansson, K. H., & Nunes, J. L. R. (1998). A multivariable laboratory process with an adjustable zero. Paper presented at the 17th American Control Conference, Philadelphia, PA.

Juang, C. F., & Chen, J. S. (2001). A recurrent neural fuzzy network controller for a temperature control system. Proceedings of the 12th IEEE International Conference on Fuzzy Systems, I, 408–413.

Juang, C. F., Huang, S. T., & Duh, F. B. (2006). Mold temperature control of a rubber injection-molding machine by TSK-type recurrent neural fuzzy network. Neuro Computing, 70, 559–567.

Lia, C., & Cheng, K. H. (2006). Recurrent neuro-fuzzy hybrid-learning approach to accurate system modeling. Fuzzy Sets and Systems, 158, 194–212.

Montague, G. A., Tham, M. T., Willis, M. J., & Morris, A. J. (1992). Predictive control of distillation columns using dynamic neural networks. Proceeding of the 3rd IFAC Symposium on Dynamics Control of Chemical Reactors, Distillation Columns, and Batch Processes, Maryland, USA, pp. 231–236.

Morris, A. J., Montague, G. A., & Willis, M. J. (1994). Artificial neural networks: Studies in process modeling and control. Trans. IChemE, 72(pt. A), 3–19.

Scott, G. M., & Ray, W. H. (1993). Creating efficient nonlinear network process models that allow model interpretation. Journal of Process Control, 3(3), 163–178.

Su, H. T., McAvoy, T. J., & Werbos, P. (1992). Long-term prediction of chemical processes using recurrent neural networks: A parallel training approach. Industrial & Engineering Chemical Research, 31, 1338–1352.

Subathra, B., & Radhakrishnan, T. K. (2010). Modeling and control of multivariable process using intelligent techniques. Sensors and Transducers, 121(10), 68–79.

Subathra, B., & Radhakrishnan, T. K. (2011a). Modeling and control of MIMO nonlinear process using recurrent neuro fuzzy networks instrumentation science and technology. Instrumentation Science and Technology, 39(2), 211–230.

Subathra, B., & Radhakrishnan, T. K. (2011b). Recurrent neuro fuzzy and fuzzy neural hybrid networks: A review. Instrumentation Science and Technology, 40(1), 29–50.

Subathra, B., Raja Rao, N., & Radhakrishnan, T. K. (2009). A comparative study of conventional and recurrent fuzzy technique for chemical processes. Instrumentation Science and Technology, 37(6), 660–675.

Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its application to modeling and control. IEEE Transaction on System, Man, Cybernetics, SMC, 15, 116–132.

Tsoi, A. C., & Back, A. D. (1994). Locally recurrent globally feedforward networks: A critical review of architectures. IEEE Transaction on Neural Networks, 5, 229–239.

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. Proceedings of the IEEE, 78, 1550–1560.

Yager, R. R., & Filev, D. P. (1994). Essentials of fuzzy modeling and control. New York, NY: Wiley.

Zhang, J. (2003). Multi-objective optimal control of batch processes using recurrent neuro-fuzzy networks. Neural Networks. Proceedings of the International Joint Conference, 1, 304–309.

Zhang, J., & Morris, A. J. (1995a). Dynamic process modeling using locally recurrent neural networks: Proceedings of the American Control Conference (Vol. 4), Seattle, WA, pp. 2767–2771.

Zhang, J., & Morris, A. J. (1995b). Fuzzy neural networks for nonlinear systems modeling. Proceeding of the Institution of Electrical Engineers, Control Theory Applications, 142, 551–561.

Zhang, J., & Morris, A. J. (1999). Recurrent neuro-fuzzy networks for nonlinear process modeling. IEEE Transaction on Neural Networks, 10, 313–326.

Zhang, J., & Morris, A. J. (2000). Long range predictive control of nonlinear processes based on recurrent neuro-fuzzy network models. Neural Computing & Applications, 9, 50–59.