Use of multilayer feedforward neural networks in identification and control of Wiener model

H. Al-Duwaish
M.N. Karim
V. Chandrasekar

Indexing terms: Neural networks, Nonlinear system identification, Wiener model

Abstract: The problem of identification and control of a Wiener model is studied. The proposed identification model uses a hybrid model consisting of a linear autoregressive moving average model in cascade with a multilayer feedforward neural network. A two-step procedure is proposed to estimate the linear and nonlinear parts separately. Control of the Wiener model can be achieved by inserting the inverse of the static nonlinearity in the appropriate loop locations. Simulation results illustrate the performance of the proposed method.

1 Introduction

System identification and modelling is a very important step in control applications since it is a prerequisite for analysis and controller design. Due to the nonlinear nature of most of the processes encountered in many engineering applications there has been extensive research covering the field of nonlinear system identification [1]. One of the most promising and simple nonlinear models is the Wiener model which is characterised by a linear dynamic part and a static nonlinearity connected in cascade as shown in Fig. 1 [2, 3]. The signal between the two parts is not available for measurement. The identification of Wiener systems involves estimating the parameters describing the linear and the nonlinear parts from input–output data. The Wiener model has been used in many important applications including pH control [4], fluid flow control [5], identification of biological systems [6], and identification of linear systems with nonlinear sensors [7]. These examples show an apparent need for algorithms able to recover nonlinearities in systems of various kinds.

Previous literature relating to the identification of the Wiener model includes a correlation analysis method to separate the identification of the linear part from the nonlinearity [8]. The input signal is assumed to be white Gaussian noise with zero mean. In [9, 4], the Hammerstein model is used for the identification of the inverse of the Wiener model. The resulting algorithm requires that the nonlinearity is invertible and the linear part must be minimum phase, restrictive conditions in some applications. A method for approximating the nonlinearity by a piecewise linear functions is discussed in [10]. Some *a priori* knowledge about the nonlinearity must be known to define the approximating functions. Methods for identifying systems with known static nonlinearity are given in [11].

IEE Proceedings online no. 19960376

Paper first received 7th April 1995 and in revised form 1st February 1996

H. Al-Duwaish is with the Department of Electrical Engineering, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia

M.N. Karim is with the Department of Chemical and Bioresource Engineering, Colorado State University, Fort Collins, CO 80523, USA

V. Chandrasekar is with the Department of Electrical Engineering, Colorado State University, Fort Collins, CO 80523, USA

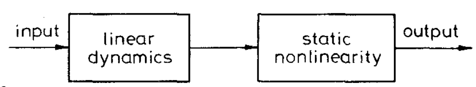

Fig. 1 *Wiener model*

In this study, we have developed a two-step method for identification of the Wiener model. In the first step, a small signal which ensures linear perturbation of the nonlinear system is applied to identify the linear dynamics using the recursive least square (RLS) algorithm [12]. This method of identifying the linear part using small-signal analysis is proposed in [4]. Once the linear part is identified the input signal is then increased and the backpropagation (BP) algorithm [13] is used to train a multilayer feedforward neural network (MFNN) to model the static nonlinearity. Control of the Wiener model has been studied and a method is proposed based on the cancellation of the nonlinearity using the nonlinearity inverse modelled by a MFNN.

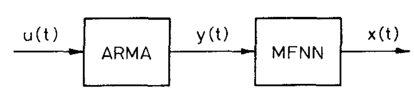

Fig. 2 *ARMA/MFNN identification method structure*

2 ARMA/MFNN model structure

The Wiener model is represented by the autoregressive moving average (ARMA) description. The MFNN model consisting of an ARMA model in series with a MFNN is shown in Fig. 2. The ARMA model is used to model the linear part and the MFNN is used to model the nonlinear part. The use of the MFNN is motivated by its ability to model any nonlinear function to any desired accuracy [12–14]. Considering single-input single-output systems, the output of the ARMA model is given by

$$y(t) = \sum_{i=1}^{n} -a_i y(t-i) + \sum_{j=0}^{m} b_j u(t-j) \quad (1)$$

where $u(t)$ is the input to the system, $y(t)$ is the output, and a_i ($i = 1, ..., n$) and b_j ($j = 0, ..., m$) are the parameters of the system.

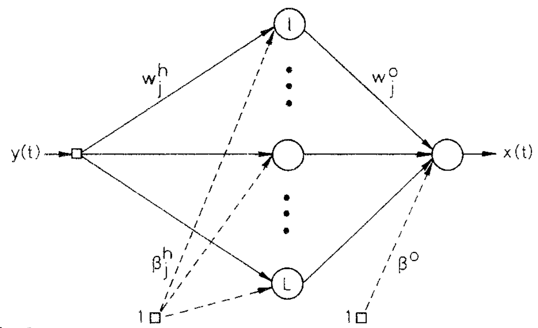

Fig. 3 *MFNN structure*

Fig. 3 shows the structure of the MFNN which is a feedforward network that is fully interconnected by layers. No feedback or bypass connections are used. Although only two layers are shown in Fig. 3 and used in the discussion, more than one hidden layer can be constructed. The input to the jth hidden unit is given by

$$net^h_j(t) = w^h_j y(t) + \beta^h_j \quad (2)$$

where w_j^h are the weights on the connection from the input unit, β_j^h are the weights on the connection from the bias unit, and $y(t)$ is the input to the network. The h superscript refers to the hidden layer. The output of the jth hidden unit is

$$z_j(t) = f^h_j(net^h_j(t)) \quad (3)$$

where f_j^h is the activation function. The equations for the output node are

$$net^o(t) = \sum_{j=1}^{L} w^o_j z_j(t) + \beta^o \quad (4)$$

$$x(t) = f^o(net^o(t)) \quad (5)$$

where the o superscript denotes the output, L is the number of neurons in the hidden layer, w_j^o are the weights on the connections from the jth hidden node to the output, β^o is the weight on the connection from the bias to the output, f^o is the activation function of the output node and $x(t)$ is the output of the network. The identification method involves the estimation of the weights of the neural network and the parameters of the linear system. This problem is solved by first estimating the linear part parameters using RLS algorithm with small input signals to ensure linear perturbation of the nonlinear system. The second step involves increasing the amplitude of the input signal and then estimating the weights of the MFNN from the outputs of the identified linear part and the outputs of the actual process using the BP learning algorithm.

3 Identification algorithm

The proposed identification algorithm is based on the well known BP and the RLS algorithms. First, the BP and the RLS algorithms are reviewed briefly; details may be found in [15, 16]. The BP is used to update the weights of MFNN and is derived based on the minimisation of the following error measure:

$$E_n(t) = \frac{1}{2}(x_d(t) - x(t))^2 \quad (6)$$

where $x_d(t)$ is the desired output and $x(t)$ is the output of the MFNN. The weights and biases of the output layer are updated according to the following equations:

$$w^o_j(t+1) = w^o_j(t) + \eta(o_d(t) - o(t))f'^o(net^o(t))z_j(t) \quad (7)$$

$$\beta^o(t+1) = \beta^o(t) + \eta(o_d(t) - o(t))f'^o(net^o(t)) \quad (8)$$

where η is the learning rate parameter, and

$$f'^o(net^o(t)) = \frac{\partial f^o}{\partial(net^o(t))} \quad (9)$$

Also, the weights and biases of the hidden layer are updated according to the following equations:

$$w^h_j(t+1) = w^h_j(t) + \eta\delta^h_j(t)y(t) \quad (10)$$

$$\beta^h_j(t+1) = \beta^h_j(t) + \eta\delta^h_j(t) \quad (11)$$

where

$$\delta^h_j(t) = f'^h_j(net^h_j(t))(x_d(t) - x(t))w^o_j \quad (12)$$

and

$$f'^h_j(net^h_j(t)) = \frac{\partial f^h_j}{\partial(net^h_j(t))} \quad (13)$$

Thus if the error at the output layer is known it can be propagated backward and used to update the weights on the hidden layers.

The initial weights and biases are usually selected as small random numbers. The motivation for starting from small weights is that large absolute values of weights cause hidden nodes to be highly active or inactive for all training samples, and thus insensitive to the training process. The randomness is introduced to prevent nodes from adopting similar functions.

The RLS is a standard procedure which has been derived to estimate the parameters of linear systems based on a set of input–output data [16]. The RLS is derived to minimise the following error measure:

$$E_c(t) = \frac{1}{N}\sum_{t=1}^{N} \lambda(t)(y_a(t) - y(t))^2 \quad (14)$$

where $y_a(t)$ is the actual output of the system, $y(t)$ is the estimated output, $\lambda(t)$ is the forgetting factor and N is the number of data points used in the identification. Now, $y(t)$ can be written as the output of a linear time-invariant system. That is,

$$y(t) = \psi(t)^T\theta(t-1) \quad (15)$$

where

$$\psi^T(t) = [-y(t-1)... - y(t-n)u(t)...u(t-m)] \quad (16)$$

is the data vector and

$$\theta^T(t-1) = [a_1...a_n \, b_0...b_m] \quad (17)$$

is the parameter vector. The parameters vector are updated according to the following recursive equations

$$\theta(t) = \theta(t-1) + \gamma(t)[y_a(t) - \psi^T(t)\theta(t-1)] \quad (18)$$

where the correcting vector $\gamma(t-1)$ is given by

$$\gamma(t-1) = \frac{1}{\psi^T(t)v(t-1)\psi(t) + \lambda(t)}v(t-1)\psi(t) \quad (19)$$

and

$$v(t) = \frac{[I - \gamma(t-1)\psi^T(t)]v(t-1)}{\lambda(t)} \quad (20)$$

To start the recursive algorithm one usually sets

$$\theta(0) = 0 \tag{21}$$

$$v(0) = \alpha I \tag{22}$$

where α is a large positive scalar. The convergence is usually fast. Discussions regarding identifiability problems and persistent excitation of the input signal can be found in [16].

The proposed identification algorithm for the Wiener model can be summarised in the following steps:

(i) Apply small input signal to the system being identified and record the output signal. Make sure that the amplitude of the input signal is small enough to ensure linear perturbation of the nonlinear system.

(ii) Use the RLS and the input-output data obtained in step (i) to estimate the parameters of the ARMA model.

(iii) Increase the amplitude of the input signal and apply it to the system being identified and record the output signal.

(iv) Apply the same input signal generated in step (iii) to the ARMA model identified in step (ii) to compute the signal between the linear and the static nonlinearity.

(v) The computed signal in step (iv), together with the recorded output of step (iii), can now be used to identify the static nonlinearity using the BP algorithm.

(vi) Terminate the training of the MFNN when an acceptable sum of square errors is achieved.

(vii) The parameters of the ARMA model obtained in step (ii) and the weights of the MFNN from step (vi) represent the ARMA/MFNN model.

4 Controller structure for Wiener model

Nonlinear Wiener systems represented by the ARMA/MFNN model can be controlled by a control loop structure, Fig. 4. Compensation for the nonlinear part of the model is possible in the controller by inversion of the MFNN in the model using another MFNN. This compensation will cancel the effect of the nonlinear MFNN and hence a linear controller can be designed for the ARMA model using well-known linear control theory.

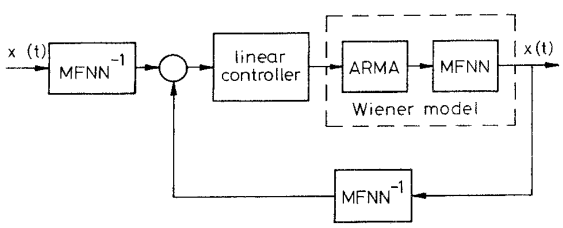

Fig.4 *Control loop structure for Wiener model*

5 Simulation results

In this example, the proposed identification method is applied to a model that describes a valve for control of fluid flow described in [10]. In this model, $u(t)$ represents the pneumatic control signal applied to the stem and $y(t)$ represents the stem position. The linear dynamics describe the dynamic balance between the control signal, a counteractive spring force and friction. The resulting flow through the valve is given by the nonlinear function $f_n(y(t))$ of the stem position.

$$y(t) = 1.4138y(t-1) - 0.6065y(t-2) + 0.1044u(t-1) + 0.0883u(t-2) \tag{23}$$

$$f_n(y(t)) = \frac{0.3163y(t)}{\sqrt{(0.10 + 0.90(y(t))^2}} \tag{24}$$

Now, the proposed identification method is applied to identify the linear and nonlinear parts of the model. To identify the linear part, an ARMA model structure similar to the process is used which is given by

$$y(t) = -a_1y(t-1) - a_2y(t-2) + b_0u(t) + b_1u(t-1) + b_2u(t-2) \tag{25}$$

The small signal analysis is used to identify the parameters of the ARMA model. A uniform random variable in the interval [–0.05, 0.05] is used as the identification input. The output of the system $f_n(y(t))$ is recorded. Using the RLS algorithm, the parameters of the ARMA model converged to 1.4137, –0.6065, 0, 0.1042, and 0.0882, respectively.

The second step is the identification of the nonlinear part which is done by increasing the amplitude of the input to a uniform random variable in the interval [–4.0, 4.0] and applying it to both the actual model of the process and to the identified linear part. The training data set of the MFNN consists of the output of the identified ARMA model as the input and the output of the actual system as the output. Using the BP algorithm, the actual and identified nonlinearity is shown in Fig. 5.

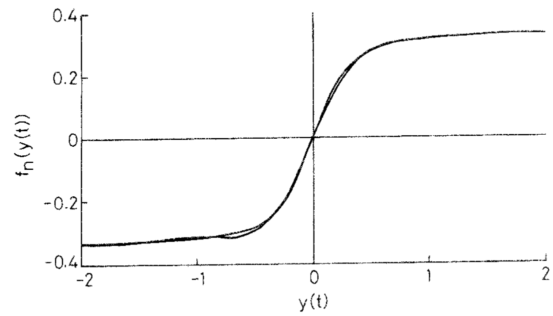

Fig.5 *Actual and identified nonlinearity*
——— actual
– – – – identified

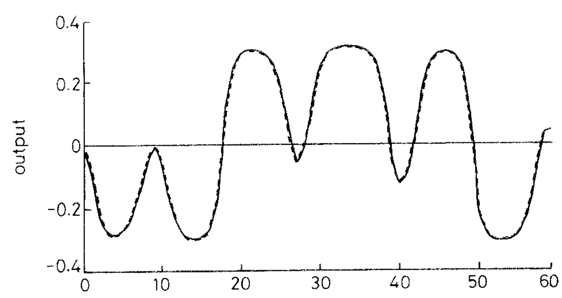

Fig.6 *Outputs of actual process and identified model*
——— actual
– – – – identified

To test the validity of the identified model, a test input given by $0.5\cos(t) + 0.5\cos(5t)$ is applied to both the actual and the identified models. The outputs of the two models are shown in Fig. 6, which shows very good agreement between the actual and the identified models. The performance of the 'best' linear model is compared with the actual model in Fig. 7 which indicates that linear modelling of this process will give poor results.

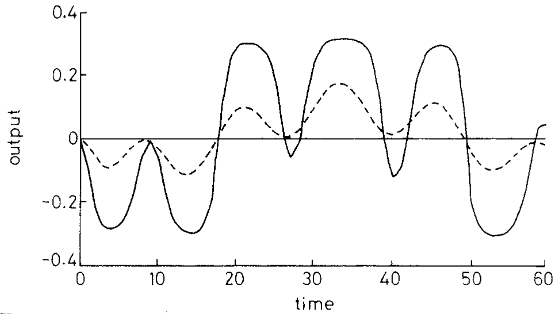

Fig. 7 *Outputs of actual process and 'best' linear model*
——— actual
– – – – best linear

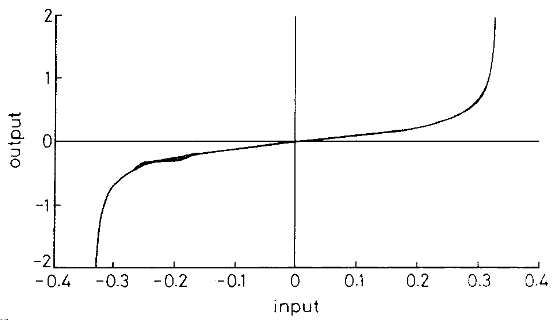

Fig. 8 *Actual and identified nonlinearity inverse*
——— actual
– – – – identified

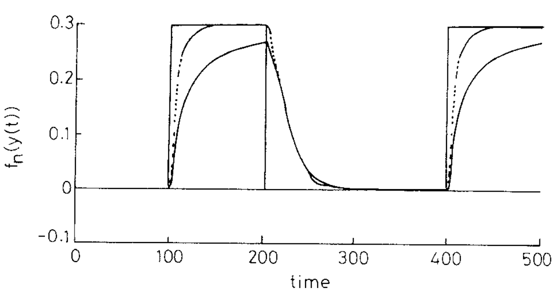

Fig. 9 *Step response with proposed and linear controllers*
——— desired output
– – – – proposed controller
——— linear controller

The control loop structure discussed in Section 4 is implemented. First, a MFNN is trained to model the nonlinearity inverse, the result is shown in Fig. 8. A linear controller is then designed for the linear part of the process. A step input of magnitude 0.3 is applied to the control loop structure of Fig. 4 and the response is shown in Fig. 9. The step response is compared to the performance of the linear controller without using the inverse of the nonlinearity. Fig. 9 shows that the performance of the proposed controller outperform the sluggish performance of the linear controller alone.

6 Conclusions

A new method for the identification and control of the nonlinear Wiener model has been proposed. The proposed identification method estimates the linear and nonlinear parts of the Wiener model separately. First, an input signal with small amplitude to ensure linear perturbation of the process is applied. The input–output data obtained in the first step is used with the RLS algorithm to identify the linear part of the process. Then, the amplitude of the input signal is increased and the identified linear part is used to compute the signal between the linear and nonlinear parts. The output of the linear part and the output of the Wiener model are used to identify the static nonlinearity using the BP algorithm. The main contribution of this method is the use of the MFNN to model the static nonlinearity of the Wiener model. Thus, systems with very violent nonlinearities can be identified. Also, this approach has the advantage of using standard tools from linear system identification and neural networks. Control of the Wiener model can be achieved by inserting the nonlinearity inverse in the appropriate loop locations. The inverse of the static nonlinearity is modelled by another MFNN. This approach is restricted to systems with invertible nonlinearities. Simulation results are included to demonstrate the effectiveness of the proposed identification and control algorithms.

7 References

1 BILLINGS, S.A.: 'Identification of nonlinear systems – a survey', *IEE Proc. D*, 1980, **127**, (6), pp. 272–285
2 BILLINGS, S.A., and FAKHOURI, S.Y.: 'Identification of a class of nonlinear systems using correlation analysis', *IEE Proc. D*, 1978, **125**, (7), pp. 691–697
3 BILLINGS, S.A., and FAKHOURI, S.Y.: 'Theory of separable processes with applications to the identification of nonlinear systems', *IEE Proc. D*, 1978, **125**, (9), pp. 1051–1058
4 PAJUNEN, G.: 'Application of model reference adaptive technique to the identification and control of Wiener type nonlinear processes'. Doctoral dissertation, Department of Electrical Engineering, Helsinki University of Technology, Helsinki, Finland
5 SINGH, M., ELLOY, J., MEZENCEV, R., and MUNRO, N.: 'Applied industrial control' (Pergamon Press, Oxford, UK)
6 HUNTER, I.W., and KORENBERG, M.J.: 'The identification of nonlinear biological systems: Wiener and Hammerstein cascade models', *Biol. Cybern.*, 1986, **55**, pp. 135–144
7 DOEBLIN, E.O.: 'Measurment systems' (McGraw Hill Int., Auckland, New Zealand, 1983)
8 BILLINGS, S.A., and FAKHOURI, S.Y.: ' Identification of nonlinear systems using the Wiener model', *Electron. Lett.*, 1978, **13**, (17), pp. 502–504
9 GREBLICKI, W.: 'Nonparametric identification of Wiener systems', *IEEE Trans.*, 1992, **IT-38**, (5), pp. 1487–1493
10 WIGREN, T.: 'Recursive prediction error identification using the nonlinear Wiener model', *Automatica*, 1993, **29**, (4), pp. 1011–1025
11 WIGREN, T.: 'Convergence analysis of recursive identification algorithms based on the nonlinear Wiener model', *IEEE Trans.*, 1994, **AC-39**, (11)
12 ITO, Y.: 'Representation of functions by superpositions of a step or sigmoid functions and their applications to neural network theory', *Neural Netw.*, 1991, **4**, pp. 385–394
13 HORNIK, K.: 'Approximation capabilities of multilayer feedforward networks', *Neural Netw.*, 1991, **4**, pp. 251–257
14 BLUM, E.K., and LI, L.K.: 'Approximation theory and feedforward networks', *Neural Netw.*, 1991, **4**, pp. 511–515
15 FREEMAN, J.A., and SKAPURA, D.M.: 'Neural networks: algorithms, applications, and programming techniques' (Addison -Wesley, 1992)
16 LJUNG, L., and SODERSTROM, T.: 'Theory and practice of recursive identification' (MIT Press)