Published in *Special Issue on Advances in Information Engineering, Engineering Letters*, **13**, (2), pages 185-194, 2006.

Modelling, simulation and proportional integral control of a pneumatic motor

R. Marumo and M.O. Tokhi

***Abstract*—Researchers have shown a considerable amount of interest in the control of pneumatic drives over the past decade, for two main reasons, firstly, the response of the system is very slow and it is difficult to attain set points due to hysteresis and secondly, the dynamic model of the system is highly non-linear, which greatly complicates controller design and development. To address these problems, two streams of research effort have evolved and these are: (i) using conventional methods to develop a modelling and control strategy, (ii) adopting a strategy that does not require mathematical model of the system. This paper presents an investigation into the modelling and control of an air motor incorporating a pneumatic equivalent of the electric H-bridge. The pneumatic H-bridge has been devised for speed and direction control of the motor. The system characteristics are divided into three regions, namely low speed, medium speed and high speed. The system is highly nonlinear in the low speed region, for which neuro-modelling, simulation and control strategies are developed.**

***Index Terms*—Modelling, neural networks, pneumatic motor, simulation.**

I. Introduction

Industrial processes, in general, require objects to be moved, manipulated or subjected to some force. The use of electrical equipment, such as DC motors, or mechanical equipment via devices driven by air (pneumatics) or liquid (hydraulics) normally achieves these tasks. Air motors are compact, lightweight sources of smooth vibration-less power. They start and stop almost instantly, and are not affected by continuous stalling or overload, and thus are suitable for intermittent operation. Air motors are relatively cheap, easy to maintain, and have the versatility of variable speed, high starting torque, are intrinsically safe in hazardous areas, and can operate in exceptionally bad environments. Detailed literature on the advantages of air motors over electric motors can be found in [1-3]. Since air motors do not require electric power, they can be used in volatile atmospheres. □□Air motors generally have high power density, so a smaller air motor can deliver the same power as its electrical counterpart. □□Unlike electric motors, many air motors can operate without the need for auxiliary speed reducers. Overloads that exceed stall torque generally cause no harm to air motors. With electric motors, overloads can trip circuit breakers, so an operator must reset them before restarting the equipment. In contrast to electric motors, which utilise expensive and complicated speed controls, speed of an air motor can be regulated through simple flow control valves. The motor torque can vary simply by regulating the input pressure. Air motors do not need magnetic starters, overload protection, or the host of other support components required by electric motors, and air motors generate much less heat than electric motors.

Manuscript received April 11, 2006.

R. Marumo was with the University of Sheffield. Department of Automatic Control and Systems Engineering. Sheffield. England. U.K. He is now with University of Botswana. Department of Mechanical Engineering. Private Bag UB0061. Gaborone. Tel: +267 3554307/4206. Fax: +267 3952309
e-mail: marumorr@mopipi.ub.bw

M.O. Tokhi is with the University of Sheffield. Department of Automatic Control and Systems Engineering. Sheffield. England. U.K.

A. Related work

Many attempts have been made to introduce simplified models in order to construct a model-based air motor controller [4]. A common method has been to approximate non-linear dynamics of the air motor into linear (ideal) models assumed to have sufficiently small uncertainty [5]. Studies on modelling of pneumatic systems, especially locally linearised modelling, can be found in the literature [6]. Linear and nonlinear dynamic models of an pneumatic actuator form the platform and launching pad points of the motion control algorithms of the air motor system in this study [7]. There are numerous researchers who have focused their efforts on different issues of modelling of pneumatic servo systems. The issues include but are not limited to the following: □ Air flow: a normal pneumatic valve does not behave like a simple nozzle. The mathematical model of the valve airflow must be produced specifying the flow capacity of the pneumatic fluid power valves. Valve modelling: there is little work found in the literature on this topic. The valve's input/output behaviour has significant influence on the servo control system. Analysis of pneumatic valve model parameters reveals that, the valve model contains two friction parts, namely static part and dynamic part. Friction parameters may be identified using evolutionary strategies [8, 9]. This paper addresses modelling, simulation and control of an air motor using neural networks. The rest of the paper is structured as follows: Section 2 provides a brief description of the experimental set up utilised in this study. Section 3 briefly describes the modelling approach. Section 4 discusses the

design and implementation of a PI control strategy for the air motor, and the paper is concluded in Section 5.

B. System set up

The control system for the air motor is shown schematically in Fig. 1. The personal computer (PC) with auxiliary hardware is used to source out and read all plant devices. All electrical devices are externally powered. Coding the control algorithm is straightforward. However, it is always advisable to consider factors such as realisation, actuator nonlinearities and computational delay to minimise controller sensitivity to errors.

The motor speed is measured by a shaft encoder, which represents the measured speed in terms of frequency. The frequency to voltage (F2V) converter transforms the frequency from the shaft encoder to a voltage in the range 0-5 V. This analogue voltage is then converted into binary form by an A/D converter, which the computer can then read. The control algorithm uses this measured speed along with other variables to generate a control signal. A D/A converter convert the control signal from binary into analogue voltage. This analogue voltage when applied to the pressure control valve (PCV) either increases or decreases the air pressure to the motor, thus controlling the speed of the motor.

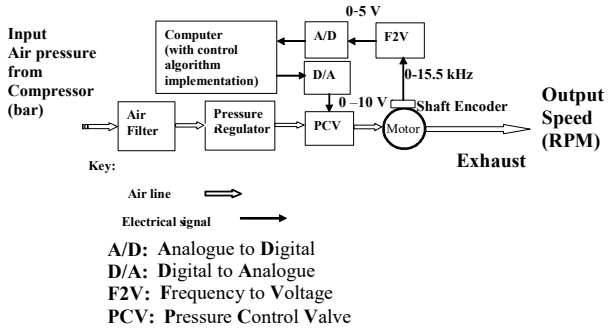


Figure 1 Schematic of an air motor system

II. SYSTEM IDENTIFICATION

System identification is one of the most fundamental requirements for many engineering and scientific applications. The objective of system identification is to find exact or approximate models of dynamic systems based on observed input and output data. Well developed techniques exist for parameter estimation of linear models and linear-in-the-parameters nonlinear models. Techniques for selection of structure and for non-linear-in-the-parameters estimation are still the subject of ongoing research. These input and output data can be obtained through experimental work, simulation or directly collected from the plant. The procedure for identifying a dynamic system consists of the basic steps as shown in Fig. 2.

Once a model of the physical system is obtained, it can be used for solving various problems such as, to control the physical system or to predict its behaviour under different operating conditions. A number of techniques have been devised by many researchers to determine models that best describe input / output behaviour of a system.

In many cases when it is difficult to obtain a model structure for a system with traditional system identification techniques, intelligent techniques are desired that can describe the system in the best possible way [10]. The air motor system incorporates a pneumatic H-bridge. The pneumatic H-bridge has been devised for speed and direction control of the motor. The main objective of this paper is to carry out modelling of the air motor using neural networks. The input (u) is the voltage signal initiated by compressed air while the output (y) is the rotor speed. A set of data is collected and divided into two halves, one set for training and the other for testing. It is important to use the test data for validation to ensure that the neural network model does replicate the input/output dynamic behaviour of the air motor system in general than memorise a specific data set.

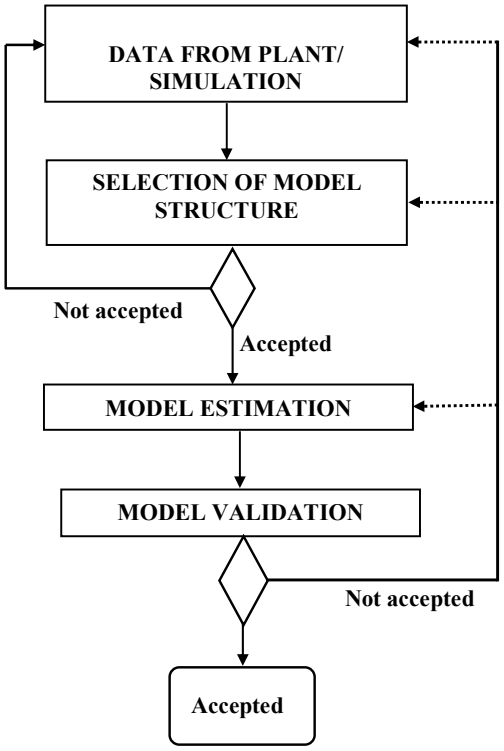


Figure 2 System identification procedures

A black box identification approach was adopted for modelling the system. This involved several tests using data obtained from a speed of 0 to 380 rev/min, termed the low speed region. There are a number of nonlinear models that are potentially suited to this problem. In this investigation, a neural network with input data structure of nonlinear autoregressive model with exogenous inputs (NARX) type, which provides a concise representation of a wide class of non-linear systems, is employed. The NARX model is also referred to in the literature by various other names such as one-step-ahead predictor or as

series-parallel model.

A. Parametric identification techniques

The air motor system falls into the class of many real world processes that are not amenable to mathematical modelling because of the following:

i. The process is too complex to represent mathematically
ii. Process model is difficult and expensive to evaluate
iii. There are uncertainties in process operation
iv. The process is non-linear, distributed, incomplete and stochastic in nature

The need to cope with significant un-modelled and unanticipated changes in the plant further complicates the control objectives. This will involve the use of advanced decision-making processes to generate control actions so that a certain level of performance is achieved and maintained even though there are drastic changes in the operating conditions. Given their inherent ability to approximate any non-linear continuous function without requiring any priori knowledge, neural networks (NNs) are remarkable choice, particularly for systems with un-modelled dynamics. Modelling of systems with non-linearities and little physical insight such as pneumatic drives is a domain of black box modeling, constituting universal approximating capabilities such as NN models. Models of dynamic systems are used for various purposes such as analysis, fault diagnostic and controller design.

Processes with complex non-linear characteristics, friction and or uncertainty in parameters make theoretical modelling difficult and sometimes even impractical. The theoretical approach applies fundamental interaction. A fundamental interaction is a mechanism by which particles interact with each other, and which cannot be explained by another more fundamental interaction. It results in a model description in terms of linear and/or non-linear differential equations, in general. For many technical plants physical knowledge is not sufficient for a successful theoretical or semi-physical modelling [11]. This is the case in pneumatic drives.

The system considered in this paper is a rotary vane air motor equipped with two servo-valves. Nonlinearities resulting from friction and thermodynamic laws concerning the state of the air are difficult to handle as analytical description and parameters are unknown. Theoretical modelling is not suitable for this approach [12]. The approach used here, is the full experimental technique called identification. Identification means computing the parameters of a given model structure assessing input/output data of the system considered. Various model structures are known. In general, linear models are easy to handle but will not yield satisfactory performance if the model validity is not restricted to small deviations from fixed operating point. Non-linear conventional methods such as state polynomial, e.g. bilinear or quadratic, require apriori knowledge for the choice of an appropriate model structure. In case of uncertain nonlinearities a mismatch between the nonlinearities of the model and process may result in large deviation between predictions and true values. In contrast, NN models present a rather general and flexible approach. They describe the input/output behaviour of the system using a set of weights. Such models can be interpreted as a weighted combination of several local models resulting in a non-linear global model. Hence the mismatch between the nonlinearities of local models and process is less significant compared with single non-linear model. Therefore, NN modelling has been applied especially to modelling tasks with uncertain nonlinearities, uncertain parameters and or high complexity.

B. Non-parametric identification techniques

Various modelling techniques can be used with neural networks to identify nonlinear dynamic systems. Nonlinear autoregressive moving average process with exogenous input (NARMAX) model (also known as error model) is one of them. Literature has revealed that, if the plant input and output data are available, the NARMAX model is a suitable choice for modelling nonlinear systems with suitable neuro-learning algorithms. The NARMAX model is mathematically expressed as:

$$\hat{y}(t) = f[(y(t-1), y(t-2), \cdots, y(t-n_y),\ u(t-1), u(t-2), \cdots, u(t-n_u),\ e(t-1), e(t-2), \cdots, e(t-n_e)] + e(t) \quad (1)$$

where: $\hat{y}(t)$, $u(t)$ and $e(t)$ represent the output vector determined by the past values of the system input and output, input vector and noise respectively, n_y, n_u, and n_e represent associated maximum lags respectively, $f(\cdot)$ represents the system mapping constructed with an NN such as multi-layered perceptron (MLP) together with appropriate activation function and learning algorithm.

If the model is good enough to identify the system without including the noise term, then it can be represented as NARX model and expressed as:

$$\hat{y}(t) = f[(y(t-1), y(t-2), \cdots, y(t-n_y),\ u(t-1), u(t-2), \cdots, u(t-n_u)] + e(t) \quad (2)$$

Fig. 3 gives a graphical representation of the NARX model identification structure using NNs.

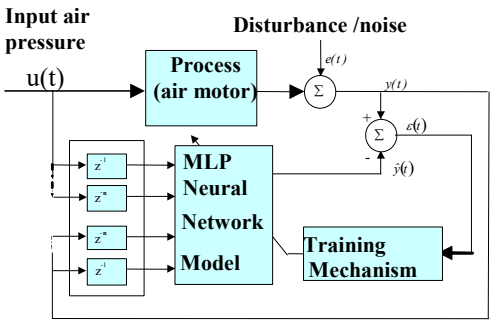


Figure 3 NARX Model identification structure

C. Model validation

A common measure of predictive accuracy of model used in system identification is to compute the one-step-ahead (OSA) prediction of the system output. From the model shown in Fig. 3, this can be expressed as:

$$\hat{y}(t) = f(u(t), u(t-1),\cdots,u(t-n_u), y(t-1),\cdots,y(t-n_y)) \quad (3)$$

where: $f(\square)$ represents a nonlinear function, u and y are the input and output samples respectively. The residual or error between the output and its prediction is given by:

$$\varepsilon(t) = y(t) - \hat{y}(t) \quad (4)$$

Often $\hat{y}(t)$ will be a relatively good prediction of $y(t)$ over the estimation set, even if the model is biased, because the model was estimated by minimising the prediction errors. Another method to evaluate the predictive capability of the fitted model is to compute the model predicted output (MPO). This can be expressed as:

$$\hat{y}_d(t) = f(u(t), u(t-1),\cdots,u(t-n_u), \hat{y}_d(t-1)\cdots,\hat{y}_d(t-n_y)) \quad (5)$$

$$\varepsilon_d(t) = y(t) - \hat{y}_d(t) \quad (6)$$

If only lagged inputs are used to assign network input nodes, then:

$$\hat{y}(t) = \hat{y}_d(t) \quad (7)$$

The implication that if the fitted model behaves well for OSA and MPO does not necessarily imply that the model is unbiased. The prediction over a different set of data often reveals that the model could be significantly biased. One way to overcome this problem is by splitting the data set into two sets, estimation set and test set. The first half (estimation set) is used to train the NN and the output computed. The NN usually tracks the system output well and converges to a suitable error minimum. New inputs (test set) are presented to the trained NN and the predicted output is observed. If the fitted model is correct, then the network will predict well for the prediction (test) set. In this case the model will have captured the underlying dynamics of the system. If both OSA and MPO of a fitted model are good over the estimation and prediction data sets, then most likely the model is unbiased. A more convincing method of model validation is to use correlation tests. If a model is adequate then the prediction error $\varepsilon(t)$ should be unpredictable from (uncorrelated with) all linear and nonlinear combinations of past inputs and outputs. This can be tested by means of the following correlation functions [13]:

$$\phi_{\varepsilon\varepsilon}(\tau) = E[\varepsilon(t-\tau)\varepsilon(t)] = \delta(\tau) \quad (8)$$

$$\phi_{u\varepsilon}(\tau) = E[u(t-\tau)\varepsilon(t)] = 0 \quad \forall\tau \quad (9)$$

$$\phi_{u^2\varepsilon}(\tau) = E[(u^2(t-\tau) - u^2(t))\varepsilon(t)] = 0 \quad \forall\tau \quad (10)$$

$$\phi_{u^2\varepsilon^2}(\tau) = E[(u^2(t-\tau) - u^2(t))\varepsilon^2(t)] = 0 \quad \forall\tau \quad (11)$$

$$\phi_{\varepsilon(u\varepsilon)}(\tau) = E[(\varepsilon(t)\varepsilon(t-1-\tau) - u(t-1-\tau)] = 0 \quad \tau \geq 0 \quad (12)$$

where $\phi_{u\varepsilon}(\tau)$ indicates the cross-correlation function between $u(t)$ and $\varepsilon(t)$, and $\varepsilon u(t) = \varepsilon(t+1)u(t+1)$ is an impulse function. All five tests defined in equations (8) to (12) should be satisfied if the $u(\bullet)$**'s** and $y(\bullet)$**'s** are used as network input nodes. In practice normalized, correlations are computed. In general, if the correlation functions in equations (8) to (12) are within the 95% confidence intervals, $\pm 1.96/\sqrt{N}$, where, N is the total number of data points, the model is regarded as satisfactory.

III. NEURAL NETWORK TRAINING

In this section neuro approaches for modelling the air motor are presented. Artificial NNs were first studied by a desire to understand and imitate the function of the human brain [14]. It has been recognised since early days that NNs offer a number of potential benefits for applications in the field of control engineering, particularly in modelling nonlinear systems. Some appealing features of NNs are: their ability to learn through examples; they do not require a priori knowledge and can approximate any arbitrary nonlinear continuous function well [9]. Many kinds of NNs have been proposed, developed and currently extensively used from varying standpoints. Amongst the most popular NNs are the multilayered perceptron (MLP), Elman recurrent network (ENN), radial basis function (RBF), Hopfield, cellular and adaptive resonance theory networks. In this investigation, MLP and ENN are used to model the air motor system. Artificial networks used in this study have been chosen after a series of simulations, experimentation and information from related work (literature review). It was found that they give better results. MLP and ENN give good results and faster convergence.

A. MLP network

Multi-layered perceptron NNs are extensively used in numerous applications including pattern recognition, prediction and control. An MLP is capable of forming arbitrary decision boundaries and representing Boolean functions [15]. The network can be made up of any number of layers with reasonable number of neurons in each layer, based on the nature of application. The layer, to which the input data is supplied, is called the input layer and the layer from which the output is taken is called the output layer. All other intermediate layers are called hidden layers. Layers are fully interconnected which means that each processing unit (neuron) is connected to every neuron in the previous and succeeding layers. However, the neurons in the same layer are not connected to each other. A neuron performs two functions of combining and activation.

Different types of function such as threshold, piecewise linear, sigmoid, tan sigmoid and Gaussian are used for activation. The most common learning algorithm used with MLP is the back propagation. The NN training may get stuck in a shallow local minimum with standard back propagation. In order to avoid entering the local minimum, the learning parameters, number of hidden neurons or initial values of the connecting weights could be changed. Also using Levenberg-Marquardt, which is a modified version of standard back propagation, can solve the shallow local minimum problem associated with standard back propagation.

B. Levenberg-Marquadt

While back propagation is a steepest descent algorithm, Levenberg-Marquardt algorithm is an approximation to Newton's method. Consider a function,

$$E(\underline{x}) = \sum_{j=1}^{N} e_j^2(\underline{x}) \tag{13}$$

and assume that it is required to minimise with respect to parameter vector $\underline{x}$. Then according to Gauss-Newton method the update would be:

$$\Delta\underline{x} = \left[J^T(\underline{x})J(\underline{x})\right]^{-1} J^T(\underline{x})e(\underline{x}) \tag{14}$$

where $J(\underline{x})$ is the Jacobian matrix. The Levenberg-Marquardt modification to the Gauss-Newton method is,

$$\Delta\underline{x} = \left[J^T(\underline{x})J(\underline{x}) + \mu I\right]^{-1} J^T(\underline{x})e(\underline{x}) \tag{15}$$

The parameter μ is multiplied by some factor (β) whenever a step would result in an increase in $E(\underline{x})$. When a step reduces $E(\underline{x})$, μ is divided by (β). Note that when is large the algorithm becomes steepest descent $\left(\text{with step } \frac{1}{\mu}\right)$ while for infinitesimal μ the algorithm becomes Gauss-Newton [16]. The Levenberg-Marquardt algorithm can be considered a trust region modification of the Jacobian matrix. For NN mapping problem the terms in the Jacobian matrix can be computed by a simple modification to the back propagation algorithm [17].

A neuro-model was set up with 15 hidden layer neurons with tansig activation function and a single saturated linear function in the output layer. The network training was carried out off-line using the Levenberg-Marquadt optimisation. The network used was of the prediction error type, so the algorithm essentially seeks to minimize the prediction error over the training data set. The network contained the following parameters:

- Levenberg-Marquardt back propagation training rule
- 15 hidden neurons
- Number of delayed plant input = 2
- Number of delayed plant output = 2
- Variable learning rate ('traingda' & 'traingd')
- logarithmic tang-sigmoid transfer function, tansig for the hidden neurons outputs
- Purelin linear transfer function for the output neuron
- Epochs (number of training data that NN has never seen during plant identification) = 500
- Maximum reference input value (upper bound on the random reference input training) = 5
- Minimum reference input value (lower bound on the random reference input training) = -5
- Maximum interval value (maximum interval over which the random reference input will remain constant) = 5 seconds
- Minimum interval value (maximum interval over which the random reference input will remain constant)= 1second
- Number of segments (segments the training data will be divided into) = 10
- Epochs (number of iterations per training segment) = 30

The system was excited with a pseudo random binary sequence (PRBS) signal, and the input/output data was recorded and used to train the network. Fig. 4 shows the algorithm convergence and fig. 5 the NN output tracking the plant output. It was noted that the fitted model behaved well for OSA and MPO. However, this does not necessarily guarantee that the model is unbiased. The prediction over a different set of data has to be carried out to ensure that the model could not be significantly biased. To overcome this problem, the best approach is to split the data into two sets, namely, estimation set and test set (prediction set). One half is used to train the NN and the output computed. This would reveal if the NN tracks the system output well i.e. converged to a suitable error minimum. Then, new inputs (another half that the NN has never seen) are presented to the trained NN and the predicted output observed. If the fitted model is correct, i.e. correct assignment of lagged input and output then the network will predict well for the prediction set.

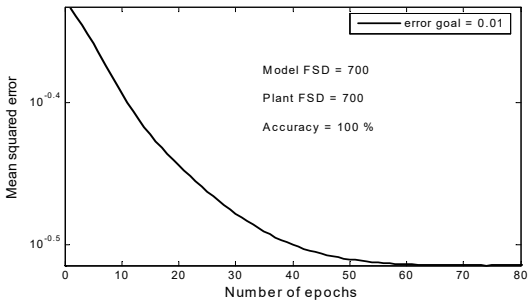


Figure 4 Mean square error

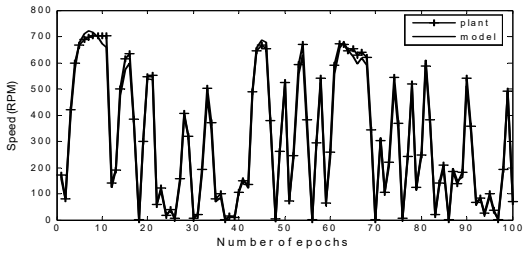


Figure 5 Actual and predicted output

Table 1 Model training results

Error type	NN type		
	ENN	MLP	RBF
ITAE	8.6550×10^{-15}	8.9858×10^{-15}	8.9725×10^{-15}
IAE	8.5250×10^{-15}	8.8850×10^{-15}	8.8850×10^{-15}
ISE	8.3250×10^{-15}	8.8255×10^{-15}	8.7850×10^{-15}
SSE	8.4550×10^{-15}	8.7350×10^{-15}	8.8756×10^{-15}
MSE	8.2250×10^{-15}	8.6723×10^{-15}	8.7122×10^{-15}

The model training results are shown in Table 1, where ITAE is the integral of time multiplied by absolute square error, IAE is the integral of absolute magnitude of the error, ISE is the integral of the square error, SSE is the sum square error and MSE is the mean square error.

Validation and cross validation consist of applying the training and test data to the neural identification model in order to see how closely it fits the experimental data from the air motor in each case. Figs. 6-10 show the five model validity correlation tests described by equations (8) to (12) for the developed NN model. It is important to note that only the first few lags are significant. Each lag in Fig. 6 is equivalent to a sample period, which was set to 0.55 seconds. It is noted that an adequate model fit was achieved.

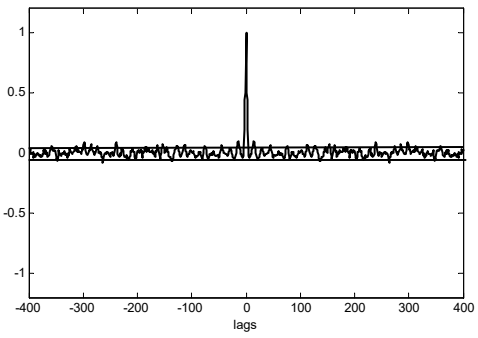


Figure 6 Auto-correlation of residuals

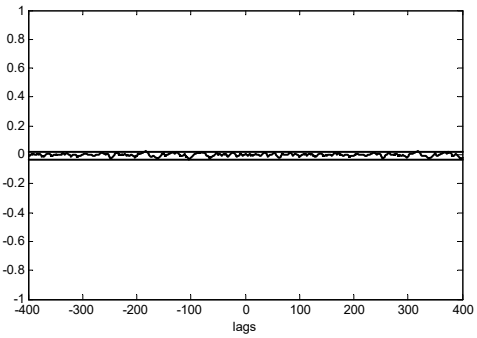


Figure 7 Cross-correlation of residuals and input

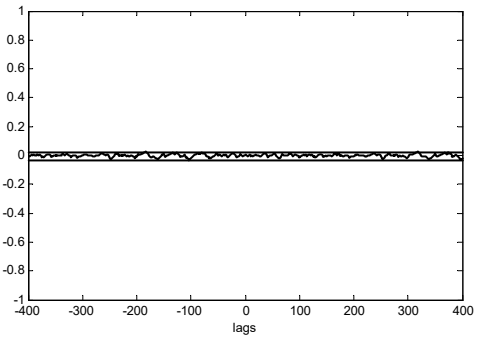


Figure 8 Cross-correlation of residuals and input square

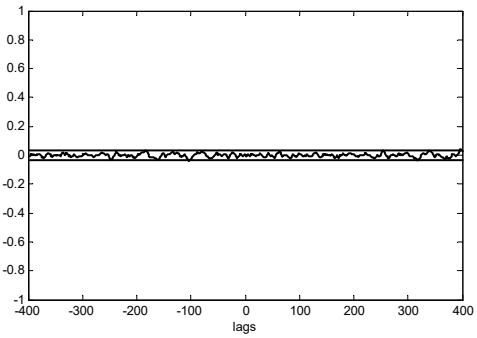


Figure 9 Cross correlation of residuals square and input square

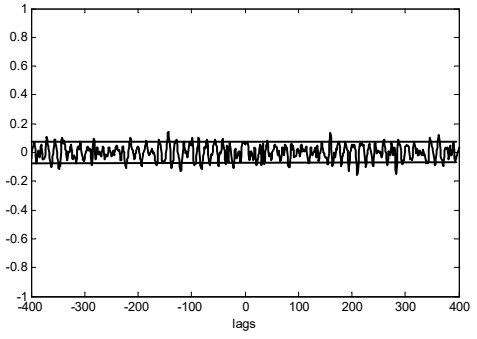
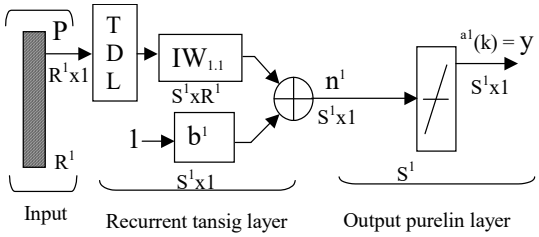


Figure 10 Cross-correlation of residuals and (input*residuals)

C. *Elman neural network*

Elman's network is a partially recurrent NN. The connections are mainly feed-forward but also include a set of carefully chosen connections that let the network remember cues from the recent past. As the feedback connections are fixed, back-propagation can be used for training of the feed-forward connections. Early NN research in language learning showed that, the network was both able to handle large amounts of data and provided evidence that abstract knowledge could emerge from statistical properties of a representative population of data and simple NN learning rule [18, 19]. The network is able to recognise sequences and also to produce short continuations of known sequences. Generalisation to new data sets arises from the spatial nature of the internal representation used by the network, allowing the new data sets to be encoded close to data sets that have already been learned in the hidden unit space of the network. The results are counter to the argument that learning algorithms based on weight adaptation after each data representation, cannot in principle extract symbolic knowledge from positive examples as prescribed by prevailing human linguistic and evolutionary psychology. This study has shown that Elman's network can learn to mimic an existing finite state automation where different states of hidden units will represent the internal states of the automaton. Fig.11 shows the structure of Elman network used for training in this study.



$$a^1(k) = \text{purelin}(WL_{1.1}a^1(k) = b^1)$$

Figure 11 Elman NN architecture

The symbols in Fig. 11 have the following definitions:

PR represents an $R \times 2$ matrix defining the minimum and maximum values of R inputs
TDL represents tapped delay lines
IW represents the new input weight matrix
Q represents the number of neurons in the layer
LW represents the new $Q \times R$ weight matrix
b represents a new $Q \times 1$ bias vector
n represents the number of network layers
y represents the network output

The generalisation capability of the Elman recurrent network is demonstrated in Fig. 12. In this case the network had 500 training epochs and 15 neurons in the recurrent layer. It is noted from the difference between neurons output error signal and the target output signals how well the NN generalises. Unfortunately, it is difficult to know off hand how large a network should be for a specific application [20]. Generalisation can be improved by other methods such as neuron regularisation and early stopping. The training strategy was implemented as follows: The entire input data set was presented to the network. Its outputs were calculated and compared with the target data to generate an error output. For each time step, the error was back propagated to find the gradients of errors for each weight and bias. The gradient was then used to update the weights with the back propagation training function chosen by the user.

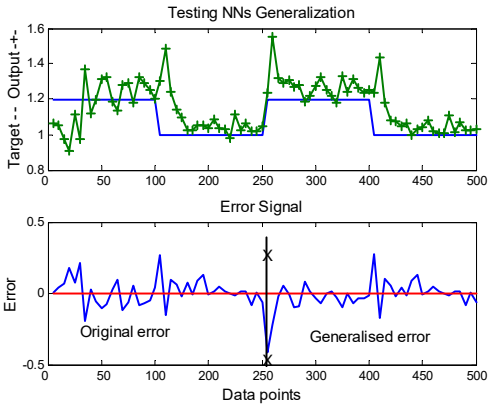


Figure 12 NNs showing generalisation capability

Table 2 Model training results

Error type	NN type	
	ENN	MLP
ITAE	8.6550×10^{-15}	8.9858×10^{-15}
IAE	8.5250×10^{-15}	8.8850×10^{-15}
ISE	8.3250×10^{-15}	8.8255×10^{-15}
SSE	8.4550×10^{-15}	8.7350×10^{-15}
MSE	8.2250×10^{-15}	8.6723×10^{-15}

A summary of the results obtained after training the NN models is as shown in Table 2, where ITAE is the integral of time multiplied by absolute square error, IAE is the integral of absolute magnitude of the error, ISE is the integral of the square error, SSE is the sum square error and MSE is the mean square error.

D. *Adaptive Neuro-Fuzzy Inference System*

In this section an adaptive neuro-fuzzy inference system (ANFIS) is utilised for modelling the system. This constitutes an NN architecture with fuzzy inference mechanism for processing the input data to the NN. As mentioned earlier, hardware-based velocity acquisition methods do not yet exist as high-volume products and high order differential equations

representing the dynamics of low speeds inhibit their analytical modelling. A natural alternative is to directly differentiate the existing velocity signal from a tachometer or optical pulse encoder output using the back-difference approximation:

$$\tilde{v}(k) = \frac{\tilde{v}(k) - \tilde{v}(k-1)}{T_s} \tag{16}$$

where: $\tilde{v}(k)$ represents the measured speed and T_s the sampling period. The back-propagation (BP) NN with the multi-perceptron structure is the most widely applied network model, due to its simple structure and computational algorithm with universal capability [21]. It has been proved that a BP NN can approximate any nonlinear function arbitrarily well [22]. A simplified BP NN with three layers, namely input, hidden and output layers is illustrated in Fig. 13. where $\left\{x_1^{(k)}, x_2^{(k)}, \cdots x_M^{(k)}\right\}$ and $\left\{y_1^{(k)}, y_2^{(k)}, \cdots y_L^{(k)}\right\}$ represent the input and output of the network at iteration k, M and L represent number of input and output nodes respectively. The learning algorithm of the BP NN utilises the well known gradient descent principle:

$$w^{(k+1)} = w^{(k)} + \Delta w^{(k)} \tag{17}$$

$$\Delta w^{(k)} = \alpha - \frac{\partial E^{(k)}}{\partial w^{(k)}} \tag{18}$$

where $E^{(k)}$ represents the sum squared error of the BP NN, which is usually defined as a quadratic function:

$$E^{(k)} = \frac{1}{2}\sum_{i-1}^{L}\left(Y_i^{(k)} - y_i^{(k)}\right)^2 \tag{19}$$

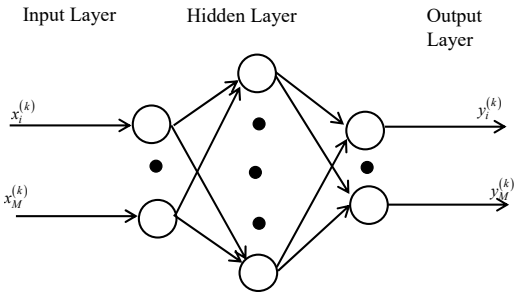


Figure 13 A three-layer neural network

The main drawback of the BP NN is the black box structure as well as slow convergence. Jang proposed a fuzzy NN model ANFIS [23, 24], the conceptual structure of which is shown in Fig. 14. Generally structure identification constitutes two problems. Firstly, to find input variables from a number of input candidates by a heuristic method based on experience and/or common sense knowledge. Secondly, to find input-output relations in the form of if-then rules. In a fuzzy model, the structure identification of this kind is stated in a different way. A fuzzy model consists of a number of if-then rules. The number of rules, N, in a fuzzy model corresponds to the order of the model in a conventional method. There are two parts in an if-then rule: the antecedent part and consequent part. The antecedent of a fuzzy rule defines a local fuzzy region, while the consequent describes the behaviour within the region via various constituents. The Takagi-Sugeno & Kang (TSK) model structure strategy with linear function as consequent and a non-linear function estimator was selected by fuzzy rules. The antecedents are similar to the Mamdani fuzzy system and the consequents can be any. Fig. 14 shows a schematic representation of such a network with three inputs, one output and three rules. The rules are in the following form:

R1: if x is A1 and y is B1 and z is C1 then f1
R2: if x is A2 and y is B2 and z is C2 then f2
R3: if x is A3 and y is B3 and z is C3 then f3

where (A1, A2…An, B1, B2…Bn, C1, C2… Cn) represent the input fuzzy sets (i.e. low, medium and high speed data respectively) and (f1, f2…fn) represent the output fuzzy sets. The nodes in the first layer compute the membership degree of the inputs in the antecedent fuzzy sets. The product nodes in the second layer represent the antecedent conjunction operator. In the consequent part, the fuzzy mean, the normalization node N and the summation operator are realised. By using smooth antecedent membership functions, such as the Gaussian functions, the following relationship can be applied:

$$uA_{ij}(x_j; c_{ij}, \sigma_{ij}) = \exp\left(-\left(\frac{x_j - c_{ij})^2}{2\sigma_{ij}}\right)\right) \tag{20}$$

The c_{ij} and σ_{ij} parameters can be adjusted by gradient-descent learning algorithms, such as back-propagation. This allows for a fine-tuning of fuzzy model to the available data for optimisation and prediction accuracy. The fuzzy output can be expressed as:

$$f = \frac{\bar{w}_1 f_1 + \bar{w}_2 f_2 + \bar{w}_3 f_3}{w_1 + w_2 + w_3} = \bar{w}_1 f_1 + \bar{w}_2 f_2 + \bar{w}_3 f_3 \tag{21}$$

where f_1, f_2 and f_3 are the outputs of the three sub-models, for the low speed, medium speed and high speed, of the air motor respectively.

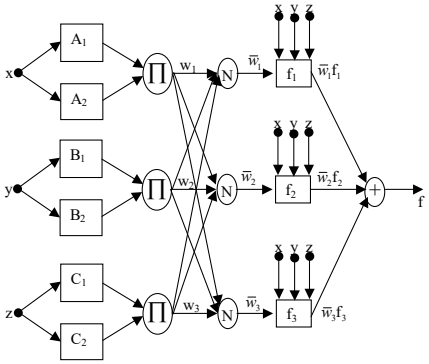


Figure 14 TSK-ANFIS structure

For training purposes, the input patterns (collected data) were normalized to unit length to ensure they fell within the required range of -1 to 1. Each network was trained on the profile of a normal event data set, split into training, verification and test set, to allow precise network prediction accuracy. Subsequently the networks were given a series of dataset that they had never seen before, to determine their arbitrary pattern generalisation and their ability to track the desired output. The first layer was designed to receive a set of input data for the first 500 data points (training data). The second layer received inputs ranging from 501 to 1000 data points (testing data). The first layer had tangsig output in the hidden layer and the second layer had a logsig transfer function in the hidden layer output.

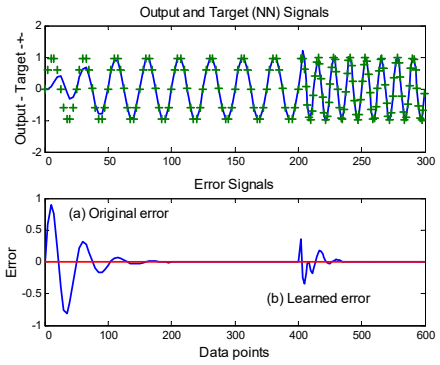


Figure 15 Adaptation using ANFIS

Fig. 15 shows the prediction capability of the ANFIS network in representing the system output. In this exercise a sampling interval of 55 milliseconds was selected. The input was chosen to be a PRBS shifting between -850 and -650 counts. These limits define boundaries of low speed region. Parameters of ANFIS network were: number of data points, 500; type of membership function, Gbell; number of membership functions 20 and number of epochs were up to 500.

IV. PI CONTROL OF THE AIR MOTOR

In this section a preliminary design of a PI controller for the air motor at low speed region is considered. The approach is adopted for more sophisticated control strategies in the future. Fig. 16 shows the control structure utilised, where for reasons of simplicity, the ANFIS model of the air motor developed in the previous section is used as a test bed simulating the system in the low-speed region.

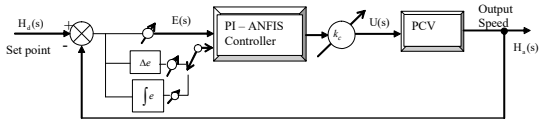


Figure 16 PI-ANFIS control system scheme

In Fig. 16, no mathematical model for the air motor system is available. The input-output data of the system is measured and the error and change in error derived, from which the integral sum of error obtained:

$$e(k) = H_d - H_a(k) \tag{22}$$

$$\Delta e(k) = e(k) - e(k-1) \tag{23}$$

$$\sum e(k) = \sum_{i=1}^{k} e(i) \tag{24}$$

where: H_d represents the desired speed, e represents the error, Δe represents the change in error and $\sum e$ represents the sum error of speed.

Fig. 17 shows the step response of the system and corresponding control signal achieved with and without anti-windup action. It is observed that the overshoot magnitude for the without anti-windup is 44%, the overshoot magnitude for 'with anti-wind up' is 11%. It is noted that response overshoot reduces significantly with anti-windup action.

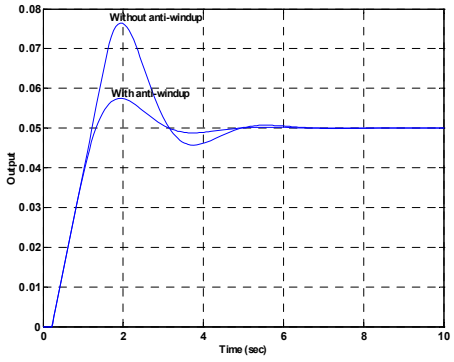


Figure 17 System response

As depicted in Fig. 18, in case of 'without anti-wind up, a limit of 33% was imposed in the control signal. When an integral action is used in a system with saturation the phenomenon of wind up will happen. When the integrator output is greater than the saturation level, then the system output is also limited and the error signal that is driving the controller cannot reduce. The integrator thinks it needs to do more and continues for a long time and cause the controller to behave badly. Thus, precautions have to be taken to limit the control signal so that the actuator is not damaged.

V. CONCLUSION

In this paper, nonlinear neuro-modelling approaches and a preliminary control system design for the low-speed region of an air motor have been presented. Three types of neural networks, namely, MLP with Levenberg-Marquardt back propagation, Elman network and ANFIS have been used to model the system. The developed models have been validated with various tests including OSA, MPO, estimation and test sets and correlation tests. Good dynamic prediction capability has been demonstrated with each of these model types, demonstrating the suitability of neural networks in modelling the air motor in the low speed region.

A preliminary design of a PI controller for the system has been carried out and tested within a simulation environment of the system. It has been demonstrated that the system can be controlled well within this speed region. However, precaution has to be taken against control signal not to go excessively high. The control approach adopted will be used for design of more sophisticated controllers in the future.

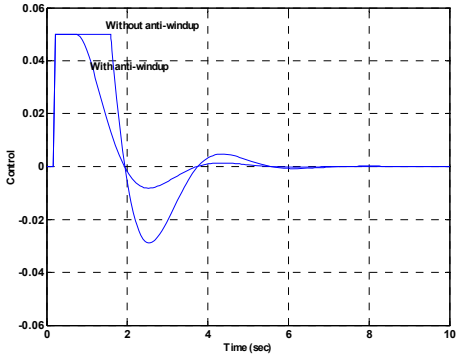


Figure 18 Control signal

REFERENCES

[1] Automation air motors, Air motor flow control system–development project (Automation Air motors Ltd., Barnsley, UK, 1998)

[2] A.L. Hitchcox, Performance insurance for air motors, Hydraulics & Pneumatics, 1995, 48, (10), pp. 63-68

[3] J. Mahanay, Gerotor air motor: new motion for low-speed output, Machine Design, 1986, 58, (3), pp. 75-77

[4] J.-S. R., Jang, C.-T. Sun, and E. Mizutani, Neuro-Fuzzy and Soft Computing, Prentice Hall, Upper Saddle River, NJ., 1997.

[5] M .O., Tokhi, I. N. Reynolds and M. Brisland, Real-time control of a radial piston air motor. IFAC World Congress, Barcelona, 21 – 26 July 2002.

[6] R. Marumo and M.O. Tokhi, Modelling and Control of a Pneumatic Motor, Proceedings of the First African Control Conference, December, Cape Town, South Africa pp. 100-112, 2003.

[7] R. Marumo and M.O. Tokhi, Modelling and Simulation of an Air Motor using Elman Neural Networks, Proceedings of the Fourth IASTED International Conference on Modelling, Simulation, and Optimization, August, Kauai, Hawaii, USA pp. 160-164, 2004.

[8] M. Sorli and S. Parstorelli, Performance of a pneumatic force controlling servo system: influence of valves conductance. Robotic and autonomous systems, 30, p 283-300, 2000.

[9] S.H. Choi, C.O. Lee and H.S. Cho, Friction compensation control of an electro pneumatic servo valve by using an evolutionary algorithm, Proc. of the Institute of Mechanical engineers, part I: Journal of systems and control engineering, vol. 214, No. 3, p 173-184, 200

[10] S.V.T Elanayar., and C.S. Yung, Radial basis function neural networks for approximation and estimation of non-linear stochastic dynamic systems. The IEEE Transaction on Neural networks, 1994, 5, (4), pp. 594-603

[11] R.W. Simnett, and E. Anderson, Air motor drives for small pumps, Chemical Engineering, 1983, 90, (25), pp. 73-75

[12] M. O. Tokhi, M. AL-Miskiry, and M. Brisland, Real-time control of air motors using a pneumatic H-bridge, Control Engineering Practice, 2001, 9, (4), pp. 449–457

[13] S.A Billings, & W. Voon, Correlation-based model validity tests for nonlinear models. International Journal of Control 44, 1986, 235-244

[14] M. L. Minsky, and S. Papert, Perceptrons, Cambridge, 1969, MA: MIT Press

[15] K. Hornik, M. Stinchmbe, and H. White, Multilayer feed forward networks are universal approximators, Neural Networks, 1989, 2, pp. 359-366

[16] M. O. Tokhi, M.H Shaheed and H. Poerwanto, Dynamic modelling of a flexible manipulator using multi-layered perceptron neural networks. Asia/Pacific International Congress on Engineering Computational Modelling and Signal Processing, Bandung, Indonesia, 24-26 November, 1999, pp. 185-194

[17] R Battiti, First and second order methods for learning: between steepest descent and Newton's method, Neural computation, 1992, 4, (2), pp141-166

[18] M. T. Hagan, M. B. Menhaj, Training feed forward networks with Marquardt algorithm. The IEEE Trans. on Neural Networks, 1994, 5, (6), pp. 989-993

[19] S.J. Hanson and J. Kegl, A connectionist network that learns natural language grammar from exposure to natural languages sentences, in Proceedings of the 9th annual conference on cognitive science, Seattle, WA, 1987

[20] J Elman, Finding structures in time, Cognitive science, 14, 1990, pp. 179-211
[21] H. Demuth and M. Baele Neural networks toolbox user's guide v.4 (The Math Works, Inc. 2000)
[22] S. Haykin, Neural Networks, a Comprehensive Foundation 2nd edition (upper Saddle River, NJ: Prentice-Hall), 1999
[23] J.S.R., Jang, C. T. Sun and E. Mizutani, Neuro-fuzzy and soft computing; a computational approach to learning and machine intelligence. Upper Sadle River: Prentice –Hall, 1997
[24] J.S.R., Jang, and C. T. Sun, "ANFIS: Adaptive-network-based fuzzy inference systems". IEEE Transactions on Systems, Man & Cybenetics, 1993, 23(3), pp. 665-685