ARCHIVES OF ACOUSTICS
29, 2, 259–273 (2004)

INTELLIGENT METHODS FOR ACTIVE NOISE AND VIBRATION CONTROL

M. O. TOKHI

The University of Sheffield
Department of Automatic Control and Systems Engineering
Sheffield, United Kingdom
e-mail: o.tokhi@sheffield.ac.uk

(*received 10 May 2004; accepted 13 May 2004*)

This paper presents an overview of intelligent soft computing techniques within the framework of active control of noise and vibration. Tools considered include genetic algorithms (GAs), neural networks (NNs) and fuzzy logic (FL). The paper highlights associated merits and potential benefits of the approaches in modelling and control of dynamic systems. These are demonstrated in the control of noise in free-field propagation and vibration suppression in 1D and 2D flexible structures. The paper shows that the potential benefits of the individual components can be exploited and approaches for design and development of hybrid soft-computing algorithms devised for modelling and control of dynamic systems. It is demonstrated that significant benefits in terms of performance can be gained with such hybrid algorithms.

Key words: Active noise control, active vibration control, fuzzy logic, genetic algorithms, neural networks, modelling of dynamic systems.

1. Introduction

A recent trend in research is towards producing machines with increasingly higher machine intelligence quotient (MIQ). The term soft computing (SC) has been introduced in conjunction with MIQ [1]. An increase in MIQ is achieved in two totally separate ways. On the one hand, there are machines based on classical logic and, thus, on precision, which are evolving towards faster and faster machines with an increasing degree of parallelism. On the other hand, new forms of logic such as fuzzy logic (FL), neural networks (NNs), and probabilistic reasoning comprising genetic algorithms (GAs), are being developed the strengths of which (by contrast) lie in their capacity to deal with uncertainty and imprecision. In the former case solutions are found for problems that are very precise and, consequently, have a high computational cost. This is referred to as hard computing. In the latter case, referred to as soft-computing, imprecise or uncertain solutions can be found at a much lower cost in terms of calculation

effort. There are a number of cases in which excessive precision is quite useless, so a non-traditional approach is preferable. In some problems, this is the only way because the computational complexity of a classical approach would be prohibitive.

Figure 1 shows, at least as far as FL, NNs, and GAs are concerned, how the various components of soft computing can be approximately ordered on a time scale and on a scale relating to their learning capacity, where the time scale is ordered according to the learning time. Fuzzy logic is not capable of learning anything. Neural networks and GAs, on the other hand, have such capability, although it can be said that, on average, pure GAs generally need longer learning times. From another viewpoint, the order is inverted. GAs do not need *priori* knowledge, NNs need very little and fuzzy logic at times needs quite detailed knowledge of the problem to be solved.

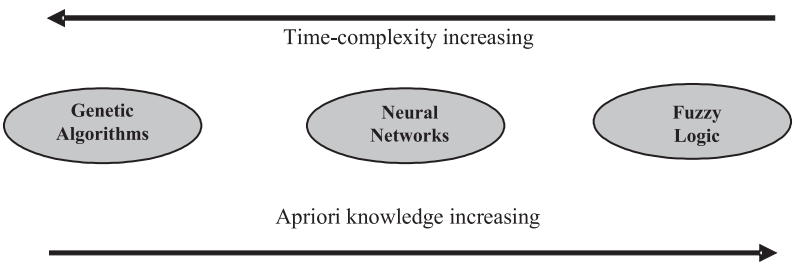

Fig. 1. Soft computing paradigm.

With reference to Fig. 1, each area of soft computing has its advantages and disadvantages. Fuzzy logic does not share the inherent concept of NNs, i.e., automatic learning. So it is impossible to use FL when experts are not available. It does, however, have significant advantage over the other two techniques. Expressed according to fuzzy canons, the knowledge base is computationally much less complex and the linguistic representation is very close to human reasoning. There are two reasons for using FL in an application. Firstly, in certain circumstances the definition of the problem is vague and uncertain. The information available does not lend itself readily to precise mathematical reasoning as in rule-based systems. A second class of applications is well defined but precise solution is not necessary; the tolerance for imprecision can be exploited to simplify the solution. Most applications of fuzzy logic today fall into the second category.

Neural networks are quite different, at least in the context of the typical features of gradient descendent learning networks. Firstly, they were conceived of specifically for learning. They are, therefore, fundamental when only some significant examples of the problem to be solved are available. There are two evident disadvantages in using NNs. In general, they can learn correctly from examples, but what is learned is not easy for humans to understand, i.e., the knowledge base extracted from them does not have such an intuitive representation as that provided, for example, by FL. Secondly, the types of functions that can be used in NNs have to possess precise regularity features and the derivative of these functions has to be known *a priori*.

Similar considerations as above hold for GAs, with certain clarifications. Their learning speed is usually slower. However, they have two great advantages over NNs. The functions that can be used in GAs can be much more general in nature, and knowledge of the gradient of the functions is not usually required. Finally, as these algorithms explore in several directions at the same time, they are much less affected than NNs by the problem of local extrema; that is, a GA has far less likelihood than an NN of finding a local extreme. Even if the extreme found is not a global one, it is likely to correspond to a less significant learning error.

Based on the above considerations, it is the opinion in this work that a technique, which makes use of a combination of soft computing features, i.e., GAs, NNs, and FL, would be a useful prospect. A hybrid technique, in fact, would inherit all the advantages, and not the less desirable features of individual soft computing components.

Although a large number of applications of soft computing have concentrated around the application of neural-fuzzy and neural-genetic techniques, the utilisation of the concept of FL to make a GA adaptive is limited. Moreover, the reported concepts do not exploit the reasoning capabilities of FL fully. Although GAs constitute powerful tools for solving difficult problems involving huge search spaces and are easy to implement, they usually require human supervision, for the features to be exploited successfully. It appears that FL techniques can help reduce the amount of human intervention needed to use GAs. Therefore, to evaluate and assess the performance of soft computing techniques, suitable combinations of GA, FL and NNs may be considered. In this manner, the characteristic features of each constituent component; i.e. reasoning capabilities of FL, learning capabilities of NNs and adaptation capabilities of GAs, can be utilised and exploited to achieve an optimal blend of soft computing components. Such features are investigated in this paper, and the performances of the proposed techniques are assessed at modelling and control levels within noise propagation and flexible structure environments. The platforms utilised include three dimensional (3D) noise propagation, flexible 1D and 2D structures.

2. Control structure

A schematic diagram of the geometric arrangement of a single-input multi-output (SIMO) active control structure is shown in Fig. 2a. The (unwanted) primary (noise/vibration) signal is detected by a detector (sensor), located at a distance r_e relative to the primary source and distance r_{fi} relative to secondary (cancelling) source i ($i = 1, ..., k$). The detected signal is processed by a SIMO controller of suitable characteristics and fed to a set of k secondary sources. The secondary signals thus generated interfere with the primary signal so that to achieve a reduction in the level of the noise/vibration at and in the vicinity of observation points j ($j = 1, ..., k$), located at distances r_{gj} relative to the primary source and r_{hij} relative to secondary source i, in the medium.

A frequency-domain equivalent block diagram of the control structure is shown in Fig. 2b, where, $\mathbf{E} = [e]$ is a 1×1 matrix representing the transfer characteristics of the path, through the distance r_e, between the primary source and the detector,

$\mathbf{F}$ is a $k \times 1$ matrix representing the transfer characteristics of the paths, through the distances r_{fi} $(i = 1, ..., k)$, from the secondary sources to the detection point, $\mathbf{G}$ is a $1 \times k$ matrix representing the transfer characteristics of the paths, through the distances r_{gj} $(j = 1, ..., k)$, from the primary source to the observation points, $\mathbf{H}$ is a $k \times k$ matrix representing the transfer characteristics of the paths, through the distances r_{hij}, from the secondary sources to the observation points, $\mathbf{M} = [m]$ is a 1×1 matrix representing the transfer characteristics of the detector, $\mathbf{L}$ is a $k \times k$ diagonal matrix representing the transfer characteristics of the secondary sources and $\mathbf{C}$ is a $1 \times k$ matrix representing the transfer characteristics of the controller;

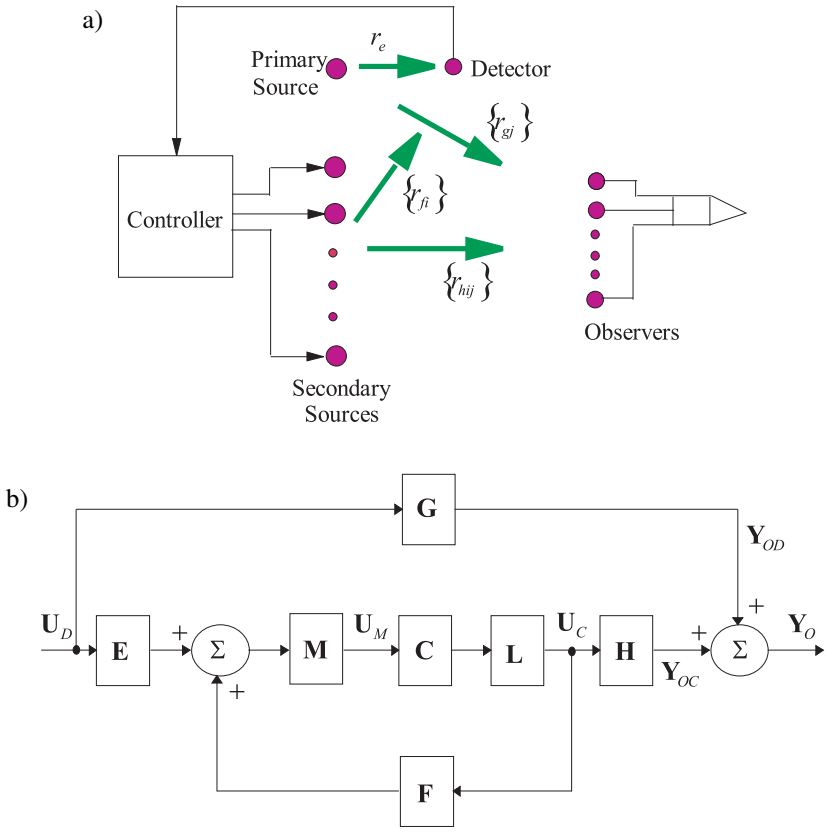

Fig. 2. Active feedforward control structure: a) schematic diagram; b) block diagram.

$$\mathbf{F} = [f_1 \;\; f_2 \;\; \cdots \;\; f_k]^T, \qquad \mathbf{G} = [g_1 \;\; g_2 \;\; \cdots \;\; g_k], \qquad \mathbf{C} = [c_1 \;\; c_2 \;\; \cdots \;\; c_k],$$

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1k} \\ h_{21} & h_{22} & \cdots & h_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ h_{k1} & h_{k2} & \cdots & h_{kk} \end{bmatrix}, \qquad \mathbf{L} = \begin{bmatrix} l_1 & 0 & \cdots & 0 \\ 0 & l_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & l_k \end{bmatrix}, \tag{1}$$

$\mathbf{U}_D$ is a 1×1 matrix representing the (primary) signal at the source, $\mathbf{Y}_{OD}$ is a $1 \times k$ matrix representing the primary signal at the observation points, $\mathbf{U}_C$ is a $1 \times k$ matrix representing the control (secondary) signals at the source, $\mathbf{Y}_{OC}$ is a $1 \times k$ matrix representing the control signals at the observation points, $\mathbf{U}_M$ is a 1×1 matrix representing the detected signal and $\mathbf{Y}_O$ is a $1 \times k$ matrix representing the observed signals.

The objective in Fig. 2 is to achieve full (optimum) cancellation at the observation points. This is equivalent to the minimum variance design criterion in a stochastic environment. This requires the primary and secondary signals at each observation point to be equal in amplitudes and have a phase difference of 180° relative to one another;

$$\mathbf{Y}_O = \mathbf{Y}_{OD} + \mathbf{Y}_{OC} = 0. \tag{2}$$

Using the block diagram in Fig. 2b and the design criterion in Eq. (2), the design rules for the controller to achieve optimum cancellation of the noise/vibration at the observation points can be expressed as [2]:

$$c_i = Q_i \left[\sum_{p=0}^{k} Q_p \right]^{-1}, \qquad i = 1, \ldots, k, \tag{3}$$

where,

$$Q_0 = \begin{vmatrix} q_{11} & q_{12} & \cdots & q_{1k} \\ q_{21} & q_{22} & \cdots & q_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ q_{k1} & q_{k2} & \cdots & q_{kk} \end{vmatrix}, \qquad Q_i = (-1)^i \begin{vmatrix} q_{01} & q_{02} & \cdots & q_{0k} \\ q_{11} & q_{12} & \cdots & q_{1k} \\ \vdots & \vdots & \vdots & \vdots \\ q_{(i-1)1} & q_{(i-1)2} & \cdots & q_{(i-1)k} \\ q_{(i+1)1} & q_{(i+1)2} & \cdots & q_{(i+1)k} \\ \vdots & \vdots & \vdots & \vdots \\ q_{k1} & q_{k2} & \cdots & q_{kk} \end{vmatrix},$$

q_{0j} $(j = 1, ..., k)$ represents transfer characteristics of system model between the detection point and observation point j when all the secondary sources are *off* and q_{ij} $(i = 1, 2, ..., k;\ j = 1, 2, ..., k)$ represents transfer characteristics of the system model between the detection point and observation point j when all the secondary sources are off except secondary source i. The design rule thus obtained can be realised with suitable estimation/learning algorithm, within either noise or vibration propagation media.

3. Genetic algorithms

The approaches described in this section constitute models and controllers in linear parametric form developed with GAs. Genetic algorithms form one of the prominent members of the broader class of evolutionary algorithms, inspired by the mechanism of natural biological evolution, i.e., the principles of survival of the fittest [3]. The operating mechanism of a GA can be described through the stages shown in Fig. 3. These comprise the following:

1. Creation of initial population: An initial population of potential solutions is created. Each element of the population is mapped onto a set of strings (the chromosome) to be manipulated by the genetic operators.
2. Evaluation and selection: The performance of each member of the population is assessed through an objective function imposed by the problem. This establishes the basis for selection of pairs of individuals that will mate during reproduction. For reproduction, each individual is assigned a fitness value derived from its raw performance measure, given by the objective function. This value is used in the selection to bias towards more fit individuals. Highly fit individuals, relative to the whole population, have a high probability of being selected for mating, whereas less fit individuals have a correspondingly low probability of being selected [4].
3. Genetic manipulation: Genetic operators such as crossover and mutation are used to produce a new population of individuals (offspring) by manipulating the "genetic information" usually called genes, possessed by the members (parents) of the current population. The crossover operator is used to exchange genetic information between pairs, or larger groups, of individuals. Mutation is generally considered to be a background operator. It introduces new genetic structures, which ensures that the search process is not trapped at a local minimum.

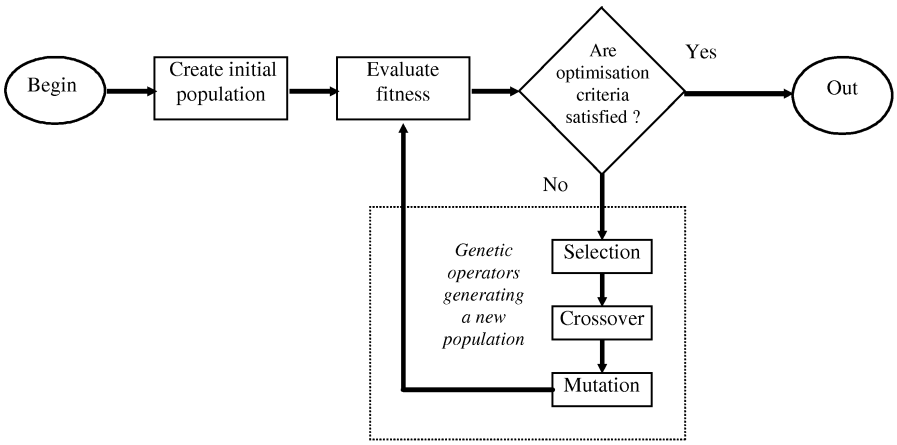

Fig. 3. Genetic algorithm-simple working principles.

After manipulation by the crossover and mutation operators, the individual strings are then, if necessary, decoded, the objective function evaluated, a fitness value assigned to each individual and individuals selected for mating according to their fitness, and so the process continues through subsequent generations. In this way, the average performance of individuals in a population is expected to increase, as good individuals are preserved and breed with one another and the less fit individuals die out. The GA is terminated when some criteria are satisfied, e.g., a certain number of generations completed or when a particular point in the search space is reached.

In this investigation, randomly selected parameters are optimised for different, arbitrarily chosen, order to fit to the system by applying the working mechanism of GAs as described above. The fitness function utilized is the sum-squared error between the actual output, $y(n)$, of the system and the predicted output, $\hat{y}(n)$;

$$f(e) = \sum_{i=1}^{n} \left(|y(n) - \hat{y}(n)|\right)^2, \tag{4}$$

where n represents the number of input/output samples. With the fitness function given above, the global search technique of the GA is utilised to obtain the best set of parameters among all the attempted orders for the system.

3.1. Control algorithms

Using the design rule in Eq. (3) an estimation learning algorithm with GAs can be formulated as follows:

Algorithm–1 (optimum disturbance reduction):

(i) Switch off all secondary sources, estimate transfer functions q_{0j} $(j = 1, 2, ..., k)$.
(ii) Switch on secondary source $i(i = 1, 2, ..., k)$, estimate transfer functions q_{ij} $(j = 1, 2, ..., k)$.
(iii) Use Eq. (3) to obtain the transfer function of the controller c_i $(i = 1, 2, ..., k)$.
(iv) Implement the controller, to generate the control signals.
(v) Measure system performance and compare with pre-specified index, if within specified range then go to (iv) otherwise go to (i).

Note that adaptation in Algorithm–1 is initiated by the supervisory level control upon detection of degradation in system performance (due to a change in system characteristics). This means that the system transfer characteristics q_{0j} and q_{ij} are re-estimated and the controller re-designed according to Eq. (3). Note further that, to design the controller with k secondary sources, a total of $k(k+1)$ models are required to be estimated. This implies that with a large number of secondary sources the computational burden on the processor implementing the algorithm will be significantly high. This will have a corresponding impact on the requirements of the computational capabilities of the processor. Thus, it is important during the process of realisation of the controller, to make sure that the processor meets the on-line sampling requirements of the control scheme as well as provides adequate computing speed for the adaptation mechanism.

Algorithm–2 (direct GA optimisation):

An alternative to the strategy outlined in Algorithm–1 is to achieve adaptation of the controller characteristics on the basis of minimising the mean square of the measured signal (as error) at each observation point. With reference to Eq. (4) this will correspond to $\hat{y}(n)$ representing the observed signal and $y(n) = 0$ as the desired signal. The optimisation capabilities of GAs can be utilised in this respect to realise such a strategy.

3.2. Implementation and results

To evaluate the performances of Algorithm–1 and Algorithm–2 simulation environments characterising the governing dynamic behaviours of a cantilever beam and of a flexible plate were developed using finite difference methods. Details of development of these environments can be found in [2] and [5]. The flexible beam comprises an aluminium type cantilever beam of length $l = 0.635$ m and mass $m = 0.03745$ kg, in fixed-free mode, and the beam is discretised along its length into 20 equal-length sections. The flexible plate simulation environment, on the other hand, comprises an aluminium type 1×1 m^2 square plate clamped at all four edges, and the plate is discretised into 20×20 equal-size segments.

A single-input single-output (SISO) control structure was utilised with Algorithm–1 within the flexible plate simulation environment. Investigations were carried out using the GA optimisation with different initial values and operator rates. From the work carried out it was found that satisfactory results were achieved with generation gap of 0.9, crossover rate of 0.06, mutation rate of 0.01. The deflection model was observed with different orders. The best result was achieved with an order 10. The GA was designed with 100 individuals in each generation. The maximum number of generations was set to 1000. The algorithm achieved MSE levels of 0.0016 and 0.0030 for Q_0 and Q_1, respectively in the 1000-th generation. Figure 4 shows output prediction with GA modelling of the system. The system performance at the observation point with the uniformly distributed white noise input signal as observed at the observation point is shown in Fig. 5. It is noted that the spectral attenuation achieved at the first, second and third modes were 11.858 dB, 21.712 dB and 9.9845 dB, respectively.

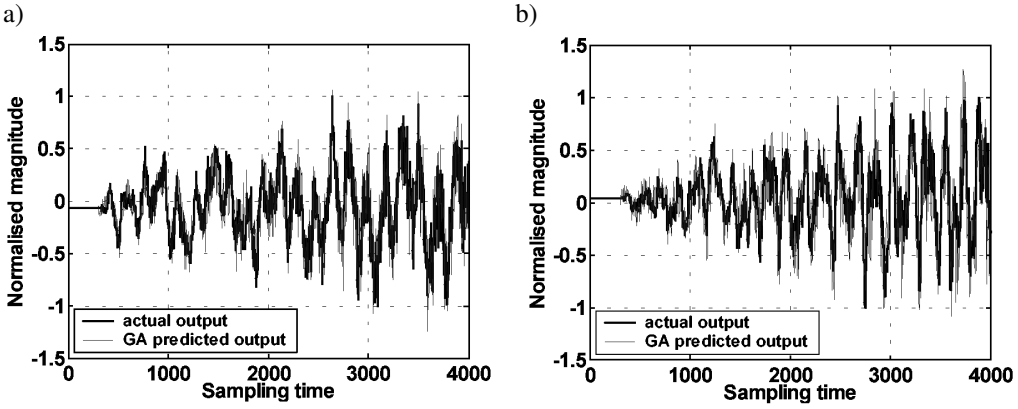

Fig. 4. Output prediction with GA characterisation – flexible plate: a) Q_0; b) Q_1.

The performance of the SISO active control system with Algorithm–2 was investigated within the flexible beam simulation environment. The performance as monitored at the observation point is shown in Figure 6. It is noted that a comparable level of cancellation to that with Algorithm–1 of the disturbance was achieved at the observation

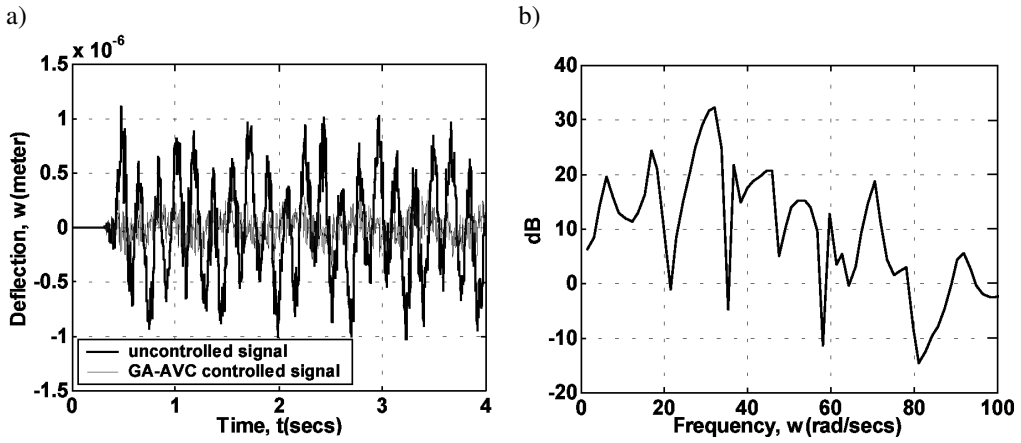

Fig. 5. Performance of the GA based SISO system at the observation point – flexible plate: a) time domain; b) spectral attenuation.

point. The advantage of Algorithm–2 is that it estimates the controller characteristics implicitly, bypassing estimation of the system model. However, the convergence of the algorithm could be slower.

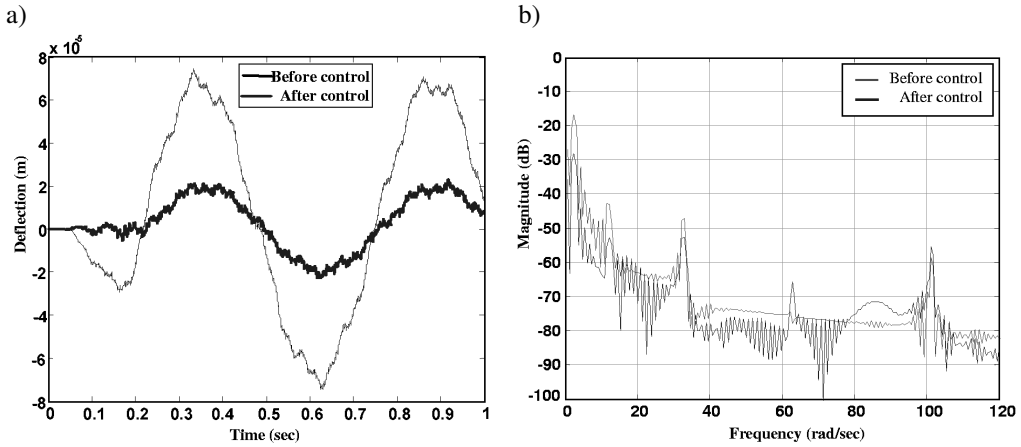

Fig. 6. Performance of the direct-GA based SISO system at the observation point – flexible beam: a) time domain; b) spectral attenuation.

4. Neuro and neuro-fuzzy control

In this section approaches based on NNs and combined NN and FL for control of noise and vibration are described. Various modelling techniques can be used with NNs to identify non-linear dynamic systems. These include state-output model, recurrent state model and Non-linear AutoRegressive Moving Average process with eXogeneous

input (NARMAX) model. It is evident from the literature that if the input and output data of the plant are available, the NARMAX model is a suitable choice for modelling nonlinear systems with suitable neuro-learning algorithms. Mathematically the model is described as [6]:

$$\begin{aligned}\hat{y}(t) \;=\; & f[(y(t-1),\, y(t-2),\, \cdots,\, y(t-n_y),\\ & u(t-1),\, u(t-2),\, \cdots,\, u(t-n_u),\\ & e(t-1),\, e(t-2),\, \cdots,\, e(t-n_e)] + e(t),\end{aligned} \tag{5}$$

where $\hat{y}(t)$is the output vector determined by the past values of the system input vector, output vector and noise with maximum lags n_y, n_u and n_e, respectively, $f(\cdot)$ is the system mapping constructed through multi-layer perceptron (MLP) or radian basis function (RBF) neural networks with an appropriate learning algorithm. The model is also known as NARMAX equation error model. However, if the model is good enough to identify the system without incorporating the noise term or considering the noise as additive at the output the model can be represented in a NARX form as [6, 7]:

$$\begin{aligned}\hat{y}(t) \;=\; & f[(y(t-1),\, y(t-2),\, \cdots,\, y(t-n_y),\\ & u(t-1),\, u(t-2),\, \cdots,\, u(t-n_u)] + e(t).\end{aligned} \tag{6}$$

This is shown in Fig. 7. Multi-layer perceptron NNs with backpropagation learning algorithm [6, 8, 9] are used in this work at modelling and control levels.

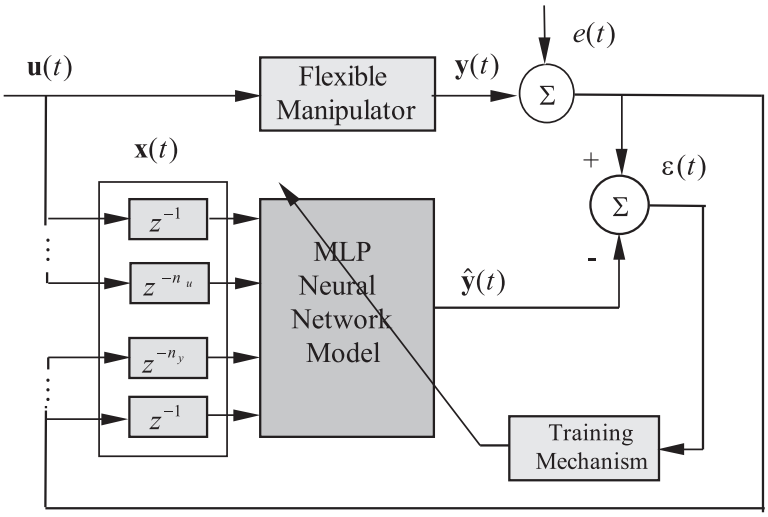

Fig. 7. NARX model identification with neural networks.

Adaptive neuro-fuzzy inference system (ANFIS) represents a combined NN and FL paradigm which takes advantage of the capabilities of FL for pre-processing the input

data to an NN. This hybrid combination allows to deal with both the verbal and the numeric power of intelligent systems, and as such it helps the NN to converge and learn more efficiently. As is known from the theory of fuzzy systems, different fuzzification and defuzzification mechanisms with different rule bases can propose various solutions to a given task [10].

4.1. Control algorithms

Considering a SISO active control structure, the controller design rule in Eq. (3) can be expressed as:

$$C = \left[1 - Q_1 Q_0^{-1}\right]^{-1}. \tag{7}$$

To allow nonlinear dynamics of the system be incorporated into neuro-training and design, the above can be realised through the training mechanism shown in Fig. 8. The corresponding control algorithm can thus be formulated as below.

Algorithm–3 (neuro/ANFIS approach):

(a) Switch off the secondary source, train an NN/ANFIS network to characterise the inverse of the system between the detection and observation point. This gives characterisation of Q_0^{-1}.
(b) Switch on the secondary source, train an NN/ANFIS network to characterise the system between the detection and observation point. This gives characterisation of Q_1.
(c) Train an NN/ANFIS network according to Fig. 8. This gives the required NN/ANFIS-controller.
(d) Implement the controller, to generate the control signal.
(e) Measure system performance and compare with pre-specified index, if within specified range then go to (d) otherwise go to (a).

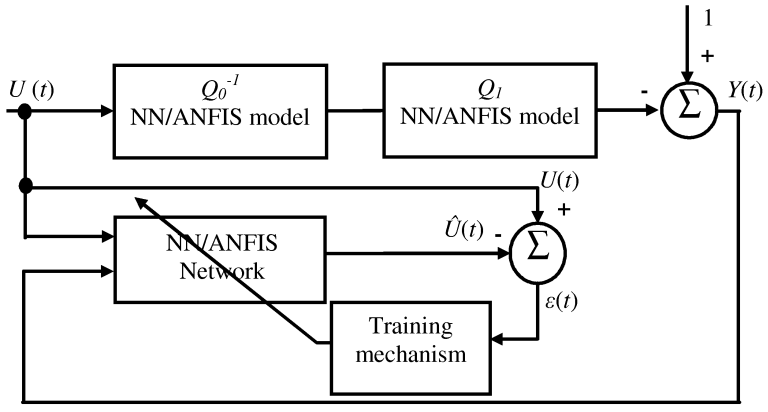

Fig. 8. Training the NN/ANFIS controller.

4.2. Implementation and results

The NN/ANFIS algorithm is evaluated and tested in this section within 3D noise propagation and 2D flexible structure environments. The noise propagation medium was constructed on the basis of physical properties of a free-field medium using measured data. Details of the development of the environment are found in [2]. The flexible structure environment constitutes the flexible plate structure described earlier in Sec. 3.

To test and verify the neuro-control strategy in noise cancellation, the noise propagation simulation environment characterising an active noise control (ANC) system was utilised and coded within MATLAB. A tansigmoid function, representing the nonlinear dynamics of the system was also incorporated into the simulation environment. A pseudorandom binary sequence (PRBS) simulating a broadband signal in the range $0 \sim 500$ Hz was utilised as the primary noise source. The amplitude of the signal was varied to excite the various dynamic ranges of the system. An MLP network incorporating two hidden layers each with nine neurons and $n_u = n_y = 9$ was trained to characterise Q_0^{-1}. A further MLP network incorporating two hidden layers each with four neurons and $n_u = n_y = 14$ was trained to characterise Q_1. Figure 9 shows the output prediction achieved with these networks. A further MLP network incorporating two hidden layers with 25 neurons in each layer and $n_u = n_y = 18$ was trained to characterise the controller according to the scheme in Fig. 8. The MLP neuro-controller thus obtained was implemented within the ANC system and its performance assessed. The performance of the system as monitored at the observation point is shown in Fig. 10. It is noted that an average level of cancellation of around 40 dB was achieved with the system.

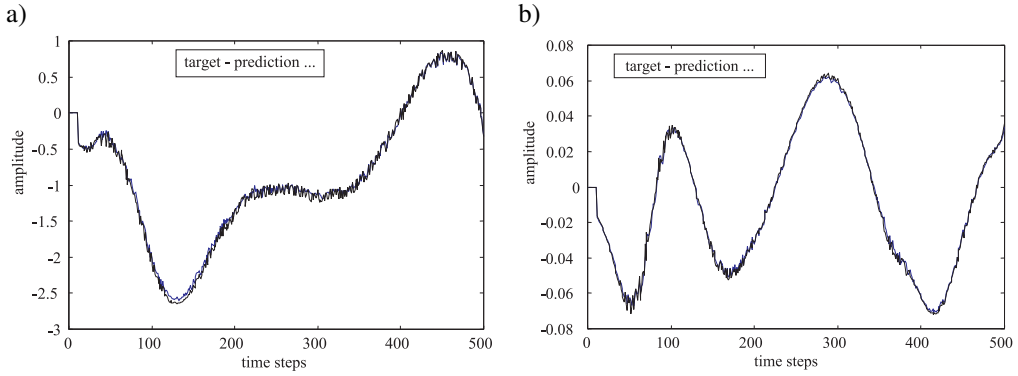

Fig. 9. Output prediction with MLP NN characterisation: a) Q_0^{-1}; b) Q_1.

To assess and verify the ANFIS algorithm, a uniformly distributed white noise was used as the primary source within the flexible plate environment. This type of input is chosen to ensure that the dynamic range of interest of the simulated plate system is captured. The ANFIS structure with first-order Sugeno model containing 36 rules was considered. Gaussian membership functions with product inference rule were used

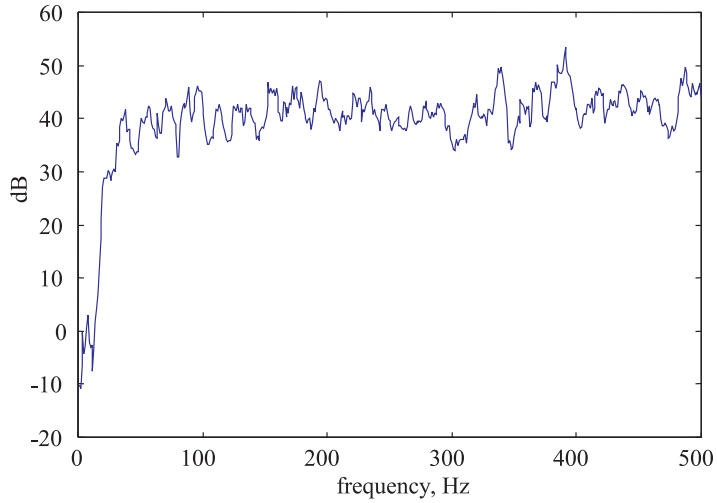

Fig. 10. Cancelled noise spectrum at observation point with MLP neuro-controller.

at the fuzzification level. The fuzzifier outputs the firing strengths for each rule. The vector of firing strengths is normalized. The resulting vector is defuzzified by utilizing the first-order Sugeno model. At the modeling level, the fuzzifier possessed two inputs, the rule base contained 36 rules and the defuzzifier had one output. Figure 11 shows the performance of the ANFIS models in characterising Q_0^{-1} and Q_1. It is noted that the network gave a very good output prediction with an MSE of 8.6723×10^{-15} and 8.2250×10^{-15} in characterising Q_0^{-1} and Q_1, respectively.

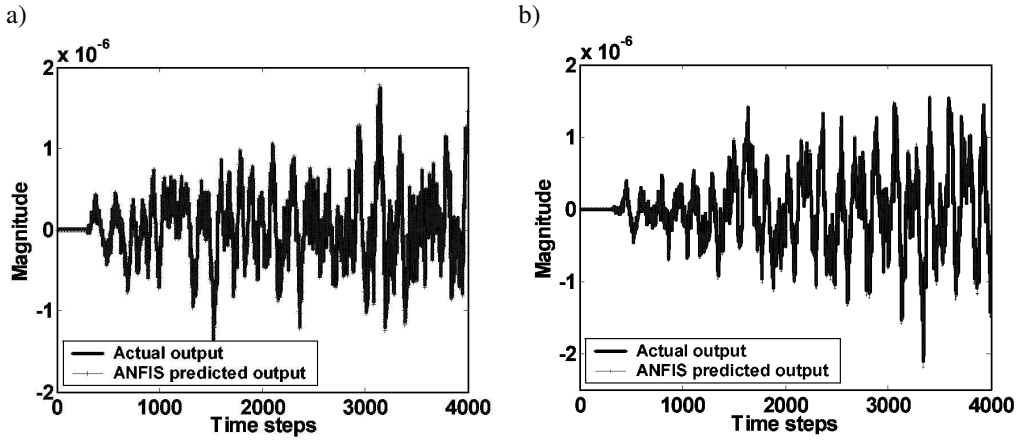

Fig. 11. Output prediction with ANFIS characterisation – flexible plate: a) Q_0^{-1}; b) Q_1.

To obtain the ANFIS controller, another first-order Sugeno model ANFIS structure containing 36 rules was considered. The bell-shaped membership functions with product inference rule were used at the fuzzification level. In this process the fuzzifier out-

puts the firing strengths for each rule. The vector of firing strengths is normalized. The resulting vector is defuzzified by utilising the first-order Sugeno model. To characterise the controller, an ANFIS network with the fuzzifier possessing two inputs, the rule base containing 36 rules and the defuzzifier comprising one output was trained. The ANFIS controller thus obtained was implemented within the system and its performance assessed in vibration reduction of the plate structure. Figure 12 shows the performance of the ANFIS active control system in suppressing the vibration of the plate at the observation point. It is noted that, with the uniformly distributed white noise input, the spectral attenuation achieved at the first, second and third modes were 15.2360 dB, 27.2721 dB, and 10.7773 dB, respectively. These as compared to the results in Sec. 3.1 are noticeably more than those achieved with GAs. Investigations have also shown that Algorithm–3 with ANFIS results in better performance than NN alone.

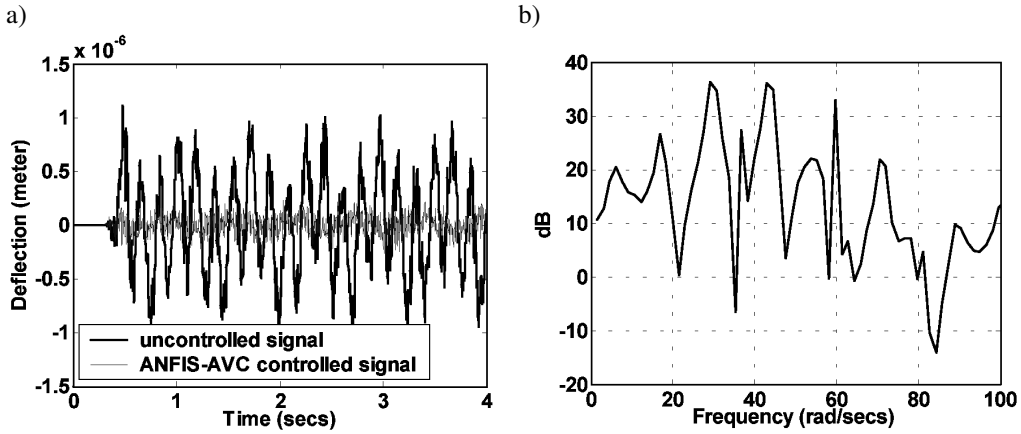

Fig. 12. Performance of the system at the observation point – flexible plate: a) time domain; b) spectral attenuation.

The above demonstrates that a suitable blend of soft computing components can lead to significant performance enhancement in the modelling and control of dynamic systems. Further hybrid paradigms reported in modelling and control of flexible manipulators include combined NNs, FL and GAs, where the NN is used for modelling and control purposes, the GA as a learning algorithm for the NN and FL in guiding the GA towards feasible solutions [11]. A further approach uses FL in a collocated control loop, GAs for optimisation of the rule base, and NN with GA within a non-collocated control loop [12]. It has been shown that with such approaches the performance of the control system enhances significantly.

A common problem in noise and vibration control applications arises due to non-minimum phase behaviour of the system model, where the controller design criterion requires inversion of the plant model, and hence unstable controller results. It has previously been demonstrated that such problems can be avoided with the soft-computing approaches investigated and verified in this paper [13].

5. Conclusion

An investigation into the development of intelligent soft computing algorithms for modelling and control of dynamic systems within the framework of active control of noise and vibration has been presented. It has been demonstrated that NNs, FL and GAs each have associated features that suit certain requirements of an application and accordingly potential benefits can be gained if such capabilities are matched with requirements of the application. In this respect an approach involving suitable blend of soft computing paradigms has been proposed that allows exploitation of potential benefits of the constituent components in such a manner that they complement one another.

References

[1] ZADEH, L. A., *Fuzzy logic, neural networks and soft computing*, Communications of ACM, **37**, 3, 77–84 (1994).

[2] TOKHI M. O., *Active adaptive control of noise and vibration*, Archives of Control Sciences, **13**, 2, 117–139 (2003).

[3] HOLLAND J., *Adaptation in natural and artificial systems*, University of Michigan Press, USA 1975.

[4] CHIPPERFIELD A. J., FLEMING P. J., POHLHEIM H., FONSECA C., *A genetic algorithm toolbox for MATLAB*, Proceedings of the International Conference on Systems Engineering, September, Coventry, UK 1994.

[5] MAT DARUS I. Z., TOKHI M. O., *Modelling of a flexible plate structure using finite difference methods*, Proceeding of the 2nd World Engineering Conference, Malaysia 2002.

[6] LUO F.-L., UNBEHAUEN R., *Applied neural networks for signal processing*, Cambridge University Press, New 1997. York

[7] SZE T. L., *System identification using radial basis neural networks*, PhD thesis, Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK 1995.

[8] MINSKY M., PAPERT S., *Perceptrons: An introduction to computational geometry*, MIT Press, Cambridge 1969.

[9] OMATU S., KHALID M., YUSOF R., *Neuro-control and its applications*, Springer, London 1996.

[10] JANG J. S. R., SUN C. T., MIZUTANI E., *Neuro-fuzzy and soft computing*, Prentice Hall, New Jersey 1997.

[11] SHARMA S. K., TOKHI M. O., *Neural network optimisation using genetic algorithm: A hierarchical fuzzy method*, Proceedings of AARTC00: IFAC Workshop on Algorithms and Architectures for Real-time Control, Palma de Mallorca (Spain), 15-17 May 2000, pp. 71–76.

[12] SIDDIQUE M. N. H., TOKHI M. O., *GA-based neuro-fuzzy controller for flexible-link manipulator*, Proceedings of IEEE International Conference on Control Applications, Glasgow, 18–20 September 2002, pp. 471–476.

[13] TOKHI M. O., SHAHEED M. H., *Adaptive neuro-inverse-dynamic active vibration control of a flexible manipulator*, Proceedings of International Ecological Congress, St. Petersburg, 14-16 June 2000, pp. 47–54.