Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER Applied Mathematics and Computation xxx (2005) xxx–xxx
APPLIED MATHEMATICS AND COMPUTATION
www.elsevier.com/locate/amc

Gas turbine diagnostics using a soft computing approach

Rajeev Verma [a], Niranjan Roy [b], Ranjan Ganguli [b,*]

[a] *Department of Mechanical Engineering, National Institute of Technology, Warangal 506004, India*
[b] *Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560012, India*

Abstract

A fuzzy system is developed using a linearized performance model of the gas turbine engine for performing gas turbine fault isolation from noisy measurements. By using a priori information about measurement uncertainties and through design variable linking, the design of the fuzzy system is posed as an optimization problem with low number of design variables which can be solved using the genetic algorithm in considerably low amount of computer time. The faults modeled are module faults in five modules: fan, low pressure compressor, high pressure compressor, high pressure turbine and low pressure turbine. The measurements used are deviations in exhaust gas temperature, low rotor speed, high rotor speed and fuel flow from a base line 'good engine'. The genetic fuzzy system (GFS) allows rapid development of the rule base if the fault signatures and measurement uncertainties change which happens for different engines and airlines. In addition, the genetic fuzzy system reduces the human effort needed in the trial and error process used to design the fuzzy system and makes the development of such a system easier and faster. A radial basis function neural network (RBFNN) is also used to

* Corresponding author.
E-mail address: ganguli@aero.iisc.ernet.in (R. Ganguli).

doi:10.1016/j.amc.2005.02.057

preprocess the measurements before fault isolation. The RBFNN shows significant noise reduction and when combined with the GFS leads to a diagnostic system that is highly robust to the presence of noise in data. Showing the advantage of using a soft computing approach for gas turbine diagnostics

Keywords: Soft computing; Fuzzy logic system (FLS); Genetic algorithm (GA); Genetic fuzzy system (GFS); Radial basis function neural network (RBFNN); Diagnostics

1. Introduction

Several researchers have proposed model based engine condition monitoring systems for gas turbine engines over the past few years. A recent review of some of this work is given in [1]. These systems were initially developed for predicting the long-term deterioration in gas turbine engines which occurs due to operating in a harsh aero-thermodynamic environment [2,3]. Because of the high levels on uncertainty in gas path measurements [4], researchers have tried to estimate the engine state from measurement deltas, which are deviations in the measurement from a baseline good engine. Since many older engines which are in service have limited instrumentation, with high levels of noise in the data, the fault isolation problem is a hard inverse problem and is difficult to address. While commercial software tools tend to use Kalman filter and weighed least square type approaches [5–8], researchers have also focused on soft computing based methods in recent years [9–12]. Soft computing encompasses genetic algorithms, fuzzy logic, neural networks and Bayesian networks among others and has emerged as a powerful approach in automated reasoning [13,14].

Recently, some work has also been directed at finding a fault in the engine once a measurement change in the form of a trend shift has been identified. This work is motivated by the realization that many engine faults are preceded by a sharp change in the measurement deltas and occur because of a fault in one module [15]. The isolation of these so-called "single faults" from gas path measurement deltas has been studied by neural network [15,16], Kalman filter [17] and fuzzy logic [18] based methods.

Fuzzy systems are also universal function approximations in a manner similar to neural networks [19]. However, fuzzy systems have the added advantage that they are expressed in linguistic terms that are easy to understand [20]. Several researchers [14,21–23] have used fuzzy logic in their work. Fuzzy systems also address the issue of uncertainty using a built in fuzzifier whereas a neural network learns the noise characteristics of the data through training. It has been shown that fuzzy systems provide accurate fault isolation results for gas turbine diagnostics [18]. However, the neural and fuzzy methods for diagnostics are highly configuration dependent, meaning that if the underlying

Nomenclature

EGT	exhaust gas temperature
FC	flow capacity
FP4	high-pressure turbine area
FP45	low-pressure turbine area
m	midpoint of fuzzy set
MAE	mean absolute error
$N1$	low rotor speed
$N2$	high rotor speed
N_{R}	noise reduction
T	set of terms
U	universe of discourse of fuzzy set
WF	fuel flow
x	elements of fuzzy sets
$\boldsymbol{x}$	design variables for GA
$\boldsymbol{y}$	module faults
z	measurement deltas
Δ	change from baseline "good" engine
η	efficiency
$\mu_A(x)$	degree of membership of x in fuzzy set A
σ	uncertainty as standard deviation
L	length of universe of discourse
N	number of fuzzy sets
$N^{(\max)}$	maximum number of fuzzy sets
N_{gen}	number of generations of GA
$N_{\mathrm{gen}}^{(\max)}$	maximum number of generations of GA

model used to obtain fault signatures or the measurement uncertainties of the signal changed, the diagnostic systems have to be redeveloped. Since there are many different engines operating with different airlines, there are likely to be many possible combinations of fault signatures and measurement uncertainties for the fault isolation systems which need to be developed. Very often the process of redeveloping the underlying numerics or rules for the diagnostic system is a trial and error process that can be very tedious and require considerable human effort.

Another way to address uncertainty in diagnostics systems is to filter the measurement deltas prior to fault isolation. Neural network [10,15] and median filter [23] based methods have been suggested as alternatives to the moving average and exponential average filters for gas path measurement deltas.

In this paper, we propose a genetic fuzzy system [25–27] that allows for easy development of the rule base for an engine given fault signature and measurement uncertainties. Unlike conventional fuzzy logic applications, where rules are generated based on operator's experience or general knowledge of the system in a heuristic way, in such a system, optimization techniques such as genetic algorithms are used to tune the fuzzy membership functions and rules. Typically, if Gaussian fuzzy sets are used, the number of fuzzy sets, their midpoints and standard deviations can be used as design variables. Genetic algorithms are used in the study to maximise the performance of a fuzzy system through automatically selecting the number of fuzzy sets and membership functions based on the fault signatures of the engine and measurement uncertainties to achieve the goal of minimizing the number of design variables. The genetic fuzzy system thus automates the creation of fuzzy system, greatly reducing the human effort needed. Furthermore, a radial basis function neural network (RBFNN) preprocessor is studied for denoising signals typical of path measurements. The advantages of using such a signal processing algorithm prior to fault isolation by a genetic fuzzy system is shown.

2. Gas turbine fault isolation

Consider a twin spool gas turbine with five modules: fan, low-pressure compressor (LPC), high-pressure compressor (HPC), high-pressure turbine (HPT) and low-pressure turbine (LPT). Most damages to the engine manifest themselves as changes in either the module efficiency or flow capacity/area. The FAN, LPC and HPC modules have efficiencies and the flow capacities associated with them, while the HPT and LPT modules have efficiencies and areas associated with them. The fingerprints or fault signatures relating a change in measurements deltas for four basic parameters with the faulty module is shown in Table 1 [18].

The four basic parameters are found in almost all engines and are exhaust gas term (EGT), low rotor speed ($N1$), high rotor speed ($N2$) and fuel flow

Table 1
Signature for module faults

Module faults	Measurement Deltas			
	ΔEGT (°C)	ΔN1 (%)	ΔN2 (%)	ΔWF (%)
FAN	−7.72	1.35	−0.59	−1.40
LPC	9.09	0.28	0.57	1.32
HPC	13.60	0.10	−0.11	1.60
HPT	21.77	0.15	−1.13	2.58
LPT	2.38	−1.96	1.27	−1.92

(WF). They are also called cockpit parameters as they are displayed to the pilot of a jet engine aircraft. The fault signatures in Table 1 assume the following couplings between module efficiencies and flow capacities [17]:

1. FAN Coupled FAN (−2% ., −2.5 FC)
2. LPC Coupled LPC (−2% ., −2.2% FC)
3. HPC Coupled HPC (−2% ., −1.6 FC)
4. HPT Coupled HPT (−2% ., −1.5 FP4)
5. LPT Coupled LPT (−2% ., +3.3% FP45)

Each fault is modeled as a 2% decrease in efficiency from the baseline "good" engine. Since the fault signatures are derived from influence coefficients, they are only approximately correct because they do not account for uncertainties in the measurement process. Each gas path measurement is associated with an uncertainty. One measure of this uncertainty is the standard deviations from revenue service data. As given in [17,18], typical standard deviations for ΔEGT, $\Delta N1$, $\Delta N2$, and ΔWF as 4.23 °C, 0.25%, 0.17% and 0.50%, respectively. These numbers are obtained from an analysis of airline monitoring data.

3. Neural signal processing

Since gas turbine measurements are often contaminated with noise and outliers, it is useful to perform a data cleaning function prior to fault isolation. In this study, we use a radial basis function neural network (RBFNN) for removing noise from simulated signals. Radial basis networks are an alternative to the more widely used multilayer perceptron networks trained using the backpropagation algorithm and take much less computer time for training [28–30].

The RBFNN model consists of three layers: an input layer, a hidden (kernel) layer and an output layer. The nodes within each layer are fully connected to the previous layer. The input variables are each assigned to a node in an input layer and pass directly to the hidden layer without weights. The hidden nodes or units contain the RBF, also called transfer functions.

An RBF is symmetrical about a given mean or center point in a multidimensional space. In the RBFN, a number of hidden nodes with RBF activation functions are connected in a feed forward parallel architecture. The parameters associated with the RBF's are optimized during training. These parameter values are not necessarily the same throughout the network nor are they directly related to or constrained by the actual training vectors. When the training vectors are presumed to be accurate, i.e. non stochastic, and it is desirable to perform a smooth interpolation between them, then linear combinations of RBF's can be found which give no error at the training vectors. The methods of fitting RBF's to data, for function approximation, are closely related to distance weighted regression. The RBF expansion for one hidden layer and an arbitrary RBF is represented by the equation

$$y_k(x) = \sum_{i=1}^{H} w_{ki} \exp(-\|c_i - x\|/\sigma_i^2),$$

where y_k = kth output, w_{ki} = weight from the ith kernel node to the kth output node, c_i = centroid of the ith kernel node, σ_i = width of the ith kernel node and H = number of kernel nodes. The parameters of the RBF w_{ki}, c_i and σ_i are commonly chosen by first selecting randomly or uniformly the c_i and then using singular value decomposition (SVD) to solve for w_{ki} and σ_i. This approach is not the most satisfactory. A better approach, suggested by Leonard et al. [28], involves using K-means clustering to determine the c_i, a K-nearest heuristic to determine the σ_i and multiple linear regressions to determine the w_{ki}. The K-means clustering algorithm finds a set of cluster centers and a partition of the training data into subsets. Each cluster center is then associated with one of the H kernels or centers in the hidden layer. After the centers are established the width of each kernel is determined to cover the training points to allow a smooth fit of the desired network outputs.

4. Fuzzy logic system

A fuzzy logic system (FLS) is a nonlinear mapping of an input feature vector into a scalar output [18]. A typical FLS maps crisp inputs to crisp outputs using four basic components: rules, fuzzifier, inference engine, and defuzzifier. Once the rules driving the FLS have been fixed, the FLS can be expressed as a mapping of inputs to outputs. Rules can come from experts or can be obtained from numerical data.

The fuzzifier maps crisp input numbers into fuzzy sets. An inference engine of the FLS maps fuzzy sets to fuzzy sets and determines the way in which the fuzzy sets are combined. In several applications, crisp numbers are needed as an output of the FLS. In those cases, a defuzzifier is used to calculate crisp values from fuzzy values.

A fuzzy set generalizes the concept of an ordinary set whose membership function only takes two values, zero and unity. The most commonly used shapes for membership functions $\mu(x)$ are triangular, trapezoidal, piecewise linear or Gaussian. Rules for the fuzzy system can be expressed as:

$$R_i : \text{IF } x_1 \text{ is } F_1 \text{ AND } x_2 \text{ is } F_2 \text{ AND } \ldots x_m \text{ is } F_m \text{ THEN } y = C_i,$$
$$i = 1, 2, 3 \ldots, M,$$

where m and M are the number of input variables and rules, x_i and y are the input and output variables, and $F_i \cdot V_i$ and $C_i \cdot W$ are fuzzy sets characterized by membership functions $\mu_{Fi}(x)$ and $\mu_{Ci}(x)$, respectively. Each rule can be viewed as a fuzzy implication $F_{1,2,3\ldots m} = F_1 \times F_2 \times \cdots F_m \cdot C_i$, that is a fuzzy set in $V \times W = V_1 \times V_2 \times V_3 \times \cdots \times V_m \times W$ with membership function given by

$$\mu_{Ri}(x,y) = \mu_{F1}(x_1) * \mu_{F2}(x_2) * \cdots * \mu_{Fm}(x_m) * \mu_{Ci}(y),$$

where * is the product with $x = [x_1 x_2 \cdots x_m] \cdot V$ and $y \cdot W$. In pattern recognition problem the outputs are often crisp sets, and $\mu_{Ci}(y) = 1$ is often used for the product inference formula. Popular defuzzification methods include maximum matching and centroid defuzzification. In our study, we keep the output as fuzzy sets as they are easier to interpret linguistically for diagnostic and prognostic action. Rules for the fuzzy system are obtained by fuzzification of the numerical values in the fingerprint charts using the following procedure:

Algorithm 1.

1. Each measurement delta is divided into N fuzzy sets whose geometry is selected by the designer.
2. A set of four measurements delta corresponding to a given module fault is input to the FLS and the degree of membership of the elements of the ˙EGT, ˙WF, ˙$N2$ and ˙$N1$ are obtained.
3. Each measurement delta is then assigned to the fuzzy set with the maximum degree of membership.
4. One rule is obtained for each module fault by relating the measurement deltas with maximum degree of membership to a module fault.

For any given input set of measurement deltas, the fuzzy rules are applied using product implication. Once the fuzzy rules are applied for a given measurement, we have degree of membership for FAN, LPC, HPC, HPT and LPT. For fault isolation, we are interested in the most likely fault. The fault with the highest degree of membership is selected as the most likely fault.

The main problem in Algorithm 1 is in the selection of the number and type of fuzzy sets in Step 1. Typically, designers select the number and geometry of the fuzzy sets based on knowledge of the problem. For example, the measurements may be classified into five fuzzy sets named very low, low, medium, high and very high. In case Gaussian functions are selected as membership functions, the midpoints and standard deviations associated with each Gaussian fuzzy set needs to be selected so that the entire measurement range is spanned by the fuzzy sets and there is some intersection between the sets. Thus, the designer must manually iterate over Algorithm 1 to obtain a fuzzy system which has good performance. This is a trial and error process. Genetic algorithms are one way of automating this process.

5. Genetic algorithm

Genetic algorithms (GA) are a probabilistic search method. A brief introduction to GA is given below. Goldberg [31] and recent papers [32–35], give more details about genetic algorithms. The genetic algorithm is motivated by the hypothesized natural process of evolution in biological populations, where genetic information stored in chromosomal strings evolve over generations to adapt favorably to a static or changing environment. The algorithm is based on elitist reproduction strategy, where members of population, which are deemed most fit, are selected for reproduction, and are given the opportunity to strengthen the chromosomal makeup of progeny generation. This approach is facilitated by defining a fitness function or a measure indicating the goodness of a member of the population in the given generation during the evaluation process.

To represent designs as chromosome-like strings, the design variable is converted to its binary equivalent and thereby mapped into a fixed length string of 0's and 1's. A number of such strings constitute a population of designs, with each design having a corresponding fitness value. This fitness value could be the objective function $F(\boldsymbol{X})$ for a function maximization problem. Thus, the GA can be used to solve optimization problems of the form,

$$\begin{aligned} &\text{Maximize} \quad F(\boldsymbol{X}) \\ &\text{Subject to} \quad X_i^{(\min)} \cdot X_i \cdot X_i^{(\max)}. \end{aligned}$$

The starting population is selected randomly in the domain lying between the minimum and maximum values of $\boldsymbol{X}$ and then the following genetic operators applied to improve results.

1. *Reproduction*. Individuals are selected and the probability of selection is based on their fitness value. The new population pool has higher average fitness value than the previous pool.
2. *Crossover*. In the two-point crossover approach, two mating parents are selected at random; the random number generator is invoked to identify two sites on the strings, and the strings of 0's and 1's enclosed between the chosen sites are swapped between the mating strings.
3. *Mutation.* A few members from the population pool are taken according to probability of mutation p_m, and a 0 to 1 or vice versa are switched at randomly selected mutation site on the chosen string.

The process of reproduction, crossover and mutation constitute one generation of the GA. After several generations the GA is stopped and the best point among the values taken as the optimal point. Being a probabilistic search method, GA's are very good at finding global maxima. Furthermore, GA's need only function values and not gradient information, which makes them easy to use for real systems where accurate gradient information is difficult to obtain, and local minima may occur. However, they are computationally expensive.

6. Genetic fuzzy system

There are two main problems in the generation of fuzzy systems [25]. The first is that it is difficult to select the appropriate number of fuzzy sets. The second is selection of the membership functions. For a given number of fuzzy sets and type of membership functions the rules need to be created. However, if the number of fuzzy sets or type of membership function changes, the rules can change. Most fuzzy systems are designed using a trial and error process. Therefore, any change in the membership functions or the number of fuzzy sets leads to a change in the rule base; the process of designing a fuzzy system is iterative and can become very cumbersome for a human designer. It is therefore desirable to create an automated procedure for the design of fuzzy systems.

A genetic algorithm is used to facilitate the design of the fuzzy system. The approach is discussed below:

Algorithm 2.

1. Define maximum and minimum values for a measurement delta $\cdot z$ by $\cdot z^{(\max)}$ and $\cdot z^{(\min)}$ respectively.
2. Define the universe of discourse for $\cdot z$ to be the set of real numbers between the minimum and maximum values, $U(\cdot z) = [\cdot z^{(\min)}, \cdot z^{(\max)}]$.
3. Define $L(\Delta z) = \cdot z^{(\max)} - \cdot z^{(\min)}$ as the length of the universe of discourse.
4. Divide U into N Gaussian fuzzy sets $F_1, F_2, \ldots, F_N$ and define the midpoint of fuzzy point F_1 by $\cdot z^{(\min)}$ and of fuzzy set F_N by $\cdot z^{(\max)}$, respectively.These fuzzy sets can be defined using the following equation:

   $$\mu(x) = \mathrm{e}^{-0.5\left(\frac{x-m}{\sigma}\right)^2},$$

   where m is the midpoint of the fuzzy set and **.** is the uncertainty (standard deviation) associated with the variable.
5. Assuming the fuzzy sets are equally spaced, calculate the mid points of fuzzy set F_2 as $\cdot z^{(\min)} + \cdot m$, of set F_3 as $\cdot z^{(\min)} + 2* \cdot m$ and set F_i as $\cdot z^{(\min)} + (i-1) \cdot m$ where

   $$\Delta m = \frac{L(\Delta z)}{N-1}.$$

6. Allow the fuzzy sets for the measurement delta Δz to move together along the number line by an amount x. This allows the midpoints of the fuzzy sets to change, along with the values $\cdot z^{(\min)}$ and $\cdot z^{(\max)}$. However, the distance $L(\Delta z)$ remains constant. With this definition, the midpoints of the fuzzy sets are defined once N and x are selected.

7. Select the standard deviation of the fuzzy set for measurement Δz as the measurement uncertainty of Δz.

The above approach can now be applied to the four measurement deltas considered in this study. This procedure is discussed in the algorithm below.

Algorithm 3.

1. Define the maximum and minimum values for each measurement ΔEGT, ΔN1, ΔN2 and ΔWF from the fault signatures shown in Table 1. Thus for ΔEGT, the maximum and minimum values are 21.77 °C and −7.72 °C, respectively.
2. Define the range spanned by each variable as $L_1 = L(\Delta\text{EGT})$, $L_2 = L(\Delta N1)$, $L_3 = L(\Delta N2)$, and $L_4 = L(\Delta\text{WF})$.
3. Choose N fuzzy sets to partition each measurement. To start the algorithm, use $N = 2$.
4. Let x_1, x_2, x_3 and x_4 define the tuning variables associated with ΔEGT, ˙N1, ˙N2 and ˙WF respectively. To start the algorithm, select random values satisfying $-25\%L_i \cdot x_i \cdot 25\%L_i$, $i = 1, 4$. Choose σ for ΔEGT, ΔN1, ΔN2 and ΔWF as 4.23 °C, 0.25%, 0.17% and 0.50%, respectively.
5. Generate the fuzzy system from the numerical data using the conventional procedure outlined before in Algorithm 1.
6. Using a sample of 100 noisy data points, calculate success rate as

   $$S = 100\frac{N_\text{C}}{N_\text{T}},$$

   where N_C is the number of correct classifications and N_T is the total number of classifications.
7. Use GA to solve the optimization problem by taking the best solution from $N_\text{gen}^{(\max)}$ generations:

   $$\begin{aligned} &\text{Maximize} \quad S(x_1, x_2, x_3, x_4) \\ &\text{Subject to} \quad -25\%L_i \cdot x_i \cdot 25\%L_i, \quad i = 1, 4. \end{aligned}$$

8. Increase N by 1,
   (a) if $N < N^{(\max)}$
      (i) Go to 3,
   (b) else
      (i) Select N with highest success rate S (if highest S is obtained by more than one value of N, select the lowest N that gives the highest S).

The only values, which need to be the input of the genetic fuzzy system (GFS), are the values of measurement deltas corresponding to each fault, and the fault signature based on the linearized influence coefficients at the current operating point. For the standard deviations of the Gaussian fuzzy sets, we use the measurement uncertainty data that can be obtained by a statistical analysis of engine data. If the measurement uncertainties change, the GFS can be tuned to the different numerics. Thus we get an automatic system that greatly reduces the need of manual manipulation.

7. Numerical results

In this study, a maximum of nine generation of the GA are used for each N values of the fuzzy sets. The population size, crossover probability and mutation probability are chosen as 20, 0.8 and 0.1, respectively. The maximum number of fuzzy sets is selected as 10.

Since genetic algorithms are computationally intensive, the issue about computation time is important for practical implementation. As an example, the code implementing the algorithm in this study takes about 3–5 min to run on Matlab on a Pentium 4 PC with the full nine generations of GA. However, in many cases, the convergences occur in 2–3 generations given that we use only four design variables and have a starting population of 20 for each variable. Each design variable is represented by a 10 bit string.

As mentioned earlier, a standard approach in the design of the optimal fuzzy system is to consider the midpoints and standard deviations of each fuzzy set as design variables. If there are N fuzzy sets and M measurements, the maximum number of midpoint design variables is $N * M$ and the maximum number of standard deviation design variables is $N * M$. The total number of design variables is therefore $2 * N * M$. For the case with $N = 6$ and $M = 4$, we would have a total of $2 * 6 * 4 = 48$ design variables, leading to high computer time requirements.

The algorithm in this study uses some prior knowledge of the problem to reduce the number of design variables dramatically. The standard deviations are thus selected to be equal to the measurement uncertainties. In this manner, the fuzzifier is able to act as a filter which addresses noise in the data in a direct manner. By making the requirement that the universe of discourse only spans the neighborhood of the measurements, the region where fuzzy set discretization is needed is optimized. Using a uniform distribution of fuzzy sets leads to so-called design variable linking in optimization and allows the midpoints to be defined using only two variables for each measurement: the number of fuzzy sets N and the translation variable $\boldsymbol{x}$. For a given number of fuzzy sets, the number of design variables is equal to the number of measurements which is four in this case.

The fuzzy system is tested using simulated data developed from the fault signatures shown in Table 1. For each module, 100 noisy data sets are generated for module faults with 2% deterioration in efficiency. Noise is added to the simulated measurement deltas using the typical standard deviations for ˙EGT, ˙*N*1, ˙*N*2, and ˙WF as 4.23 °C, 0.25%, 0.17% and 0.50%, respectively.

Fig. 1 shows the success rate for the optimal GFS as the number of fuzzy sets is increased from 2 to 9. For each value of *N* in this figure, the optimal values of ***x*** are calculated using Algorithm 3. For only 2 fuzzy sets, the success rate is about 80% and quickly rises as the number of sets increases. The number $N = 6$ is selected by Algorithm 3 as the point where the GFS is optimal with a minimum number of sets. Fig. 2 shows the success rate of the fuzzy system with six sets as the GA generations' progress. In this case, only two generations

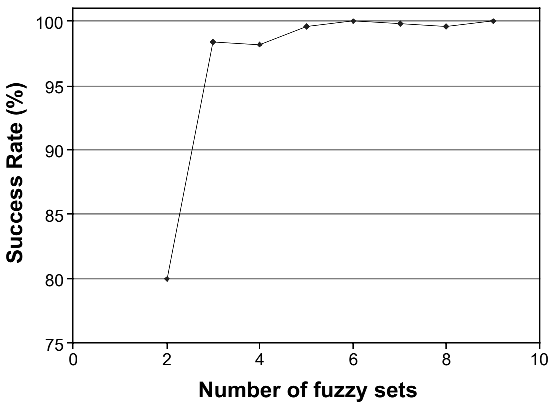

Fig. 1. Change in fault isolation success rate with increasing number of fuzzy sets.

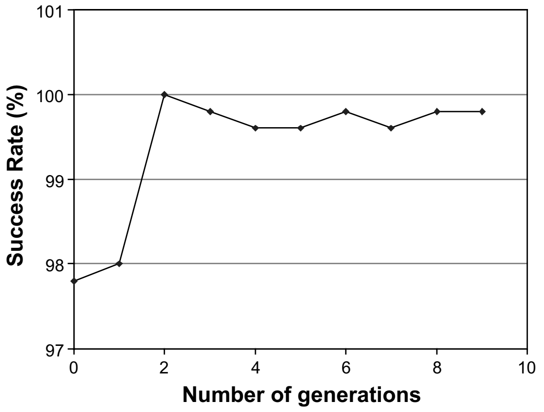

Fig. 2. Evolution of success rate for fuzzy system with 6 sets with generations of genetic algorithm.

were needed to achieve a success rate of 100% and the values of $\boldsymbol{x}$ corresponding to the second generation of GA is selected by Algorithm 3 as the optimal fuzzy system.

Tables 2–6 provide the midpoints of the fuzzy sets for the four measurements as the number of fuzzy sets increases from two to six. The starting values in Table 2 show two fuzzy sets with midpoints centered near the maximum and minimum values of the measurements. The values in Table 6 correspond to the case where $N = 6$ in Fig. 1 and $N_{gen} = 2$ in Fig. 2. Figs. 3–7 show the evolution of the fuzzy system using the fuzzy sets for exhaust gas temperature as an example. Fig. 3 shows the starting case with two fuzzy sets which is a crude

Table 2
Midpoints of two fuzzy sets

ΔEGT (°C)	−7.69	21.80
ΔN1 (%)	−1.93	1.38
ΔN2 (%)	−1.10	1.30
ΔWF (%)	−1.89	2.61

Table 3
Midpoints of three fuzzy sets

ΔEGT (°C)	−8.16	6.58	21.33
ΔN1 (%)	−2.40	−0.75	0.91
ΔN2 (%)	−1.57	−0.37	0.83
ΔWF (%)	−2.36	−0.11	2.14

Table 4
Midpoints of four fuzzy sets

ΔEGT (°C)	−8.31	1.52	11.35	21.18
ΔN1 (%)	−2.55	−1.45	−0.35	0.76
ΔN2 (%)	−1.72	−0.92	−0.12	0.68
ΔWF (%)	−2.51	−1.01	0.49	1.99

Table 5
Midpoints of five fuzzy sets

ΔEGT (°C)	−7.82	−0.44	6.92	14.30	21.67
ΔN1 (%)	−2.06	−1.23	−0.40	0.42	1.25
ΔN2 (%)	−1.23	−0.63	−0.03	0.57	1.17
ΔWF (%)	−2.02	−0.89	0.23	1.36	2.48

Table 6
Midpoints of six fuzzy sets

	VL	L	ML	MH	H	VH
ΔEGT (°C)	−9.62	−3.72	2.17	8.07	13.97	19.87
ΔN1 (%)	−2.23	−1.56	−0.90	−0.24	0.42	1.08
ΔN2 (%)	−1.21	−0.72	−0.25	0.23	0.71	1.19
ΔWF (%)	−2.25	−1.35	−0.45	0.45	1.35	2.25

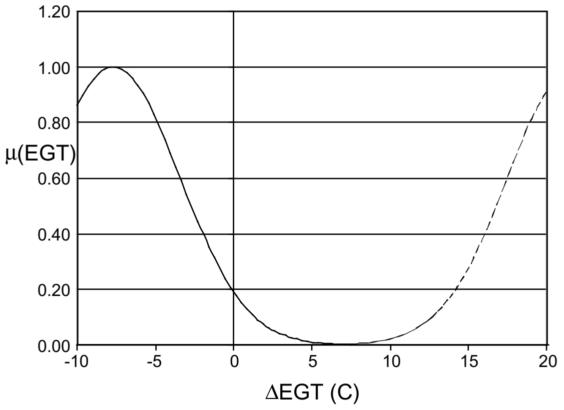

Fig. 3. Discretization of universe of exhaust gas temperature using two fuzzy sets.

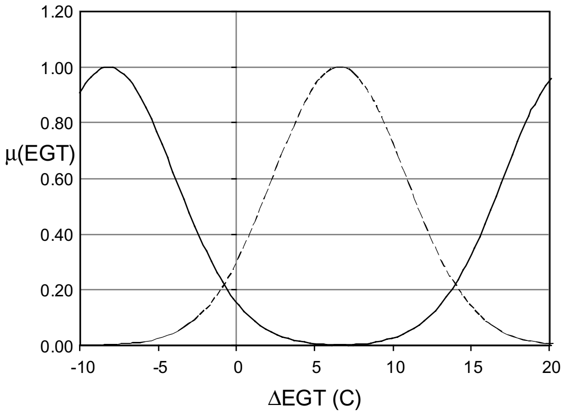

Fig. 4. Discretization of universe of exhaust gas temperature using three fuzzy sets.

descretization. In Fig. 7, the optimal level of discretization with six fuzzy sets is achieved.

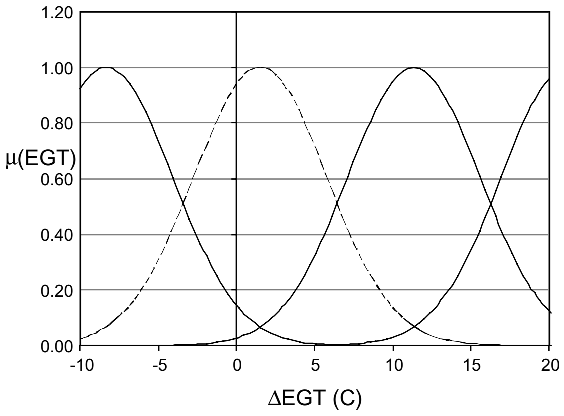

Fig. 5. Discretization of universe of exhaust gas temperature using four fuzzy sets.

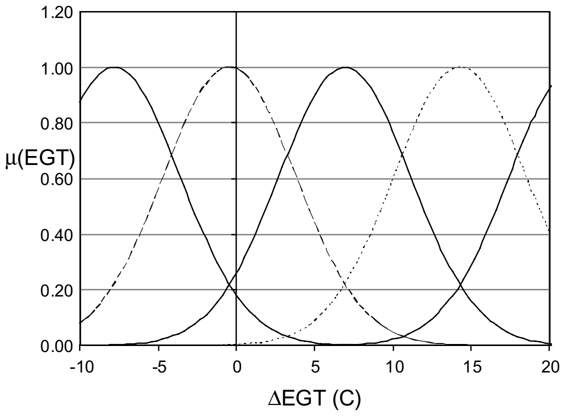

Fig. 6. Discretization of exhaust gas temperature using five fuzzy sets.

In Table 6 each fuzzy set is assigned a linguistic value of very low (VL), low (L), medium-low (ML), medium-high (MH), high (H) and very high (VH). These "linguistic measures" are shown in Fig. 7 for the six ΔEGT fuzzy sets. The fuzzy rule base for the case with six fuzzy sets is shown in Table 7. Table 7 is the result of fuzzification of the numerical data in Table 1. These rules can be read as follows for the FAN module:

IF

·EGT is very low AND

·$N1$ is very high AND

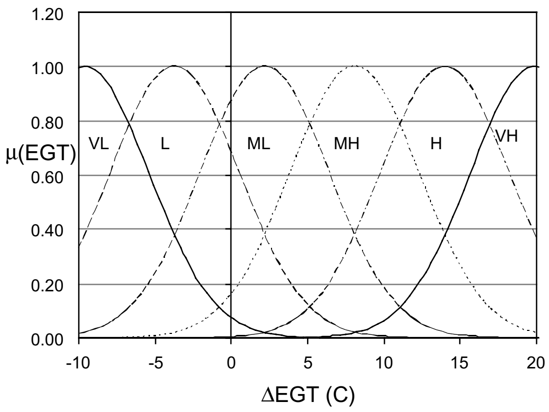

Fig. 7. Discretization of universe of exhaust gas temperature using six fuzzy sets.

Table 7
Rules for optimal fuzzy system with six fuzzy sets

	ΔEGT	ΔN1	ΔN2	ΔWF
FAN	VL	VH	L	VL
LPC	ML	MH	H	MH
HPC	MH	MH	ML	H
HPT	VH	MH	VL	VH
LPT	L	VL	VH	VL

˙N2 is low AND
˙WF is very low
THEN
Problem in FAN module

The rules for the other modules can be similarly interpreted. These rules provide a knowledge base and represent how a human engineer would interpret data to isolate an engine fault using fingerprint charts.

Table 8 shows the success rate of the fuzzy set with 100 noisy data points. The noisy data points for testing are different from data used for developing the rule base of the fuzzy system. The average success rate is 100%, compared to 98.2% for the manually designed fuzzy system in Ref. [18]. The manually designed fuzzy system showed some problems in differentiating between faults in the LPC and those in the HPC. It is clear that GFS is able to identify the correct fault despite the presence of considerable uncertainty in measurements.

The effect of noise on the GFS is shown in Fig. 8 and the results are compared with data from the fuzzy system from Ref. [18]. Here the noise ratio is

Table 8
Results for optimal fuzzy system and manually designed system

Module	Success rate (%)	Success rate (%)[a]
HPC	100	94
HPT	100	100
LPC	100	97
FAN	100	100
LPT	100	100
Average success rate	100	98.2

[a] From [18].

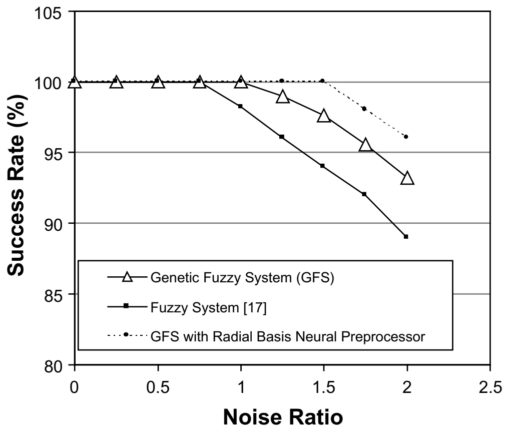

Fig. 8. Success rate in fault isolation with increasing noise levels in data.

defined as σ/σ_0 where σ_0 is the baseline noise level used for developing the GFS and σ is the noise level in the simulated data used for testing. It is clear that both the systems show a decline in the average fault isolation success rate with increasing noise levels in the data. However, the GFS appear to show a somewhat better performance as the noise level increases. This is due to the "optimal" nature of the fuzzy system developed and the use of formal optimization methods rather than a trial and error process in maximizing the success rate. The result of applying a neural network preprocessor to the GFS is discussed below.

To study the signal processor, we assume time series of 100 discrete points. From $k = 0$ to $k = 50$, the signal changes linearly from 0 to $\text{sign}(\Delta z)\sigma_0/2$. From $k = 50$ to 51, the signal changes by Δz. From $k = 51$ to 100 the signal changes from Δz to $\Delta z + \text{sign}(\Delta z)\sigma_0/2$. This simulates a "single fault" situation, where a step jump equal to the measurement deltas corresponding to the module faults is added to a linearly varying signal. As an example, the ΔEGT variation for an HPC fault is simulated using a linear variation from 0 °C at $k = 1$ to 4.23/2 = 2.115 °C at $k = 50$, followed by a change to 13.6 + 4.23/2 = 15.715 °C at $k = 51$, and a linear variation thereafter to 13.6 + 4.23 = 17.83 °C. Fig. 9 shows the noisy signal and RBF filtered signal.

For determining the RBF unit centers, we use a 'K-means' clustering algorithm. The 'K-means' clustering algorithm finds a set of clusters each with centers from the given training data. The cluster centers become the centers of the RBF units. The number of clusters is a design parameter and determines the number of RBF units, i.e. nodes, in the hidden layer. We have used $H = 20$. When the RBF centers have been established, the widths of each RBF can be calculated. The width of any RBF distance to the nearest p RBF units, where p is a design parameter for the RBFN, for unit t is given by

$$\sigma_i = \sqrt{\left[\frac{1}{p}\sum_{j=1}^{p}\sum_{k=1}^{r}(x_{\hat{k}i} - x_{\hat{k}j})^2\right]},$$

where $x_{\widehat{ki}}$ and $x_{\widehat{kj}}$ are the kth entries of the centers of the ith and jth hidden units. We have used $p = 5$. When the centers and widths of the RBF units have been chosen, then the $N = 100$ training samples are processed through the hidden nodes to generate an $H \times N$ matrix, called A. Let T be the $M \times N$ desired output matrix for the training patterns and $M = 100$ is the number of output nodes. The objective is to find the weights that minimize the error between the actual output and the desired output of the network. Essentially, we are trying to minimize the objective (cost) function

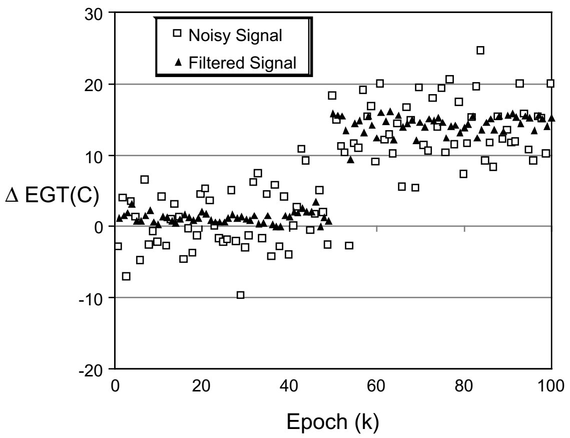

Fig. 9. Noisy and filtered ΔEGT signal simulating HPC fault.

$$\|T - WA\|,$$

where W is the $M \times H$ matrix of weights on the connections between the hidden and output nodes of the network. We train the RBF network with added Gaussian noise at $\sigma_0 = 4.23$ °C, 0.25%, 0.17% and 0.50%, respectively for ΔEGT, ΔN1, ΔN2, ΔWF.

Noise is added to the ideal signal using a baseline value σ_0 of typical standard deviations for ˙EGT, ˙N1, ˙N2, and ˙WF as 4.23 °C, 0.25%, 0.17% and 0.50%, respectively. The filtered signal in Figs. 9 and 10 show considerable noise reduction while preserving the nature of the step edge. This data represents one noisy signal for each measurement. The visual quality of the data is considerably improved. Similar results are obtained for all the signals corresponding to the faults in Table 1. To summarize these results concisely, the following noise reduction measure is defined based on the mean absolute error (MAE) criteria.

$$\mathrm{MAE}^{(\mathrm{noisy})} = \sum_{i=1}^{N} \frac{1}{N} \left| \Delta z_i^{(\mathrm{noisy})} - \Delta z_i^{(\mathrm{ideal})} \right|,$$

$$\mathrm{MAE}^{(\mathrm{filtered})} = \sum_{i=1}^{N} \frac{1}{N} \left| \Delta z_i^{(\mathrm{filtered})} - \Delta z_i^{(\mathrm{ideal})} \right|,$$

$$N_{\mathrm{R}} = 100 \frac{\mathrm{MAE}^{(\mathrm{noisy})} - \mathrm{MAE}^{(\mathrm{filtered})}}{\mathrm{MAE}^{(\mathrm{noisy}})}.$$

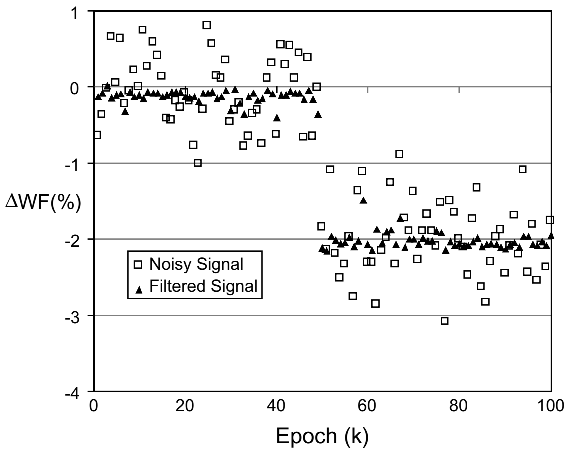

Fig. 10. Noisy and filtered ΔWF signal simulating LPT fault.

Table 9
Noise reduction using radial basis neural network

	ΔEGT (°C)	ΔN1 (%)	ΔN2 (%)	ΔWF (%)
HPC	78.84	67.03	67.38	81.87
HPT	84.24	72.07	83.38	83.71
LPC	77.50	74.34	78.48	77.95
FAN	74.80	82.43	79.04	80.62
LPT	68.76	83.83	84.68	82.95
Average	76.83	75.94	78.59	81.42

For each signal, 100 samples of noisy test data are created and the noise reduction calculated. These values are summarized in Table 9 and show a noise reduction averaging between 75% and 81%. Results in this paper clearly demonstrate the power of the soft computing framework for automated decision making under uncertainty. The approach uses the concept of "hybridization in soft computing" where using different techniques such as neural networks, genetic algorithms and fuzzy logic together gives better results than if each method is used individually [36]. The "hybridization" process uses the strengths of each different approach to attack the problem.

8. Conclusions

A novel genetic fuzzy system (GFS) is developed in this study for fault isolation in gas turbine engines. The GFS has better performance than a manually designed fuzzy system because GA's automatically selects the number of fuzzy sets and membership functions based on the fault signatures of the engine and measurement uncertainties. The GA searches for the optimum solution given a comparatively small number of rules compared to all possible. This minimizes the computational demand of the model generation and allows problems with realistic dimensions to be considered. The fault signatures are derived from influence coefficients. A radial basis function neural network (RFBNN) is also studied for data cleaning prior to fault isolation. RBFNN share the universal approximation capability, and take much less training time and offer much better performance than the traditional linear filter.

The following conclusions can be drawn from this study.

1. For simulated faults considered in this study, the GFS achieved a success rate of 100% for the five module faults (HPC, LPC, FAN, HPT, and LPT) and four measurements (˙EGT, ˙N1, ˙N2, ˙WF). In contrast, a manually developed fuzzy system achieved a success rate of 98% with some confounding between the LPC and HPC module faults.

2. The trial and error process used to design a fuzzy system leads to considerable human labor and is often sub optimal. Different aircraft engines operated by different airlines can have different numerics such as influence coefficients and measurement uncertainties and it is a tedious process to develop a fuzzy system for each case. The GFS automates the process of design of the fuzzy system.
3. As noise levels in data increase, the GFS retains its edge over the manually designed fuzzy system, giving 2–5% higher success rate with the same numerics.
4. A radial basis neural network prefilter achieved 75–81% noise reduction for simulated signals with linear deterioration and step changes. When the neural network is used to prefilter signals prior to fault isolation, the accuracy of the GFS is further improved for lower quality data by 2–4%.

References

[1] Y.G. Li, Performance analysis based gas turbine diagnostics: a review, Journal of Power and Energy 216 (A5) (2002) 363–377.

[2] R. Kurz, K. Brun, Degradation in gas turbine systems, ASME Journal of Engineering for Gas Turbine and Power 123 (2001) 344–349.

[3] K. Mathioudakis, P. Kamboukos, A. Stamasis, Turbofan performance deterioration tracking using non-linear models and optimization techniques, Journal of Turbomachinery 124 (4) (2002) 344–349.

[4] M. Pinelli, P.R. Spina, Gas path field performance determination: sources of uncertainties, ASME Journal of Engineering for Gas Turbine and Power 124 (1) (2002) 155–160.

[5] A.J. Volponi, Gas Path Analysis: An Approach to Engine Diagnostics, Time-Dependent Failure Mechanisms and Assessment Methodologies, Cambridge University Press, 1983.

[6] L.A. Urban, A.J. Volponi, Mathematical methods of relative engine performance diagnostics, SAE 1992 Transactions, 101, Journal of Aerospace, Technical Paper 922048, 1992.

[7] D. Doel, TEMPER—a gas path analysis tool for commercial jet engines, ASME Journal of Engineering for Gas Turbine and Power 116 (1) (1994) 82–89.

[8] D. Doel, Interpretation of weighted least squares gas path analysis, ASME Journal of Engineering for Gas Turbine and Power 125 (3) (2003) 624–633.

[9] S. Sampath, S. Ogaji, R. Singh, D. Probert, Engine fault diagnostics: an optimization procedure, Applied Energy 73 (1) (2003) 47–70.

[10] P.J. Lu, T.C. Hsu, Application of autoassociative neural network on gas-path sensor data validation, Journal of Propulsion and Power 18 (4) (2002) 879–888.

[11] S. Ogaji, R. Singh, Advanced engine diagnostics using artificial neural networks, Applied Soft Computing 3 (3) (2003) 259–271.

[12] C. Romesis, A.K. Mathioudakis, Setting up of a probabilistic neural network for sensor fault detection including operation with component faults, ASME Journal of Engineering for Gas Turbine and Power 125 (3) (2003) 634–641.

[13] P. Hajela, Soft computing in multidisciplinary aerospace design-new directions for research, Progress in Aerospace Sciences 38 (1) (2002) 1–21.

[14] M. Jamshidi, Autonomous control of complex systems: robotic applications, Applied Mathematics and Computation 120 (2001) 15–29.

[15] H. DePold, F.D. Gass, The application of expert systems and neural networks to gas turbine prognostics and diagnostics, ASME Journal of Engineering for Gas Turbine and Power 121 (4) (1999) 607–612.

[16] P.J Lu, T.C. Hsu, M.C. Zhang, J. Zhang, An evaluation of engine fault diagnostics using artificial neural networks, ASME Journal of Engineering for Gas Turbine and Power 123 (2) (2001) 240–246.

[17] A.J. Volponi, H. Depold, R. Ganguli, C. Daguang, The use of Kalman filter and neural network methodologies in gas turbine performance diagnostics: a comparative study, ASME Journal of Engineering for Gas Turbine and Power 125 (4) (2003) 917–924.

[18] R. Ganguli, Application of fuzzy logic for fault isolation of jet engines, ASME Journal of Engineering for Gas Turbines and Power 125 (3) (2003) 617–623.

[19] X.L. Hong, P.C.L. Chen, The equivalence between fuzzy logic systems and feedforward neural networks, IEEE Transactions of Neural Networks 11 (2) (2000) 356–365.

[20] L.A. Zadeh, Fuzzy logic = computing with words, IEEE Transactions on Fuzzy Systems 4 (2) (1996) 101–103.

[21] Y. Iwakoshi, T. Furuhashi, Y. Uchikawa, A fuzzy classifier system for evolutionary learning of robot behaviors, Applied Mathematics and Computation 91 (1998) 73–81.

[22] J.-J. Boreux, L. Duckstein, A fuzzy approach to the definition of standardized visibility in fog, Applied Mathematics and Computation 61 (1994) 287–299.

[23] R.M. Faye, S. Sawadogo, C. Lishou, F. Mora-Camino, Long-term fuzzy management of water resource systems, Applied Mathematics and Computation 137 (2003) 459–475.

[23] R. Ganguli, Jet engine gas-path measurement filtering using center weighted idempotent median filters, Journal of Propulsion and Power 19 (5) (2003) 930–937.

[25] P. Pawar, R. Ganguli, Genetic fuzzy system for damage detection in beams and helicopter rotor blades, Computer Methods in Applied Mechanics and Engineering 192 (16–18) (2003) 2031–2057.

[26] J. Gao, M. Lu, Fuzzy quadratic minimum spanning tree problem, Applied Mathematics and Computation 164 (3) (2005) 773–788.

[27] C.S. Shieh, Genetic fuzzy control for time-varying delayed uncertain systems with a robust stability safeguard, Applied Mathematics and Computation 131 (2002) 39–58.

[28] J. Leonard, M. Kramer, L.H. Unger, Using radial basis functions to approximate a function and its error bounds, IEEE Transactions on Neural Networks 3 (1992) 624–627.

[29] F.J. Hickernell, Y.C. Hon, Radial basis function approximations as smoothing splines, Applied Mathematics and Computation 102 (1999) 1–24.

[30] X. Li, On simultaneous approximations by radial basis function neural networks, Applied Mathematics and Computation 95 (1998) 75–89.

[31] D. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1998.

[32] P. Hajela, Nongradient methods in multidisciplinary design optimization-status and potential, Journal of Aircraft 36 (1) (1999) 255–274.

[33] L. Wei, M. Zhao, A niche hybrid genetic algorithm for global optimization of continuous multimodal functions, Applied Mathematics and Computation 160 (2005) 649–661.

[34] Z.Y. Lee, Applying the double side method of weighted residual to the solution of shell deformation problem, Applied Mathematics and Computation, in press, doi:10.1016/j.amc.2003.10.064.

[35] P.K. Gudla, R. Ganguli, An automated hybrid genetic-conjugate gradient algorithm for multimodal optimization problems, Applied Mathematics and Computation, in press, doi:10.1016/j.amc.2004.08.026.

[36] C.W. De Silva, The role of soft computing in intelligent machines, Philosophical Transactions of the Royal Society A 361 (1809) (2003) 1749–1780.