Available online at www.sciencedirect.com
ScienceDirect
Mechanical Systems and Signal Processing 21 (2007) 994–1007
Mechanical Systems and Signal Processing
www.elsevier.com/locate/jnlabr/ymssp

Genetic algorithm with an improved fitness function for (N)ARX modelling

Q. Chen[a,*], K. Worden[b], P. Peng[a], A.Y.T. Leung[c]

[a]College of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics, 29 Yudao Street, Nanjing 210016, China
[b]Department of Mechanical Engineering, University of Sheffield, Sheffield S1 3JD, UK
[c]Department of Building and Construction, City University of Hong Kong, Hong Kong, China

Received 4 January 2005; received in revised form 10 January 2006; accepted 25 January 2006
Available online 27 March 2006

Abstract

In this article a new fitness function is introduced in an attempt to improve the quality of the auto-regressive with exogenous inputs (ARX) model using a genetic algorithm (GA). The GA is employed to identify the coefficients and the number of time lags of the models of dynamic systems with the new fitness function which is based on the prediction error and the correlation functions between the prediction error and the input and output signals. The new fitness function provides the GA with a better performance in the evolution process. Two examples of the ARX modelling of a linear and a non-linear (NARX) simulated dynamic system are examined using the proposed fitness function.

Keywords: ARX model; Genetic algorithm; Fitness function; Model structure

1. Introduction

In engineering dynamics, control engineering and many other areas, auto-regressive with exogenous inputs (ARX) models are widely utilised for describing dynamic data regimes for linear and non-linear systems [1,2]. The model structure of an ARX model comprises a number of time lags of each processing variable and a set of coefficients. Many algorithms have been employed to identify the parameters of the ARX and non-linear ARX (NARX) models. Among these algorithms, the least squares (LS) algorithm, maximum likelihood (ML) method and genetic algorithm (GA) are the most often used [3]. The genetic algorithm proposed here has the property that it simultaneously carries out both parameter estimation and structure detection. In the case of a linear-in-the-parameters model, these two stages can be accomplished using least-squares or maximum likelihood methods and, for example, orthogonal forward selection, and there is arguably no real advantage in adopting a GA approach. However, the intention of this program of research is to ultimately demonstrate a GA methodology suitable for non-linear-in-the-parameters systems and this will be the subject of a further paper which will consider exponential-ARMA models and threshold models. For the moment though, the concern is how the objective function for the GA is conformed and the approach presented here can be

*Corresponding author.
E-mail address: q.chen@nuaa.edu.cn (Q. Chen).

0888-3270/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ymssp.2006.01.011

demonstrated adequately on linear-in-the-parameters models. In most cases of ARX modelling, only the coefficients of the processing variables in the time series equations are identified using one of those algorithms, and the number of time lags is to be given beforehand. However, like the other modelling requirements, the model structure is a vital and difficult issue to address. The number of time lags in a time series model represents the model structure or model order. In practical systems it is hard to guess a correct model order for ARX models if little information is known of the system for the model orders depend on the system's physical characteristics and the data acquisition process. An inappropriate model structure would degrade the performance of the ARX modelling process badly no matter which algorithm is used. Therefore, determining a proper number of time lags of the processing variables is as critical as fitting the model coefficients. Researchers have been seeking a method to identify both the coefficients and the optimum model structures automatically for many years. However, there are still no analytical results or specific guidelines available. Experience and a priori knowledge are the only reliable sources [3]. The same situation is experienced by various modelling techniques. Many articles have studied the statistics and conditions that a quality model should meet, while there is no mention of how to improve the model structures automatically in the modelling process [4,5]. Since the GA's evolutionary process can make the adapted parameters fade in, it is possible that both the coefficients and the model orders could be identified in the evolutionary process. Sarimveis et al. have successfully employed a GA to optimise the model structure of radial basis function neural networks [6], although only one hidden layer is allowed and the only model structure parameter is the number of neurons in the hidden layer. Therefore, the GA has the potential to solve the optimum model structure problem for ARX modelling procedure. But the model structures may still be far away from the correct ones even using a GA if the fitness function is not properly defined. Normally, only the prediction error is used in the fitness function, with this the ARX model often fails to evolve into the one with the correct model structure. The form of the fitness function is thus one of the key issues for the modelling problem using a GA. By 'correct model structure', it is implicitly meant that the numbers of time lags in a model satisfy the AIC or AICC criterion. Or more practically, a correct model structure should have accurate-enough output predictions from given inputs that have not been seen during model training process. In this paper, to identify the correct number of time lags in an ARX, model correlation function data between the prediction residuals and the response variables and predictor variables are introduced into the fitness function. Since these correlation functions strongly link to the model structures, the new fitness function significantly improves the accuracy of the model structure and the model coefficients. Two examples of the ARX modelling of a linear and a non-linear simulated dynamic system are examined using the proposed fitness function in the GA. The results demonstrate that the new fitness function has better performance.

2. A quick review of ARX modelling and the GA

2.1. ARX model

For a dynamic system's response to a stationary random excitation to be a stationary random process, the system must be strictly stable. The response time series is a sequentially correlated signal, i.e. the value of a variable at an instant depends on its past values and the input process. Such a process is termed as a dynamic process. In general, a dynamic process can be described using a discrete-time series model, such as an auto-regressive (AR) model, a moving average (MA) model, or the combination of the two models, an ARMA model. When the input variable is a known or measurable quantity the model is usually termed as NARX. The general form is as follows:

$$y(t) = F(y(t-\tau),\ldots,y(t-p\tau),\ u(t-\tau),\ldots,u(t-q\tau)),$$

where $F(\bullet)$ represent a non-linear function, and can specify any finite non-linear system [7]. If separating the linear part from the non-linear part, it can be written as

$$\begin{aligned} y(t) = {} & \phi_1 y(t-\tau) + \phi_2 y(t-2\tau) + \cdots + \phi_p y(t-p\tau) \\ & + \theta_0 u(t) + \theta_1 u(t-\tau) + \cdots + \theta_q u(t-q\tau) \\ & + \varphi_s(y(t-\tau),\ldots,y(t-m\tau),u(t-\tau),\ldots,u(t-n\tau)), \end{aligned} \tag{1a}$$

where $y(t)$ is the response of the system and $u(t)$ the driving noise acting as an excitation signal. Only stationary random processes are considered as the system input signals. $\varphi_s(\bullet)$ is the non-linear part if a non-linear system is dealt with and the model is then usually termed non-linear ARX or NARX. $\phi_1 \dots \phi_p, \theta_0 \dots \theta_q$ are the model parameters to be identified, and, p, q and s are the numbers of time lags of the processing terms, and τ is the time lag of each step. In the following sections y, u, ϕ, θ and p, q have the same meaning as described above.

One-step-ahead (OSA) prediction of the model can be determined by

$$\hat{y}(t) = \phi_1 y(t-\tau) + \phi_2 y(t-2\tau) + \cdots + \phi_p y(t-p\tau) + \theta_0 u(t) + \theta_1 u(t-\tau) + \cdots + \theta_q u(t-q\tau) + \varphi_s(y), \tag{1b}$$

where $\hat{y}(t)$ is the predicted response at instant t. Full prediction is obtained by

$$\hat{y}(t) = \phi_1 \hat{y}(t-\tau) + \phi_2 \hat{y}(t-2\tau) + \cdots + \phi_p \hat{y}(t-p\tau) + \theta_0 u(t) + \theta_1 u(t-\tau) + \cdots + \theta_q u(t-q\tau) + \varphi_s(\hat{y}). \tag{1c}$$

The difference for full prediction is that all processing terms of response should be previously predicted values. Clearly, p starting values are needed to start up the prediction process of the model, which are normally the first p measured values.

There are other forms of time series models for non-linear systems, such as the NARMAX model [7], or AVD model [8]. They also have a model structure selection problem, therefore, the conclusions should apply to those modelling techniques. Rodriguez-Vazquez et al. discussed a structural selection technique in [9] using multi-objective genetic programming. They use the correlation functions between the residuals and input and output as one of the objective functions for model validity test in the programming process to determine the best model structure. In fact reference [9] is simply the latest in a long programme of work by Rodriguez-Vazquez, Fleming and Fonseca on model-term selection using evolutionary algorithms. The rather modest aim which distinguishes the present work is to express all the requirements for model selection within a single objective function. The GA is used here, although the approach could be adopted for genetic programming.

2.2. Genetic algorithm

The fundamentals and applications of the GA can be found in many articles such as [10–12]. The GA refers to evolutionary computing based on the ideas of genetics. The GA is inspired by Darwin's theory of evolution which solves a problem by an algorithm imitating an evolutionary process. The space of all feasible solutions is referred to as the search space. With a GA the problem to be solved is projected onto the search space, and the best solution among a number of possible solutions is identified during the evolution procedure. A chromosome consisting of a string of genes represents a solution of the problem. Encoding a chromosome is the first thing to do to, which will be very specific for a particular problem. A chromosome should in some way contain information about the solution that it represents. The most-often used way of encoding is a binary string. In genetics a chromosomes consists of genes, i.e. blocks of DNA. So a gene is a segment of a chromosome and carries some particular genetic information. According to these genes a fitness function and a coding function are established. The coding function translates the genes (binary codes) into a real solution. The fitness function evaluates the solution's fitness based on the current solution and the training data set. The general forms of a fitness function and coding function are represented as

$$V_f = f(t_i, X), \tag{2}$$

$$t_i = C(ch_i), \quad i = 0, 1, 2 \dots n, \tag{3}$$

where V_f is the fitness value, and normally it should be no less than zero, X the training data set and t_i the coding value of a chromosome, ch_i.

Operations are performed on these codes that make the whole population evolve. The elementary operators of the GA effect the population as follows:

- *Selection*: select one chromosome that has a high fitness value from the current population, and copy it into the new population.
- *Cross-over*: select two chromosomes that have high fitness values from the current population, exchange some bits of them and copy them into the new population. The locations of these bits are random.
- *Mutation*: select one chromosome that has a high fitness value from the current population, alter some bits of it and copy it into the new population.

3. ARX modelling using a GA

3.1. The general process

To fit the parameters of the ARX model shown as Eq. (1) one can design a genetic algorithm as shown in Fig. 1.

Two indices, the speed of convergence and the root mean square (RMS) of the prediction error sequence, can be used to observe the performance of a GA process.

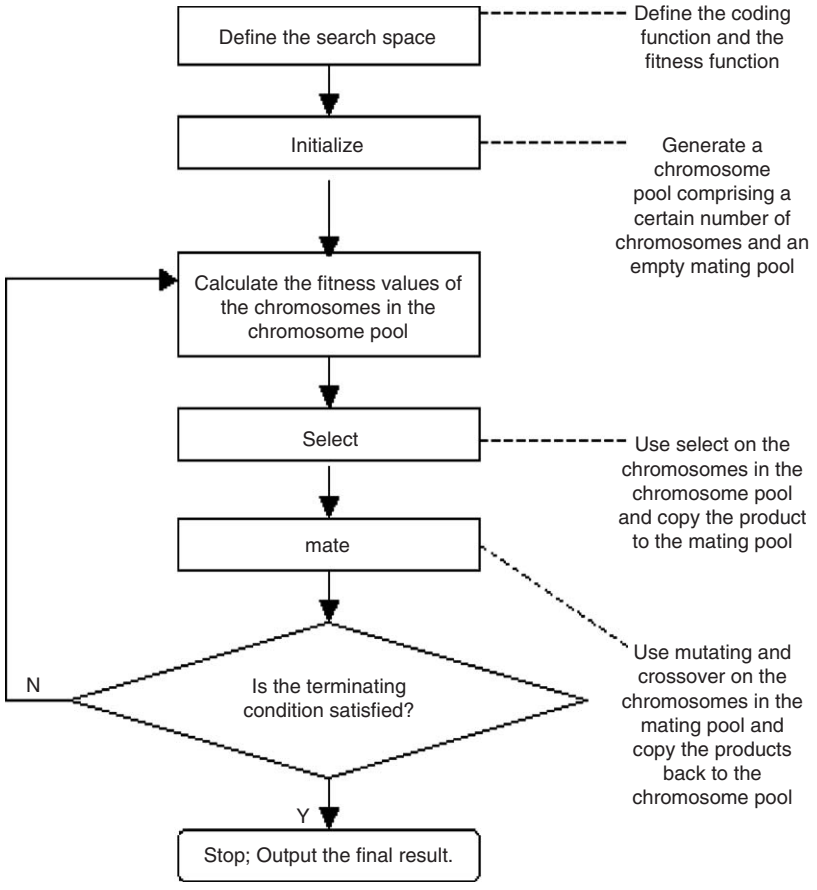

Fig. 1. Flowchart of using GA in ARX modeling.

3.2. Coding function

As explained before, each solution (or chromosome) is denoted by a binary string. To fit all the parameters of an ARX model the binary string of a solution is divided into some sub-strings which are called genes and each gene is translated into a parameter of the ARX model. In the translation of this article a binary coding will be used. The exact number of parameters is not known beforehand because the correct number of time lags cannot be determined before the search process started. So a minimum length of the binary string is prescribed for the chromosome in which the last set of bits denotes the time lags of y and u in integer format. In the run of the algorithm, the number of parameters is updated and recorded in the last bits in each chromosome during the search process. The coefficients of the model are coded in the rest of the chromosome in floating-point-number format to adapt to any possible values in the evolution process. The chromosome structure is as shown in the following table:

	ϕ_1	ϕ_2	…	ϕ_p	Buffer	Buffer	θ_1	θ_2	…	θ_q	Buffer	Buffer	⇒
⇒	Ψ_1	Ψ_2	…	Ψ_s	Buffer	Buffer	p	q	s				

In the table, buffer cells are empty in most of situations depending on the model structural parameters p and q. In this paper s is the number of non-linear terms in simplest form, for the implementation of the process without affecting the point made in this paper. If considering general non-linear terms with many possible combinations, the structure of the chromosome would be a lot more complicated and its size would be too large to bear for a personal computer. The buffer cells are specified to fix the locations of each parameter in the corresponding sections of the chromosome so that the mutation or cross-over operations will be always on the same parameter during the evolution process. However, it must be ensured that the maximum values of p and q will not exceed the maximum number of genes for the corresponding to the parameters. A typical coefficient, say ϕ_i, will be coded in the floating-point-number format similar to the binary coding format in computing process:

1	0101	1011	1110	1101	0101	0	0000	0011
coefficient sign 0→ + 1→ -	a_4	a_3	a_2	a_1	a_0	exponent sign 0→+ 1→ -	$\overbrace{\quad\quad\quad\quad}^{k}$	
	coefficient digits after decimal point						exponent Digits	

The transform between hexadecimal and decimal system is performed to obtain the parameter value:

$$\phi_i = \pm\left(\tfrac{1}{16}\times a_4 + \tfrac{1}{16^2}\times a_3 + \tfrac{1}{16^3}\times a_2 + \tfrac{1}{16^4}\times a_1 + \tfrac{1}{16^5}\times a_0\right)\times 2^{\pm k}.$$

The reason that 2 is used as the base of the power rather than 16 is because one found that this can speed up the search process and make the search steps are finer so there is less possibility of being trapped into a local minimal. The binary coding is used for integer p:

000	000	000	011
b_3	b_2	b_1	b_0

The value of the integer is evaluated between octal and decimal system

$$p = 8^0\times b_0 + 8^1\times b_1 + 8^2\times b_2 + 8^3\times b_3.$$

The uniform cross-over is adopted across the length of the chromosome. And the probability of mutation operation is related to the lengths of the underline genes.

3.3. Fitness function

The fitness function has a great affect on the convergence speed of a GA process. To drive the GA search process towards the location of the best solution, the fitness function should be able to reflect the key properties of the model. If the fitness function contains inadequate information about the model, it will not be capable of identifying a chromosome (solution) with superior characters. It would therefore take a longer time to advance in the evolution. As described above, this will reduce the convergence speed of the process. If the convergence is too slow, the search process will not be able to find the best solution in an acceptable time period. According to the rule of the GA, the fitness value should be no less than zero, and it should index the fitness of chromosomes and rise during the evolution. In the application to ARX modelling, the fitness value should describe the quality of an ARX model. In the literature of ARX modelling, prediction error is often used as an index for model quality. Generally the normalised model OSA prediction error can be defined as follows:

$$e = \sqrt{\frac{\sum_{i=1}^{n}(y(i)-\hat{y}(i))^2}{\sum_{i=1}^{n}y(i)^2}}, \tag{4}$$

where $y(i)$ is the measured response, and $\hat{y}(i)$ the model prediction of the response $y(i)$. To make sure that the fitness value increases for better generations or models during the evolutionary procedure, one can define the fitness function as

$$f_{\mathrm{I}}(\{y\},\{u\}\{\phi\},\{\theta\},p,q) = \frac{1}{e} = \sqrt{\frac{\sum_{i=1}^{n}y(i)^2}{\sum_{i=1}^{n}(y(i)-\hat{y}(i))^2}}. \tag{5}$$

The f_{I} cannot capture all the critical information of a time series model, for the *RMS* of the prediction error sequence is unable to justify the model structure, especially for the NARX models. One denotes the OSA prediction error or residual as

$$\varepsilon(i) = y(i) - \hat{y}(i),$$

where $\hat{y}(i)$ is the OSA prediction from Eqs. (1b) and (1c). If there are processing terms unmodelled, their influences will be left in the prediction residuals. Therefore, the prediction error ε will be correlated with y or u, or both depending on the particular case [13]. The correlation function between them can be evaluated by

$$\begin{aligned} R_{y,\varepsilon}(\tau) &= \frac{1}{n-\tau}\sum_{i=0}^{n-\tau} y(i)\varepsilon(i+\tau), \\ R_{u,\varepsilon}(\tau) &= \frac{1}{n-\tau}\sum_{i=0}^{n-\tau} u(i)\varepsilon(i+\tau). \end{aligned} \tag{6}$$

The OSA prediction error is included in the correlation functions $R_{y,\varepsilon}$ and $R_{u,\varepsilon}$ which are used to validate a fitted model in time series modelling. Therefore, OSA prediction will be used in the following new fitness function as well. The new fitness function is defined as

$$f_{\mathrm{II}}(\{y\},\{u\}\{\theta\},\{\phi\},p,q) = \frac{1}{e+\rho_{y,\varepsilon}+\rho_{u,\varepsilon}}, \tag{7}$$

where

$$\rho_{y,\varepsilon} = \sqrt{\sum_{\tau=1}^{n-1} R_{y,\varepsilon}^2(\tau) \Big/ \sum_{i=1}^{n} y(i)^2},$$

$$\rho_{u,\varepsilon} = \sqrt{\sum_{\tau=1}^{n-1} R_{u,\varepsilon}^2(\tau) \Big/ \sum_{i=1}^{n} y(i)^2}.$$

It is evident that the best model structure and coefficients will maximise f_{II} because in this case e, $\rho_{y,\varepsilon}$ and $\rho_{u,\varepsilon}$ are at their minimum. For linear models, the fitness function of Eq. (7) will be adequate for

a correct model structure, but not enough for NARX models. To validate a non-linear time series model, more correlation functions are needed to test the model [1,7]:

$$R_{\varepsilon,\varepsilon y}(\tau) = \frac{1}{n-\tau}\sum_{i=0}^{n-\tau}\varepsilon(i)\varepsilon(i+\tau)y(i+\tau),$$
$$R_{\varepsilon,y^2}(\tau) = \frac{1}{n-\tau}\sum_{i=0}^{n-\tau}\varepsilon(i)y^2(i+\tau),$$
$$R_{\varepsilon^2,y^2}(\tau) = \frac{1}{n-\tau}\sum_{i=0}^{n-\tau}\varepsilon^2(i)y^2(i+\tau). \tag{8}$$

For a correct NARX model structure the following relations should hold:

$$R_{\varepsilon,\varepsilon}(\tau) = \delta(\tau),$$
$$R_{u,\varepsilon}(\tau) = 0,$$
$$R_{\varepsilon,\varepsilon y}(\tau) = 0,$$
$$R_{\varepsilon,y^2}(\tau) = 0,$$
$$R_{\varepsilon^2,y^2}(\tau) = 0. \tag{9}$$

Therefore, the following fitness function for non-linear models should give better performance:

$$f_{\mathrm{II}}(\{y\},\{u\}\{\theta\},\{\phi\},p,q,s) = \frac{1}{e+\rho_{y,\varepsilon}+\rho_{u,\varepsilon}+\rho_{\varepsilon,\varepsilon y}+\rho_{\varepsilon,y^2}+\rho_{\varepsilon^2,y^2}}, \tag{10}$$

where

$$\rho_{\varepsilon,\varepsilon y} = \sqrt{\sum_{i=0}^{n-1}R^2_{\varepsilon,\varepsilon y}(\tau)\Big/\sum_{i=0}^{n}y(i)^2}, \quad \rho_{\varepsilon,y^2} = \sqrt{\sum_{i=0}^{n-1}R^2_{\varepsilon,y^2}(\tau)\Big/\sum_{i=0}^{n}y(i)^2},$$
$$\rho_{\varepsilon^2,y^2} = \sqrt{\sum_{i=0}^{n-1}R^2_{\varepsilon^2,y^2}(\tau)\Big/\sum_{i=0}^{n}y(i)^2}.$$

4. Example

4.1. Simulated dynamic system

To demonstrate the performance of the new fitness function, a single degree of freedom dynamic system was simulated. Two cases were investigated: linear and non-linear.

Case 1: A linear system. The equation of motion of the system is

$$\ddot{y} = a_1\dot{y} + a_0 y + b_0 u \tag{11}$$

in which y represents the system response and u the driving signal which is a stationary random noise with mean = 0 and RMS = 1.0. The physical parameters of the system are given as

$$a_1 = -1.445,\ a_0 = -5221,\ b_0 = 1.$$

The equivalent discrete time equation is

$$y(k) = 1.467y(k-1) - 0.986y(k-2) + 9.928\times10^{-5}u(k-1).$$

Case 2: A non-linear system. For the non-linear system the only difference in its equation of motion from the linear one is a squared term

$$\ddot{y} = a_1\dot{y} + a_0 y + a_{11}y^2 + b_0\ddot{u}, \tag{12}$$

where

$$a_{11} = -783.152.$$

Again the equivalent discrete time equation is

$$y(k) = 1.467y(k-1) - 0.986y(k-2) - 7.335 \times 10^{-2}y^2(k-1) + 9.928 \times 10^{-5}u(k-1).$$

The input signal is a limited band Gaussian noise with frequency band 5–15 Hz. The sampling frequency is 100 Hz considering the harmonic components of the non-linear term.

4.2. ARX models

For the linear system, the following ARX model is fitted:

$$\hat{y}(k) = \begin{cases} y(k) & k \leqslant p, \\ \phi_1 y(k-1) + \cdots + \phi_p y(k-p) & \\ +\theta_1 u(k-1) + \cdots + \theta_q u(k-q) & k > p, \end{cases} \tag{13}$$

in which $\hat{y}(k)$ is the model prediction of the system response, p and q the number of time lags of the input and the output of the model which are the parameters determining the model structure. The non-linear system is modelled with a non-linear ARX model

$$\hat{y}(k) = \begin{cases} y(k) & k \leqslant p, \\ \phi_1 y(k-1) + \cdots + \phi_p y(k-p) & \\ +\phi_{11} y^2(k-1) \ldots \phi_{s1} y^2(k-s) & k > p. \\ +\theta_1 u(k-1) + \cdots + \theta_q u(k-q & \end{cases} \tag{14}$$

The model prediction errors are calculated by the following formula:

$$e = \sqrt{\frac{\sum [y(k) - \hat{y}(k)]^2}{\sum y(k)^2}}. \tag{15}$$

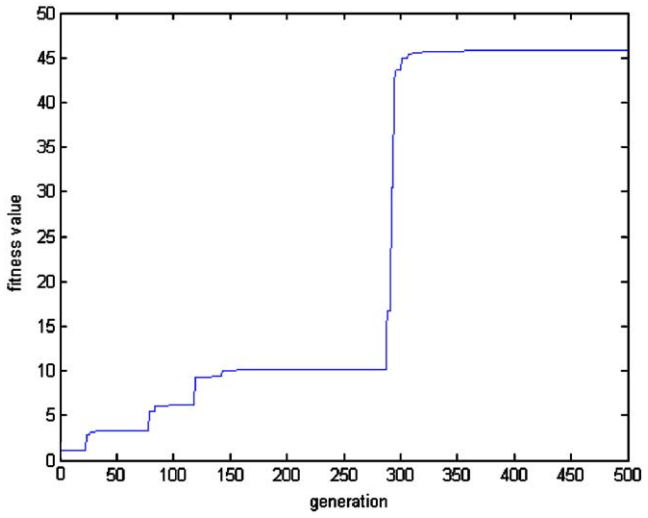

Fig. 2. Trend of f_1 for linear model.

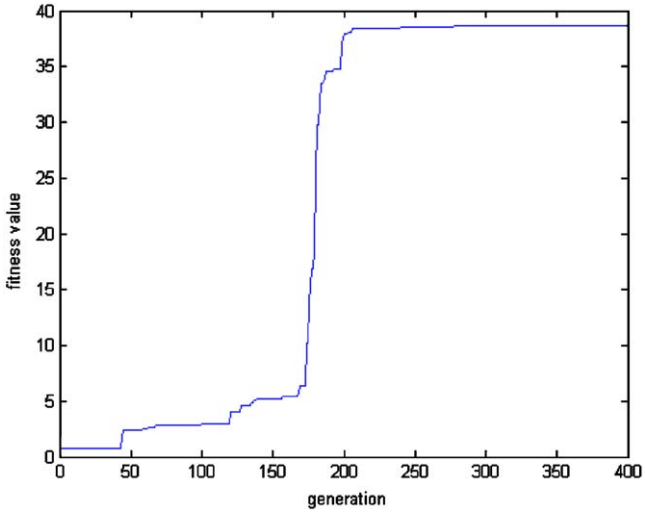

Fig. 3. Trend of f_{II} for linear model.

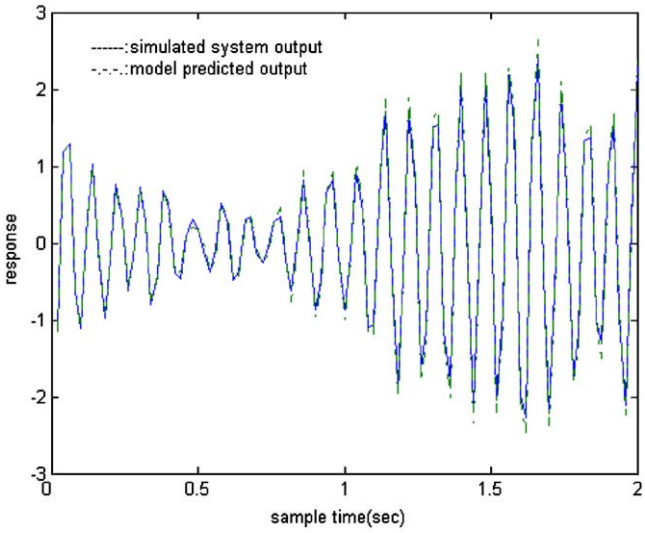

Fig. 4. Prediction of the linear model with (f_I).

4.3. Model performance and results

The results of the ARX model in the form of Eq. (13) for the linear system governed by Eq. (11) are shown in Figs. 2–5 and Table 1. Figs. 2 and 3 give the trends of the performance of the two fitness functions. The new fitness function f_{II}, shows a clear advantage over the old one f_I, in convergence speed. Figs. 4 and 5, in which the solid line represents the simulated response and the broken line the predicted response by the model, show the prediction errors of the two models based on f_I and f_{II}, respectively. A better prediction from the model with f_{II} is observed in terms of the *RMS* of the prediction error sequence. This simple example fails to show the new fitness function's potential power in optimising the model structure for both fitness functions can pick up the best model structure in such a simple linear system. However, f_{II}does make the evolution process reach its best generation faster than f_I From Fig. 2, it can be seen that the genes mapping the number of time lags get the best fitness value among the population at the 286th generation. From Fig. 3, it is observed that it took

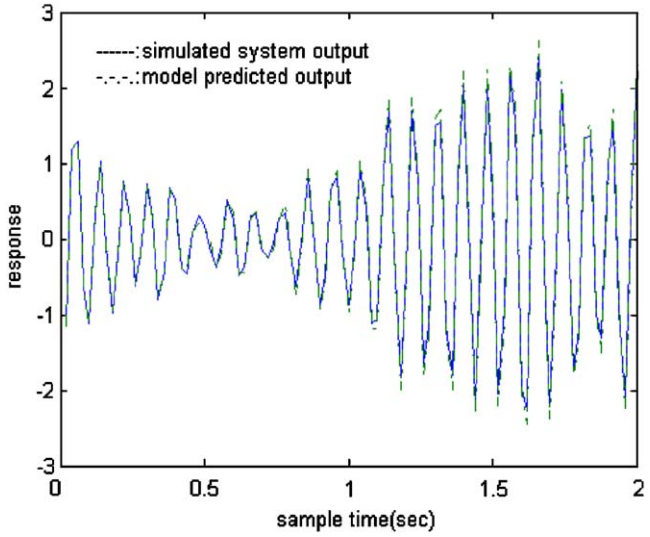

Fig. 5. Prediction of the linear model with (f_{I}).

Table 1
Linear ARX model parameters

Fitness function	ϕ_1	ϕ_2	θ_1	p	q	*RMS*
f_{I}	1.48437	−0.97850	9.155250e−5	2	1	0.09870
f_{II}	1.467297	−0.98565	9.928268e−5	2	1	0.09211

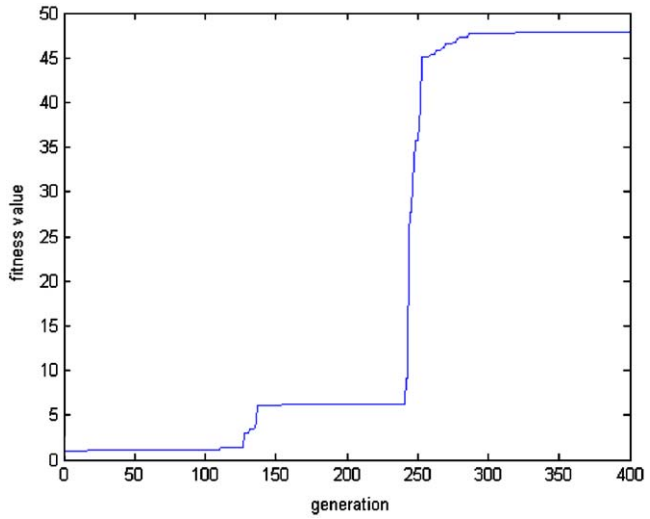

Fig. 6. Trend of f_{I} for non-linear model.

GA 201 generations for the genes of the number of time lags to get the best fitness value. Table 1 gives the parameter values and prediction errors in RMS of the two models for comparison.

For the non-linear system of Eq. (12), a non-linear ARX model similar to Eq. (14) was obtained with the two fitness functions. Figs. 6–9 are the fitness function performance trends and system response prediction of the two non-linear models based on f_{I} and f_{II}. Table 2 shows the model parameters and their errors for

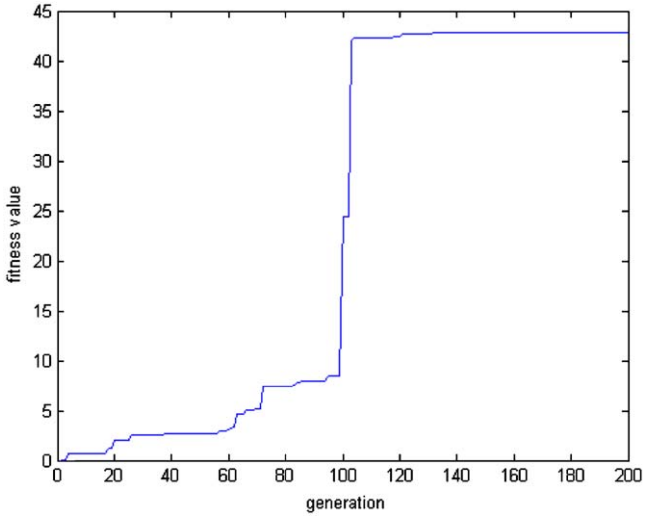

Fig. 7. Trend of f_{II} for non-linear model (f_I).

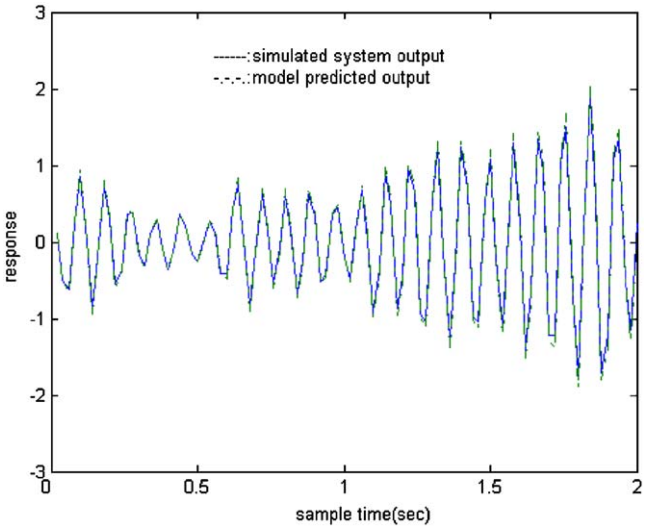

Fig. 8. Prediction of non-linear model (f_I).

comparison. Fig. 7 shows that with the fitness function f_{II} the genes mapping the number of time lags get the best fitness function value at the 100th generation. In contrast, with f_I, the same genes get the best fitness function value at about the 280th generation which is about half the speed of the new fitness function (Fig. 6). It can also be seen from these two figures that once the genes for the correct number of time lags emerge in the population, the best fitness function value increases dramatically and a speedy convergence occurs soon after. It can also be observed from Figs. 8 and 9 and Table 2 that the GA can also correctly identify the number of non-linear items in the non-linear ARX model (denoted by s in Eq. (14)). It should be pointed out that the RMS of the prediction error of the non-linear model with f_{II} is notably smaller than the model with f_I because the coefficient of the non-linear term is inaccurate for the model using f_I, which clearly shows the benefit of the new fitness.

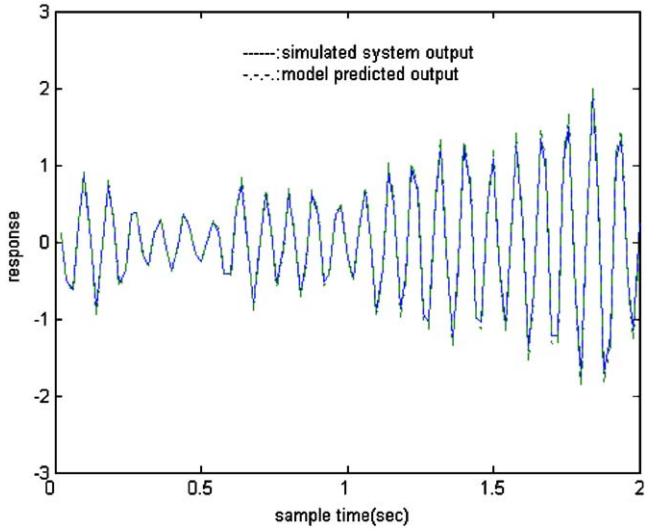

Fig. 9. Prediction of non-linear model (f_{II}).

Table 2
Nonlinear ARX model parameters

Fitness function	ϕ_1	ϕ_2	ϕ_{11}	θ_1	p	q	s	RMS
f_I	1.49106	−9.92171e−1	−3.88737e−2	9.52587e−5	2	1	1	1.431
f_{II}	1.467297	−9.856537e−1	−7.775347e−2	9.928268e−5	2	1	1	1.343

Table 3
GA results of linear ARX model parameters from ten runs with f_{II}

ϕ_1	ϕ_2	θ	p	q
1.489820	−9.843464e−1	9.155262e−5	2	1
1.489845	−9.843750e−1	9.155180e−5	2	1
1.492188	−9.879799e−1	9.180000e−5	2	1
1.492188	−9.876347e−1	9.155250e−5	2	1
1.489845	−9.843750e−1	9.155250e−5	2	1
1.489788	−9.843750e−1	9.196158e−5	2	1
1.489792	−9.843750e−1	9.196147e−5	2	1
1.489788	−9.843750e−1	9.196019e−5	2	1
1.489626	−9.841604e−1	9.197538e−5	2	1
1.489775	−9.843750e−1	9.196275e−5	2	1
1.4902655	**−9.8503714e−1**	**9.1783079e−5**	**2**	**1**

The bold values are the averages of the parameters.

To verify the stability and robustness of the GA process, ten runs were completed for both linear and non-linear ARX models. The results show the sound stability and convergence in the values of the parameters (Tables 3 and 4) and the fitness function plots (Figs. 10 and 11). The parameter values in the last rows of Table 3 and 4 are the means of the corresponding values. In the linear model case, the average number of convergence generations is 650, and the average final fitness function value is 157.7. While in the non-linear model case, the average number of convergence generations is 1000 with average final fitness function value 342.7.

Table 4
GA results of NARX model parameters from ten runs with f_{II}

ϕ_1	ϕ_2	ϕ_{11}	θ	p	q	s
1.489744	−9.942663e−1	−6.238163e−2	9.155250e−5	2	1	1
1.489744	−9.842911e−1	−6.249994e−2	9.155250e−5	2	1	1
1.489836	−9.843740e−1	−6.238163e−2	9.155262e−5	2	1	1
1.490171	−9.843740e−1	−6.249517e−2	9.195495e−5	2	1	1
1.489782	−9.843750e−1	−5.026096e−2	9.195495e−5	2	1	1
1.489838	−9.843740e−1	−1.015624e−1	9.155262e−5	2	1	1
1.489744	−9.843140e−1	−5.026096e−1	9.155250e−5	2	1	1
1.492188	−9.877920e−1	−2.243992e−3	9.179092e−5	2	1	1
1.484373	−9.771919e−1	−6.249971e−2	9.231572e−5	2	1	1
1.489820	−9.843349e−1	−1.015624e−1	9.155262e−5	2	1	1
1.489524	**−9.849687e−1**	**−1.0704974e−1**	**9.173319e−5**	**2**	**1**	**1**

The bold values are the averages of the parameters.

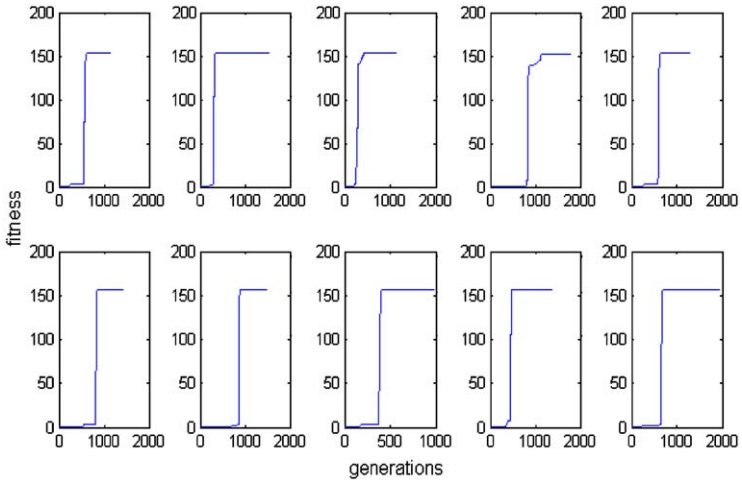

Fig. 10. Plots of fitness function f_{II} from ten runs on ARX model.

5. Conclusion

It can be concluded from the work reported in this article that the new fitness function based on the correlation function sequences between the prediction errors, the system response variables and the predictor variables can improve the performance of the GA in the ARX modelling process. There are two aspects: (i) better estimation of the model parameters and structures; (ii) faster convergence speed. This is because the new fitness function contains more information about the model quality. This work implies that more terms based on model validation functions of the time series models [4], could also be introduced into the fitness function for modelling non-linear dynamic systems for better model structures and more accurate model parameters. It should be pointed out that this paper is a largely heuristic one dealing with the application of GAs to dynamic modelling issues. Rigorous mathematical analysis and proof are not a major concern of the paper. The idea of the extended fitness function is quite straightforward and it is not really necessary to introduce mathematical procedures for this simple numerical step. We are conscious that the paper might be perfected if a rigorous mathematic procedure could be introduced. However, there are formidable problems in carrying out a rigorous analysis of the GA (the exception to this being Vose's analysis of the Simple GA). The idea of the extension of the fitness function is a simple one, and we consider this is one of the advantages of the idea. The idea of using arbitrary weighting factors is not necessary here, the point is that the method works with the trivial weighting scheme. More general weightings may be considered in future work.

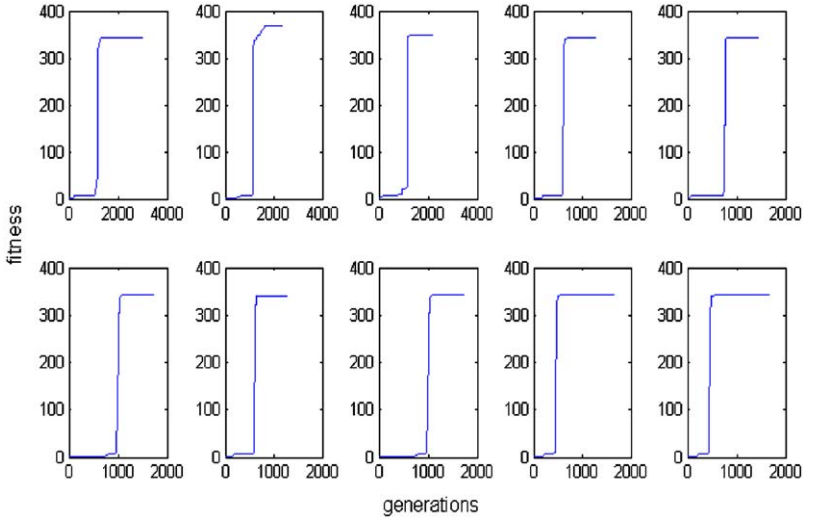

Fig. 11. Plots of fitness function f_{II} from ten runs of NARX model.

Acknowledgements

This work is a part of the project funded by Natural Science Foundation Of China (Project Grant No. 60234010). Authors would like to thank Mr. W Cheng of College of Computing Science, who helped to improve the software code and carried out the stability test of GA process.

References

[1] K. Worden, G.R. Tomlinson, Nonlinearity in Structural Dynamics, IOP Publishing Ltd., Amsterdam, 2001.
[2] Q. Chen, G.R. Tomlinson, Parametric identification of systems with dry friction and nonlinear stiffness using a time series model, Journal of Vibration and Acoustics 118 (1996) 252–263.
[3] S.F. Masri, A.G. Chassiakos, T.K. Caughey, Structure-unknown non-linear dynamic systems: identification through neural networks, Smart Materials and Structures 1 (1992) 45–56.
[4] G.C. Goodwin, M. Gevers, B. Ninness, Quantifying the error in estimated transfer functions with applications to model order selection, IEEE Transactions on Automatic Control 37 (1992) 913–929.
[5] L. Ljung, L. Guo, The role of model validation for assessing the size of the unmodelled dynamics, IEEE Transactions on Automatic Control 42(9) (1997).
[6] H. Sarimveis, A. Alexandridis, S. Mazarakis, G. Bafas, A new algorithm for developing dynamic radial basis function neural network model based on genetic algorithms, Computers and Chemical Engineering 28 (2004) 209–217.
[7] S. Chen, S.A. Billings, Representations of nonlinear systems: the NARX model, International Journal of Control 49 (1989) 1013–1032.
[8] Q. Chen, G.R. Tomlinson, A new type of time series model for the identification of nonlinear system, Mechanical Systems and Signal Processing 8 (5) (1994) 531–549.
[9] K. Rodriguez-Vazquez, C.M. Fonseca, P.J. Fleming, Identifying the structure of nonlinear dynamic systems using multiobjective genetic programming, IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans 34(4) (2004).
[10] G.J.E. Rawlins, Foundations of Genetic Algorithms, Morgan Kaufmann Publishers, San Meteo, CA, 1991.
[11] J. Baker, J. Grefenstette, How genetic algorithms work: a critical look at implicit parallelism, in: Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, Palo Alto, CA, 1989, pp. 20–27.
[12] D.E. Goldberg, Genetic Algorithms in Search, Optimisation and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[13] S.A. Billings, W.S.F. Voon, Correlation based validity tests for non-linear models, International Journal of Control 44 (1) (1986) 235–244.