DOI 10.1007/s10115-004-0184-3

Knowledge and Information Systems (2005) 8: 235–256

Knowledge and Information Systems

Evolution of mathematical models of chaotic systems based on multiobjective genetic programming

Katya Rodríguez-Vázquez[1], Peter J. Fleming[2]

[1]DISCA, IIMAS, Universidad Nacional Autónoma de México, México City, México
[2]Automatic Control and Systems, Engineering, University of Sheffield, United Kingdom

Abstract. This work is concerned with the identification of models for nonlinear dynamical systems using multiobjective evolutionary algorithms. Systems modelling involves the processes of structure selection, parameter estimation, model performance and model validation and involves a complex solution space. Evolutionary Algorithms (EAs) are search and optimisation tools founded on the principles of natural evolution and genetics, which are suitable for a wide range of application areas. Due to the versatility of these tools and motivated by the versatility of genetic programming (GP), this evolutionary paradigm is proposed for this modelling problem. GP is then combined with a multiobjective function definition scheme. Multiobjective genetic programming (MOGP) is applied to multiple, conflicting objectives and yields a set of candidate parsimonious and valid models, which reproduce the original system behaviour. The MOGP approach is then demonstrated as being applicable for system modelling with chaotic dynamics. The circuit introduced by Chua, being one of the most popular benchmarks for studying nonlinear oscillations, and the Duffing–Holmes oscillator are the systems to test the evolutionary-based modelling approach introduced in this paper.

Keywords: Chaotic dynamic systems; Genetic programming; Multiobjective optimisation; System modelling

1. Introduction

Evolutionary computation (EC) is a fascinating field that has attracted the attention of many researchers from diverse backgrounds. This interest in evolution-based search and optimisation algorithms has motivated new developments and applica-

Received 15 March 2003
Revised 20 February 2004
Accepted 1 June 2004
Published online 22 October 2004

tions. In this paper, genetic programming (GP), an EC paradigm, is applied where the computer structures that undergo adaptation are themselves represented as computer programs (Banzhaf et al. 1998; Koza 1992). Since the emergence of the GP paradigm, there have been an increasing amount of research working on both theory (Poli 2001; Rosca and Ballard 1999) and applications (Chen and Yeh 1996; Gray et al. 1996; Miller et al. 2000; Nikolaev and Iba 2001; Nordin 1994; Ryan and Ivan 1999; Spector and Alpern 1995; Witbrock and Neil-Reilly 1999), amongst other domains. One of this application areas is system modelling. Modelling can be defined as an experimental process of building a mathematical model from input–output observations from a system. In other words, it is the art of creating a mathematical description of an unknown dynamic system. In the context of GP, this problem is formulated as requiring the discovery of a computer program (mathematical expression of a nonlinear or linear model) that provides an acceptable prediction (near-optimal model) of a system under investigation from a given set of input–output data. System modelling is important in the study of a wide class of signal-processing problems, where applications range from analysis of marketing data (Chen and Yeh 1996; Kaboudan 1999) through modelling and prediction (Evans et al. 2001; Koza 1992; Nikolaev and Iba 2001; Oakley 1994). In each case, a model considering the best possible approximation to the dynamic system is estimated based on observation (or historical) data. However, it has long been realised that the responses of many nonlinear dynamic systems do not follow simple, regular, and predictable trajectories, but swirl around in a random-like and seemingly irregular behaviour. As long as the process involved is nonlinear, even a simple strictly deterministic model may develop such complex behaviour. This behaviour is known and defined as *chaos* and has led to developments in study of nonlinear systems. A special feature of chaos is its fundamental property, known as extreme sensitivity to initial conditions, in the sense that two sets of similar initial conditions can give rise to two dramatically different asymptotic states of the system trajectory. A second feature of chaos is its inability to predict long-term behaviour. Some studies of GP applied for chaotic systems modelling and prediction have been undertaken. Examples of them are Koza (1992), Mulloy et al. (1996) and Oakley (1994), which have the same foundations, where the problem was formulated as a symbolic regression problem and the function set was defined as

$$F = \{+, -, *, \%, sin, cos, exp, log\}$$

(% represents the protected division function).

In previous work (Evans et al. 2001; Rodríguez-Vázquez and Fleming 1999), the use of genetic programming in polynomial NARMAX structures modelling has been studied. Results have been shown to work well. Therefore, these works are the basis of this paper, which considers the use of a multiobjective fitness function evaluation applied to dynamic systems exhibiting chaotic behaviour. It is also pointed out that knowledge about the system under analysis can be included into the multiobjective cost function and, thus, a search process can be addressed to a specific region of the entire search space. The paper is then organised as follow. In Sect. 2.1, the definition of genetic programming is given. The optimisation and multiobjective optimisation concepts are defined in Sect. 2.2. How to formulate the nonlinear system modelling problems using genetic programming is answered in Sect. 2.3. The concepts of term clustering, cluster coefficients and fixed-points, features that can be known a priori and are useful information to determine the nonlinearities and address the search process into a feasible region, are presented in Sect. 3. Numerical examples to il-

lustrate the application of this alternative system modelling method are provided in Sect. 4. Finally, conclusions are drawn in Sect. 5.

2. Multiobjective genetic programming (MOGP)

In this section, the concepts of genetic programming, multiobjective optimisation and how genetic programming encoding can be applied to represent nonlinear dynamical models are discussed. The application of this nonconventional system modelling method, named here as multiobjective genetic programming (MOGP), is illustrated in Sect. 4.

2.1. Genetic programming

GP (Banzhaf et al. 1998; Koza 1992) is a subclass of genetic algorithms, GAs (Goldberg 1989). GAs are search and optimisation tools that use computational models of evolutionary processes in the design of computer-based problem solving. They work by maintaining a population of structures that evolve according to rules of selection and other operators such as recombination and mutation. In a similar way to that of natural evolution and heredity, the simple GA has a structure as shown in Program 2.1. Thus, this algorithm works on a population of *N binary fixed-length individuals*, $P(t) = \{x_1^t, \ldots, x_N^t\}$, representing search points in the space of potential solutions of a given problem. How well each individual x_i^t adapts each generation t to the problem under investigation is provided by a quality measure called the *fitness*. The population evolves, generation by generation, toward better regions of the search space by means of genetic processes, such as *selection, recombination* and *mutation*. The selection process uses the fitness measure to choose individuals of the previous generation, $(P(t-1))$, to be reproduced, favouring those of higher quality. The recombination, or crossover, operation promotes the exchange of genetic information between parent individuals, thereby producing descendants. The simplest crossover operator takes two parent individuals that exchange a portion of their chromosomes to produce two offspring. The mutation operation alters the genetic information by introducing some changes into the population. In the case of binary encoding, mutation swaps 0's with 1's, and vice versa. The evaluation process is repeated until a predefined termination criterion is met, or alternatively, until a maximum number of generations (iterations) is reached.

Program 2.1 Evolution-based algorithm

```
PROGRAM Evolution-Based Algorithm
   t=0
   Create Initial Population P(t)
   Evaluate Initial Population P(t)
   While(not termination_criterion) do
      t=t+1
      Select Individuals for Reproduction P(t) from P(t-1)
      Alter P(t)
      Evaluate New Population P(t)
   end
```

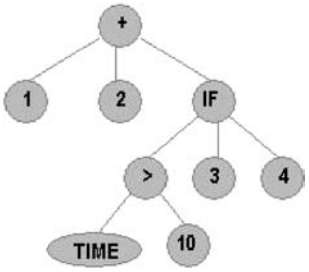


Fig. 1. A mathematical expression represented hierarchically by its parse tree

In the case of GP, the individuals (potential solutions) of the population are expressed as programs instead of individuals represented as bit strings. The fact that many problems can be expressed as computer programs makes GP a more powerful tool than its predecessor, the GA. Here, these programs, which are composed of functions and terminals appropriate to the problem domain, are encoded as hierarchical tree structures, providing a dynamic and variable representation. To illustrate the hierarchical encoding used for GP, Fig. 1 gives a simple example where the operations $F = \{+, -, *, \%, >, <, =, \text{IF}\}$ belong to the function set and the variables and constants $T = \{1, 2, 3, \ldots, 5, \text{TIME}\}$ constitute the terminal set. This hierarchical tree represents the following mathematical expression:

$$f = \begin{cases} 1+2+3=6 & \text{if TIME} > 10 \\ 1+2+4=7 & \text{otherwise}. \end{cases}$$

2.1.1. Genetic operators

Like the standard GA, the two main operators are *crossover* and *mutation*. The *crossover*, a sexual operator, works by first selecting a pair of structures from the current population. Then, a node rooted from each parent is randomly selected. These nodes become the roots for the substructures located below the crossover point. In the next step, the substructures are exchanged between the parents, producing two new structures. Because of the dynamic representation used in genetic programming, the parents are typically of different size, shape and content, and the offspring are also generally different. Figure 2 illustrates the crossover operation where the marked nodes are the roots (crossover nodes) for the subexpressions.

Mutation operates by randomly selecting a node, which can be either a terminal or internal point, and replacing the associated substructure with a randomly generated subtree up to a maximum size.

The *fitness* measure of each computer program is assigned in terms of how well it performs in the particular problem domain. This fitness value depends on the problem but is generally defined as the error produced by the computer program. In conventional genetic programming approaches, and, in general, in any evolutionary algorithm, the assigned fitness measure is based on the evaluation of a scalar function. But these population-based methods possess the characteristic of simultaneously searching for multiple solutions and, more, can evaluate several aspects of the problem. For this reason, this work presents an extension of the conventional or single fitness measure genetic programming mapping into a multiobjective genetic programming approach described in the next section.

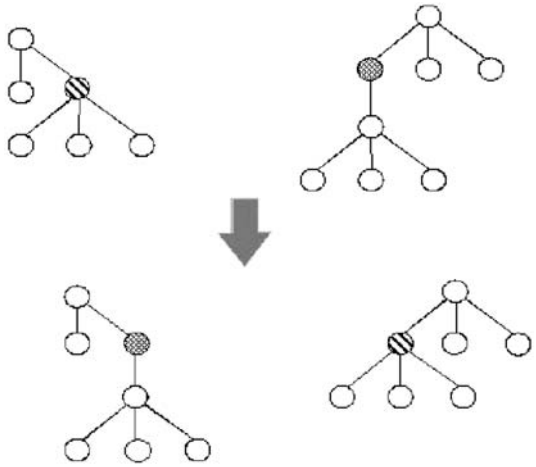


Fig. 2. GP-crossover operation

2.2. Multiobjective optimisation

As has been mentioned, nonlinear system modelling problems typically consist of multiple and competing characteristics to be evaluated. Some approaches do not consider all aspects of the identification or these are aggregated into a single objective function (e.g. Akaike information criterion [AIC] (Akaike 1974)). However, these approaches have significant drawbacks and it is preferable to treat the multiple objectives separately. Hence, multiobjective optimisation has become a useful tool for describing real problems, in general, that cannot be effectively represented in a unidimensional space. Multiobjective (also called multicriteria, multiperformance or vector) optimisation is then defined as the problem of finding:

a vector of decision variables that satisfies constraints and optimises a vector function, the elements of which represent the objective functions. These functions are usually in conflict with each other. Hence, the term optimise means finding such a solution that would give values for all the objective functions acceptable to the designer.

Expressing this definition mathematically, it can be stated as finding the vector $\overline{x}^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$ that satisfies the m inequality constraints,

$$g_i(\overline{x}) \geq 0 \quad i = 1, 2, \ldots, m, \tag{1}$$

the p equality constraints,

$$h_i(\overline{x}) = 0 \quad i = 1, 2, \ldots, p, \tag{2}$$

and optimises the vector function,

$$\overline{f}(\overline{x}) = [f_1(\overline{x}), f_2(\overline{x}), \ldots, f_k(\overline{x})], \tag{3}$$

where $\overline{x} = [x_1, x_2, \ldots, x_n]^T$ is the vector of decision variables.

The constraints given by 1 and 2 define the feasible region X and any point $\overline{x}$ in X defines a feasible solution. The k components of the vector $\overline{f}(\overline{x})$ represent

the noncommensurable criteria that must be considered. The constraints $g_i(\overline{x})$ and $h_i(\overline{x})$ represent the restrictions imposed on the decision variables. The vector $\overline{x}^*$ denotes the optimal solution set. In multiobjective optimisation (MO), the solution is a set of alternative solutions rather than a single optimal one. Thus, the simultaneous evaluation of model complexity, model performance and model validation of the identification procedure are performed for each member of the population of potential mathematical models. The fitness value of each population member is assigned by means of a rank-based fitness method (Fonseca and Fleming 1995). This fitness evaluation is based on the definition of Pareto-optimality or nondominance, which constitutes by itself the origin of research in multiobjective optimisation. Considering an optimisation (minimisation) problem and given two n components objective function vectors, $\overline{f}_u$ and $\overline{f}_v$, we can say that $\overline{f}_u$ dominates $\overline{f}_v$ (is Pareto-optimal) if

$$\forall i \in \{1, \ldots, n\},\ f_{u_i} \leq f_{v_i} \wedge \exists i \in \{1, \ldots, n\},\ f_{u_i} < f_{v_i}, \tag{4}$$

producing a set of possible and valid solutions known as the Pareto-optimal or admissible solution set of the problem. Selection in genetic programming is made using a method of ranking that favours nondominant individuals of the population (Fonseca and Fleming 1995).

Current research on evolutionary multiobjective optimisation algorithms propose alternative multiobjective evaluation schemes. Examples of these works are the nondominated sorting genetic algorithm (NSGA) proposed and extended by Deb et al. (2000), the Pareto archived evolution strategy (PAES) proposed by Knowles and Corne (2000) and the micro-genetic algorithm by Coello-Coello and Toscano-Pulido (2001), amongst others. A survey of research on evolutionary multiobjective optimisation (EMO) techniques can be found in Coello-Coello et al. (2002). However, these efforts have been mainly focused on simple biobjective constrained and unconstraind test problems of parameter optimisation. In our case, the main point is to solve a practical multiobjective problem evaluating attributes of model structure, performance and validation. It is important to point out that we use a Pareto-based fitness assignment based on previous work by Fonseca and Fleming (1995) but we are also able to integrate any other multiobjective evaluation technique.

2.3. Difference equation program model representation

The well-known NARMAX (nonlinear autoregressive moving average with exogenous inputs) model that is an extended ARMAX description for representing nonlinear systems has been introduced by Leontaritis and Billings (1985). This model is given by a nonlinear function, $F^{\ell}(\bullet)$, of the output, $y(k)$, the input, $u(k)$, and the possible noise disturbance, $e(k)$. Thus,

$$\begin{aligned} y(k) = F^{\ell}(y(k-1), \ldots, y(k-n_y), & u(k-1), \ldots u(k-n_u), \\ & e(k-1), \ldots e(k-n_e)) + e(k), \end{aligned} \tag{5}$$

where n_y, n_u and n_e are the maximum lags considered for the output, input and noise terms, respectively, and ℓ is the degree of nonlinearity of the model structure. Note that, if $\ell = 1$, the resulting model is a linear structure. As has been stated in previous work by Chen and Billings (1989), the most typical choice for $F^{\ell}(\bullet)$ in (5) is a polynomial expansion. The model is linear in the parameters and can, therefore, be

estimated by means of a least-squares algorithm. Then, based on the polynomial representation of the NARMAX model, the GP population consists of tree-structured individuals that readily represent alternative structures for the application of the NARMAX approach. Potential models are encoded as hierarchical tree structures, thus providing a dynamic and variable representation, and these constitute members of a population of different model structures. These structures consist of functions (internal nodes) and terminals (leaf nodes) appropriate to the problem domain. Hence, the function set is here defined as $F = \{ADD, MULT\} = \{+, *\}$ and the terminal set as $T = \{X_0, X_1, \ldots, X_{ny}, X_{ny+1}, \ldots, X_{ny+nu}, X_{ny+nu+1}, \ldots, X_{ny+nu+ne}\} = \{c, y(k-1), \ldots, y(k-n_y), u(k-1), \ldots, u(k-n_u), e(k-1), \ldots, e(k-n_e)\}$. This hierarchical tree representation of polynomial NARMAX models is interpreted in Polish notation.[1] An example is the following expression: (ADD (ADD X1 X5) (MULT (ADD X2 X3)(ADD X1 X4))). This is equivalent to the polynomial nonlinear model defined as

$$\begin{aligned} y(k) = {} & \theta_0 + \theta_1 e(k-1) + \theta_2 y(k-1) + \theta_3 y(k-2) \\ & + \theta_4 y(k-1)u(k-1) + \theta_5 y(k-2)u(k-1), \end{aligned} \tag{6}$$

where $\{X1, X2, X3, X4, X5\} = \{1.0$(the constant term), $y(k-1)$, $y(k-2)$, $u(k-1)$, $e(k-1)\}$. A least-squares algorithm is applied to compute the parameter vector θ_i to minimise the residual of errors $\varepsilon(k)$ between the measured output $y(k)$ and the predicted output $\hat{y}_k$ that is given by

$$\varepsilon(k) = y(k) - \hat{y}(k, \hat{\theta}). \tag{7}$$

This parameter estimation algorithm works by first calculating the process terms coefficients and, by means of (7), the residuals are computed. Once the residuals are known, these are incorporated into the model and a new set of parameters is estimated.

3. Term clustering and fixed points

The concepts of term clustering and cluster coefficients are useful to extract some dynamical properties such as the number of fixed points. The fixed points are directly related to the degree of nonlinearity of the output, as described below, and these provide useful information that can be integrated into the MOGP method. In order to analyse the relationship between the structure of NARMAX polynomials and their respective fixed points, some definition are given as follow.

3.1. Term clustering and fixed points

In term clustering (Aguirre and Mendes 1996), the deterministic part of a NARMAX model, that is the NARX model, is expanded as the summation of terms with degrees of nonlinearity in the range $1 \leq m \leq \ell$. Each mth order term can contain a pth order

[1] This reverse polish expression can be also represented as a tree being the root node $y(k)$, lagged input, output and noise terms and constant representing the terminal set and the function set consisting of addition and products.

factor in $y(k-n_i)$ and a *(m-p)*th order factor in $u(k-n_i)$ and is multiplied by a cluster coefficient Cp, $m\text{-}p(n_1, \ldots, n_m)$ as follows:

$$y(k) = \sum_{m=0}^{\ell} \sum_{p=0}^{m} \sum_{n_1,n_m}^{n_y,n_u} C_{p,m-p}(n_1, \ldots, n_m) \prod_{i=1}^{p} y(k-n_i) \prod_{i=p+1}^{m} u(k-n_i), \tag{8}$$

where

$$\sum_{n_1,n_m}^{n_y,n_u} = \sum_{n_1=1}^{n_y} \cdots \sum_{n_m=1}^{n_u} \tag{9}$$

and the upper limit is n_y if the summation refers to factors in $y(k-n_i)$ or n_u for the case of factors in $u(k-n_i)$.

For example, expanding the summation up to second order, that is, $\ell = 2$ gives

$$\begin{aligned} y(k) = {} & C_{0,0} + \sum_{n_1}^{n_y} C_{1,0}(n_1) y(k-n_1) + \sum_{n_1}^{n_u} C_{0,1}(n_1) u(k-n_1) \\ & + \sum_{n_1}^{n_y} \sum_{n_2}^{n_y} C_{2,0}(n_1, n_2) y(k-n_1) y(k-n_2) \\ & + \sum_{n_1}^{n_y} \sum_{n_2}^{n_u} C_{1,1}(n_1, n_2) y(k-n_1) u(k-n_2) \\ & + \sum_{n_1}^{n_u} \sum_{n_2}^{n_u} C_{0,2}(n_1, n_2) u(k-n_1) u(k-n_2) \,. \end{aligned} \tag{10}$$

Definition 1 (Cluster coefficients (Mendes 1995)). The constants $\sum_{n_1,n_m}^{n_y,n_u} C_{p,m-p}(n_1, \ldots, n_m)$ are the coefficients of the *term clusters* $\Omega y^p u^{m-p}$, which contain terms of the form $y(k-i)^p u(k-j)^{m-p}$ for $m = 0, \ldots, \ell$ and $p = 0, \ldots, m$. Such coefficients are called cluster coefficients represented as $\sum_{y^p u^{m-p}}$.

From the last definition, one can say that the set of all candidate terms for a NARX model is the union of all possible clusters up to degree ℓ. That is,

$$\begin{aligned} \{\text{All possible terms}\} = {} & \bigcup_{\substack{p=0,\ldots,m \\ m=0,\ldots,\ell}} \Omega_{y^p u^{m-p}} \\ = {} & \text{constants} \cup \Omega_y \cup \Omega_u \cup \Omega_{y^2} \cup \Omega_{yu} \quad \cup \Omega_{u^2} \cup \ldots \cup \\ & \text{all possible combinations up to degree } \ell. \end{aligned}$$

Definition 2 (Fixed points). Defining a vector $\overline{x}$ as the state of the dynamical system and the function f that describes how the system moves, there are special circumstances where the system does not move, but stays in a special state. These states are called *fixed points* (or equilibrium points) of the dynamical system.

To illustrate this definition, an example is considered.

Given the nonlinear discrete time system

$$x(k+1) = [x(k)]^2 - 6 \tag{11}$$

and given the state $x(k) = 3$, the value at the next instance is computed as

$$x(k+1) = x(k)^2 - 6 = 3^2 - 6 = 3\,. \tag{12}$$

We observe that the system is again at state $x = 3$. This value is a fixed point of the system because, if we are ever in state 3, we remain there for all time.

In the context of the NARMAX model representation, this concept has been applied by Aguirre and Mendes (1996). The fixed points are defined as those points for which $y(k) = y(k+1), i \in Z$, and usually constitute the starting point in an analysis of nonlinear system models.

In what follows, the fixed points will be computed for the autonomous version of the system under investigation. If the original polynomial is nonautonomous, then $u(k-i) = 0, i = 0, 1, \ldots$. All possible clusters of an autonomous polynomial with degree of nonlinearity ℓ are $\Omega =$ constant, Ω_y, Ω_{y^2},..., Ω_{y^ℓ}. Hence, the fixed points of an autonomous polynomial with degree of nonlinearity are given by the roots of the clustered polynomial, expressed as

$$\begin{aligned} y(k) = C_{0,0} + y(k)\sum_{n_1=1}^{n_y} C_{1,0}(n_1) + y(k)^2 \sum_{n_1,n_2}^{n_y,n_y} C_{2,0}(n_1,n_2) + \ldots \\ + y(k)^\ell \sum_{n_1,n\ell}^{n_y,n_y} C_{\ell,0}(n_1,\ldots,n_\ell)\,. \end{aligned} \tag{13}$$

By using the definition of cluster coefficients and eliminating the argument k, the last equation can be rewritten as follows:

$$\sum_{y^\ell} y^\ell + \ldots + \sum_{y^2} y^2 + (\sum_{y} -1)y + \sum_{0} = 0\,. \tag{14}$$

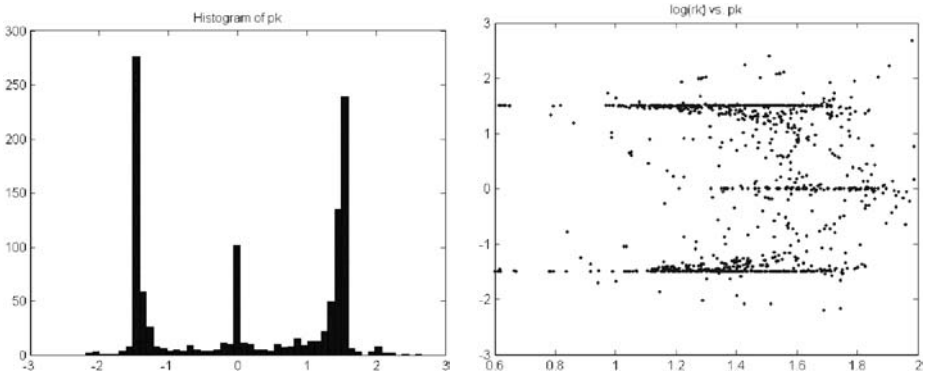
From the last equation, it becomes clear that an autonomous polynomial with degree of nonlinearity ℓ will have ℓ fixed points if $\sum_{y^\ell} \neq 0$.

Moreover, the fixed points can be directly extracted from the data. Relevant information about the system dynamics can then be obtained before the modelling procedure takes place. A routine to perform this task has been written by Mendes (1995), which is based on the algorithm proposed by Glover and Mees (1993). This procedure is explained in the following subsection.

3.2. Fixed points from data

Looking at the time series of Fig. 7, lines have been drawn inferring a possible fixed point and its position. The idea of the Glover and Mees algorithm (Glover and Mees 1993) says that an estimation of a fixed point of a linear model is given by the average value of the time series. However, in this case, the time series only behaves in a linear way for a limited time, the trajectory eventually leaves the vicinity of the fixed point and higher order terms become important. Then, there are k short sections and the process for obtaining fixed points from data starts by modelling these short sections, or windows, of the time series. If the window is $\{x_1, x_2, \ldots, x_{2m+1}\}$ then,

$$\sum_{i=k}^{k+m} c_i x_i = 1 \quad k = 1, \ldots, m+1\,. \tag{15}$$



(a) Histogram of p_k (b) Plot of $log(r_k)$ vs. p_k for $m = 6$

Fig. 3. Each k produces a single point

This AR(m) model has $m+1$ parameters and then $m+1$ equations for a window of length $2m+1$. Equation (15) can then be rewritten as,

$$X_c = 1_{m+1}, \tag{16}$$

where matrix X has elements $X_{i,j} = x_{i,j-1}$ and 1_{m+1} is a vector the elements of which are one. The solution is given by $c = X^{-1}1_{m+1}$. The fixed point of the model corresponds to a constant time series $x_i = p$ and replacing this in (15) gives

$$p^{-1} = \sum_{i=1}^{m+1} c_i = \sum_{i,j=1}^{m+1} X_{i,j}^{-1}\,. \tag{17}$$

For each window, there is a different value of p (fixed points). Then, for each k, there is a p_k to be the value of the fixed point for the linear model of the window $\{x_k, x_{k+1}, \ldots, x_{k+2m}\}$.

In order to graphically present this information, the distance between the m-dimensional points $v_k = (x_k, \ldots, x_{k+m+1})$ and the m-dimensional fixed points $q_k = (p_k, \ldots, p_k)$ is calculated. Points near the fixed point are also close to their corresponding fixed-point estimates. Thus, $r_k = ||v_k - q_k||$ is defined and a plot of $log(r_k)$ vs. pk from time series of Fig. 7 is presented (see Fig. 3). It is seen from Fig. 3 that there is a strong accumulations around ± 1.5 and a weaker one near 0.

4. Simulation results

This section provides two examples concerning the use of multiobjective genetic programming (MOGP) in modelling nonlinear dynamic systems. These examples possess the characteristics of having a chaotic motion. The first example, the well-known Duffing–Holmes oscillator (Holmes 1979), illustrates how MOGP can be applied in the identification of NARMAX models. The second example discusses the identification of models from chaotic data sets.

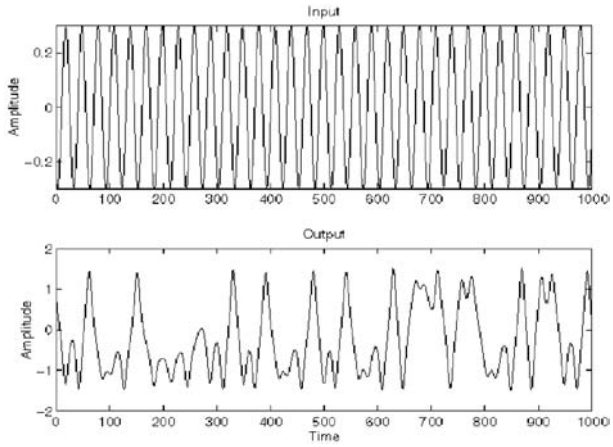


Fig. 4. Input–output data from the Duffing–Holmes equation exhibiting a chaotic regime

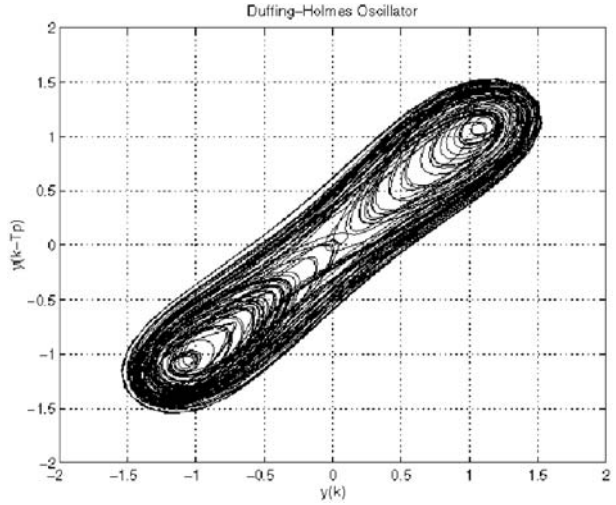


Fig. 5. Reconstructed attractor from measured data of the Duffing–Holmes oscillator

4.1. Example 1. Duffing–Holmes oscillator

The well-known Duffing–Holmes equation is commonly used to model mechanical oscillations. This system is characteristic of many structural nonlinearities encountered in practice (Hunter 1992). This system is modelled by means of the following equation:

$$\ddot{y} + \delta\dot{y} - \beta y + \alpha y^3 = Au\,. \tag{18}$$

In this equation, $\delta = 0.15$, $\beta = 1.0$, $\alpha = 1.0$, $A = 0.30$, $u = cos(\omega k)$ and $\omega = 1$ rad/s. This set of parameters drives the Duffing–Holmes equation to exhibit chaotic motion (see Figs. 4 and 5).

Table 1. GP parameters used in nonlinear systems identification

Parameter name	Description
Objective	Find a program to model the relationship of input-output data gathered from the system in study.
Terminal set	c, $y(k-1), \ldots, y(k-n_y), u(k-1), \ldots, u(k-n_u)$.
Function set	ADD, MULT.
Fitness cases	The number of pairs of data.
Fitness function	The fitness function is given in a multiobjective fashion (equations 1 and 14) and considered attributes related to the model complexity, model performance and the number and location of fixed points which constrain the search space.
GP-parameters	MaxGen = 200, PopSize = 100.
GP-operators	Crossover: 0.9, Mutation: 0.1.
Termination criterion	Maximum number of generations.

Because a priori knowledge about the system can be obtained (the number and location of fixed points extracted from the original data and also expressed in equation (18)), this information is introduced as objectives because it has a direct relationship to the dynamic model degree. The inclusion of this objective has the aim of searching in the region of models exhibiting these characteristics. Nevertheless, a disadvantage that the MOGP framework exhibits is the inability to deal straightforward with equality constraints. Therefore, in order to overcome this weakness, the objective related to model degree was redefined as

$$Obj_{DEG} = |DEG - No.FXP|, \tag{19}$$

where the number of fixed points are (the variable No. FXP from (19)) defined to be 3 as shown from Figs. 4 and 5. Thus, the distance between solution value and the constraint value must be a minimum, where the minimum value of (19) will be zero and corresponds to the correct number of fixed points of the system. MOGP can then deal directly with the minimisation of this objective. Alternative algorithms to treat equality constraints combined with evolutionary algorithms have appeared in the literature. GENOCOP (genetic algorithm for numerical optimisation for constrained problems) introduced by Michalewicz and Nazhiyath (1995) uses some ideas of repair algorithms. In the approach by Hinterding (2001), equality constraints are replaced by inequalities $h(\overline{x}) \leq \delta$ and $h(\overline{x}) \geq \delta$, where δ is adaptively adjusted based on the proportion of the population that is in the feasible region rather than being assigned a sufficiently small value.

The Duffing–Holmes system shows three equilibrium points located at $(-\sqrt{\beta}, 0, \sqrt{\beta})$ (Mendes 1995), one trivial and two symmetrical. Notice that, for $\beta = 1$, the equilibrium are located at $(-1, 0, 1)$.

Several runs of the multiobjective GP approach were carried out. Maximum lag values were set to be $n_y = n_u = 10$. The GP parameters were set up as describe in Table 1 and the set of objectives and constraints for this problem were defined as shown in Table 2. Performance error was defined as (7), while fixed point number was defined as (19) and set as constraint (the search is addressed to the region of cubic models). The long-term prediction error (LTPE) was measured in order to generate stable models.

Table 2. Description of the objectives considered in the MOGP-identification procedure

Attribute	Objective name	Description	Type
Model complexity	Model size	Number of process terms	Objective (minimise)
	Model degree	Fixed points no.	Constraint (minimise)
	Model lag	Maximum lagged input and output terms	Objective (minimise)
Model Performance	Residual variance	Variance of predicted error between measured and estimated output	Objective (minimise)
	LTPE	Long-term predictive error	Objective (minimise)

Table 3. Performance of Duffing–Holmes models obtained by means of MOGP identification method (σ_ε^2 is the residual variance)

Model	P	DEG	n_y	n_u	σ_ε^2	Fixed points
1	3	3	3	5	5.2599×10^{-3}	(−1.0782, 0, 1.0782)
2	3	3	5	7	2.2504×10^{-3}	(−1.0044, 0, 1.0044)
3	4	3	3	2	2.4610×10^{-5}	(−0.9796, 0, 0.9796)
4	5	3	2	2	6.5465×10^{-6}	(−0.9939, 0, 0.9939)
5	8	3	3	2	1.6745×10^{-8}	(−1.0015, 0, 1.0015)
6	10	3	5	2	4.5810×10^{-9}	(−1.0000, 0, 1.0000)
7	11	3	7	2	3.2077×10^{-9}	(−1.0001, 0, 1.0001)

4.1.1. Numerical results

The multiobjective GP-NARMAX method previously described is an innovative method for modelling dynamic systems. Recapitulating, the multiobjective approach has the aim of optimising different measure requirements within the modelling process. Hence, the NARMAX modelling problem is formulated as the problem of finding the vector of decision variables (the vector of model regressors, linear and nonlinear terms), $\overline{x} = [x_1, \ldots, x_m]$, that minimise the n components vector function

$$\overline{F}(\overline{x}) = [f_1(\overline{x}), \ldots, f_n(\overline{x})], \tag{20}$$

where $\overline{F}$ is the fitness function (cost function), which can include measurements related to model structure (e.g. nonlinearity degree), the goodness to fit of the estimated model, validation criteria or any criteria (e.g. location of fixed points) to evaluate the identified models.

Then, models with 3 and up to 11 terms were generated by means of this approach (see Table 3). Note that, in order to have a trivial and symmetrical nontrivial fixed points, the identified autonomous part of the models should possess terms from the clusters Ω_y and Ω_{y^3}. The cluster Ω_u was also identified; this is confirmed in (12). The inclusion of the correct terms in the model structure is detailed in Table 4.

Models 5, 6 and 7 possess terms that belong to the cluster Ω_{yu}. This is considered to be a spurious cluster due to the small values of the cluster coefficients (Table 5).

Table 4. Duffing–Holmes clusters and model structures

Model	Ω_y	Ω_u	Ω_{yu}	Ω_{y3}
1	$y(k-1)$	$u(k-5)$		$y(k-1)y(k-2)y(k-3)$
2	$y(k-1)$	$u(k-7)$		$y(k-2)y(k-4)y(k-5)$
3	$y(k-1), y(k-2)$	$u(k-2)$		$y(k-1)^2y(k-3)$
4	$y(k-1), y(k-2)$	$u(k-2)$		$y(k-1)^2y(k-2)$, $y(k-1)y(k-2)^2$
5	$y(k-1), y(k-2)$, $y(k-3)$	$u(k-1), u(k-2)$	$y(k-1)u(k-2)$	$y(k-1)^3$, $y(k-1)^2y(k-2)$
6	$y(k-1), y(k-2)$, $y(k-3)$	$u(k-1), u(k-2)$	$y(k-1)u(k-2)$	$y(k-1)^3$, $y(k-1)^2y(k-2)$, $y(k-1)^2y(k-5)$, $y(k-1)y(k-2)y(k-5)$
7	$y(k-1), y(k-2)$, $y(k-3)$	$u(k-1), u(k-2)$	$y(k-1)u(k-2)$	$y(k-1)^3$, $y(k-1)^2y(k-2)$, $y(k-1)^2y(k-5)$, $y(k-1)^2y(k-7)$, $y(k-1)y(k-2)y(k-5)$

Table 5. Duffing–Holmes cluster coefficients

Model	Ω_y	Ω_u	Ω_{yu}	Ω_{y3}
1	1.1442×10^{0}	2.8242×10^{-1}		-1.2403×10^{-1}
2	1.1463×10^{0}	1.9213×10^{-1}		-1.4503×10^{-1}
3	1.0436×10^{0}	4.8607×10^{-2}		-4.5474×10^{-2}
4	1.0435×10^{0}	5.1880×10^{-2}		-5.1965×10^{-2}
5	1.0357×10^{0}	3.5589×10^{-2}	-2.7461×10^{-5}	-3.5655×10^{-2}
6	1.0260×10^{0}	2.6017×10^{-2}	-4.4668×10^{-5}	-2.6004×10^{-2}
7	1.0282×10^{0}	2.8197×10^{-2}	-1.3635×10^{-5}	-2.8171×10^{-2}

The comparison of the original system and the identified models is presented in Fig. 6, where the attractor is reconstructed using these models. Estimated output $\hat{y}(k)$ is computed as

$$\hat{y}(k) = f^{\ell}(\hat{y}(k-1), \ldots, \hat{y}(k-n_y), u(k-1), \ldots, u(k-n_u)) \,. \tag{21}$$

The first n_y output points are taken from measured data, and the remaining points are calculated by means of models given in Table 4. Although models 1 and 2 show the desired number and location of equilibrium, their associated attractors suggest that, in order to describe the dynamics of the system, models must include the output linear terms and lag of input terms must be small.

These results show how MOGP can address simultaneously model performance and model structure of system identification problems. Only information about the number of fixed points (nonlinearity degree of autonomous part of the model) was considered in the MOGP framework in order to direct the search into the feasible zone (search space of cubic models). No information about term clustering was involved; nonetheless, MOGP was capable of generating a family of solutions that

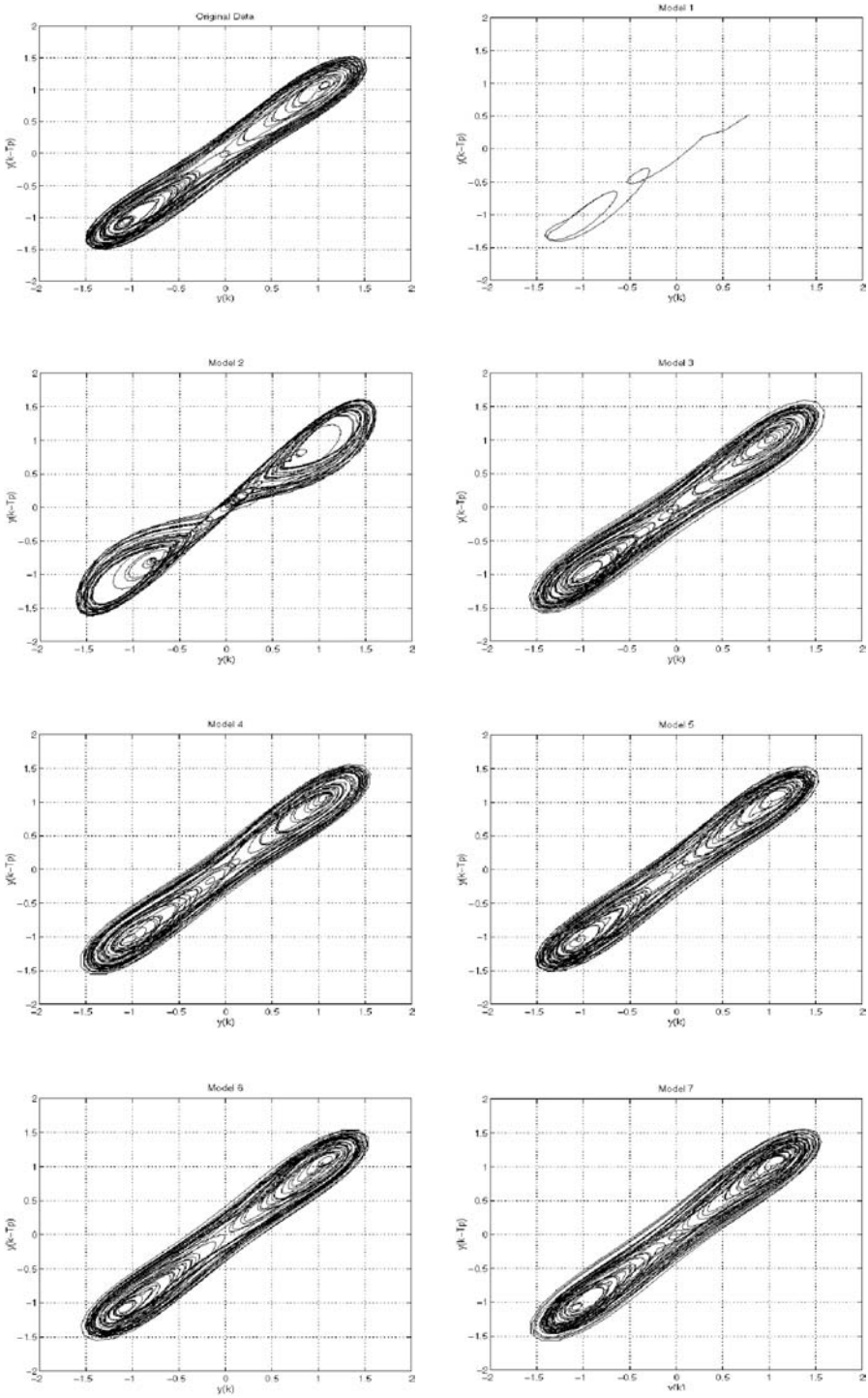


Fig. 6. Duffing–Holmes embedded attractors reconstructed from identified models

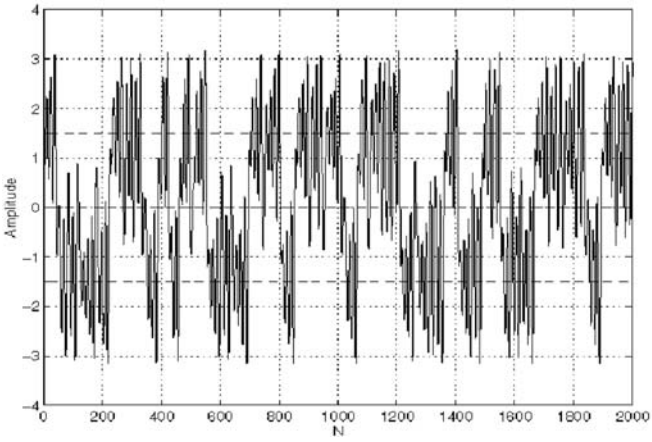


Fig. 7. Z-coordinate in the Chua circuit

included the correct clusters. As seen from these results, MOGP generates a family of equivalent solutions (based on the concept of Pareto optimality). It is seen from Tables 3 and 4 that there are no solutions better in all objectives. These objectives are in conflict with each other. Thus, these solutions are equivalent, nondominant and satisfy the prescribed goals.

4.2. Example 2. Chua circuit

This section discusses the modelling of nonlinear polynomial models from chaotic data sets. The Chua circuit (Chua and Hasler 1993) is one of the most popular benchmarks for studying nonlinear oscillations. The normalised equations of the Chua circuit can be written as (Chua and Hasler 1993; Chua et al. 1986):

$$\begin{cases} \dot{x} = \alpha(y - h(x)) \\ \dot{y} = x - y + z \\ \dot{z} = -\beta y \end{cases} \qquad h(x) = \begin{cases} m_1 x + (m_0 + m_2) & x \geq 1 \\ m_0 x & |x| \leq 1 \\ m_1 x - (m_0 + m_1) & x \leq -1, \end{cases} \tag{22}$$

where $m_0 = -1/7$ and $m_1 = 2/7$. The variation of α and β parameters drives the system to display several regular and chaotic regimes. The well-known double-scroll attractor is obtained for $\alpha = 9.0$ and $\beta = 100/7$. Based on these values, 22 was simulated using a Runge–Kutta method of 4th-order with step size of 0.001.[2] The z-coordinate of the double-scroll attractor was then chosen for this example (see Figs. 7 and 8).

Selecting the input terms in the NARMAX model formulation to be zero gives the nonlinear auto regressive moving average (NARMA) model description (Leontaritis and Billings 1985),

$$y(k) = F^{\ell}(y(k-1), \ldots, y(k-n_y), e(k-1), \ldots e(k-n_e)) + e(k)\,. \tag{23}$$

The identification technique previously described in this paper is used to identify nonlinear polynomial models of this form.

[2] Thanks are given to Eduardo Mendes, who performed this simulation and provided the data.

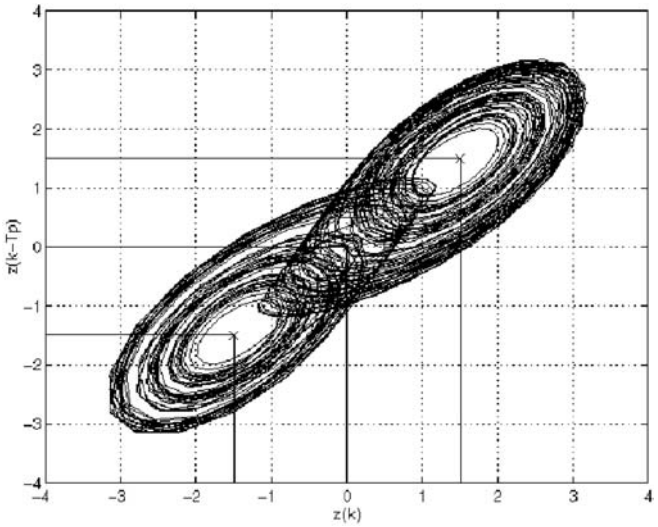


Fig. 8. Double-scroll attractor obtained from the simulation of (21)

4.2.1. Validation

Besides the inclusion of fixed points in the multiobjective cost function, statistical validation functions are also considered in this example. Thus, the multiobjective identification method is applied and a family of nonlinear candidate models based on the evaluation of the performance, complexity and validity criteria can emerge. Regarding the validation criteria, Billings and Tao (1991) have introduced time-series validation tests based on general correlations. These are given by

$$\begin{cases} \Phi_{\varepsilon'\varepsilon'}(\tau) = E\left[(\varepsilon(k) - \overline{\varepsilon(k)})(\varepsilon(k-\tau) - \overline{\varepsilon(k)})\right] = \delta(t) \\ \Phi_{\varepsilon'(\varepsilon'^2)}(\tau) = E\left[(\varepsilon(k) - \overline{\varepsilon(k)})(\varepsilon^2(k-\tau) - \overline{\varepsilon^2(k)})\right] = 0 \\ \Phi_{(\varepsilon'^2)(\varepsilon'^2)}(\tau) = E\left[(\varepsilon^2(k) - \overline{\varepsilon^2(k)})(\varepsilon^2(k-\tau) - \overline{\varepsilon^2(k)})\right] = \delta(\tau)\,. \end{cases} \tag{24}$$

Thus, the objectives considered in this example are as defined in Table 2, adding three validation criteria given by (24) and defined as constraints to be minimised. For the validation purpose, it is desirable that the fixed points of the estimated models be as close as possible to the fixed points of the original system. Based on the concept of symmetry of fixed points, it is stated in Mendes (1995) that the fixed points of a cubic polynomial with a $(-z, 0, z)$ symmetry are obtained from the cluster polynomial (see (14)), where $\sum_0 = 0$. This guarantees that the respective dynamical polynomial model should not have any terms taken from the clusters Ω_0 and Ω_{y^2}. In this case, the symmetrical fixed points are at

$$\overline{y} = \pm\sqrt{\frac{\sum_y - 1}{\sum_{y^3}}}\,. \tag{25}$$

Table 6. Set of polynomial models used to reconstruct the double-scroll attractor plotted in Fig. 8

Model	p	DEG	LAG	σ_ε^2 ($\times 10^{-5}$)	Fixed point location
1	9	3	3	3.5222	(−1.5323, 0, 1.5323)
2	10	3	4	3.1718	(−1.5341, 0, 1.5341)
3	12	3	4	1.3427	(−1.5096, 0, 1.5096)
4	13	3	4	1.2139	(−1.5013, 0, 1.5013)
5	14	3	4	1.2133	(−1.5023, 0, 1.5023)
6	14	3	5	1.1768	(−1.5028, 0, 1.5028)
7	14	3	8	1.1668	(−1.5048, 0, 1.5048)

It is interesting to note that the double-scroll attractor possesses these characteristics and the results show that the dynamical polynomial models identified using GP only possess linear and cubic terms in their structures. Table 6 gives the location of the fixed points for each model and details of their structures. An additional feature for the validation of chaotic models should be to verify if such models settle to attractors that resemble the geometry of the original data. Thus, the embedded attractors of the identified models are shown in Fig. 9.

As mentioned above, three additional criteria defined as constraints were evaluated in this example.

From these results, the most parsimonious model (model 1) that was identified and can reproduce the double-scroll attractor consists only of nine terms. The structure is given as

$$\begin{aligned} z(k) = {} & 2.9579z(k-1) - 2.9369z(k-2) + 1.0354z(k-3) - 0.6932z(k-1)^3 \\ & + 2.3307z(k-1)^2z(k-2) + 1.9162z(k-1)z(k-2)z(k-3) \\ & - 0.4732z(k-1)z(k-3)^2 - 1.1913z(k-1)^2z(k-3) \\ & - 1.9134z(k-1)z(k-2)^2 . \end{aligned} \tag{26}$$

5. Conclusions

Genetic programming is a technique that has been developed for the purpose of evolving programs. Combining this technique with the general NARMAX representation has proved to be a powerful tool for nonlinear system modelling, demonstrating effective performance. The application area of this technique has been tested, in this paper, on two typical benchmark modelling problems that exhibit chaotic motions.

Using validation and additional information extracted from the original data prior to the identification, MOGP was able to find models that could reproduce the dynamics presented in the system under investigation. In the case of the Chua circuit, the set of models bred using MOGP was able to reproduce the attractors that resemble the geometry of the original data.

In the study of the Duffing–Holmes oscillator, spurious terms were included in some of the nondominated models. Nevertheless, the resulting models were able to reproduce the original attractor trajectories. The two most parsimonious models possessed the correct number and approximate location of the equilibrium points. However, they were unable to describe the chaotic regime and reproduce the attractor of

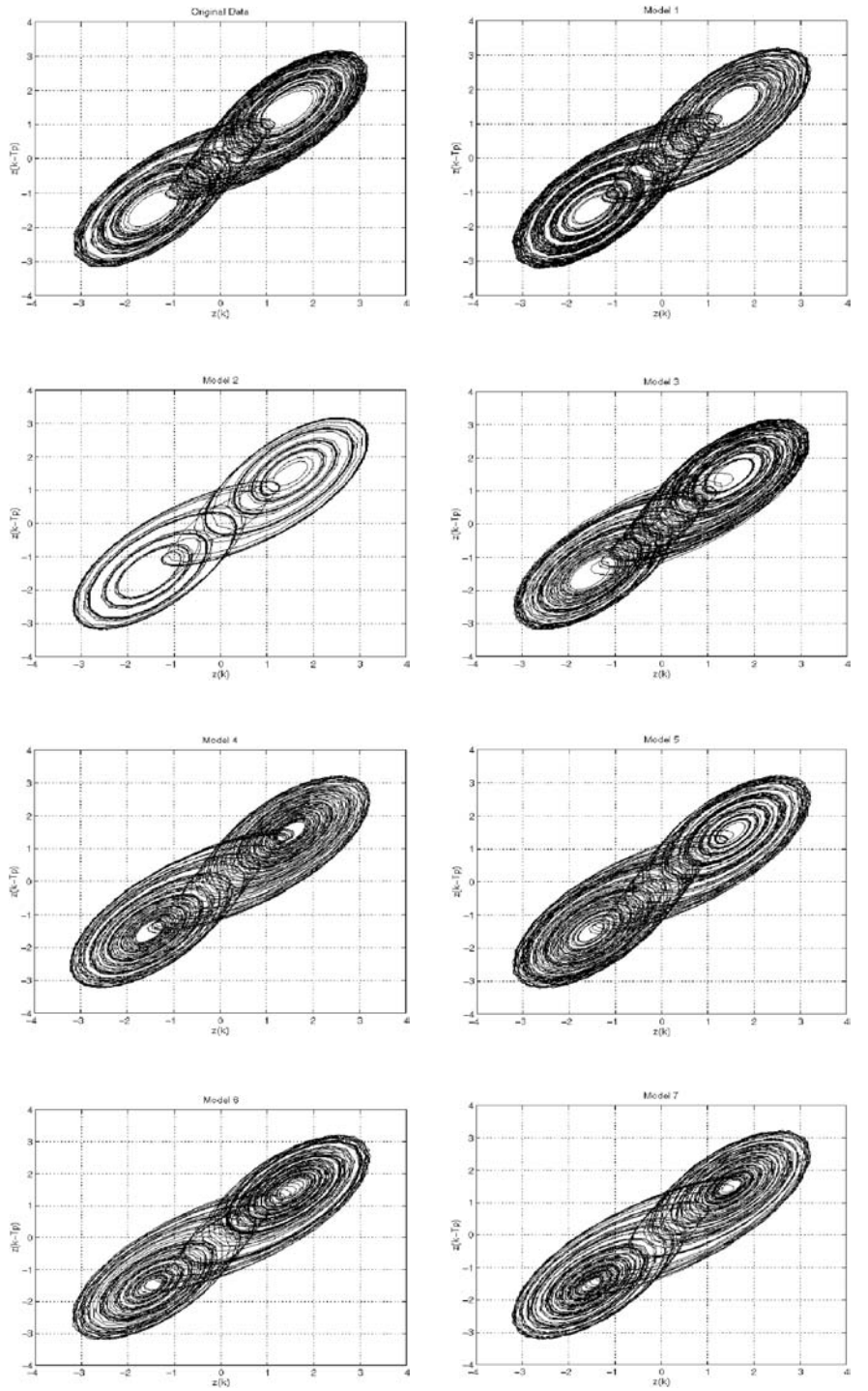


Fig. 9. Double-scroll attractor reconstructed using models identified by means of the NARMA-GP approach

the original system. Further work could include not only the evaluation of statistical but also dynamical validation tools, such as the Lyapunov exponent and correlation dimension (Mendes 1995). These criteria could add a degree of selectivity to the identification procedure and point to more accurate models. The results presented in this paper demonstrate that formulating the modelling process as a multiobjective optimisation problem can provide a more practical way for solving them. On the other hand, genetic programming gives an alternative approach for generating simple and near-optimal solutions of practical and complex problems.

Acknowledgements. Katya Rodríguez-Vázquez gratefully acknowledges the financial support of Consejo Nacional de Ciencia y Tecnología (CONACyT), México, under the projects J34900-A and 40602-A. The authors gratefully acknowledge Dr. Eduardo M. Mendes for his valuable discussion and Eng. Francisco Javier Cárdenas for his technical support.

References

Aguirre LA, Mendes EMAM (1996) Global non-linear polynomial models: Structure, terms clusters and fixed points. Int J Bifurcation Chaos 6:279–294

Akaike H (1974) A new look at the statistical model identification. IEEE Trans Automat Control AC-19:716–723

Banzhaf WP, Nordin P, Keller RE, Francone FD (1998) Genetic programming: An introduction. Kaufmann, San Mateo, CA

Billings SA, Tao QH (1991) Model validity tests for non-linear signal processing applications. Int J Control 54:157–194

Chen S, Billings SA (1989) Representation of non-linear systems: the NARMAX model. Int J Control 49(3):1013–1032

Chen S-H, Yeh C-H (1996) Toward a computable approach to the efficient market hypothesis: an application of genetic programming. J Econ Dynamics Control 21:1043–1064

Chua LO, Hasler M (1993) Special issue on chaos in non-linear electronic circuits. IEEE Trans Circuits Syst 40:10–11

Chua LO, Kamuro M, Matsumoto T (1986) The double scroll family. IEEE Trans Circuits Syst 33:1072–1118

Coello-Coello CA, Toscano-Pulido G (2001) Multiobjective optimization using a micro-genetic algorithm. In: Zitzler E, Deb K, Thiele L, Coello-Coello CA, Corne D (eds) First international conference on evolutionary multi-criterion optimization. Lecture note in computer science. Springer, Berlin Heidelberg New York, pp 126–140

Coello-Coello CA, Van Veldhuizen DA, Lamont GB (2002) Evolutionary algorithms for solving multiobjective problems. Kluwer

Deb K, Agrawal S, Pratab A, Meyarivan T (2000) A fast elitism non-dominated sorting genetic algorithm for multiobjective optimization NSGA-II. In: Schoenauer M, Deb K, Rudolph G, Yao X, Lutton E, Merelo JJ, Schwefel H-P (eds), Proceedings of the parallel problem solving from nature VI conference. Lecture notes in computer science. Springer, Berlin Heidelberg New York, pp 849–858

Evans DC, Fleming PJ, Hill DC, Norton JP, Pratt I, Rees D, Rodríguez-Vázquez K (2001) System identification techniques in aircraft gas-turbine engines. Control Eng Pract IFAC 9:135–148

Fonseca CM, Fleming PJ (1995) Multiobjective optimization and multiple constraint handling with evolutionary algorithms I: a unified formulation. Research Report 564, Dept of automatic control and systems engineering. University of Sheffield, UK

Glover J, Mees A (1993) Reconstructing the dynamics of Chua's circuit. J Circuits Syst Comput 3:201–214

Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley

Gray HF, Maxwell RJ, Martínez-Pérez I, Arus C, Cerdan S (1996) Genetic programming for classification of brain tumors from nuclear magnetic resonance biopsy spectra. In: Koza JR, Goldberg DE, Fogel DB, Riolo RL (eds) Proceedings of the first annual conference on genetic programming. Stanford University, CA, pp 424

Hinterding R (2001) Constrained parameter optimisation: equality constraints. In: IEEE Proceedings of congress on evolutionary computation 2001, vol 1, 27–30 May, Seoul, Korea, pp 687–692

Holmes P (1979) A non-linear oscillator with a strange attractor. Philos Trans Royal Soc London A 292:419–448

Hunter NF (1992) Applications of non-linear time series models to driven systems. In: Casdagli M, Eubank S (eds) Non-linear modeling and forecasting, pp 467–491
Kaboudan MA (1999) Genetic evolution of regression models for business and economic forecasting. In: IEEE congress on evolutionary computation 2:1260–1267
Knowles JD, Corne DW (2000) Approximating the non-dominated front using the Pareto archived evolution strategy. Evolution Comput 8:149–172
Koza JR (1992) Genetic programming: on the programming of computers by means of natural selection. MIT Press, Cambridge, MA
Leontaritis IJ, Billings SA (1985) Input-output parametric models for non-linear systems, part I and part II. Int J Control 41:304–344
Mendes EMAM (1995) Identification of non-linear discrete systems with intelligent structure detection. PhD thesis, Department of Automatic Control and Systems Engineering, University of Sheffield, UK
Michalewicz Z, Nazhiyath G (1995) Genocop III: A co-evolutionary algorithm for numerical optimization problems with non-linear constraints. In: Fogel DB (ed) Proceedings of the second IEEE conference on evolutionary computation, Piscataway, NJ, pp 647–651
Miller JF, Job D, Vassilev VK (2000) Principles in the evolutionary design of digital circuits. Gen Program Evolv Mach 1:7–36
Mulloy BS, Riolo RL, Savit RS (1996) Dynamics of genetic programming and chaotic time series prediction. In: Koza JR, Goldberg DA, Fogel DB, Riolo RL (eds) Proceedings of the first annual conference on genetic programming, 13–16 July, Stanford University, CA, pp 157–165
Nikolaev NI, Iba H (2001) Accelerated genetic programming of polynomials. Gen Program Evolv Mach 2:231–275
Nordin P (1994) A compiling genetic programming system that directly manipulates the machine code. In: Kinnear KE (ed) Advances in genetic programming, chapter 14. MIT Press, Cambridge, MA, pp 311–331
Oakley EHN (1994) Two scientific applications of genetic programming: stack filters and non-linear equation fitting to chaotic data. In: Kinnear KE (ed) Advances in genetic programming, chapter 16. MIT Press, Cambridge, MA, pp 369–389
Poli R (2001) Exact schema theory for genetic programming and variable-length genetic algorithms with one-point crossover. Gen Program Evolv Mach 2:123–163
Rodríguez-Vázquez K, Fleming PJ (1999) Genetic programming for dynamic chaotic systems modelling. In: IEEE proceedings of congress on evolutionary computation, vol 1, 6–9 July, Washington, DC, pp 22–28
Rosca JP, Ballard DH (1999) Rooted-tree schemata in genetic programming. In: Spector L, Langdon WB, O'Reilly U, Angeline PJ (eds) Advances in genetic programming 3, chapter 11. MIT Press, Cambridge, MA, pp 243–271
Ryan C, Ivan L (1999) An automatic software re-engineering tool based on genetic programming. In: Spector L, Langdon WB, O'Reilly U, Angeline PJ (eds) Advances in genetic programming 3, chapter 2. MIT Press, Cambridge, MA, pp 15–40
Spector L, Alpern A (1995) Induction and recapitulation of deep musical structure. In: Proceedings of international joint conference on artificial intelligence, IJCAI'95 workshop on music and AI, Montreal, Quebec
Witbrock M, Neil-Reilly S (1999) Evolving genetic art. In: Bentley P (ed) Evolutionary design by computers, chapter 10. Kaufmann, San Francisco, CA, pp 251–259

Author biographies

Katya Rodríguez-Vázquez received her Licenciatura in Computing Engineering degree from the National University of México in 1994. She obtained the degree of Ph.D. from the University of Sheffield in 1999. She is currently a research associate at the Institute of Applied Mathematics and Systems Engineering, National University of Sheffield. She has published a number of papers in international journals and conferences and has been a member of the Technical Program Committee for conference related to evolutionary computation. Her research interests are evolutionary computation, multiobjective optimisation and its applications.

Peter J. Fleming is Professor of Industrial Systems and Control at the University of Sheffield and Director of the Rolls-Royce University Technology Centre for Control and Systems Engineering. He is Vice-President of the International Federation of Automatic Control. His research interests in control systems, optimization and evolutionary computation have led to close links with industries in sectors such as aerospace, power generation, food processing, pharmaceuticals and manufacturing.

Correspondence and offprint requests to: Katya Rodríguez-Vázquez, IIMAS-UNAM, Circuito Escolar, Ciudad Universitaria, 04510 Coyoacán, México City, México. Email: katya@uxdea4.iimas.unam.mx