Int. J. Appl. Math. Comput. Sci., 2015, Vol. 25, No. 4, 863–875
DOI: 10.1515/amcs-2015-0062

NONLINEAR SYSTEM IDENTIFICATION WITH A REAL–CODED GENETIC ALGORITHM (RCGA)

IMEN CHERIF [a,*], FARHAT FNAIECH [a]

[a] Laboratory of Signal, Image and Energy Mastery (SIME)
National Higher Engineering School of Tunis (ENSIT), 5 Av. Taha Hussein, 1008, Tunis, Tunisia
e-mail: imcherif2001@yahoo.fr, fnaiech@ieee.org

This paper is devoted to the blind identification problem of a special class of nonlinear systems, namely, Volterra models, using a real-coded genetic algorithm (RCGA). The model input is assumed to be a stationary Gaussian sequence or an independent identically distributed (i.i.d.) process. The order of the Volterra series is assumed to be known. The fitness function is defined as the difference between the calculated cumulant values and analytical equations in which the kernels and the input variances are considered. Simulation results and a comparative study for the proposed method and some existing techniques are given. They clearly show that the RCGA identification method performs better in terms of precision, time of convergence and simplicity of programming.

Keywords: blind nonlinear identification, Volterra series, higher order cumulants, real-coded genetic algorithm.

1. Introduction

In the real world, many systems exhibit nonlinearities that cannot be ignored such as in nonlinear control systems, equalization of nonlinear communication channels, poly-spectrum modeling and many others (Attar and Dowell, 2005; Tan and Cloe, 2000; Ozertem and Erolagnus, 2009; Kalouptsidis and Konkoulas, 2005; Tsoulkas *et al.*, 2001). This is why the popularity of nonlinear signal processing has increased over the last decade.

Modeling with Volterra series proved to be a viable approach for nonlinear physical systems. These series can describe any input-output relationship in a nonlinear domain with reasonable accuracy. Each term of a Volterra series specifies a particular order of nonlinearity of the identified system. For example, in digital communication systems, the communication channels are usually impaired by a nonlinear intersymbol interference (ISI). Channel identification allows us to compensate the ISI effects at the receivers. This nonlinearity is in general modeled by a finite Volterra series.

The great advantage of modeling with Volterra series is essentially related to the following facts:

- The output of the Volterra series is linear from the viewpoint of model kernel parameters.
- The signal nonlinearity can be represented through multidimensional operators existing in the products of the input samples.

Unfortunately, modeling with Volterra series involves excessive computational requirements, even when using a truncated version of the series and after taking into consideration the symmetry of the different order of kernels, too.

Glentis *et al.* (1999) considered the problem of nonlinear filtering and identification based on finite support Volterra models. They developed efficient algorithms and persistent excitation conditions for this purpose. It is shown that the normal equations (expressed in terms of cumulants) for a finite support Volterra system excited by a zero mean Gaussian input have a unique solution if, and only if, the power spectral process of the input signal is nonzero at least at m distinct frequencies (where m is the memory of the system) (Stoica and Soderstrom, 1982).

Tsoulka *et al.* (2001) considered the problem of direct identification of a discrete bilinear system governed by an i.i.d. input, with additive noise at the output. It was shown that the bilinear model parameters can be determined via linear equations using cross-cumulants

*Corresponding author

between the output and the input up to the third order. The proposed identification scheme avoids some well-known problems associated with nonlinear least squares algorithms and especially the normal matrix inversion.

Koukoulas and Kalouptsidis (1997; 2000) employed cumulants and poly-spectra to compute the symmetric Volterra kernels, and presented closed-form solutions in the case of a stationary, Gaussian, zero mean random process or an i.i.d. one.

However, in many real applications, the above described methods are not easily applicable for many reasons, and especially for the large number of parameters to be determined and the high computational complexity. In the last years, some intelligent approaches have been adopted to overcome some of the underlined problems, such as neural networks (for their parallelism and learning capacity), fuzzy logic and genetic algorithms.

Kalouptsidis and Koukoulas (2005) proposed blind identification approach of quadratic nonlinear models using neural networks with higher order cumulants. This approach used a weight-decoupled extended Kalman filter training algorithm to estimate the model parameters. Although the proposed identification method yields acceptable kernel estimates, its computational complexity is relatively high and the algorithm becomes intractable in the case of a large number of estimates.

Stathaki and Scohyers (1997) used an unconstrained Lagrange programming neural network to identify the Volterra kernels. This method needs a large number of iterations to converge.

Recently, genetic algorithms (GAs) have been proposed to solve many nonlinear optimization problems and to address the issue of the complexity associated with Volterra models (Cherif *et al.*, 2007; Vasconcelos *et al.*, 2001; Herrera *et al.*, 1998). They are known as powerful optimization tools. A genetic algorithm is a quasi-stochastic search process based on the law of natural selection and genetics. It is important to pinpoint here their major features, especially

- robustness and ability to escape from local minimum points;
- no need for any complex computation as in the conventional optimisation methods, such as the computation of the gradient vectors, the normal matrix inverse, the Fourier transform and its inverse, etc.

For using genetic algorithms for a nonlinear identification problem, in this work we shall propose a new RCGA method for blind identification of quadratic Volterra systems. This method profits from the multiple features of the GA and uses it to solve a set of nonlinear equations where the unknown parameters are the kernels of a discrete time Volterra series with finite memory. These equations are performed by considering the output cumulant up to the third order. The major advantage of the proposed identification method is that it can be applied successfully to a great number of unknown parameters, and without any excessive computational complexity.

In order to make a comprehensive comparison study by inspecting other methods in the literature (Glentis *et al.*, 1999; Koukoulas and Kaloptsidis, 2000), it is quite difficult to find works dealing with the same examples and the same experimental conditions such as the input sequences, the same benchmark parameters, the same computing conditions (e.g., computer speed, memory size, etc.). Consequently, the comparison will be limited to two methods presented by Tan and Clow (2000) as well as Stathaki and Scohyers (1997).

The paper is structured as follows. Section 2 defines the problem of blind system identification with a real-coded genetic algorithm. Section 3 presents a statistical study of the system output and discusses the identifiability of the Volterra model with the output cumulants up to the third order. Section 4 presents the new approach to identify the unknown quadratic kernels with the RCGA. A comparison with some existing methods is given in Section 5. Finally, in Section 6, computer simulations and comparison are carried out resulting in good solutions of the proposed nonlinear identification genetic method in terms of precision, convergence speed and computation simplicity.

2. RCGA blind identification problem formulation

Let us denote the output of a discrete nonlinear time-invariant quadratic system driven by an input random sequence. A quadratic Volterra representation of the input output relationship can be expressed as

$$y(n) = \sum_{i=0}^{N} \sum_{j=0}^{N} h_2(i,j)\, x(n-j) + e(n). \qquad (1)$$

The noise signal is assumed to be a white Gaussian sequence, zero mean, and independent of the input. Furthermore, it is assumed that the unknown quadratic kernels are causal, absolutely summable and symmetric, that is, $h_2(i,j) = h_2(j,i)$.

Also, N is assumed to be known beforehand; this prevents the user from launching a structure identification procedure based on the Fisher test, the Aikake test, etc. The problem of blind identification may be formulated as follows: Based only on the knowledge of the output measurement and the input statistics, the unknown system kernels $h_2(i,j)$ are estimated using a real-coded genetic algorithm. The fitness function of the RCGA is defined as the difference between calculated cumulant values and

analytical equations in which the kernels and the input variances are considered. These analytical functions depend strongly on the character of the input sequence (a Gaussian sequence or an i.i.d. one). In the next section, we shall study both the cases.

3. Statistical analysis

Recall the definition of the cumulants of a random variable given by Kalouptsidis and Koukoulas (2005). The p-th order cumulant sequence of a stationary random signal is denoted by

$$c_{p,y}(k_1,\ldots,k_{p-1}) = \operatorname{cum}\{y(n),\ldots,y(n+k_{p-1})\}. \quad (2)$$

The function $c_{p,y}$ is symmetric and completely determined in the space $\mathbb{Z}^{p-1}$ by its values in the domain Δ defined as

$$\Delta = \left\{(k_1,\ldots,k_{p1}) \in \mathbb{Z}^{p-1}; 0 \le k_{p-1} \le k_i\right\}. \quad (3)$$

For every $l \in \mathbb{N}$, we define

$$\Delta_l = \{(k_1,k_2,\ldots,k_{p-1}) \in \Delta; k_i > l\}. \quad (4)$$

To determine the parameters of the quadratic Volterra system, we will generate sufficient output cumulant information up to the third order. The first order cumulant is defined by $c_{1,y} = E(y(n))$, where $E(\cdot)$ stands for the expectation operator.

Taking into account that the noise $e(n)$ signal is a white Gaussian sequence, zero mean, and independent of the input, the first order cumulant can be written as

$$c_{1,y} = \sum_i \sum_j h_2(i,j)\, E(x(n-i)\,x(n-j)). \quad (5)$$

The second order cumulant is given by

$$c_{2,y}(m) = \operatorname{cum}\{y(n), y(n+m)\}. \quad (6)$$

In view of the Volterra model (1), $c_{2,y}(m)$ can be expressed as

$$\begin{aligned} c_{2,y}(m) &= \operatorname{cum}\Big\{\sum_{i,j} h_2(i,j)\, x(n-i)x(n-j), y(n+m)\Big\} \\ &= \operatorname{cum}\Big\{\sum_{i,j} h_2(i,j)\, x(n-i)x(n-j), \\ &\quad \sum_{i,j} h_2(i,j)\, x(n+m-i)x(n+m-j)\Big\} \end{aligned} \quad (7)$$

The first and second order cumulants of $y(n)$ given by (5) and (7) are not sufficient to solve the identification problem because the number of the unknown parameters in these equations is greater than that of useful samples of $c_{2,y}(m)$.

For this reason, we derive the third order cumulant of the output $y(n)$ which is given by

$$c_{3,y}(m,k) = \operatorname{cum}\{(y(n), y(n+m), y(n+k)\}. \quad (8)$$

The development of these equations changes depending on the distribution type of the excitation signal. In the following, we shall derive useful expressions for (5), (7) and (8) for the following cases of the input sequence:

- $x(n)$: stationary zero mean white Gaussian process,
- $x(n)$: stationary independent identically distributed (i.i.d) process.

3.1. Case of stationary Gaussian input excitation.

Let us assume that $x(n)$ is an unknown zero mean white Gaussian signal with variance $\gamma_{2x} = E\{x^2(n)\} \neq 0$.

For simplicity, we assume that the output is zero mean, which leads to $\sum_i h_2(i,j) = 0$. Consequently,

$$c_{1,y} = 0. \quad (9)$$

Under the same assumptions, the second order cumulant will be written as

$$c_{2,x}(n) = \gamma_{2,x}\delta(n) = \begin{cases} \gamma_{2,x} & \text{if} \quad n = 0, \\ 0 & \text{otherwise}, \end{cases} \quad (10)$$

and

$$c_{k,x}(n_1, n_{2,\ldots,}n_{k-1}) = 0 \quad \text{for all } k > 2.$$

In this paper, if the output is not zero mean, then we make it so by subtracting its mean from all samples prior to any further processing.

From (7) and (10), we get

$$\begin{aligned} &c_{2,y}(m) \\ &= \sum_i \sum_j h_2(i,j)\, h_2(i+m, j+m) \\ &\quad \times \operatorname{cum}\{x(n-i)x(n-j), x(n-i)x(n-j)\} \\ &= 2\gamma_{2x}^2 \sum_i \sum_j h_2(i,j)\, h_2(i+m, j+m). \end{aligned}$$

It is known that the average of the product of an odd number of zero mean jointly Gaussian random variables is identically zero irrespective of their mutual correlation. Moreover, the average of the product of an even number of zero mean jointly Gaussian random variables is equal to the sum over all distinct ways of partitioning the random variables into products of averages of pairs (Vasconcelos

et al., 2001). For example, if $x_1, x_2, x_3, x_4, x_5, x_6$ are zero mean jointly Gaussian random variables, then

$$E\{x_1, x_2, x_3, x_4, x_5\} = 0, \tag{11}$$

$$\begin{aligned} &E\{x_1x_2x_3x_4x_5x_6\} \\ &= E\{x_1x_2\}E\{x_3x_4x_5x_6\} \\ &\quad + E\{x_1x_3\}E\{x_2x_4x_5x_6\} \\ &\quad + E\{x_1x_4\}E\{x_2x_3x_5x_6\} \\ &\quad + E\{x_1x_5\}E\{x_2x_3x_4x_6\} \\ &\quad + E\{x_1x_6\}E\{x_2x_3x_4x_5\}. \end{aligned} \tag{12}$$

Based on the fact that $y(n)$ is a zero mean random process, it follows that the third order cumulants is identical with the third order moment. Accordingly, from (8) and (13) we get

$$c_{3,y}(m,k) = E\{(y(n)y(n+m)\,y(n+k)\}, \tag{13}$$

$$c_{3,y}(m,k) = 8\gamma_{2,x}^3\phi_2(k, k-m, m), \tag{14}$$

where the function $\phi_2(\cdot)$ is defined by

$$\begin{aligned} &\phi_2(k,p,q) \\ &= \sum_i \sum_j \sum_k h_2(i,j)\,h_2(i+k, k+p)\,h_2(j+q,k). \end{aligned}$$

Equations (11) and (15) provide a sufficient number of equations with respect to the unknown kernels number, forming a system of nonlinear equations. Before presenting the genetic approach to solve these systems of equations, let us discuss the case of stationary i.i.d. excitation.

3.2. Case of a stationary independent identical distributed input (i.i.d. excitation). In this section, we assume that $x(n)$ is a zero mean i.i.d signal satisfying

$$\gamma_{4,x} = E\{x^4(n)\} \neq 0,$$

$$\gamma_{3,x} = E\{x^3(n)\} \neq 0,$$

$$\gamma_{2,x} = E\{x^3(n)\} \neq 0.$$

Using the properties of cumulants (Kalouptsidis and Koukoulas, 2005), if a sequence is an i.i.d. zero mean random process, then

$$\begin{aligned} c_{p,x}(k_1, \ldots, k_{p-1}) &= \gamma_{p,x}\delta(k_1, \ldots, k_{p-1}) \\ &= \begin{cases} \gamma_{p,x} & \text{if } k_1 = \cdots = k_{p-1} = 0, \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{15}$$

From this equation and (5), the first order cumulant of the output will be given by

$$c_{1,y} = \gamma_{2,x}\sum_{i=0}^{N} h_2(i,i). \tag{16}$$

In view of the Volterra model (1), $c_{2,y}(m)$ can be written as

$$\begin{aligned} &c_{2,y}(m) \\ &= \sum_i h_2(i,i)h(i+m, i+m) \\ &\quad + 4\gamma_{2,x^2}\sum_{j>0,i} h_2(i+j,i)h(i+j+m, i+m), \end{aligned} \tag{17}$$

where $\gamma_{2,x^2} = \gamma_{4,x} + 2\gamma_{2,x}^2$.

Finally, the third order cumulant of $y(n)$ given by (8) will be expressed by

$$\begin{aligned} c_{3,y}(m,k) &= \gamma_{6,x}\phi_0(m,k) + \gamma_{4,x}\gamma_{2,x}\phi_1(m,k) \\ &\quad + \gamma_{3,x}^2\phi_2(m,k) + \gamma_{2,x}^3\phi_3(m,k), \end{aligned} \tag{18}$$

where the functions $\phi_i(\cdot)$ the are defined by

$$\begin{aligned} &\phi_0(m,k) \\ &= \sum_i h_2(i+m, i+m)h_2(i+k, i+k)\,h_2(i,i), \\ &\phi_1(m,k) \\ &= w_1(m,k) + w_1(m-k, -k) + w_1(k-m, -m) \end{aligned}$$

with

$$\begin{aligned} &w_1(m,k) \\ &\quad = 4\sum_{j,i} h_2(i+j+m, i+m) \\ &\qquad \times h_2(i+j+k, i+k)\,h_2(i,j), \\ &\phi_2(m,k) \\ &\quad = \Psi_2(m,k) + w_2(m,k) + w_2(m-k, -k) \\ &\qquad + w_2(k-m, -m), \\ &\Psi_2(m,k) \\ &\quad = 4\sum_{j,i} h_2(i+j+m, i+m) \\ &\qquad \times h_2(i+j+k, i+k)\,h_2(i+j, i), \end{aligned}$$

and

$$\begin{aligned} w_2(m,k) &= 2\sum_{j,i} h_2(j+m, j+m) \\ &\quad \times h_2(i+k, i+k)\,h_2(i,j), \\ \phi_3(m,k) &= 8\sum_{i,j,l} h_2(i,j) \\ &\quad \times h_2(m+j, m+l)\,h_2(k+l, k+i). \end{aligned}$$

Since the Volterra kernels have a finite range of the order N, the third order cumulant has only one

nonvanishing term in Δ_N; this term is given by

$$\begin{aligned} & c_{3,y}(m,k) \\ & = \gamma_{3,x}^2 w_2(m-k,-k) \\ & = 2\gamma_{3,x}^2 \sum_{i,j} h_2(m-k+j, m-k+j) \\ & \quad \times h_2(i-k, i-k)\, h_2(i,j). \end{aligned} \tag{19}$$

Equations (17)–(19) provide a sufficient number of equations forming a system of multi-nonlinear equations with multi-unknowns which are Volterra kernels. In the following, we shall present a new approach to blind identification with a genetic algorithm.

4. Blind identification with a real-coded genetic algorithm

Genetic algorithms are inspired by the natural search and selection processes leading to the survival of the best individuals. The elements of the search space can be coded using two possibilities: binary coding (yielding a binary coded genetic algorithm, BCGA) or real coding, which would seem natural when dealing with optimization problems with variables in continuous domains (yielding a real-coded genetic algorithm). The basic steps of a general genetic algorithm are depicted in the flowchart of Fig. 1.

In order to apply the genetic techniques to estimate the unknown kernels set and the r-th moment of $x(n)$ denoted by $E[x^{\gamma}(n)] = \gamma_{r,x}$, we used the RCGA. Consequently, we defined a chromosome as a real vector formed by the entire number of the kernels $h_2(i,j)$ and $\gamma_{r,x}$ with $r = 2$ for the Gaussian case (respectively for the i.i.d case). Thus, in the case of Gaussian excitation, the general j-th chromosome in a generation (population) will be expressed by the vector

$$\mathrm{Gch}_j = [h_2(0,1) \ldots h_2(N-1,N)\ \gamma_{2,x}]. \tag{20}$$

In the case of i.i.d. input, the general j-th chromosome is then given by

$$\begin{aligned} \mathrm{Ich}_j = \Big[& h_2(0,0)\ \ h_2(0,1) \ldots h_2(N,N) \\ & \gamma_{2,x}\ \ \gamma_{3,x}\ \ \gamma_{4,x} \Big]. \end{aligned} \tag{21}$$

From these representations, each kernel or moment is viewed as a gene and is assumed to be real valued. In the following, we develop each step of the RCGA for the proposed application.

4.1. Fitness and cost functions. The fitness function is required to be non-negative. In this work, we defined the fitness function by

$$F_j = \frac{1}{1+I_j}, \tag{22}$$

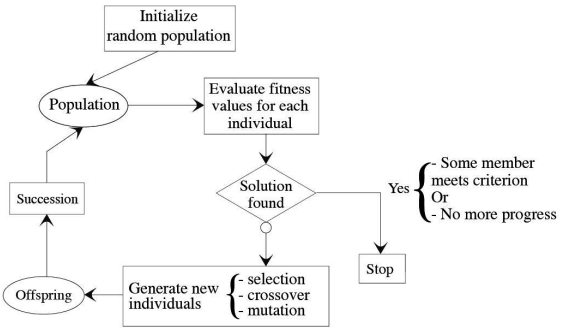

Fig. 1. Flowchart illustrating the basic steps of a general genetic algorithm.

where I_j is the cost function to be minimized. It is defined by

$$\begin{aligned} I_j = & \left(\widehat{c}_{1,y} - c_{1,y}\right)^2 + \sum_m \left(\widehat{c}_{2,y}(m) - c_{2,y}(m)\right)^2 \\ & + \sum_m \sum_k \left(\widehat{c}_{3,y}(m,k) - c_{3,y}(m,k)\right)^2. \end{aligned} \tag{23}$$

The subscript j stands for the number of chromosomes in the current population, while $c_{1,y}$, $c_{2,y}(m)$ and $c_{3,y}(m,k)$ are the values of the first, second and third order cumulants, respectively. These cumulants are computed directly using Eqns. (9), (11) and (15) for the Gaussian case (or Eqns. (17), (18), and (20) for the i.i.d case) and with the different values of the kernels given by each chromosomes Gch_i (or Ich_j) in the current iteration. As cumulants involve expectation operations, they cannot be computed in an exact manner from the available real output signals. The computation of cumulants can be approximated (estimated) consistently by replacing expectations by sample averages

$$\widehat{c}_{1,y} = \frac{1}{N} \sum_n y(n), \tag{24}$$

$$\widehat{c}_{2,y} = \frac{1}{N} \sum_n y(n) y(n+m), \tag{25}$$

$$\widehat{c}_{3,y}(m,k) = \frac{1}{N} \sum_n y(n) y(n+m) y(n+k). \tag{26}$$

Once the fitness function is defined and the population set is initialized, one should apply the different operators of the genetic algorithm.

4.2. Selection operator. The selection operator is the basic operator to form the mating set from the old one. The roulette wheel is one of the most often used stochastic selection techniques (Herrera,1998). Its evaluation is given by the following steps:

(i) Compute the total fitness of the population by

$$F = \sum_{j}^{\text{pop-size}} F_j,$$

where F_j is the fitness value of the chromosome j and 'pop-size' is the population size.

(ii) Compute the probability of selection for each individual: $P_j = F_j/F$.

(iii) Compute the cumulative probability

$$q_j = \sum_{i=1}^{j} p_i$$

for each individual j.

(iv) Generate a random number $\mu \in [0, 1]$. If $\mu < q_1$, then select the individual 1. Otherwise, select the j-th individual such that $2 \leq j \leq$ pop-size and $q_{j-1} < \mu \leq q_j$.

Note that in the case of a large number of unknown variables it will be better to use the tournament selection operator to avoid the large scatter around the expected number of the copies of an individual.

4.3. Crossover operator. The crossover operator is used to get a later generation by exchanging characteristics between multiple couples of individuals (parents) selected stochastically from the mating set. In our application, we applied the one-point crossover type with a different crossover probability. (This probability is chosen after several runs in order to maximize the algorithm performance.)

- Evaluate the fitness values by Eqns. (23) and (24) to form the mating set.
- Inspect the different values: if there is at least one less than a thresholding value, then stop, otherwise go to Step 7 of Algorithm 1.

4.4. Mutation operator. The mutation operator randomly changes some genes in a chromosome with a certain mutation probability. This probability cannot be too large so as not to make the GA degenerate into a simple stochastic search algorithm. In our simulations, the mutation probability is selected experimentally after several runs. In this work, we used the random mutation that consists in replacing the gene $x_i \in [a, b]$ to be mutated by a random number (uniform) chosen in the same range.

Finally, Algorithm 1 summarizes the different steps of the new procedure of blind nonlinear system identification with the RCGA.

Algorithm 1. Different steps of blind nonlinear system identification with the RCGA.

Step 1. Input data and fix input data and fix parameters kernel.

Step 2. Select a quadratic Volterra kernel.

Step 3. Construct a chromosome as indicated by Eqns. (21) (Gaussian case) or (22) (i.i.d. case).

Step 4. Compute the output system cumulants which are approximated by Eqns. (25)–(27).

Step 5. Form a first population by generating different chromosomes randomly.

Step 6. For each chromosome compute values of different cumulants using Eqns. (9), (10), (15) (Gaussian case) or (17), (18), (19) (i.i.d. case).

Step 7. For each chromosome. Apply the different operators of the genetic algorithm for generating a new population, and go back to Step 3.

5. Comparison with the existing methods

The proposed identification method has been compared with the methods proposed by Tan and Clow (2000) as well as Stathaki and Scotyos (1997).

Tan and Clow (2000) solve the problem of blind identification using only the third order output cumulant values with a feed-forward neural network. In most feed-forward neural network applications, weight connexions are trained in such a way that the network establishes a relationship between the system inputs and outputs, but in their work, the functional approximation characteristics of a neural network were applied to identify the parameters of the nonlinear system. In other words, for a general nonlinear system modeled by its output signal $y(t) = f(\phi(t-1), h)$, where h stands for the system parameters and $\phi(t-1)$ denotes measurable signals up to time $(t-1)$, there exists a neural network NN so that the relation between the parameter estimates and the input and output can be given by $h(t) = \text{NN}(\phi(t-1), y(t))$. Consequently, for a given $\phi(t-1)$ and $y(t)$, the neural network can be trained such that its output converges to the desired parameters. Then, given $c_{3,y}(m, k)$, the authors constructed a neural network (h-NN) so that their outputs converge to the exact kernel values. For more details, see the works of Tan and Clow (2000) as well as Annaswamy and Yu (1996).

Stathaki (1997) tried to determine the different Volterra kernels and the variance of the input from the autocorrelation estimates $\rho[k]$ and the third order moment estimates $\mu[k, l]$ of the system output. For this purpose, they used a Lagrange programming neural network (LPNN) to solve a system of nonlinear equations with multiple unknowns. As the LPNN is essentially designed for general nonlinear programming, the author

formulated the identification problem as follows

$$\begin{cases} \text{minimize } L(f) = \sum_i \sum_j \left(\mu\left[i,j\right] - M\left[i,j,f\right]\right)^2 \\ \text{subject to } \rho\left[i\right] = R\left[i,j\right], \end{cases}$$

where $R\left[i,j\right]$ is the autocorrelation function of the real process $y\left[n\right]$ and $M\left[i,j\right]$ is its third order moment sequence, f is the vector formed by the unknown parameters of the Volterra model and the unknown variance of the driving noise. Thus the Lagrangian function will be written as

$$L(f,\lambda) = L(f) + \sum_i \lambda_i \left(\rho\left[i\right] - R\left[i,f\right]\right).$$

To improve the convergence and the precision of the algorithm, we extended the preceding function by defining the augmented Lagrangian function such as

$$L(f,\lambda) = L(f) + \sum_i \lambda_i \left(\rho\left[i\right] - R\left[i,f\right]\right) + \alpha \sum_i \left(\rho\left[i\right] - R\left[i,f\right]\right)^2,$$

where α is a penalty parameter. Thus, the back-propagation algorithm can be established using the Lagrange multiplier.

These two methods overcome the difficulty of programming related to the dimensions of the model, but they introduce other problems, especially the multiple tuning parameters of neural networks, a large number of iterations needed for convergence, and multiple tests to search for good initialization parameters, which make the Monte-Carlo procedure heavier.

The proposed genetic method is characterized by the simplicity of programming, a reduced number of initial parameters to choose at the beginning of simulations, and a high speed of convergence.

6. Simulation results

In these experiments, two quadratic models are used to validate our proposed methodology. The probability of crossover and mutation is chosen experimentally after several tests. Note that the RCGA uses random chromosomes initialisation in the range $[-5,+5]$. To avoid a dependence on realizations, estimates were averaged over 100 Monte-Carlo runs.

The noise signal $e(n)$ is assumed to be a white Gaussian sequence and independent of the input. The parameter estimation was performed for two different signal to noise ratio (SNR) levels: 20 dB and 3 dB. In each run, 128000 samples for $e(n)$ are generated. The SNR is computed using the following expression:

$$\text{SNR} = \frac{E\left(\left(y(n) - e(n)\right)^2\right)}{E\left(e^2\left(n\right)\right)}. \tag{27}$$

For each simulation, we summarize results in different tables giving the mean of each estimate along the 100 Monte-Carlo runs and the corresponding standard deviation.

6.1. Model 1. The first model to be tested here is given by the following quadratic model:

$$y(n) = 0.4x(n)x(n-1) + 0.2x(n)x(n-2) - 0.5x(n-2)x(n-1) + e(n), \tag{28}$$

In each simulation the output signal is set to zero mean by subtracting its mean from all samples prior to any further processing.

- First a Gaussian random sequence $x(n)$ with zero mean is generated and serves as the input signal. Figure 2 shows an example of the evolution of the best parameters (i.e., a chromosome of the population having the best fitness values) with respect to the number of iterations for one run and for SNR $= 3$ dB.
- Second, an i.i.d. signal with an exponential distribution input with zero mean is generated to construct the system output. Figure 3 shows an example of the evolution of the best parameters with respect to the iterations number for one run and for SNR $= 3$ dB.

Comparison results (the means and the standard deviations) of the new algorithm with respect to the h-NN and LPNN algorithms are summarized in Tables 2–5 for different SNR levels and for 100 Monte-Carlo runs. Table 6 includes a comparison of the convergence times needed for each algorithm for 100 Monte-Carlo runs.

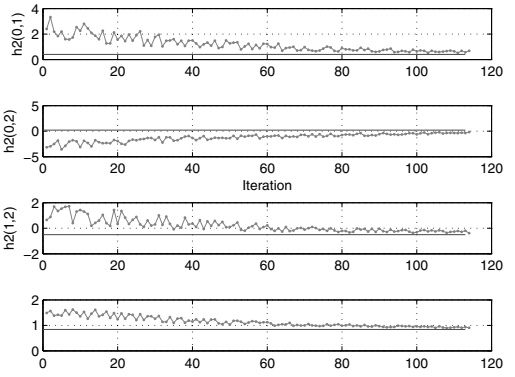

Fig. 2. Evolution of the estimates with respect to the number of iterations for one run (Model 1, Gaussian excitation, SNR $= 3$ dB).

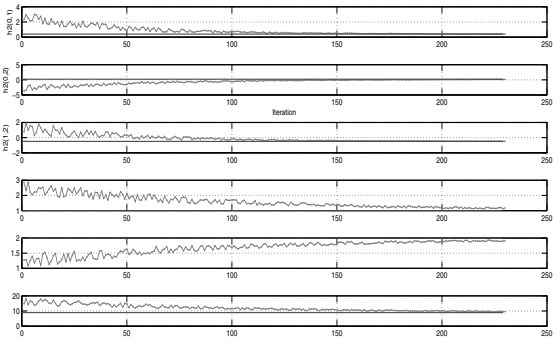

Fig. 3. Evolution of the estimates with respect to the iteration number for one run (Model 1, i.i.d. excitation, SNR = 3 dB).

Note that for simulations we used an Intel®Core(TM) i3 CPU with 2.4 GHz using Matlab. Moreover, one must note that these results may slightly vary according to the initialisation procedures, programming methodologies, etc.

6.2. Model 2. The output sequence is generated using the model given by

$$\begin{aligned} y(n) &= 0.21x^2(n) - 1.62x^2(n-1) + 1.41x^2(n-2) \\ &\quad + 3.72x(n)x(n-1) + 1.86x(n)x(n-2) \\ &\quad + 0.76x(n-1)x(n-2) + e(n). \end{aligned} \tag{29}$$

The same experiments as for Model 1 are pursued in this case. In the same manner, Figs. 4 and 5 show two examples of the evolution of the best parameters with respect to the iterations number for both cases, Gaussian and i.i.d. excitation signals, respectively.

Different results for different SNR levels and for 100 Monte-Carlo runs are summarized in Tables 7–10. Table 11 gives a comparison of the convergence time needed for each algorithm for 100 Monte-Carlo runs.

6.3. Analysis of results. From Figs. 2–5 we note that despite the oscillation presented in the beginning of the simulation the RCGA usually converges to the desired parameters. These oscillations are interpreted by the stochastic aspect of the algorithm in the first research stage before finding a good set of optimal parameters.

Table 1. Comparison of the convergence time (second) for 100 Monte-Carlo runs and Model 1.

	Gaussian input (4 parameters)	i.i.d. input 6 parameters)
RCGA	3820	5543
h-NN	5378	6428
LPNN	5424	6387

Table 2. Comparison of the convergence time (second) for 100 Monte-Carlo runs and Model 2.

	Gaussian input (7 parameters)	i.i.d. input(9 parameters) (9 parameters)
RCGA	5786	7260
h-NN	7152	8356
LPNN	6990	8422

From Tables 2–5 and 7–10, one can observe that the RCGA based on the cumulants of the output up to the third order provides good estimated Volterra kernels for both the types of the excitation input and for different levels of the SNR. The standard deviations for the new algorithm are in general less than those given by the h-NN and the LPNN algorithms, which leads to a good precision.

It is noted also that the algorithm is relatively faster than the others and does not need a large number of iterations to converge. Indeed, the number of iterations needed for convergence in all simulations did not exceed 300. Tables 6 and 11 show that the reductions in the computation time vary between 14% and 30% with respect to the time needed by the other algorithms.

The study of the consistency of the estimates is a subject for further work.

7. Conclusions

In this work, an evolutionary method based on a genetic algorithm was proposed and successfully used for blind Volterra system identification. This method does not require any excessive computations or mathematical transformations. The overall performance of the algorithm (precision, speed of convergence, computational complexity) is very satisfactory in comparison with several other algorithms.

Extending the proposed algorithm to higher order Volterra series and general nonlinear polynomial systems remains an open problem. Moreover, performances of the presented method can be compared with respect to the binary coded genetic algorithm (BCGA).

References

Annaswamy, A.S. and Yu, H. (1996). Adaptive neural networks: A new approach to parameter estimation, *IEEE Transactions on Neural Networks* **7**(4): 907–918.

Attar, P.J. and Dowell, E.H. (2005). A reduced order system ID approach to the modelling of nonlinear structural behavior in aeroelasticity, *Journal of Fluids and Structures* **21**(5–7): 531–542.

Cherif, I., Abid, S., Fnaiech, F. and Favier, G. (2004). Volterra kernels identification using higher order moments for different input signals, *IEEE-ISCCSP, Hammamet, Tunisia*, pp. 845–848.

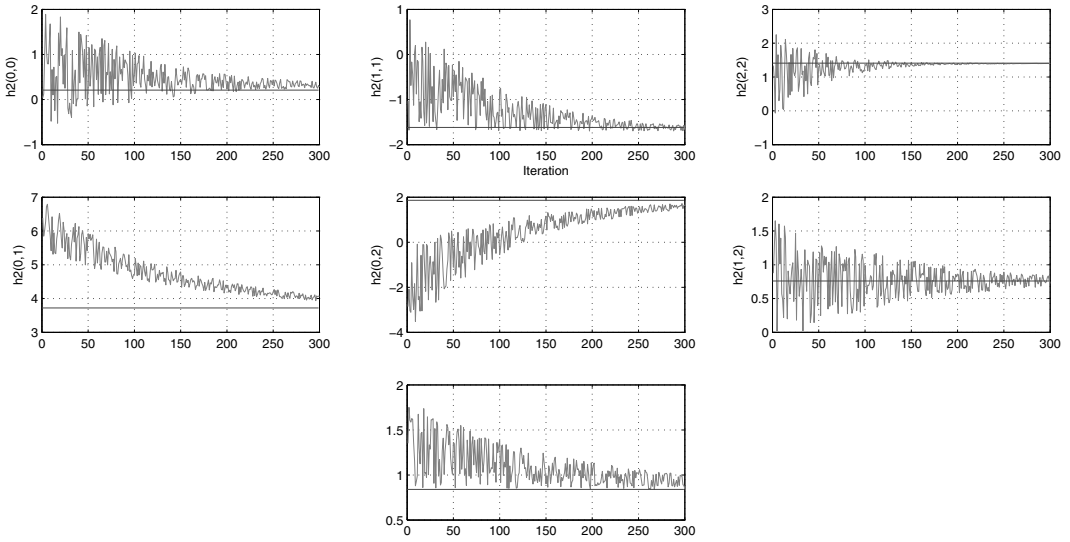

Fig. 4. Evolution of the estimates with respect to the iteration number for one run (Model 2, Gaussian excitation, SNR = 3 dB).

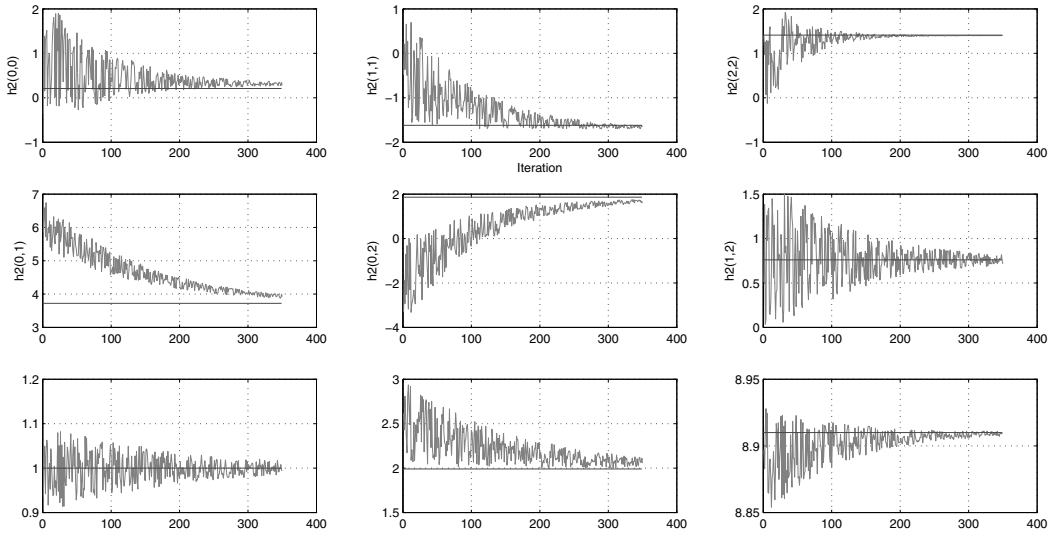

Fig. 5. Evolution of the estimates with respect to the iteration number for one run (Model 2, i.i.d. excitation, SNR = 3 dB).

Cherif, I., Abid, S. and Fnaiech, F. (2005). Blind identification of quadratic systems under i.i.d. excitation using genetic algorithms, *8th International Symposium on Signal Processing and its Applications ISSPA'2005, Sydney, Australia*, pp. 463–466.

Cherif I., Abid S., and Fnaiech, F. (2007). Blind nonlinear system identification under Gaussian and/or i.i.d. excitation using genetic algorithms, *IEEE International Conference on Signal Processing and Communication ICSPC 2007, Dubai, UAE*, pp. 644–647.

Cherif, I., Abid, S., Fnaiech, F. (2012). Nonlinear blind identification with three dimensional tensor analysis, *Mathematical Problems in Engineering* **2012**, Article ID: 284815.

Glentis, G.O.A., Koukoulas, P. and Kalouptsidis, N. (1999). Efficient algorithms for Volterra system identification, *IEEE Transactions on Signal Processing* **47**(11): 3042–3057.

Greblicki, W. (2001). Recursive identification of Wiener systems, *International Journal of Applied Mathematics and Computer Science* **11**(4): 977–991.

Herrera, F., Lozano, M., and Verdegay, J.L. (1998). Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis, *Artificial Intelligence Review* **12**(4): 265–319.

Koukoulas, P. and Kalouptsidis, N. (1996). On blind identification of quadratic systems, *EUSIPCO'96, Trieste, Italy*, pp. 1929–1932.

Koukoulas, P. and Kalouptsidis, N. (1997). Third order system identification, *International Conference on Acoustic Speech and Signal Processing: ICASSP'97, Munich, Germany*, pp. 2405–2408.

Table 3. Model 1: comparison of the identified Volterra parameters with respect to the real ones (under Gaussian excitation). SNR $= 3$ dB, probability of crossover $= 0.08$, probability of mutation $= 0.01$.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,1)$	0.4	0.3156	0.1323	0.4282	0.1421	0.4333	0.1356
$h_2(0,2)$	0.2	0.1834	0.0576	0.1795	0.0677	0.1912	0.0664
$h_2(1,2)$	-0.5	-0.4823	0.1287	-0.4879	0.1185	-0.5346	0.1234
$\gamma_{2,x}$	0.84	0.834	0.0478	0.871	0.0243	0.778	0.0436

Table 4. Model 1: comparison of the identified Volterra parameters with respect to the real ones (under Gaussian excitation). SNR $= 20$ dB, probability of crossover $= 0.08$, probability of mutation$= 0.01$;

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,1)$	0.4	0.4557	0.0429	0.4176	0.0784	0.4478	0.0899
$h_2(0,2)$	0.2	0.1987	0.0456	0.1883	0.0454	0.2125	0.0447
$h_2(1,2)$	-0.5	-0.5228	0.1136	-0.5322	0.1052	-0.5290	0.1878
$\gamma_{2,x}$	0.84	0.880	0.0493	0.758	0.0543	0.946	0.0555

Table 5. Model 1: comparison of the identified Volterra parameters with respect to the real ones (under i.i.d. excitation). SNR $= 3$ dB, probability of crossover $= 0.07$, probability of mutation $= 0.02$.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,1)$	0.4	0.4525	0.1576	0.3367	0.2154	0.4339	0.1868
$h_2(0,2)$	0.2	0.2343	0.0449	0.1821	0.1701	0.3011	0.3458
$h_2(1,2)$	-0.5	-0.4664	0.0451	-0.5551	0.1000	-0.4591	0.0855
$\gamma_{2,x}$	1.00	0.9977	0.0287	1.1046	0.0877	1.1054	0.0554
$\gamma_{3,x}$	1.99	2.0551	0.1099	1.8488	0.2225	2.1199	0.1616
$\gamma_{4,x}$	8.91	9.1644	0.2876	9.0907	0.3999	9.2366	0.4097

Table 6. Model 1: comparison of the identified Volterra parameters with respect to the real ones (under i.i.d. excitation). SNR $= 20$ dB, probability of crossover $= 0.07$, probability of mutation $= 0.02$.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,1)$	0.4	0.4110	0.0121	0.3958	0.1044	0.4339	0.1133
$h_2(0,2)$	0.2	0.2248	0.0265	0.1954	0.1011	0.3011	0.4544
$h_2(1,2)$	-0.5	-0.5147	0.0133	-0.4976	0.1565	-0.4591	0.2252
$\gamma_{2,x}$	1.00	0.9962	0.0114	1.0225	0.1087	1.1054	0.1412
$\gamma_{3,x}$	1.99	2.0174	0.0775	1.9956	0.1111	2.1199	0.1445
$\gamma_{4,x}$	8.91	9.0299	0.1648	8.9335	0.1543	9.2366	0.3165

Koukoulas, P. and Kalouptsidis, N. (2000). Second order Volterra system identification, *IEEE Transactions on Signal Processing* **48**(12): 3574–3577.

Kalouptsidis, N. and Koukoulas, P. (2005). Blind identification of Volterra–Hammerstein systems, *IEEE Transactions on Signal Processing* **53**(8): 2777–2787.

Mathlouthi H., Abderrahim K., Msahli F. and Gerard F. (2009). Crosscumulants based approaches for the structure identification of Volterra models, *International Journal of Automation and Computing* **6**(4): 420–430.

Mendel J.M. (1991). Tutorial on higher order statistics (spectra) in signal processing and system theory: Theoretical results and some applications, *Proceedings of the IEEE* **79**(3): 278–305.

Ozertem, U. and Erdogmus, D. (2009). Second-order Volterra system identification with noisy input-output measurements, *IEEE Signal Processing Letters* **16**(1): 18–21.

Orjuela, R., Marx, B., Ragot, J. and Maquin D. (2013). Nonlinear system identification using heterogeneous multiple models, *International Journal of Applied Mathematics and Computer Science* **23**(1): 103–115, DOI: 10.2478/amcs-2013-0009.

Phan, M.Q., Longman, R.W., Lee, S.C. and Lee, J.-W. (2003). System identification from multiple-trial data corrupted by non-repeating periodic disturbances, *International Journal of Applied Mathematics and Computer Science* **13**(2): 185–192.

Table 7. Model 1: comparison of the identified Volterra parameters with respect to the real ones (under i.i.d. excitation). SNR = 20 dB, probability of crossover = 0.07, probability of mutation = 0.02.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,0)$	0.21	0.2554	0.1366	0.2523	0.1444	0.2334	0.1504
$h_2(1,1)$	-1.62	-1.6244	0.2146	-1.4767	0.5111	-1.7567	0.3895
$h_2(2,2)$	1.41	1.4267	0.2623	1.6551	0.3459	1.5644	0.3123
$h_2(0,1)$	3.72	3.8866	0.2861	3.5988	0.2674	3.6790	0.2716
$h_2(0,2)$	1.86	1.9382	0.0682	2.2113	0.2215	2.1723	0.1473
$h_2(1,2)$	0.76	0.8145	0.1969	0.5866	0.1862	0.6877	0.2566
$\gamma_{2,x}$	0.84	0.786	0.0971	0.884	0.0657	0.899	0.0855

Table 8. Model 2: comparison of the identified Volterra parameters with respect to the real ones (under Gaussian excitation). SNR = 20 dB, probability of crossover = 0.095, probability of mutation = 0.02.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,0)$	0.21	0.2326	0.0911	0.1970	0.1151	0.2265	0.1310
$h_2(1,1)$	-1.62	-1.6441	0.0251	-1.7221	0.0878	-1.8111	0.2442
$h_2(2,2)$	1.41	1.3955	0.1827	1.5778	0.2222	1.5794	0.2657
$h_2(0,1)$	3.72	3.7732	0.2491	3.5652	0.1999	3.7278	0.2643
$h_2(0,2)$	1.86	1.8901	0.0606	1.7888	0.0578	1.8259	0.1116
$h_2(1,2)$	0.76	0.7088	0.0824	0.8033	0.1565	0.6928	0.2245
$\gamma_{2,x}$	0.84	0.8511	0.0354	0.919	0.0522	0.855	0.0551

Stathaki, T. and Scohyers, A. (1997). A constrained optimisation approach to the blind estimation of Volterra kernels, *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-97, Munich, Germany*, pp. 2373–2376.

Stoica, P. and Soderstorm, T. (1982). Instrumental variable methods for identification of Hammerstein systems, *International Journal of Control* **35**(3): 459–476.

Tan, H.Z. and Chow, T.W.S. (2000). Blind identification of quadratic nonlinear models using neural networks with higher order cumulants, *IEEE Transactions on Industrial Electronics* **47**(3): 687–696.

Tseng, C.H. and Powers, E.J. (1995). Identification of cubic systems using higher order moments of i.i.d. signals, *IEEE Transactions on Signal Processing* **43**(7): 1733–1735.

Tsoulkas, V., Koukoulas, P. and Kalouptsidis, N. (2001). Identification of input-output bilinear system using cumulants, *IEEE Transactions on Signal Processing* **49**(11): 2753–2761.

Vasconcelos, J.A., Ramirez, J.A., Takahashi, R.H.C. and Saldanha, R.R. (2001). Improvement in genetic algorithms, *IEEE Transactions on Magnetics* **37**(5): 3414–3417.

Zhang, S. and Constantinides, A.G. (1992). Lagrange programming neural networks, *IEEE Transactions on Circuits and Systems: Analog and Digital Signal Processing* **39**(7): 441–452.

Imen Cherif received the B.Sc. degree in mathematics from the University of Sciences of Tunis in 1999, and the D.E.A. degree in electrical engineering from the High School of Science and Engineering of Tunis (ESSTT Tunisia) in 2002. She is currently preparing her Ph.D. in nonlinear system identification and is an assistant professor in the University of Elmanar, Tunisia. Her research interests are focused on nonlinear system identification with evolutionary methods such as genetic algorithms and fuzzy logic.

Farhat Fnaiech received the B.Sc. degree in mechanical engineering in 1978 from the High School of Sciences and Techniques of Tunis, and the Master's degree in 1980. He obtained the Ph.D. degree in electrical engineering from the same school in 1983, and the *Doctorate Es Science* in physics from the Faculty of Sciences of Tunis in 1999. He is currently a professor at the High School of Science and Engineering of Tunis. He is a senior member of the IEEE and has published over 150 research papers in many journals and international conferences. He has been a general chairman and a member of the international board committee of many international conferences. He is an associate editor of *IEEE Transactions on Industrial Electronics*. He has served as an IEEE chapter committee coordination sub-committee delegate of Africa Region 8. His main research areas are nonlinear adaptive signal processing, nonlinear control of power electronic devices, digital signal processing, image processing, intelligent techniques and control.

Table 9. Model 2: comparison of the identified Volterra parameters with respect to the real ones (under i.i.d. excitation). SNR = 3 dB, probability of crossover = 0.09, probability of mutation = 0.02.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,0)$	0.21	0.2005	0.1433	0.2456	0.1756	0.3467	0.1564
$h_2(1,1)$	-1.62	-1.5810	0.2567	-1.6689	0.2679	-1.5568	0.2665
$h_2(2,2)$	1.41	1.4221	0.0647	1.4527	0.2122	1.5627	0.0722
$h_2(0,1)$	3.72	3.8642	0.1982	3.5822	0.0889	3.8202	0.3667
$h_2(0,2)$	1.86	1.8719	0.0894	2.0001	0.5213	2.1087	0.0865
$h_2(1,2)$	0.76	0.7183	0.1884	0.7557	0.2222	0.7433	0.2237
$\gamma_{2,x}$	1	1.0665	0.0656	0.8893	0.1054	0.9753	0.0842
$\gamma_{3,x}$	1.99	1.9444	0.1667	2.1234	0.2119	2.0024	0.1839
$\gamma_{4,x}$	8.91	9.0189	0.2002	9.3250	0.2880	9.1237	0.2543

Table 10. Model 2: comparison of the identified Volterra parameters with respect to the real ones (under i.i.d. excitation). SNR = 20 dB, probability of crossover = 0.09, probability of mutation = 0.02.

		New algorithm		h-NN		LPNN	
Parameters	Real values	Mean	STD	Mean	STD	Mean	STD
$h_2(0,0)$	0.21	0.2245	0.0621	0.2254	0.0665	0.2321	0.0865
$h_2(1,1)$	-1.62	-1.6434	0.0333	-1.6445	0.0618	-1.6355	0.0576
$h_2(2,2)$	1.41	1.4374	0.1154	1.4333	0.1538	1.4276	0.1533
$h_2(0,1)$	3.72	3.7258	0.0536	3.7818	0.0755	3.7500	0.0777
$h_2(0,2)$	1.86	1.9032	0.1675	1.8999	0.2657	1.8533	0.1114
$h_2(1,2)$	0.76	0.7518	0.0431	0.7654	0.1573	0.7721	0.1319
$\gamma_{2,x}$	1	1.0183	0.0311	0.9872	0.0751	1.0144	0.0616
$\gamma_{3,x}$	1.99	2.0096	0.1186	2.0002	0.1848	1.9999	0.1355
$\gamma_{4,x}$	8.91	8.9322	0.0946	9.1058	0.08332	9.0756	0.1563

Appendix

Cumulants

Let $x = [x_1, x_2, \ldots, x_k]^T$ denote a vector of real random variables. A cumulant of these random variables is defined as a coefficient of the vector $v = [v_1, v_2, \ldots, v_k]^T$ in the Taylor series expansion (if it exists) of the cumulant generating function $K(v) = \ln(E(\exp(jvTx)))$.

The cumulant is therefore defined in terms of its joint moments of orders up to. For zero-mean real random variables, the first, second, and third order cumulants are given by

$$\begin{aligned} \operatorname{cum}(x_1) &= E\{x_1\}(A-1), \qquad \text{(A1)} \\ \operatorname{cum}(x_1, x_2) &= E\{x_1 x_2\}, \\ \operatorname{cum}(x_1, x_2, x_3) &= E\{x_1 x_2 x_3\}. \end{aligned}$$

In the case of nonzero mean real random variables, one replaces x_i by $x_i - E\{x_i\}$ in these formulas. Let $\{x(n)\}$ be a zero-mean k-th order stationary discrete time random process. The k-th order cumulant of this process, denoted $c_{k,x}(\tau_1, \tau_2, \ldots, \tau_{k-1})$, is defined as the joint k-th order cumulant of the random variables $x(n), x(n+\tau_1), \ldots, x(n+\tau_{k-1})$, i.e.,

$$\begin{aligned} &c_{k,x}(\tau_1, \tau_2, \ldots, \tau_{k-1}) \\ &\quad = \operatorname{cum}(x(n), x(n+\tau_1), \ldots, x(n+\tau_{k-1})). \qquad \text{(A2)} \end{aligned}$$

Because of the stationarity assumption, the k-th order cumulant is only a function of the $k-1$ lags $\tau_1, \tau_2, \ldots, \tau_{k-1}$. The $\tau_1 - \tau_2 - \cdots - \tau_{k-1}$ space constitutes the domain of support for $c_{k,x}(\tau_1, \tau_2, \ldots, \tau_{k-1})$

From (A1) and (A2), the second and third order cumulants of zero-mean are given by

$$\begin{aligned} c_{2,x}(\tau) &= E\{x(n)x(n+\tau)\}, \\ c_{3,x}(\tau_1, \tau_2) &= E\{x(n)x(n+\tau_1)x(n+\tau_2)\}. \end{aligned}$$

Of course, the second order cumulant $C_{2,x}(\tau)$ is just the autocorrelation of $x(n)$. We shall use a more familiar notation for autocorrelation, namely, $r_x(\tau)$, interchangeably with $C_{2,x}(\tau)$.

Fundamental properties of cumulants may be summarized as follows:

(i)

$$\begin{aligned} &\operatorname{cum}[\beta_1 x_1, \beta_2 x_2, \ldots, \beta_k x_k] \\ &\qquad = \left(\prod_{i=1}^{k} \beta_i\right) \operatorname{cum}[x_1, x_2, \ldots, x_k] \end{aligned}$$

where $\{\beta_i\}_{i=1,2,\ldots,k}$ are constants.

(ii) Cumulants are symmetric functions in their arguments:

$$\operatorname{cum}[x_1, x_2, \ldots, x_k] = \operatorname{cum}[x_{i1}, xi_2, \ldots, x_{ik}],$$

where $(i_1, i_2, \ldots, i_k)$ are obtained from a permutation of $(1, 2, \ldots, k)$.

(iii) Cumulants are additive in their arguments:

$$\begin{aligned}&\operatorname{cum}[y + z, x_1, x_2, \ldots, x_k]\\ &\quad = \operatorname{cum}[y, x_1, x_2, \ldots, x_k] + \operatorname{cum}[z, x_1, x_2, \ldots, x_k],\end{aligned}$$

where y and z are two random variables.

(iv) If the random variables $\{x_i\}_{i=1,2,\ldots,k}$ can be divided into two or more groups which are statistically independent, their order cumulant is zero: $\operatorname{cum}[x_1, x_2, \ldots, x_k] = 0$.

Received: 20 February 2014
Revised: 7 September 2014