2005 WSEAS Int. Conf. on DYNAMICAL SYSTEMS and CONTROL, Venice, Italy, November 2-4, 2005 (pp148-153)

Nonlinearities Identification using The LMS Volterra Filter

Georgeta Budura, Corina Botoca
Communications Department
Faculty of Electronics and Telecommunications
Timisoara, Bd. V. Parvan, No.2
ROMANIA
http://www.utt.ro/english/index.php

Abstract: - Nonlinear adaptive filtering techniques, based on the Volterra model, are widely used for the nonlinearities identification in many applications. This paper proposes a new implementation of the third order LMS Volterra filter**.** A third order nonlinear system with memory is identified using the new LMS algorithm implementation for the Volterra kernels estimation. The accuracy of the proposed algorithm is appreciated by comparing the estimated third order kernel with the real kernel. It is demonstrated that the adaptive algorithm gives satisfactory convergence in different conditions of noise. The extension of the LMS algorithm implementation to higher order Volterra filters is also possible and involves a few simple changes.

Key-Words: - Volterra kernels estimation, LMS algorithm, Nonlinear filter identification.

1 Introduction

The current trend in the telecommunication systems design is the identification and compensation of unwanted nonlinearities. It was demonstrated that unwanted nonlinearities in the system have a determinant effect on his performance [1]. There are various ways of reducing the effects of undesired nonlinearities [2],[3],[4]. The use of nonlinear models considered in this paper to characterize and compensate harmful nonlinearities offer a possible solution. The Volterra series have been widely applied as nonlinear system modelling technique with considerable success. However, at present, none general method exists to calculate the Volterra kernels for nonlinear systems, although they can be calculated for systems whose order is known and finite. When the nonlinear system order is unknown, adaptive methods and algorithms are widely used for the Volterra kernel estimation. The accuracy of the Volterra kernels will determine the accuracy of the system model and the accuracy of the inverse system used for compensation. The speed of kernel estimation process is also important. A fast kernel estimation method may allow the user to construct a higher order model that give an even better system representation. This paper proposes a new implementation of the third order LMS Volterra filter**.** A third order nonlinear system with memory is identified using the new LMS algorithm implementation for the Volterra kernels estimation.

The proposed implementation of the third order LMS Volterra filter is based on the extended input vector and on the extended filter coefficients vector. Due to the linearity of the input-output relation of the Volterra model with respect to filter coefficients, the implementation of the LMS algorithm was realized as an extension of the LMS algorithm for linear filters.

2 The Volterra Model

This section will discuss some important aspects of the Volterra model. For a discrete-time and causal nonlinear system with memory, with input x[n] and output y[n], the Volterra series expansion is given by [4]:

$$y[n]=\sum_{r=1}^{N}\sum_{k_1=0}^{M-1}\cdots\sum_{k_r=0}^{M-1}h_r[k_1,\cdots,k_r]x[n-k_1]\cdots x[n-k_r] \tag{1}$$

where N represents the model nonlinearity degree, M is the filter memory and $h_r[k_1,\cdots,k_r]$ is the r^{th} order Volterra kernel. In the most general case Eq.(1) may use different memory for each nonlinearity order. A further simplification can be made to Eq.(1) by considering symmetric Volterra kernels. The kernel $h_r[k_1,\cdots,k_r]$ is symmetric if the indices can be interchanged without affecting its value.

Choosing $N=3$ in the Eq.(1), the input-output relationship of the third order Volterra filter can be expressed as:

$$y[n]=h_0+\sum_{k_1=0}^{M-1}h_1[k_1]x[n-k_1]+$$
$$\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}h_2[k_1,k_2]x[n-k_1]x[n-k_2]+ \quad (2)$$
$$\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}\sum_{k_3=0}^{M-1}h_3[k_1,k_2,k_3]x[n-k_1]x[n-k_2]x[n-k_3]$$

The input-output relation can also be written in terms of nonlinear operators as indicated in Eq.(3).

$$y[n]=H_0[x[n]]+H_1[x[n]]+H_2[x[n]]+H_3[x[n]] \quad (3)$$

In the above representations, the functions $h_i, i=\overline{0,3}$ represent the kernels associated to the nonlinear operators $H_i[x[n]]$. The nonlinear model described by the Equations (2) and (3) is called a third order Volterra model. Note that the above representations have the same memory for all nonlinearity orders. In this case the second order Volterra kernel is a *(M×M)* matrix. As presented in reference [5], the third order kernel is composed of *M* matrices having the dimension *(M×M)*.

If we consider symmetric kernels of memory *M*, the second order Volterra kernel requires the determination of *M(M+1)/2* coefficients, while the third order kernel needs *M(M+1)(M+2)/6* coefficients [5].

The kernel estimation accuracy becomes the major problem in the practical applications. It was shown that the Volterra operators are homogeneous and generally not orthogonal. As a consequence of this last characteristic the Volterra kernels can not be measured using the cross correlation techniques and the values of the Volterra kernels will depend on the order of the Volterra representation being used [5],[6]. If the order of the Volterra model is changed the Volterra kernels will change and they must be recalculated. However, for an input having a symmetric amplitude density function, such as the Gaussian noise, the odd order Volterra functionals are orthogonal to the even order Volterra functionals. It follows that for this type of input, a 2nd order Volterra model, with zero DC component, is an orthogonal model. This leads to direct Volterra kernel measurement by the cross-correlation method [7].

For higher order Volterra filters adaptive methods of kernels estimation are widely used. Due to the linearity of the input-output relation according to the kernels, or filter coefficients, the application of adaptive algorithms for the Volterra filters implementation is quite simple. The nonlinearity is reflected only by the multiple products between the delayed versions of the input signal.

Next we will introduce the input vectors corresponding to different orders kernels. The first order input vector, corresponding to a filter length $M=3$, is defined as follows:

$$X^{(1)T}=[x[n]\,x[n-1]\,x[n-2]] \quad (4)$$

If we consider equal memories for different orders filters, "the second order input vector" can be expressed by:

$$X^{(2)}=X^{(1)}*X^{(1)T} \quad (5)$$

For symmetric kernels only the elements $x_{i,j}$, having $i\geq j$, of $X^{(2)}$, are selected in the input-output relation of the Volterra filter. Hence "the second order input vector", written in vector form is:

$$X^{(2)T}=\left[x^2[n]\,x[n]x[n-1]\,x[n]x[n-2]\,x^2[n-1]\;\; x[n-1]x[n-2]\,x^2[n-2]\right] \quad (6)$$

and it has the dimension (1×6).

For "the third order input vector" we propose to express the multiple input delayed signal products in Eq.(2) by matrices elements. These matrices can be generated by multiplying "the second order input vector" defined according to Eq.(5) by the elements of the first order input vector. If we consider equal filters memories in Eq.(2), *M=3*, and symmetric kernels it follows:

$$X^{(2)}*x[n]=\begin{bmatrix} x^3[n] & x^2[n]x[n-1] & x^2[n]x[n-2] \\ \cdots & x[n]x^2[n-1] & x[n]x[n-1]x[n-2] \\ \cdots & \cdots & x[n]x^2[n-2] \end{bmatrix} \quad (7)$$

$$X^{(2)}*x[n-1]=\begin{bmatrix} \cdots & \cdots & \cdots \\ \cdots & x^3[n-1] & x^2[n-1]x[n-2] \\ \cdots & \cdots & x[n-1]x^2[n-2] \end{bmatrix} \quad (8)$$

$$X^{(2)} * x[n-2] = \begin{bmatrix} \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \cdots & x^3[n-2] \end{bmatrix} \quad (9)$$

Hence, "the third order input vector" consists in fact, in that case, of 3 matrices as indicated in Equations (7)÷(9) and corresponds to a symmetric third order Volterra kernel. We can write "the third order input vector" in vector form as follows:

$$X^{(3)T} = \Big[x^3[n]\, x^2[n] x[n-1] \cdots x[n] x^2[n-2]\, x^3[n-1] \cdots x[n-1] x^2[n-2]\, x^3[n-2] \Big] \quad (10)$$

Its dimension is (1×10).

The defined input vectors will be used to implement the LMS Volterra filter in a typical nonlinear system identification application.

3 Volterra Kernels Estimation by the LMS Adaptive Algorithm

A typical adaptive technique used for Volterra kernels identification is shown in Fig.1.

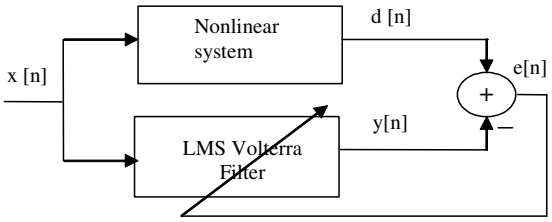

Fig.1 Volterra kernel identification by adaptive method

The Volterra filter of fixed order and fixed memory adapts to the unknown nonlinear system using one of the various adaptive algorithms. The use of adaptive techniques for Volterra kernel estimation has been intensively published. Most of the previous work considers 2nd order Volterra filters and some consider the 3rd order case.

The most commonly used algorithm uses an LMS adaptation criterion. Although the LMS algorithm has some drawbacks, such as its dependence on signal statistics, which can lead to low speed or large residual errors, it is very simple to implement and well behaved compared to the faster recursive algorithms. This section discusses the extension of the LMS algorithm to the nonlinear case using the previously defined input vectors. The discrete time impulse response of a first order, linear, system with memory span *M*, is written in vector form as in Eq.(11) and the input vector as in Eq.(12).

$$H_k^{(1)T} = \Big[h_k^{(1)}[0] \quad h_k^{(1)}[1] \quad \cdots \quad h_k^{(1)}[M-1] \Big] \quad (11)$$

$$X_k^{(1)T} = \big[x_k \quad x_{k-1} \quad \cdots \quad x_{k-M+1} \big] \quad (12)$$

In Eq.(11) the filter order is written as a superscript index. This notation will be kept consistent for the rest of the paper.

Using Eq.(11) and Eq.(12), the output of a linear system is written as:

$$y_k = H_k^{(1)T} \cdot X_k^{(1)} \quad (13)$$

At sample *k*, the desired output is d_k and the linear adaptive filter output is y_k. For the LMS algorithm, we minimize the Eq.(14).

$$E\Big[e_k^2 \Big] = E\Big[d_k - H_k^{(1)T} X_k^{(1)} \Big] \quad (14)$$

The vector H^* that minimizes the Eq.(14) may be expressed as a solution of the normal equation given in (15).

$$H^* = R_{xx}^{-1} \cdot G \quad (15)$$

where: R_{xx}^{-1} is the inverse of the input correlation matrix, $R_{xx} = E\big[X_k^{(1)} X_k^{(1)T}\big]$, containing the moments of the input signal up to 2nd order and $G = E\Big[X_k^{(1)} y_k \Big]$.

The well known LMS update equation for a first order, linear, filter is:

$$H_{k+1}^{(1)} = H_k^{(1)} + \mu e_k X_k^{(1)} \quad (16)$$

where μ is a small positive constant (referred to as the step size) that determines the speed of convergence and also affects the final error of the filter output.

The extension of the LMS algorithm to higher order, nonlinear, Volterra filters involves a few simple changes. First, the vector of the impulse

response coefficients becomes the vector of Volterra kernels coefficients. Second, the input vector, which for the linear case contains only a linear combination, for nonlinear Volterra filters, complicates.

Consider the Volterra representation with symmetric kernels. There are two parts of this representation: (1) the Volterra kernel estimations and (2) the products of the delayed input signal. If we express the Volterra kernels and the input signal products in a vector form, then we can write the adaptive Volterra filter output using the vector notation. Each Volterra kernel (estimated at sample *k*) can be written in vector form.

For simplicity we have implemented the nonlinear adaptive filter considering only first order and 3rd order Volterra kernels.

The Eq.(17) gives "the input matrix" at sample *k*, containing the first and "the third order" input vectors defined previously.

$$X_k = \begin{bmatrix} X_k^{(1)T} \\ X_k^{(3)T} \end{bmatrix} \tag{17}$$

The size of the input matrix is determined by the size of the third order input vector $X_k^{(3)}$. "The filter coefficients matrix" at sample *k* is given by:

$$H_k = \begin{bmatrix} H_k^{(1)T} \\ H_k^{(3)T} \end{bmatrix} \tag{18}$$

where $H_k^{(1)T}$ is given by the Eq.(12) and $H_k^{(3)T}$ is the third order kernel expressed in vector form as indicated in Eq.(19).

$$H_k^{(3)T} = \left[h_k^{(3)}[0,0,0] \quad h_k^{(3)}[0,0,1] \quad \cdots \quad h_k^{(3)}[M-1,M-1,M-1] \right] \tag{19}$$

The update equation for the LMS Volterra filter can be written also in matrix form:

$$\begin{bmatrix} H_{k+1}^{(1)T} \\ H_{k+1}^{(3)T} \end{bmatrix} = \begin{bmatrix} H_k^{(1)T} \\ H_k^{(3)T} \end{bmatrix} + e_k \begin{bmatrix} \mu_1 & 0 \\ 0 & \mu_3 \end{bmatrix} \begin{bmatrix} X_k^{(1)T} \\ X_k^{(3)T} \end{bmatrix} \tag{20}$$

In the nonlinear case it is possible to set different step sizes for different order kernels. Consequently we have introduced the step size matrix *M*, defined by:

$$\mu = \begin{bmatrix} \mu_1 & 0 \\ 0 & \mu_3 \end{bmatrix} \tag{21}$$

4 Experiments and results

The adaptive method already presented was implemented in a system identification application, using the MATLAB software. The identified system was a 3rd order system with memory represented in Fig.2.

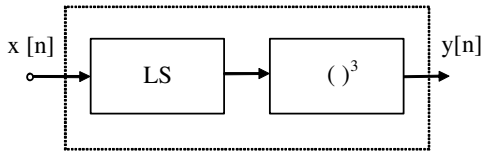

Fig.2 The nonlinear system

where LS is a linear system with impulse response:

$$h_1(t) = \frac{1256}{0.98} \exp(-251{,}2t) \sin(1231t)\sigma(t) \tag{21}$$

The discrete time impulse response of the linear system is:

$$h_1[n] = {}^{1}\!/_{T_e}\, h_1(nT_e) \tag{22}$$

where T_e denotes the sampling period. Also we have calculated the Volterra kernels associated with the nonlinear system. The results are given in Eq.(23).

$$\begin{aligned} h_1[n] &= 0 \\ h_3[n_1,n_2,n_3] &= h_1[n_1]h_1[n_2]h_1[n_3] \end{aligned} \tag{23}$$

The input signal was a 2000 samples Gaussian noise with zero mean. The filter memory used was $M = 40$. Because the kernels involved in the nonlinear filter has the same parity we have chosen $\mu_1 = \mu_3$ in Eq. (20). In order to minimize the error $e[n]$ of the adaptation algorithm, the value of μ_1 used in simulations was smaller than that imposed by theoretical considerations [1].

In Fig. 3 we have represented the evolution of the mean square error (MSE), $E\left[(d[n]-y[n])^2\right]$, considering 140 adaptation steps.

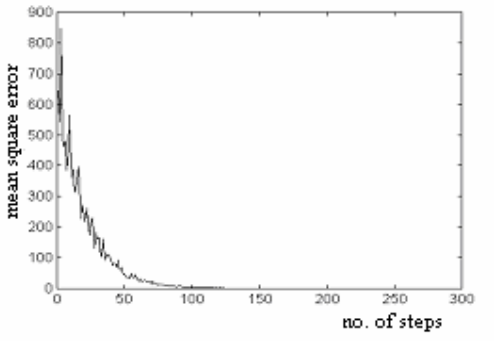

Fig.3. MSE against no. of iterations

The accuracy of the proposed algorithm was appreciate according to:

$$eh = 10\lg\left(\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}\sum_{k_3=0}^{M-1}\left(h_3[k_1,k_2,k_3]-\hat{h}_3[k_1,k_2,k_3]\right)^2\right) \quad (24)$$

where $h_3[k_1,k_2,k_3]$ are given by Eq.(23) and $\hat{h}_3[k_1,k_2,k_3]$ are experimentally determined using the LMS algorithm. The result is presented in Fig.4.

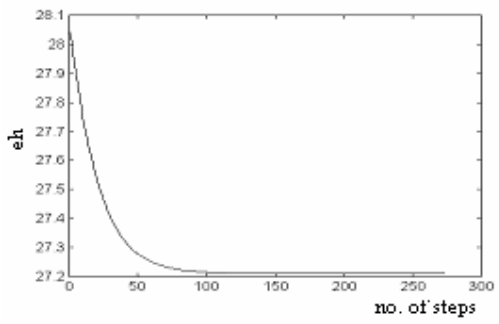

Fig.4 Third order Volterra kernel convergence against no. of iterations

We have repeated the by adding noise at the output signal y[n]. We have considered two cases:

- additive noise having $\sigma^2 = 0.1$ (Fig.5);
- additive noise having $\sigma^2 = 0.01$ (Fig.6).

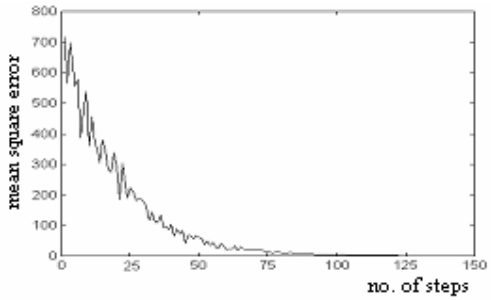

Fig.5 MSE against no. of iterations ($\sigma^2 = 0.1$)

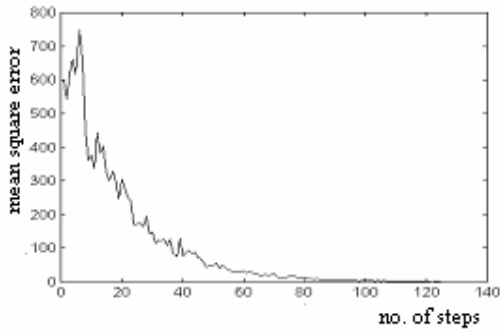

Fig 6. MSE against no. of iterations ($\sigma^2 = 0.01$)

Fig.7 shows the third order Volterra kernel convergence corresponding to:

-Adaptation with noise ($\sigma^2 = 0.1$; doted line);

-Adaptation with noise ($\sigma^2 = 0.01$; continuous line);

-Adaptation without noise (line-dot-line).

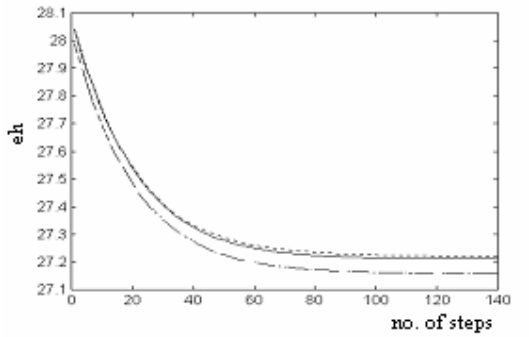

Fig.7 Compared results regarding the convergence of the third order Volterra kernel

It can be seen that the adaptive algorithm gives satisfactory convergence in all three situations.

5 Conclusion

This paper has presented a new implementation of the third order LMS Volterra filter**.** The new implementation is based on the extended input vector and on the extended filter coefficients vector.

We have defined the input vectors corresponding to different order kernels. We have proposed a new method to generate the third order kernel coefficients based on the first order input vector.

The new adaptive method was implemented in a system identification application, and the estimated third order kernel was compared with the previously determined kernel. The results indicates that the adaptive algorithm yields to a satisfactory convergence as indicated in Fig.7.

The method is easy to implement and useful in practical applications, when the capabilities of the linear filters are unsatisfactory. In a nonlinear system identification problem the presence of the second order Volterra kernel in the structure of the nonlinear adaptive filter is mandatory.

In order to guarantee the convergence of the LMS algorithm we have considered a more general form for the updating equation that offers the possibility to set different step sizes for different order kernels.

The extension of the LMS algorithm implementation to higher order Volterra filters is also possible and involves a few simple changes regarding the definition of the input vector and the definition of the filter coefficients vector.

References:

[1] J.Tsimbinos, Identification And Compensation Of Nonlinear Distortion, *Thesis*, Institute for Telecommunications Research, University of South Australia, 1995. http://www.unisa.edu.au/html, 237 pages.

[2] A.Stenger, W.Kellermann, RLS-Adapted Polynomial Filter for Nonlinear Acoustic Echo Cancelling, Proceedings of the X EUSIPCO *2000*, Tampere, Finland, pp. 1867-1870.

[3] A.Stenger, W. Kellermann, Adaptation of a Memoryless Preprocesor for Nonlinear Acoustic Echo Cancelling, *Signal Processing* , No. 9, 2000, pp. 1747-1760.

[4] F.Küch, W. Kellermann, Nonlinear Echo Cancellation using a Second Order Volterra Filter, ICCASSP 2002, {kuech,wk}@LNT.de.

[5] G. Budura, I. Naforniţa, Kernels Measurement Techniques for Constructing Nonlinear Models, *Scientific Bulletin of the „Politehnica" University of Timişoara*, Romania, Vol.I, 2002,Timişoara, pp. 190-195.

[6] M. Schetzen, *The Volterra and Wiener Theories of Nonlinear Systems*, John Wiley and Sons, New York, 1980.

[7] G. Budura, C.Botoca, La construction d'un modèle non linéaire a l'aide de séries Volterra et Wiener, *Revue Roumaine des Sciences Techniques, Serie Electrotechnique et Energetique*, accepted to be published in no.4, 2005.