SYMBOLIC COMPUTATIONS IN VOLTERRA SYSTEM IDENTIFICATION

K. Kontosis, P. Angelikopoulos, P. Koukoulas, N. Kalouptsidis, I. Emiris

Department of Informatics and Telecommunications
University of Athens, Greece
Email: kalou@di.uoa.gr

ABSTRACT

This paper is concerned with symbolic computations in Volterra system identification using higher order cumulants. An efficient method that implements the Leonov-Shiryaev theorem is introduced. The proposed method relies on the exploitation of recursive relations between cumulants. The method is applied on the problem of blind identification of Volterra-Hammerstein systems excited by stationary higher order white noise. It solves previously intractable instances.

1. INTRODUCTION

Identification of nonlinear systems is of primary importance in several application domains since many signals of interest are generated by nonlinear sources or are processed by nonlinear systems. Volterra systems constitute a popular class of models for modeling nonlinear behavior ([1]).

Volterra-Hammerstein systems provide the simplest gate to the world of nonlinear systems, yet they are general enough to be successfully used in a number of signal processing applications. They were proposed to capture the memory nonlinear effects in the power amplifiers (PA) associated with wideband signals and they are used as a model for the predistorter as well.

Cumulant based Volterra identification, conventional or blind, heavily relies on Leonov Shiryaev (LS) theorem (see [2]). The LS theorem expresses the cumulant of products of random variables in terms of products of cumulants of random variables, summed over the set of indecomposable partitions. Identification algorithms are then devised by meticulously picking output samples so that the subsequent application of the LS on the corresponding output cumulant leads to closed form expressions for the Volterra kernels.

Calculations with indecomposable partitions become fairly complicated even with low degrees of nonlinearities. Symbolic approaches can potentially provide important insight into the problem. This is the main objective of the paper. A symbolic method for the computation of the LS expression is derived and implemented in C++. Our approach outperforms all past works dealing with the application of LS Theorem ([3],[4]) as shown in Section 3.

2. LEONOV-SHIRYAEV THEOREM AND INDECOMPOSABLE PARTITIONS

The LS theorem is stated in this section. For the proof see [5]. The nature of indecomposable partitions is clarified. For a short tutorial on cumulants and their use in signal processing see [6].

Theorem 2.1 [Leonov-Shiryaev] *Consider random variables X_{ij} where $i = 1, \dots, k$, and $j = 1, \dots, m_i$. Define $Y_i = \prod_{j=1}^{m_i} X_{ij}$, then,*

$$cum(Y_1, \dots, Y_k) = \sum_P cum(X_{ij} : (i,j) \in P_1) \cdots cum(X_{ij} : (i,j) \in P_M) \quad (2.1)$$

The summation is over all indecomposable partitions $P = \{P_1, P_2, \cdots, P_M\}$ of the following two-dimensional array:

$$\begin{array}{cccc} (1,1) & (1,2) & \cdots & (1,m_1) \\ (2,1) & (2,2) & \cdots & (2,m_2) \\ \vdots & \vdots & \vdots & \vdots \\ (k,1) & (k,2) & \cdots & (k,m_k) \end{array} \quad (2.2)$$

This array shall be denoted by LS.

Definition 2.1 *A partition P, as above, is* indecomposable*, if there exists no proper subset of sets of P such that their union coincides with the union of some of the rows of the array LS.*

An equivalent definition belongs to Brillinger [7]:

Definition 2.2 *An indecomposable partition arises, if each set communicates with every other set of the partition of the two dimensional array.*

Two sets P_{i_1}, P_{i_2} of the partition *hook*, if there exists an index $(j_1, j_2) \in P_{i_1}$ and $(j_3, j_4) \in P_{i_2}$ such as $j_1 = j_3$. Two sets P_{i_j}, P_{i_k} *communicate* if $P_{i_j}, P_{i_1}, \dots P_{i,k}$ hook consecutively together.

We now give matrix and graph characterizations of indecomposable partitions. Given a partition P of array (2.2) having M elements, we define an $M \times k$ matrix A as follows:

$A(i,j) =$ Number of elements on the j_{th} line of the array that belong to the i_{th} set of the partition. A graph G with k nodes is defined from matrix A in the following way: We connect node i to node t if elements $(i,j), (t,j)$ of A are non-zero, for some j. A $n \times n$ matrix is *Reducible* if there exists a permutation matrix P of dimension $n \times n$, such as $P'AP$ is a block triangular matrix, where $'$ denotes matrix transpose. Irreducible is any matrix that is not reducible.

Theorem 2.2 *The following statements are equivalent:*

1. *P is indecomposable.*
2. *$A'A$ is irreducible.*
3. *G is connected*

Proof 2.2 $1 \leftrightarrow 3$. *Following definition 2.2 , if two sets hook, the elements $(i,j), (t,j)$ of A are non-zero. Thus we connect the corresponding nodes in G. G is obviously connected if and only if all sets communicate, which is the case of an indecomposable partition.*

$2 \leftrightarrow 3$. *Known result. See [8]*

In the sequel we shall refer to k as the order of the cumulant $cum(Y_1, \ldots, Y_k)$ and to $N = \sum_{i=1}^{k} m_i$ as the cardinality of the cumulant. The LS expression (2.1) simplifies if the random variables X_{ij} are zero-mean Gaussian. In this case, only indecomposable partitions with elements of cardinality 2 survive and the corresponding cumulant will be referred to as $Z_M G$ cumulant.

3. ALGORITHMIC APPROACHES AND NP COMPLETENESS

3.1. Brute force schemes

In this section alternative algorithms for the implementation of (2.1) are discussed. A straightforward implementation of (2.1) checks whether each partition of (2.2) is indecomposable. A slight variant of this procedure is discussed in [4]. This algorithm requires $O(Bell(N) \cdot 2^N \cdot p(N))$ steps, where the Bell Numbers $Bell(N)$ provide the number of different partitions of (2.2) and are determined via the expression:

$$e^{e^x - 1} = \sum_{N=0}^{\infty} \frac{Bell(N)}{N!} x^N \tag{3.1}$$

$p(N)$ is a small implementation-specific polynomial of N hence can be neglected. The term $O(2^N)$ rises from the testing of all all $2^{card(P)} = O(2^N)$ subsets of each partition P.

The above brute force algorithm can be improved if Theorem 2.2 is invoked. In this case set enumeration is replaced by a graph connectivity test and complexity drops to $O(Bell(N)p'(N))$.

Numeric calculations indicate that the indecomposable partitions are affluent in the set of all partitions and their number is approximated by $Bell(N)$. To cope with this situation a new expression of the LS relationship (2.1) is introduced that builds upon the recursive structure of cumulants. A new algorithm based on this expression is proposed.

3.2. Recursive relations between cumulants

Let $S = \{ (i,j),\ i = 1, \ldots, k, \text{and } j = 1, \ldots, m_i \}$ be the set of the elements of LS in (2.2).

To any $P \subseteq S$ we associate:

- a vector $B^{(P)}$ where $B_i^{(P)} =$ Number of elements from the i-th row of LS array that belong to P, for $i = 1, \ldots, k$
- The maximal sequence $a_1, a_2, \ldots, a_p$ of *distinct* indices in $\{1 \ldots k\}$ such that $B_{a_1}^{(P)} = B_{a_2}^{(P)} = \cdots = B_{a_p}^{(P)} = 0$

Theorem 3.1 *Consider the random variables X_{ij} where $i = 1, \ldots, k$ and $j = 1, \ldots, m_i$. Define $Y_i = \prod_{j=1}^{m_i} X_{ij},\ i = 1, \ldots, k$. Let v be a random variable, then,*

$$cum\left(v \cdot Y_1, Y_2, \ldots, Y_k\right) = good(S) \cdot cum\left(v, X_{ij}; (i,j) \in S\right) + \sum_{\substack{P \subset S, \\ good(P)=1}} cum\left(v, X_{ij}; (i,j) \in P\right) \cdot cum\left(\prod_{\substack{(i,j) \in \bar{P}, \\ B_i = 0}} X_{ij}, Y_{a_1}, \ldots, Y_{a_p}\right)$$

where: $\bar{P} = S - P$

$$good(P) = \begin{cases} 1 & \text{if } \bar{P} = \emptyset \text{ or } \exists\, (i,j) \in \bar{P} \text{ such that } B_i^{(P)} \neq 0 \\ 0 & otherwise \end{cases}$$

Theorem 3.1 expresses cumulants of products of variables in terms of products of smaller cardinality cumulants. The proof is omitted due to lack of space. Consecutive application of Theorem 3.1 reduces to cumulants of random variables as in (2.1). An example of three applications of Theorem 3.1 follows:

$$\begin{aligned} \mathbf{cum(u^2, u^2)} &= cum(u,u,u,u) + cum(u) \cdot cum(u^2, u) + \\ &\quad 2 \cdot cum(u,u) \cdot cum(u^2) + 3 \cdot cum(u,u,u) \cdot cum(u) \\ \mathbf{cum(u^2, u)} &= cum(u,u,u) + 2 \cdot cum(u) \cdot cum(u,u) \\ \mathbf{cum(u^2)} &= cum(u,u) + cum(u) \cdot cum(u) \end{aligned}$$

If Theorem 3.1 is applied to $Z_M G$ cumulants, the constraint $card(P) = 1$ could be added to the definition of $good(P) = 1$.

On the basis of the above theorem, the symbolic algorithm summarized below is proposed for the recursive computation of cumulants. The key point of the method is that it re-uses precalculated cumulants, as same cumulants are usually requested more than once by the recursive relation. Therefore, only c distinct cumulants are calculated, ending up with an output-sensitive complexity $O(c \cdot 2^N \cdot size \cdot p''(N))$, where $size$ is the number of terms in the final simplified solution. This is because Theorem 3.1 is applied to c cumulants by computing a sum of 2^N terms for each cumulant, and for each such term line 14 consumes $O(size)$ time because it manipulates data of $size$ terms. In the example above $c = 3$ and $size = 4$. It is remarkable that for $Z_M G$ cumulants, the complexity drops to $O(c \cdot size \cdot p''(N))$. Cumulant equivalence is used in order to minimize c. This is achieved by storing precalculated cumulants in a canonical form, which is independent of the variable names. The asymptotic growth of c is a complicated issue, however it is illustrated in Table 1.

Algorithm 1 CUM(X)

Input : cumulant array X as in Theorem 2.1
Output : vector representation of the terms in the righthand side of (2.1)

1: **if** CUM(X) is precalculated **then return** the solution
2: **else** do the following:
3: choose any element v from X, and remove it from X
4: calculate the set S and the array LS from X
5: $Solution \leftarrow 0$
6: **for** each subset $P \subset S$ **do**
7: calculate matrix $B^{(P)}$ from P
8: **if** P is good **then**
9: $X' \leftarrow X$
10: join all rows i, with $B_i^{(P)} \neq 0$ together into one row in X'
11: remove all elements X'_{ij} with $(i,j) \in P$ from X'
12: remove empty rows from X'
13: $C \leftarrow$ CUM(X')
14: $Solution \leftarrow Solution + cum\,(v, X_{ij} : (i,j) \in P) \cdot C$
15: **end if**
16: **end for**
17: $Solution \leftarrow Solution + good(S) \cdot cum(v, X_{ij} : (i,j) \in S)$
18: **return** $Solution$ and store it

3.3. Results

Table 1 shows the maximum cardinality cumulants that were calculated within a time limit of 100 seconds. *ACUMUL*, a C++ implementation of Algorithm 1 was used. V is the number of the distinct

variables of the cumulant, and N is the cumulant cardinality. Since the steps of the algorithm significantly depend on the form of array X, many extreme cases were tested to ensure these results. Values c, $size$ are representative, taken from a single test.

Fig. 1 clearly shows the superior performance of the proposed method. Single variable cumulants were calculated. Cumulants of the form $cum(u^3, u^3, \ldots, u^{n \bmod 3})$ were used, because being a hard case for our method, they point out it's effectiveness.

$Z_M G$ cumulants	
V	*N*
1	72
2	32
3	24
4	20
8	16
14	14

General cumulants			
V	*N*	*c*	*size*
1	18	247	355
2	15	1012	4839
3	14	2080	27150
4	12	946	23167
6	11	535	39595
10	10	31	102505

Table 1. Size of computed problems.

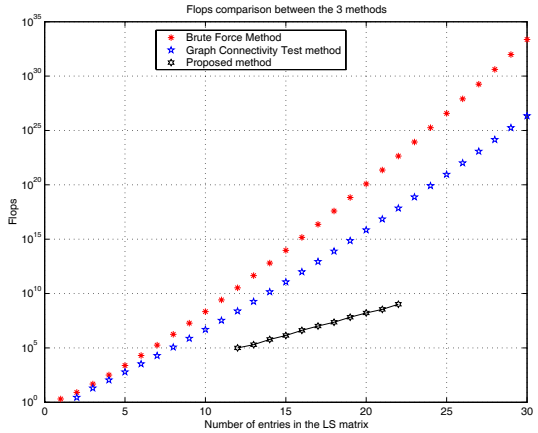

Fig. 1. Comparison of methods

4. BLIND IDENTIFICATION OF VOLTERRA HAMMERSTEIN SYSTEMS

In this section we will use the proposed method to solve the problem of blind identification of Volterra-Hammerstein (V-H) systems. As an example of the whole procedure we shall present the closed form expressions derived for a third order system.

V-H models are considered in [2] and have the following form

$$y(n) = \sum_{k=0}^{q} h_1(k)u(n-k) + \sum_{k=0}^{q} h_2(k)u^2(n-k) + \cdots + \sum_{k=0}^{q} h_p(k)u^p(n-k) + \eta(n). \quad (4.1)$$

$y(n)$ denotes the system output, $u(n)$ denotes the system input and $\eta(n)$ denotes the disturbance. q specifies the system memory while p provides the highest order of nonlinearity. $u(n)$ and $\eta(n)$ are mutually independent. Throughout this section $u(n)$ is stationary zero-mean higher-order white noise meaning that its k-th order cumulant is given by

$$c_u^k(\tau_1, \tau_2, \cdots, \tau_{k-1}) = \gamma_k \delta(\tau_1, \tau_2, \cdots, \tau_{k-1})$$

where $\delta(\tau_1, \tau_2, \cdots, \tau_{k-1})$ is the $(k-1)$-dimensional unit sample signal, which is one for $\tau_1 = \tau_2 = \cdots = \tau_{k-1} = 0$ and zero elsewhere. Likewise $\eta(n)$ is a stationary moving average process of order at most $q-1$. The determination of the Volterra kernels $h_i(k)$ is our main objective.

Let us rewriting eq. (4.1) as a linear multivariable system of the form

$$y(n) = \sum_{i=0}^{q} \mathbf{b}(i)\mathbf{w}(n-i) + \eta(n) \quad (4.2)$$

where

$$\mathbf{w}(n) = \begin{pmatrix} u(n) & u^2(n) & \cdots & u^p(n) \end{pmatrix}'$$

and

$$\mathbf{b}(i) = \begin{pmatrix} h_1(i) & h_2(i) & \cdots & h_p(i) \end{pmatrix}$$

where $'$ denotes matrix transpose. We shall impose the following normalization constraint

$$\mathbf{b}(0) = \begin{pmatrix} 1 & 0 & \cdots & 0 \end{pmatrix}$$

Following the analysis of [2] we arrive at the following equations:

$$\mathbf{c}_y^* = \mathbf{\Gamma}_w^* \mathbf{b}'(q) \quad (4.3)$$

where

$$\mathbf{c}_y^* = \begin{pmatrix} c_y^2(q) & \bar{c}_y^3(q,0) & \cdots & \bar{c}_y^p(q,0) & \bar{c}_y^{p+1}(q,0) \end{pmatrix}' \quad (4.4)$$

and

$$\mathbf{c}_y^\circ = \mathbf{\Gamma}_w^\circ \mathbf{b}'(\tau) \quad (4.5)$$

where

$$\mathbf{c}_y^\circ = \begin{pmatrix} \bar{c}_y^3(q,\tau) & \bar{c}_y^4(q,\tau) & \cdots & \bar{c}_y^{p+2}(q,\tau) \end{pmatrix}' \quad (4.6)$$

$$\mathbf{\Gamma}_w^* = \begin{pmatrix} (\bar{\mathbf{\Gamma}}_{2w})_{1\times p} \\ (\bar{\mathbf{\Gamma}}_{3w})_{1\times p} \\ \vdots \\ (\bar{\mathbf{\Gamma}}_{pw})_{1\times p} \\ (\bar{\mathbf{\Gamma}}_{(p+1)w})_{1\times p} \end{pmatrix} \quad (4.7)$$

and

$$\mathbf{\Gamma}_w^\circ = \begin{pmatrix} \mathbf{b}(q)(\bar{\mathbf{\Gamma}}_{3w})'_{p\times p} \\ \mathbf{b}(q)(\bar{\mathbf{\Gamma}}_{4w})'_{p\times p} \\ \vdots \\ \mathbf{b}(q)(\bar{\mathbf{\Gamma}}_{(p+1)w})'_{p\times p} \\ \mathbf{b}(q)(\bar{\mathbf{\Gamma}}_{(p+2)w})'_{p\times p} \end{pmatrix} \quad (4.8)$$

The matrices $\mathbf{\Gamma}_w^*$ and $\mathbf{\Gamma}_w^\circ$ consist of cumulants of products of the input variable $u(n)$. The rows of $\mathbf{\Gamma}_w^*$ are the vectors $(\bar{\mathbf{\Gamma}}_{kw})_{1\times p}$ whose j-th element, $1 \le j \le p$, is given by

$$\text{cum}\left(u(n), u(n), \cdots, u(n), u^j(n)\right) \quad (4.9)$$

The cumulant operator involves k variables where $k-1$ of them are equal to $u(n)$ while one is equal to $u^j(n)$.
Likewise the (i,j) element of the matrix $(\bar{\mathbf{\Gamma}}_{kw})_{p\times p}$, with $1 \le i,j \le p$, is given by

$$\text{cum}\left(u(n), u(n), \cdots, u(n), u^j(n), u^i(n)\right) \tag{4.10}$$

A symbolic algorithm for the solution of the blind identification problem when the input intensities γ_j, $1 \le j \le 3p$, are known involves the following 4 steps:

1. Determine the entries of the matrix $\mathbf{\Gamma}_w^*$ of the linear system solver (4.3) using the LS symbolic procedure described in section (3.2)
2. Compute the boundary vector $\mathbf{b}(q)$ applying a symbolic linear solver (4.3). This step has been analyzed in detail in the specific case of section 4.2.
3. Use a matrix by vector multiplication symbolic module to calculate $\mathbf{\Gamma}_w^\circ$ via (4.8)
4. Determine the vector $\mathbf{b}(\tau)$ by solving the linear system (4.5).

For steps 2 and 4 we rely on [9]. The parameters $\mathbf{c}_y^*$ and $\mathbf{c}_y^\circ$ are directly obtained from output cumulants. The resulting parameters $\mathbf{b}(q)$ and $\mathbf{b}(\tau)$ are rational functions of γ_j, $\mathbf{c}_y^*$ and $\mathbf{c}_y^\circ$.

It follows from the above description that the main barriers that need to be removed are related to the application of the Leonov - Shiryaev formula in the matrices $\mathbf{\Gamma}_w^*$ and $\mathbf{\Gamma}_w^\circ$ in eqs. (4.3) and (4.5). This is achieved in section 3.2.

4.1. Computational requirements

Consider a V-H system of order p. According to the previous analysis we need output cumulants of order up to $p+2$ (or of order up to $2p$ [2] if we want to ensure solvability). Due to (4.10) this will yield a cumulant cardinality $N \le 3p$. By using our tool, a V-H system of order 7 needs cumulants of cardinality $N = 21$.

4.2. Application in a Volterra-Hammerstein system of order 3

We illustrate the above symbolic procedure in a V-H system of order 3 when the input is white Gaussian. In this case the rational functions specifying the kernels are considerably simplified provided that output cumulants are taken in the range $p+2, \cdots, 2p$ rather than $3, \cdots, p+2$. Let us first consider identifiability. This case is secured if the determinant of $\mathbf{\Gamma}_w^*$ as well as the determinant $\mathbf{\Gamma}_w^\circ$ are nonzero. In the general case the determinant of $\mathbf{\Gamma}_w^*$ turns out to be:

$$\prod_{i=1}^{p} i!\gamma_2^{p(p+1)/2} \tag{4.11}$$

Consider the case $p=3$. The determinant is $12\gamma_2^6$. Clearly this is always nonzero because $\gamma_2 \neq 0$. The matrix $\mathbf{\Gamma}_w^\circ$ is triangular. Generally $\det(\mathbf{\Gamma}_w^\circ)$ is an integer multiple of $(\bar{c}_y^{p+1}(q,0))^p \gamma_2^{p(p-1)/2}$. In the case $p=3$ the determinant is $216\gamma_2^3(\bar{c}_y^4(q,0))^3$. Moreover, $\bar{c}_y^4(q,0) = 6h_3(q)\gamma_2^3$. Therefore the determinant of $\mathbf{\Gamma}_w^\circ$ is nonzero if $h_p(q) \neq 0$.

Application of step 2 provides the following rational functions for the boundary kernels.

$$\begin{aligned} h_3(q) &= \frac{\bar{c}_y^4(q,0)}{6\gamma_2^3} \\ h_2(q) &= \frac{\bar{c}_y^3(q,0)}{2\gamma_2^2} \\ h_1(q) &= \frac{\bar{c}_y^2(q)}{\gamma_2} - \frac{\bar{c}_y^2(q)}{2\gamma_2^2} \end{aligned}$$

Finally application of step 4 yields the following expressions for the remaining coefficients:

$$\begin{aligned} h_3(\tau) &= \frac{\bar{c}_y^6(q,\tau)}{36\gamma_2^2\bar{c}_y^4(q,0)} \\ h_2(\tau) &= \frac{\bar{c}_y^5(q,\tau)}{6\gamma_2\bar{c}_y^4(q,0)} - \frac{\bar{c}_y^3(q,0)\bar{c}_y^6(q,\tau)}{12\gamma_2\bar{c}_y^4(q,0)\bar{c}_y^4(q,0)} \\ h_1(\tau) &= \frac{\bar{c}_y^3(q,\tau)}{\bar{c}_y^3(q,0)} - \frac{2\bar{c}_y^3(q,0)\bar{c}_y^5(q,\tau)}{3\bar{c}_y^4(q,0)\bar{c}_y^4(q,0)} \\ &\quad + \frac{\bar{c}_y^6(q,\tau)\bar{c}_y^3(q,0)\bar{c}_y^3(q,0)}{3\bar{c}_y^4(q,0)\bar{c}_y^4(q,0)\bar{c}_y^4(q,0)} - \frac{\bar{c}_y^6(q,\tau)\bar{c}_y^2(q)}{6\bar{c}_y^4(q,0)\bar{c}_y^4(q,0)} \\ &\quad - \frac{\bar{c}_y^6(q,\tau)}{4\gamma_2\bar{c}_y^4(q,0)} \end{aligned}$$

5. CONCLUSION

In this paper a symbolic procedure for the implementation of the Leonov-Shiryeav theorem is derived and is applied to the problem of blind identification of Volterra-Hammerstein systems. Work in progress considers creating a database of all cumulants of arbitrarily high order, and their application in solving the blind identification problem of general Volterra systems.

6. REFERENCES

[1] V. J. Mathews and G. L. Sicuranza, *Polynomial Signal Processing*, Wiley, New York, 2000.

[2] N. Kalouptsidis and P. Koukoulas, "Blind identification of Volterra Hammerstein systems," *IEEE Trans. Signal Processing*, vol. 53, pp. 2777–2787, August 2005.

[3] B. Smith and C. Field, "Symbolic cumulant calculations for frequency domain time series," *Statistics and Computing*, vol. 11, no. 1, pp. 75–82, January 2001.

[4] D.F. Andrews and J.E. Stafford, "Iterated full partitions," *Statistics and Computing*, vol. 9, pp. 189–192, 1998.

[5] V.P. Leonov and A.N. Shiryaev, "On a method of calculation for semi-invariants," *Theory of Probability & its applications*, vol. 4, pp. 319–329, 1959.

[6] J.M. Mendel, "Tutorial on higher-order statistics (spectra) in signal processing and system theory: Theoretical results and some applications," *Proc. IEEE*, vol. 79, pp. 278–305, 1991.

[7] D.R. Brillinger and M. Rosenblatt, *Asymptotic Theory of K-th Order Spectra*, Spectral Analysis of Time Series. Wiley, 1967.

[8] R.S. Varga, *Matrix Iterative Analysis*, Springer, 2nd edition, 1999.

[9] I. Gohberg, P. Lancaster, and L. Rodman, *Matrix Polynomials*, Academic Press, New York, 1982.