Multidimensional Multiple-Order Complex Parametric Model Identification

Denis Kouamé, Member, IEEE, and Jean-Marc Girault, Member, IEEE

Abstract—This paper presents a way to access both the multiple-order and parameters of a multidimensional complex number autoregressive (AR) model through matrix factorization. The principle of this technique consists of the transformation of the multidimensional model to a pseudo simple-input simple-output AR model, then performing factorization of the covariance matrix of the data. This factorization then leads to a recursive form of the parameter and order estimation. This paper makes two principal contributions. The first is a generalization of one dimensional factored form algorithm, and the second is that it makes it possible to access all the possible different orders and parameters of a multidimensional complex number AR model of any dimension, whereas classical approaches are limited to at most four-dimensional models. Computer simulation results are provided to illustrate the behavior of this method.

Index Terms—Autoregressive, model, multidimensional, order, parameter.

I. INTRODUCTION

This paper is a generalization of the works recently reported in [1], where the investigations were restricted to a model with the same order in each dimension. Multidimensional parametric modeling has achieved a high degree of popularity in the literature due to its success in the areas of spectral analysis and image processing. Over 20 years in the field of multidimensional parametric modeling, two-dimensional (N = 2) autoregressive (2D-AR) modeling has received much attention in many areas, particularly in digital signal- and image-processing areas. Many works have therefore merged, e.g., [2]–[5] and relevant references therein. Although three-dimensional (N = 3) AR (3D-AR) has received some attention, studies concerning more than three dimensions are rare, particularly in contrast to their potential applications [6]–[8]. This is mainly due to the high complexity of the generalization of 2D-AR cases.

The aim of this paper is to propose a general framework for estimation of general N-dimensional (ND) AR model parameters and orders. The key point here is that, unlike the study reported in [1], the model orders may be different in each dimension. The problem of multidimensional order estimation is of great importance in several areas, and many works have been devoted to it—e.g., [7] and [9]. Based on matrix factorization, the technique presented here can yield a straightforward recursive-like form of the algorithm for both dimensions (time, space,•••) and orders. It allows easy AR modeling of, for example, three, four (or more) dimension signals. The properties of the algorithm are introduced and its behavior is illustrated with N = 2, N = 3 and N = 4 using a numerical example.

II. PRELIMINARY

The following notations will be used. Bold type will be used for vectors. Thus, when no confusion is possible

\[
\mathbf{n} = (n_1, n_2, \ldots, n_N)
\]

\[
\mathbf{k} = (k_1, k_2, \ldots, k_N)
\]

and generally any set of index variables \((t_1, t_2, \ldots, t_N)\) will be denoted \(\mathbf{t} = (t_1, t_2, \ldots, t_N)\). Consequently, the value of an N-dimensional field \(y\) at \((n_1, n_2, \ldots, n_N)\) will be denoted \(y(n_1, n_2, \ldots, n_N) = y(\mathbf{n})\). Also \(\sum_{i=1}^{n_1} \cdots \sum_{i=N}^{n_N}\) will be denoted \(\sum_{i=1}^{n_1} \cdots \sum_{i=N}^{n_N}\) and \(\sum_{i=1}^{n_1} \cdots \sum_{i=N}^{n_N}\) will be denoted \(\sum_{i=1}^{n_1} \cdots \sum_{i=N}^{n_N}\). Finally, \(\lim_{n_1 \to -\infty} \lim_{n_2 \to -\infty} \cdots \lim_{n_N \to \infty} \) will be denoted \(\lim_{\mathbf{n} \to \infty}\).

III. MULTIPLE ORDER MULTIDIMENSIONAL COMPLEX AR IDENTIFICATION

A. Methodology Presentation

The multiple-order multidimensional model problem may be formulated using multidimensional AR models. A complex-number multidimensional or ND-AR represents \(y(\mathbf{n})\), the components of the complex number signal \(y\) at location \(\mathbf{n}, \mathbf{n} = \{n_i\}_{i=1}^{N}\), as a linear combination of the complex number components \(y(n_1-k_1, n_2-k_2, \ldots, n_N-k_N)\) and an additive noise \(u(n_1, n_2, \ldots, n_N)\), where \((k_1, k_2, \ldots, k_N) \in I\) and \(I\) is a set of neighbors excluding \((0,0,\ldots,0)\). This paper is primarily concerned with developing an approach for \(N \neq 1\), although the results are also valid when \(N = 1\). We consider a second-order stationary multidimensional complex AR process (ND-AR) defined by

\[
y(\mathbf{n}) = \sum_{\mathbf{k}} a(\mathbf{k}) y(\mathbf{n} - \mathbf{k}) + u(\mathbf{n})
\]

(1)

where \(u(\mathbf{n})\) is a field of zero-mean constant variance-independent random noise and the parameters \(a(\mathbf{k})\) are complex numbers and provide a stable system. In the following, attention is focused on the first hyperplane model, without loss of gen-
erality (the methodology may be applied to the other hyperplanes), i.e., the set of neighbors is \( I = \{ k_1, k_2, \ldots, k_N \} \setminus k_i = 0, 1, 2, \ldots, p_i, i = 1, 2, \ldots, N \} \setminus (0, 0, \ldots, 0) \). Here, one can have
\[
p_i \neq p_2 \neq \cdots p_N
\tag{2}
\]
i.e., the model orders are not necessarily identical in all directions. Our aim here is to estimate both the \( p_i; i = 1 \cdots N \) and the \( a(k) \) using a recursive-like algorithm, which generalizes UD\(^T\)\(H\) matrix decomposition \([10], [11]\).

First define \( m \) such that
\[
m > \max \{ p_i; i = 1 \cdots N \}.
\]

Also define the following vectors in which elements of \( y \) and \( a \) are stacked:
\[
\theta_T^n = \begin{bmatrix} a(0, 0, \ldots, 1) \cdots a(0, 0, \ldots, m) \\
a(0, 1, 0, \ldots, 0) \cdots a(0, m, 0, \ldots, 0) \\
a(0, 0, \ldots, 0) \cdots a(m, m, m, \ldots, m) \end{bmatrix}.
\tag{3}
\]

Note that \( y(n_1, n_2, \ldots, n_N) \) and 1 are part of these vectors. Let also
\[
x_T^n(n_1, n_2, \ldots, n_N)
= \begin{bmatrix} y(n_1, n_2, \ldots, n_N - 1) \cdots y(n_1, n_2, \ldots, n_N - m) \\
y(n_1, n_2 - 1, \ldots, n_N) \cdots y(n_1, n_2 - m, \ldots, n_N) \\
y(n_1 - m, n_2 - m, \ldots, n_N - m) \end{bmatrix}.
\tag{4}
\]

Thus, (1) is equivalent to
\[
y(n) = x_T^n(n) \theta_T^n + w(n)
\tag{5}
\]
and finally
\[
\phi_T^n(n) = \begin{bmatrix} x_T^n(n) \\ y(n) \end{bmatrix}
\tag{6}
\]
where \( H \) denotes the Hermitian matrix transpose.

Remark 1: From these definitions, it is clear that the defined vectors (and also the matrices defined from these vectors) are all pseudoperiodic over \((m+1)N\) and \(N\). For \(N\), this is due to the fact the matrix elements have been stacked dimension by dimension. The periodicity over \((m+1)\) can be seen explicitly in \(\theta_T^n \) (3). It behaves as if the actual model parameters have been filled with zeros to constitute sets of \((m+1)\) elements, and thus \(x_T^n\), \(\phi_T^n\) and all resulting matrices and vectors have the same type of periodicity.

The data covariance matrix can then be defined as
\[
P_m(n) = \begin{bmatrix} \sum_{i=1}^{n} \phi_T^n(n) \phi_T^{H}(n) \end{bmatrix}^{-1}.
\tag{7}
\]
Assuming that \(p = (m+1)^N\), the size of this matrix is \(p \times p\).

\(P_m\) can then be written in factored form as follows:
\[
P_m(n) = U_m(n)D_m(n)U_m^{H}(n)
\tag{8}
\]
where \(H\) denotes the Hermitian matrix transpose and \(U\) is an upper triangular matrix with all diagonal elements equal to unity. The elements of this upper triangular matrix are column vectors with dimension 1 to \(p\) defined as follows:
\[
\begin{align*}
U_m(n) &= \begin{bmatrix} \text{col} \{ \partial_{\lambda p}(n) \} \cdots \text{col} \{ \partial_{p-i,i}(n) \} \cdots \text{col} \{ \partial_{p,0}(n) \} \end{bmatrix}\\
D_m(n) &= \begin{bmatrix} \cdots \text{col} \{ \partial_{p-1,1}(n) \} \cdots \text{col} \{ \partial_{p,0}(n) \} \end{bmatrix}
\end{align*}
\tag{9}
\]

\(\partial_{p-i,i}(n)\) is explained below.
\(D_m(n)\) in (8) is a diagonal matrix.

Remark 2:
\(\partial_{p-i,i}(n)\) is a column vector. Its dimension is \(p - i\).
\[
\begin{bmatrix} \text{col} \{ \partial_{p-i,i}(n) \} \end{bmatrix}^{-1} = \begin{bmatrix} \partial_{p-i,i}(n) \end{bmatrix}
\tag{10}
\]
is the \((p - i + 1)\)th column.

The diagonal matrix \(D_m(n)\) in (8) contains loss functions for order \(0\) to \(m\), as seen below. The components of its diagonal have \(N\) types of periodicity, each type being with period \(m+1\). This actually means by definition (see (4),(6), and (7)) that there is an \((m+1)\)-periodicity in each dimension of such a process, as explained in Remark 1.

Accordingly there is the same periodicity in the columns of \(U\) and in the components of each column.

Due to the structure of the model defined in (3), \(\partial_{p-i,i}(n)\) consists of part or all of the \(true\) parameters of the model, depending on the model order and the dimension of \(\partial_{p-i,i}(n)\), as explained below.

Equation (8) is achieved from successive decompositions. Indeed, using (6) in (7)
\[
P^{-1}_m(n) = \begin{bmatrix} 1 & -x_n(j)x_m(H) \\
-\sum_{j=1}^{m} x_n(j)x_m(H) & \sum_{j=1}^{m} x_n(j)y(j) \\
-\sum_{j=1}^{m} x_n(j)y(j) & \sum_{j=1}^{m} x_n(j)y(j) \\
\end{bmatrix}^{-1}.
\tag{11}
\]
From classical least squares estimation theory (see [11]), it is known that
\[
\partial_{p,0}(n) = \begin{bmatrix} \sum_{j=1}^{n} x_m(j)x_m(j) \end{bmatrix}^{-1} \begin{bmatrix} \sum_{j=1}^{n} x_m(j)y(j) \end{bmatrix}.
\tag{12}
\]
or similarly
\[
\partial_{p-i,i}(n) = \begin{bmatrix} \sum_{j=1}^{n-i} x_m(j)x_m(j) \end{bmatrix}^{-1} \begin{bmatrix} \sum_{j=1}^{n-i} x_m(j)y(j) \end{bmatrix}.
\tag{13}
\]
Thus (11) may also be written
\[
P^{-1}_m(n) = \begin{bmatrix} 1 & 0 \\
0 & I_{p-i,0}^*(n) \\
0 & 0 & C_{p,0}(n) \\
\end{bmatrix} \times \begin{bmatrix} I_{p-i,0}^*(n) & 0 \\
0 & \partial_{p,0}^H(n) \\
\end{bmatrix} 
\tag{14}
\]
where
\[
Q_m(n) = \sum_{i=1}^{n} x_m(i)x_m(i)
\tag{15}
\]
\[
C_{p,0}(n) = \sum_{j=1}^{n} y(j)^2 \\
- \partial_{p,0}^H(n) \sum_{j=1}^{n} x_m(j)x_m(j)\partial_{p,0}(n).
\tag{16}
\]
\( C_{p,0} \) is equivalent to a loss function. Indeed, (12) is also equivalent to
\[
\sum_{j=1}^{n} x_m(j)x_m^H(j) \quad \vartheta_{p,0}(n) = \sum_{j=1}^{n} x_m(j)y(j). \tag{17}
\]
Defining
\[
\hat{y}(n) = \sum_{j=1}^{n} x_m^H(j)\vartheta_{p,0}(n)
\]
yields (see Appendix A)
\[
C_{p,0}(n) = \sum_{j=1}^{n} |y(j) - \hat{y}(j)|^2
\]
and generally
\[
C_{p-i,d}(n) = \sum_{j=1}^{n-i} |y(j) - \hat{y}(j)|^2. \tag{20}
\]
Thus, the \( C_{p-i,d}(n) \) are positive. Now, returning to (14), \( Q_m(n) \), which is in the form of (7), is then iteratively decomposed in the same way. Ultimately, after taking the inverse of the matrices involved in the decomposition, (8) is obtained. Thus \( D_m \) and, namely, the inverse of its elements is equivalent to the loss function (see Appendix A) and, according to its periodicity, its minimum in each direction will exhibit the relevant order as in classical estimation theory.

As mentioned above, \( U_m(n) \) is an upper triangular complex number matrix, the elements of which are \( \vartheta_{p-i,d}(n) \).

\( D_m(n) \) is a diagonal matrix, the elements of which are the inverse of the loss function, as mentioned above, i.e., \( C_{p-i,d}^{-1}(n) \). It is important to note that the similarity with the case when the order is identical in all the dimensions is only apparent. Here, every thing has to be analyzed dimension by dimension. Since the components of the covariance matrix \( P_m \) have \( N \) types of periodicity, each type with period \( m_i+1 \) (see Remark 1), our model has the following characteristics (see Appendixes A and D).

- The true parameters as defined in (3) are not directly the columns of \( U_m \) but are included in these columns. This is due to the definitions of the model in (3). The true model parameter vector is filled with sets of zeros to construct \( \theta_m \).
- Indeed, assuming the order of the model is \( (p_1, p_2, \ldots, p_N) \), then the actual parameters \( a(k) \) of the model are arranged in the vector denoted \( \theta = \theta_{m_0} \), where \( m_0 \) is the number of the column containing the estimates of the true model parameter vector. That is

\[
\begin{bmatrix}
a(0,0,\ldots,1) & \cdots & a(0,0,\ldots,p_1) \\
0 & \cdots & 0 \end{bmatrix}
\]

This vector has \( \prod_{k=1}^{N} (1+p_k) - 1 \) “useful” components, which are actual parameters.

For convenience let us assume that
\[
\begin{bmatrix}
a(0,0,\ldots,1) & \cdots & a(0,0,\ldots,p_1) \\
0 & \cdots & 0 \end{bmatrix}
\]
\[\text{dim} \theta = m_0 \tag{22}\]
\[
\int \left( (p_1+1)(p_2+1)(p_3+1) - 1 \right) \frac{1}{(p_1+1)(p_2+1)}(m+1)^2 + 1
\]
where \( \text{Int} \) represents the integer part of the fraction. This expression is modulo \( (m+1) \). We illustrate this in the example given below (Section IV).

- Returning to the model order, the elements of \( D_m \) are pseudoperiodic, with a period \( m+1 \) in each direction (dimension); i.e., \( D_m \) has \( N \) types of periodicity, as explained in Remark 1. This is a direct consequence of arrangement of the element of \( \theta_m \) defined in (6). The minimum within each \( (N) \) type of periodicity constitutes the actual order \( (p_1, p_2, \ldots, p_N) \). Thus, the minimum of each periodicity has to be found to find the model order.

This is now clear that the order recursion is achieved from the above. The dimension (time, space, \ldots)-recursion may be obtained by using UDU* decomposition. This last recursion is obtained by defining \( n - 1 = (n_1 - 1, n_2 - 1, \ldots, n_N - 1) \) and writing the relevant recursions. From (7), it can be written as follows:
\[
P_m(n) = [P_m^{-1}(n-1) + \phi_m(n)\vartheta_m^H(n)]^{-1}. \tag{23}\]
Defining the variables as in the one-dimensional case [12], \( f = U_m^* (n-1) P_m (n) \), \( g = D_m (n-1) P_m \), and \( \beta (n) = 1 + f^T g \), where the asterisk denotes the complex conjugate, at the rank one update expression, the matrix \( P_m (n) \) can be expressed by

\[
P_m (n) = U_m (n) D_m (n) U_m^H (n)
= U_m (n-1) \left[ D_m (n-1) - \frac{gg^H}{\beta (n)} \right] U_m^H (n-1).
\]

(24)

From these recursions, only elements of \( U_m \) with physical meanings are retained. To speed up computations, the recursions may be applied only to the elements with physical meanings. The overall complexity of the parameter estimation algorithm, in terms of number of operations \( \chi \), for data \( y \) of size \((N_1 \max \times N_2 \max \times \ldots \times N_N \max)\), is [10]

\[
\chi = [1.5 (m+1)^2 + 1.5 (m+1)^N] \\
\times \prod_{k=1}^{N} N_k \max \text{ complex additions} \\
+ [1.5 (m+1)^2 + 5.5 (m+1)^N] \\
\times \prod_{k=1}^{N} N_k \max \text{ complex multiplications} \\
+ [(m+1)^N] \prod_{k=1}^{N} N_k \max \text{ complex divisions}
\]

(25)

where “complex operations” (additions, multiplications, divisions) means operations on complex numbers. To summarize, the computational load of the algorithm is nearly \((m+1)^2 + 5.5 (m+1)^N\) for each measurement.

\textbf{Remark 3:} The methodology developed here may be straightforwardly applied to any general one-dimensional complex models (e.g., ARX, ARMAX, etc.) to estimate their parameters and orders.

Finally, the properties in terms of uniqueness, exactness, and convergence of the parameter estimates developed above are proven in Appendixes C and D.

\section{Summary of the Proposed Algorithms}

Two algorithms can be obtained from the above. The first, referred to as the batch algorithm, deals with parameter estimation in the case of a stationary model, i.e., when model parameters do not vary. The second, referred to as the recursive algorithm, deals with recursive parameters estimation. In both cases, the order is estimated simultaneously. The proposed algorithms can be summarized as follows, given \( N \)-dimensional complex data \( y(n) = y(n_1, n_2, \ldots, n_N) \) with \( n_1 = 1, \ldots, N_1 \max, n_2 = 1, \ldots, N_2 \max, \ldots, n_N = 1, \ldots, N_N \max \) representing an AR model (1).

\begin{enumerate}
  \item \textbf{Batch Algorithm:}
  \begin{enumerate}
    \item Choose an arbitrary number \( m \) (maximum possible order)
    \item Stack elements of \( y \) according to (4) to obtain vector \( \phi_m (n) \) according to (6)
    \item Compute \( P_m (n) \) according to (7)
    \item Decompose \( P_m (n) \) according to (8) and take the inverse of \( U_m \). Any decomposition may be used. See an example in Appendix B
    \item Find the \( N \) types of periodicity in the (diagonal) elements of \( D_m \)
    \item Find the minimum in each type of periodicity
    \item Find the column of \( U \), the number of which is \( m_0 \), containing the useful parameters (i.e., the parameters of the model); the number may be chosen as (see Fig. 2)
  \end{enumerate}
  \[
  m_0 \sim \text{Int} \left\{ \frac{\prod_{k=1}^{N} (y_k + 1)}{\prod_{k=1; k \neq k_0}^{N} (y_k + 1)} \right\} + 1
  \]
  \[
  \text{where Int denotes the integer part of the fraction, } k_0 \text{ is the direction in which elements of the data } y \text{ are stacked. Due to } (m+1)-\text{periodicity, the columns } m_0 + (m+1), m_0 + 2(m+1), \ldots \text{ also contain useful parameters. Note that this number should not be greater than } (m+1)^N, \text{ the number of columns of } U.
  \]
  \item \textbf{Recursive Algorithm:} This algorithm is based on the Bierman technique [10]. For readability, we set \( U = U_m \) and \( D = D_m \).
  \begin{enumerate}
    \item Same as step 1 of batch algorithm
    \item Set \( d = (m+1)^N; r = 1; \)
    \item Initialize: \( U = I \) and \( D = \mu I \); where \( \mu \) is a large number and \( I \) is \( d \times d \) identity matrix.
    \item for \( j_1 = n + 1, \ldots, N_1 \max \) for \( j_2 = m + 1, \ldots, N_2 \max \) for \( j_3 = m + 1, \ldots, N_3 \max \) for \( j_N = n + 1, \ldots, N_N \max \) \( \phi_m = \left[ -y(j_1 - m : j_1, j_2 - m : j_2, \ldots, j_N - m : j_N) \right] \).
    \item Construct a vector \( \phi \) by stacking elements of \( \phi_m \)
    \item \( f = U^T \phi \)
    \item \( g = Df^* \)
    \item \( \alpha = r + D(1, 1) |f(1)|^2 \)
    \item \( \gamma = 1/\alpha \)
  \end{enumerate}
\end{enumerate}
9. $D(1,1) = D(1,1)r_\gamma / r$
10. for $jj = 2, \ldots, d$
   \[ \beta = \alpha \]
   \[ \alpha = \alpha + D(jj,jj)f(jj)^2 \]
   \[ \lambda = -f(jj)\gamma \]
   \[ \gamma = 1/\alpha \]
   $D(jj,jj) = D(jj,jj)\gamma * \beta / r$
11. for $ii = 1, \ldots, jj - 1$
   \[ \beta = U(ii, jj) \]
   $U(ii, jj) = \beta + \lambda g(ii)$
   \[ g(ii) = g(ii) + g(jj) \beta \]
   skip to step 11
   skip to step 10
   $\alpha, \beta, \gamma, f, g$ are intermediate variables
12. Same as step 6 of batch algorithm
13. Same as step 7 of batch algorithm

IV. NUMERICAL SIMULATIONS

For readability, we performed simulations on two-, three-, and four-dimensional cases.

A. 2-D Case

To illustrate the exactness of the parameter estimation, we compare here the proposed method and well-known Yule Walker algorithm in the complex version; see [13]. The model used was

\[ x(n_1, n_2) = \sum_{k_1=0}^{d_1} \sum_{k_2=0}^{d_2} a(k_1, k_2) \times x(n_1 - k_1, n_2 - k_2) + w(n_1, n_2) \]

where $y$ is a $256 \times 256$ complex field driven by a complex Gaussian random field with variance $\sigma^2$. We took different values of $\sigma$ in 0.01; 0.1; 1; 10 and $m = 10$. The theoretical parameter values and results are shown in Tables I and II. Tables III and IV show the extracted minima in each direction versus the standard deviation of the additive noise. $d_1$ is the minima in the first direction allowing access to $p_1$, and $d_2$ is the same in the second direction and allows access to $p_2$. The first lines of the Tables are order 0, the second lines are order 1, etc. As can be seen, for $d_1$, there is no significant change from line 3, i.e., $p_1 = 2$ and similarly $p_2 = 3$. The behavior of $D_m$ is illustrated in the 3-D example below.

From these results, it can be seen that the proposed approach has a very good numerical stability and the effect of noise is limited.

B. 3-D Case

For simplicity in the presentation of the parameters of the model, we set $N = 3, m = 6, (p_1, p_2, p_3) = (2, 3, 4)$. We considered the following 3-D complex AR model as defined in (3) by:

\[ x(n_1, n_2, n_3) = \sum_{k_1=0}^{d_1} \sum_{k_2=0}^{d_2} \sum_{k_3=0}^{d_3} a(k_1, k_2, k_3) \times x(n_1 - k_1, n_2 - k_2, n_3 - k_3) + w(n_1, n_2, n_3) \]

where $y$ is a $256 \times 256 \times 256$ complex field driven by a complex Gaussian random field with variance $\sigma^2 = 0.1$. The parameter vector is thus arranged as in Table V.


TABLE III
LOSS FUNCTION VERSUS σ IN THE 2-D CASE, WITH m = 10, p1 = 2, p2 = 3

\[ \sigma = 0.01, 0.1 \]

<table>
<thead>
<tr>
<th>( d_2 (\sigma = 0.10) )</th>
<th>( d_1 (\sigma = 0.10) )</th>
<th>( d_2 (\sigma = 10) )</th>
<th>( d_1 (\sigma = 10) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>23.9466</td>
<td>17.3058</td>
<td>2.3428 \times 10^3</td>
<td>1.6892 \times 10^3</td>
</tr>
<tr>
<td>7.9375</td>
<td>7.9375</td>
<td>7.893 \times 10^3</td>
<td>7.893 \times 10^3</td>
</tr>
<tr>
<td>4.4735</td>
<td>5.8423</td>
<td>4.445 \times 10^3</td>
<td>5.770 \times 10^3</td>
</tr>
<tr>
<td>3.6113</td>
<td>5.3419</td>
<td>3.576 \times 10^3</td>
<td>5.262 \times 10^3</td>
</tr>
<tr>
<td>3.5212</td>
<td>5.3394</td>
<td>3.467 \times 10^3</td>
<td>5.272 \times 10^3</td>
</tr>
<tr>
<td>3.5093</td>
<td>5.3426</td>
<td>3.470 \times 10^3</td>
<td>5.267 \times 10^3</td>
</tr>
<tr>
<td>3.5072</td>
<td>5.3233</td>
<td>3.465 \times 10^3</td>
<td>5.274 \times 10^3</td>
</tr>
<tr>
<td>3.5097</td>
<td>5.3561</td>
<td>3.459 \times 10^3</td>
<td>5.278 \times 10^3</td>
</tr>
<tr>
<td>3.5098</td>
<td>5.3691</td>
<td>3.459 \times 10^3</td>
<td>5.284 \times 10^3</td>
</tr>
<tr>
<td>3.5060</td>
<td>5.3655</td>
<td>3.455 \times 10^3</td>
<td>5.297 \times 10^3</td>
</tr>
<tr>
<td>3.4924</td>
<td>5.3513</td>
<td>3.452 \times 10^3</td>
<td>5.274 \times 10^3</td>
</tr>
</tbody>
</table>

TABLE IV
LOSS FUNCTION VERSUS σ IN THE 2-D CASE, WITH m = 10, p1 = 2, p2 = 3

\[ \sigma = 1, 10 \]

<table>
<thead>
<tr>
<th>( d_2 (\sigma = 1) )</th>
<th>( d_1 (\sigma = 1) )</th>
<th>( d_2 (\sigma = 10) )</th>
<th>( d_1 (\sigma = 10) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3076 \times 10^3</td>
<td>1.7549 \times 10^3</td>
<td>2.5101 \times 10^3</td>
<td>1.7708 \times 10^3</td>
</tr>
<tr>
<td>0.8049 \times 10^3</td>
<td>0.8049 \times 10^3</td>
<td>0.8226 \times 10^3</td>
<td>0.8226 \times 10^3</td>
</tr>
<tr>
<td>0.4094 \times 10^3</td>
<td>0.5937 \times 10^3</td>
<td>0.4571 \times 10^3</td>
<td>0.5093 \times 10^3</td>
</tr>
<tr>
<td>0.3601 \times 10^3</td>
<td>0.5462 \times 10^3</td>
<td>0.3602 \times 10^3</td>
<td>0.5431 \times 10^3</td>
</tr>
<tr>
<td>0.3490 \times 10^3</td>
<td>0.5478 \times 10^3</td>
<td>0.3491 \times 10^3</td>
<td>0.5421 \times 10^3</td>
</tr>
<tr>
<td>0.3494 \times 10^3</td>
<td>0.5475 \times 10^3</td>
<td>0.3481 \times 10^3</td>
<td>0.5419 \times 10^3</td>
</tr>
<tr>
<td>0.3489 \times 10^3</td>
<td>0.5481 \times 10^3</td>
<td>0.3490 \times 10^3</td>
<td>0.5439 \times 10^3</td>
</tr>
<tr>
<td>0.3488 \times 10^3</td>
<td>0.5467 \times 10^3</td>
<td>0.3490 \times 10^3</td>
<td>0.5433 \times 10^3</td>
</tr>
<tr>
<td>0.3496 \times 10^3</td>
<td>0.5499 \times 10^3</td>
<td>0.3484 \times 10^3</td>
<td>0.5419 \times 10^3</td>
</tr>
<tr>
<td>0.3505 \times 10^3</td>
<td>0.5459 \times 10^3</td>
<td>0.3476 \times 10^3</td>
<td>0.5435 \times 10^3</td>
</tr>
<tr>
<td>0.3508 \times 10^3</td>
<td>0.5453 \times 10^3</td>
<td>0.3466 \times 10^3</td>
<td>0.5444 \times 10^3</td>
</tr>
</tbody>
</table>

TABLE V
THEORETICAL PARAMETERS IN THE 3-D CASE

\[ a(\cdot, 0) = \]

\[ a(\cdot, 1) = \]

\[ a(\cdot, 2) = \]

\[ a(\cdot, 3) = \]

\[ a(\cdot, 4) = \]

and following Fig. 4, the first column containing all “useful” parameters is \( m_0 = \text{Int}(59 \times 7 \times 7/12) + 1 = 241 \).

Table VI shows the estimated parameters extracted from \( U_m; m=6 \) at column \( m_0 = 241 \).

As can be seen, the results show that the estimates are close to theoretical values.

C. 4-D Case

For readability, we show a low-order model for this four-dimensional (4-D) case. The used model was

\[
x(n_1, n_2, n_3, n_4) = \sum_{k_1=0}^{1} \sum_{k_2=0}^{1} \sum_{k_3=0}^{1} \sum_{k_4=0}^{1} a(k_1, k_2, k_3, k_4) \times x(n_1 - k_1, n_2 - k_2, n_3 - k_3, n_4 - k_4) + w(n_1, n_2, n_3, n_4)
\]

where \( y \) is a \( 32 \times 32 \times 32 \times 32 \) complex field driven by a complex Gaussian random field with variance \( \sigma = 1 \). The theoreti-
Let us therefore consider such a system, i.e., a 1-D AR model through matrix factorization. This approach is a possible procedure to decompose a matrix 

\[ \text{C} = \sum_{j=1}^{n} |y(j) - \hat{y}(j)| H(y(j) - \hat{y}(j)) \]

and therefore

\[ C = \sum_{j=1}^{n} |y(j) - \hat{y}(j)| H(y(j) - \hat{y}(j)) \]

\[ -\hat{y}(j) y(j) + \sum_{j=1}^{n} \hat{y}(j) y(j) \]

\[ = \sum_{j=1}^{n} |y(j) - \hat{y}(j)| H(j) x^H(j) \hat{\theta}(n) \]

\[ -\hat{\theta}(n) x(j) y(j) + \sum_{j=1}^{n} \hat{\theta}(n) x(j) x^H(j) \hat{\theta}(n) \]

and thus, using (27)

\[ C = \sum_{j=1}^{n} |y(j) - \hat{y}(j)| H(n) \sum_{j=1}^{n} x(j) x^H(j) \hat{\theta}(n). \]

Returning to our multidimensional model, the regression vector \( x_m \) is \((m+1)\)-pseudoperiodic, and this periodicity may be seen in each dimension. When considering the \( k \)-th dimension, \( 1 \leq k \leq N \), (32) holds, as it holds in all the other dimensions. This means that the elements of \( D_m \) will both decrease periodically with period \((m+1)\) related to each dimension.

APPENDIX B

**Batch Decomposition of \( UDU^H \)**

This is a possible procedure to decompose a matrix \( M \times M \)-dimensionual in the form \( UDU^H \).

1. for \( i = 1, 2, \ldots, M \)
2. for \( j = 1, \ldots, i - 1 \)
3. for \( k1 = 1, \ldots, j-1 \) \( A(i,j) = A(i,j) - A(i,k1) A(j,k1)^* \) skip 3 skip 2
4. for \( j2 = 1, \ldots, i - 1 \) \( s = A(i,j2) / A(j2,j2) \)
   \( A(i,j) = A(i,i) - s A(i,j2)^* \)
   \( A(i,j2) = s \)
   skip 4 skip 1
   \( U = \text{lowertri}(A); \) \( D = \text{diag}(a); \) for \( i = 1, \ldots, N U(i,i) = 1; \) end;

where \( \text{lowertri}(A) \), \( \text{diag}(A) \) and \( ^* \) denote lower triangular part of A, diagonal of A and complex conjugate symbol, respectively.

### APPENDIX A

**Loss Function**

Let us first analyze the model in a given dimension. Let us also denote the signal in the \( k \)-th dimension \( y(n) = y(k)(n) \) and \( \theta(k) = \theta; k = 1, 2, \ldots, N \). This behaves as a one-dimensional system. Let us therefore consider such a system, i.e., a 1-D \( p \)-order AR model in the form of

\[ y(n) = x^H(n) \hat{\theta}(n) + \eta(n) \]

where \( x \) denotes the regression vector. Let \( \hat{\theta}(n) \) be an estimate of \( \theta(n), \hat{y}(n) = x^H(n) \hat{\theta}(n) \) and \( e(n) = y(n) - \hat{y}(n) \) the prediction error. The search for the “true” order of the model consists of looking for the “knee” (see [11]) of the minimum value of the loss function, i.e.,

\[ C = E[\epsilon^2(n)] \]

or its variants (Akaite information criterion, final prediction error, etc.). From (26)

\[ \hat{\theta}(n) = \left[ \sum_{j=1}^{n} x(j)x^H(j) \right]^{-1} \sum_{j=1}^{n} x(j)y(j) \]

TABLE VII

**ESTIMATED PARAMETERS IN THE 3-D CASE, \( m = 6 \)**

\[
\begin{array}{c|cccc}
\hline
\theta(n) & 1.0000 & 0.2005 & -0.7000 & -0.4799 & -0.1399 & -0.0992 & +0.3383 \\
-0.7999 & -1.0001 & -0.3097 & +0.3393 & +0.2903 & +0.6396 & +0.3697 & -0.2693 \\
0.1504 & +0.4499 & -0.2949 & +0.1959 & +0.1348 & -0.1515 & -0.0501 & \\
\hline
\end{array}
\]

\[
\begin{array}{c|cccc}
\hline
\theta(n) & a(1;1) & a(1;2) & a(1;3) & a(1;4) \\
-1.2502 & +0.8003 & 0.3088 & +0.0693 & -0.2680 & -0.4212 \\
1.8798 & -0.7355 & -0.8925 & +1.1685 & -0.7993 & -0.6156 & -0.2472 & +0.6321 \\
-0.1716 & +0.6824 & -0.5124 & -0.0170 & +0.0014 & +0.0218 & -0.3798 & +0.5934 \\
0.4500 & -0.9998 & -0.5917 & -0.5792 & -0.3909 & +0.4090 & -0.3362 & +0.1829 \\
-1.5318 & +0.2061 & -0.1640 & +1.1137 & 0.7635 & +0.1151 & -0.0069 & +0.5153 \\
0.3000 & +0.0934 & -0.3492 & -0.1784 & -0.1221 & +0.2394 & 0.0064 & -0.0653 \\
-0.1129 & +0.4628 & 0.3015 & +0.1716 & -0.1101 & -0.2066 & -0.1560 & -0.0385 \\
0.5996 & -0.2459 & -0.0517 & -0.4690 & -0.3214 & -0.0347 & 0.0983 & +0.2016 \\
-0.1910 & +0.1210 & -0.1233 & +0.1092 & 0.0748 & +0.0843 & 0.0045 & -0.0653 \\
0.0250 & +0.1899 & -0.1025 & -0.0106 & -0.0076 & +0.0704 & 0.0466 & -0.0085 \\
-0.1339 & +0.1394 & 0.0703 & +0.1215 & 0.0830 & +0.0479 & -0.0469 & +0.0440 \\
0.0688 & +0.0099 & 0.0206 & +0.0446 & -0.0303 & -0.0149 & -0.0027 & +0.0224 \\
\hline
\end{array}
\]
is a lower triangular matrix. Let us assume

\[ A(i, j) = \begin{cases} 
-0.0006 & + 0.4889 \\
0.2940 & - 0.1462 \\
0.3494 & - 0.1496 \\
-0.1758 & + 0.0881 \\
0.0631 & + 0.1452 \\
-0.2941 & - 0.1764 \\
0.0693 & - 0.0819 \\
0.2520 & - 0.1260 \\
0.0024 & + 0.4201 \\
0.2520 & - 0.1250 \\
0.0013 & - 0.2983 \\
0.1167 & + 0.0894 \\
0.0685 & - 0.0811 \\
\end{cases} \]

**APPENDIX C**

**UNIQUENESS OF THE DECOMPOSITION**

Define

\[ \Omega_m(n) = \left[ \sum_{i=1}^{n} \phi_m(n) \phi^H_m(n) \right]. \tag{33} \]

Assume

\[ \Omega_m(n) = V_m(n) \Delta_m(n) V^H_m(n). \tag{34} \]

Therefore, from (7) and (8), \( \Omega_m(n) = \sum_{i=1}^{m} A(i, 0) \), and hence the model parameter matrix \( U_m(n) = V_m^{-1}(n) \), where \( V_m^{-1} \) denotes \( [V_m^H]^{-1} \).

The loss function matrix (see Appendix A) is thus \( \Delta(n) = D_m^{-1}(n) \). From (34), we have

\[ \Delta_m(n) = U_m^H(n) \sum_{i=1}^{n} \phi_m(n) \phi^H_m(n) U_m(n). \tag{35} \]

Now assume that there exists another parameter matrix \( W_m(n) \), which has exactly the same unit upper triangular structure as \( U_m(n) \) but which gives smaller loss function matrix, i.e.,

\[ U_m^H(n) \sum_{i=1}^{n} \phi_m(n) \phi^H_m(n) U_m(n) - W_m^H(n) \]

\[ \times \left[ \sum_{i=1}^{n} \phi_m(n) \phi^H_m(n) \right] W_m(n) \geq 0 \tag{36} \]

or

\[ \Delta_m(n) - W_m(n) V_m(n) \Delta_m(n) V^H_m(n) W_m(n) \geq 0. \tag{37} \]

For readability, we ignore the argument \( n \) in the matrices, and the elements of the matrices will be denoted with the relevant lower case letter, i.e., \( X = (x)_{ij} \).

Now define \( K = W_m H V_m \). By definition, \( K \) is a lower triangular matrix with all its diagonal elements \( k_{ii} = 1; \forall i = 1, \ldots, (m + 1)^N \). Thus (37) is

\[ \Delta_m - W_m^H V_m \Delta_m V^H_m W_m \geq 0. \tag{38} \]

By considering the elements of the matrices, (38) is

\[ (\Delta_m - W_m^H V_m \Delta_m V^H_m W_m)_{ij} \]

\[ = \delta_i - \sum_{j=1}^{i} k_{ij} \delta_i k_{ij} = \delta_i - \sum_{j=1}^{i} |k_{ij}|^2 \delta_i \]

\[ = \delta_i - (\sum_{j=1}^{i} |k_{ij}|^2 + |k_{ij}|^2) \]

\[ = \delta_i - (\sum_{j=1}^{i-1} |k_{ij}|^2 + 1) \]

\[ = -\delta_i \sum_{j=1}^{i} |k_{ij}|^2 \geq 0 \tag{39} \]

where \( * \) denotes the complex conjugate symbol.

Since elements of matrices \( \Delta \) are loss functions, then \( \delta_i > 0 \), as shown in Appendix A. Thus the only possibility for (39) is \( k_{ij} = 0 \forall j = 1, \ldots, i - 1, i = 1, \ldots, (m + 1)^N \). This yields \( K = W_m^H V_m = I \), where \( I \) is the identity matrix. Therefore \( W_m(n) = V_m^{-H}(n) = U_m(n) \) and thus provides uniqueness. Similar proof in a different context and in a real case may be found in [14].

**APPENDIX D**

**EXACTNESS AND ASYMPTOTIC PROPERTIES**

The theory presented in Section III-A shows that the parameters obtained, namely, the elements of \( U_m \), are estimates of the model parameters in the least squares sense. Provided the correct order is found, the model parameter can be extracted from the relevant column of \( U_m \), as shown in Section III-A.

Now assume that the data are actually generated by the model (5) via a true parameter embedded in column \( m \) of \( U_m \). Let us extract the parameters from this column and call them \( \bar{\theta} \). From (5) and for ease, we write

\[ y(n) = x_m^H(n) \bar{\theta} + w(n) \tag{40} \]

where \( w(n) \) is a field of zero-mean constant variance-independent random noise. We know from (10) and (13) that the estimated parameter is given by \( \hat{\theta}_{m0} - 1_p - m_0 + 1(n) \). For readability, define \( \bar{\theta}(n) = \hat{\theta}_{m0} - 1_p - m_0 + 1(n) \).
Thus, from (13)

\[
\hat{\theta}(n) = \left[ \sum_{j=1}^{n+m_{0}-p-1} x_m(j)x_m^H(j) \right]^{-1} \sum_{j=1}^{n+m_{0}-p-1} x_m(j)y(j).
\]

Defining \( N = \prod_{k=1}^{N} n_k \) and inserting (40) in (41) gives

\[
\hat{\theta}(n) = \left[ \sum_{j=1}^{n+m_{0}-p-1} x_m(j)x_m^H(j) \right]^{-1} \times \sum_{j=1}^{n+m_{0}-p-1} \{ x_m(j)x_m^H(j)\hat{\theta} + x_m(j)w(j) \}^{1-N} - \hat{\theta} + \left[ \sum_{j=1}^{n+m_{0}-p-1} x_m(j)x_m^H(j) \right]^{-1} \times \frac{1}{N} \sum_{j=1}^{n+m_{0}-p-1} \{ x_m(j)w(j) \}.
\]

(42)

Remembering that \( \lim_{n \to \infty} \) means \( \lim_{m_1 \to \infty} \) and \( \lim_{n_2 \to \infty} \) and..., and \( \lim_{n_N \to \infty} \), \( \lim_{n \to \infty} \) yields \( \lim_{n \to \infty} \). Since \( w(n) \) is a field of zero-mean constant variance-independent random noise, when \( \lim_{n \to \infty} \), the term \( (1/N) \sum_{j=1}^{n+m_{0}-p-1} \{ x_m(j)w(j) \} \) will converge to its expected value, that is, zero, and thus \( \hat{\theta}(n) \) will converge to \( \hat{\theta} \), the true parameter.

ACKNOWLEDGMENT

The author would like to thank the anonymous reviewers for their helpful suggestions.

REFERENCES


Denis Kouamé (M’97) graduated in computer integrated in manufacturing engineering from the École d’Ingénieurs de Tours, Tours, France, in 1992. He received the M.Sc. degree and the Ph.D. degree in signal processing and medical ultrasound imaging from the University of Tours, Tours, France, in 1993 and 1996, respectively.

After having been Associate Professor at the University of Tours, he is currently a Professor at the University of Toulouse 3, France. His research interests include mono and multidimensional signal and image processing, ultrasound and medical imaging techniques with relevant applications, in biomedical signal and image analysis.He is regularly involved in the Technical Program committees of major congress in these fields.

Jean-Marc Girault (S’98–M’00) received the master’s degree in signal processing and biological and medical imaging from the University of Angers, France, in 1996 and the Ph.D. degree in “sciences de l’ingénieur” from the University of Tours, Tours, France.

He is Lecturer at the University of Tours in signal processing in the I.U.T. of Blois. His research area concerns ultrasound signal processing in biomedical applications and particularly in emboli detection.