# NON-ITERATIVE SOLUTION FOR PARAFAC WITH A TOEPLITZ MATRIX FACTOR

*Alain Y. Kibangou and Gérard Favier*

GIPSA-Lab, CNRS
961 rue de la Houille Blanche
BP 46, 38402 Saint Martin d'Hères, France
phone: + (33) 4 76 82 62 57, fax: + (33) 4 76 57 47 90,
email: alain.kibangou@gipsa-lab.inpg.fr

Laboratoire I3S, University of Nice Sophia Antipolis, CNRS
Les Algorithmes- Bât. Euclide B, 2000 Route des lucioles,
B.P. 121, 06903 Sophia Antipolis Cedex, France
phone: + (33) 4 92 94 27 36, fax: + (33) 4 92 94 27 36,
email: favier@i3s.unice.fr

## ABSTRACT

Recently, tensor signal processing has received an increased attention, particularly in the context of wireless communication applications. The so-called PARAllel FACtor (PARAFAC) decomposition is certainly the most used tensor tool. In general, the parameter estimation of a PARAFAC decomposition is carried out by means of the iterative ALS algorithm, which exhibits the following main drawbacks: convergence towards local minima, a high number of iterations for convergence, and difficulty to take, optimally, special matrix structures into account. In this paper, we propose a non-iterative parameter estimation method for a PARAFAC decomposition when one matrix factor has a Toeplitz structure, a situation that is commonly encountered in signal processing applications. We illustrate the proposed method by means of simulation results.

## 1. INTRODUCTION

The use of tensors, or multiway arrays, in Signal Processing (SP) originated in the context of blind source separation by means of cumulant tensors [2]. Since then, the growing interest for tensor tools in the SP community was mostly linked with the development of High-Order Statistics (HOS)-based SP methods [4]. Another motivation for using the tensor formalism is due to the multidimensional nature of the signals as it is the case, for instance, in wireless communication applications [3].

The most used tensor model, PARAFAC [6], allows representing a tensor of order $N$ by means of $N$ matrices, called matrix factors. One of the main properties of PARAFAC concerns the essential uniqueness of its matrix factors, i.e. the matrix factors are unique up to column permutation and scaling. A wide range of PARAFAC-based signal processing applications including blind equalization, blind source separation, and blind channel identification among others can be found in the literature [1, 5, 9, 14].

Most of the methods proposed for fitting the PARAFAC model are iterative and based on the Alternating Least Squares (ALS) algorithm. However, the estimated parameters are often obtained after a high number of iterations. In addition, convergence to the global optimum is not guaranteed. Recently, some works were devoted to the acceleration of the algorithms for fitting the PARAFAC model. They include enhanced line search methods [10, 12] and Levenberg-Marquardt based methods [15].

In certain PARAFAC-based signal processing applications, the matrix factors are in Hankel or Toeplitz form [5, 8]. Taking the *a priori* knowledge of the algebraic structure of the matrix factors into account can help to accelerate the convergence of ALS. Unfortunately that is difficult to achieve in an optimal way.

To overcome the ALS drawbacks concerning convergence speed, local minima, and the difficulty to take special structures into account, a non-iterative or closed-form solution can be used for estimating the PARAFAC factors from noisy data. Recently a structure-independent solution has been proposed [13]. This solution consists in transforming PARAFAC into several matrix joint diagonalization problems. Although this solution is called closed-form by its authors, in fact, it is not a closed-form one because it resorts to matrix joint diagonalization methods which are often iterative. Motivated by the importance of Toeplitz matrices in signal processing applications, this paper presents a non-iterative method for estimating the parameters of a PARAFAC decomposition when at least one factor exhibits a Toeplitz structure.

The rest of this paper is organized as follows. Section 2 briefly recalls the PARAFAC decomposition and then the non-iterative method for estimating the parameters of the PARAFAC decomposition involving Toeplitz constraints is derived in Section 3. The proposed method is illustrated by means of some simulation results in Section 4, before concluding the paper in Section 5.

*Notations*: Vectors are written as boldface lower-case letters ($\mathbf{a}$,$\mathbf{b}$,$\cdots$), matrices as boldface capitals ($\mathbf{A}$,$\mathbf{B}$,$\cdots$) and tensors as blackboard letters ($\mathbb{A}$, $\mathbb{B}$,$\cdots$). $\mathbf{A}^T$, $\mathbf{A}^H$, $\mathbf{A}^*$, and $\mathbf{A}^\dagger$ stand for transpose, transconjugate, conjugate, and pseudo-inverse of the matrix $\mathbf{A}$. $\mathbf{e}_k^{(K)}$ denotes the $k$th unit vector of the Euclidean basis in $\Re^K$. $\mathbf{1}_N$ and $\mathbf{I}_N$ denote respectively the all ones vector of dimension $N$ and the identity matrix of order $N$. $\mathbf{J}_N$ stands for the exchange matrix of order $N$, i.e. the matrix with ones on the antidiagonal and zero elsewhere. The operator $vec(.)$ forms a vector by stacking the columns of its matrix argument, while $unvec_{J\times K}(.)$ is the inverse operator of $vec(.)$ that forms a $J\times K$ matrix from its vector argument of dimensions $JK\times 1$. The operator $diag(.)$ forms a diagonal matrix from its vector argument. We denote by $\mathbf{A}_{.p}$ (resp. $\mathbf{A}_{p.}$) the $p$th column (resp. row) of the matrix $\mathbf{A}$. The outer, Kronecker, and Khatri-Rao products are respectively denoted by $\circ$, $\otimes$, and $\odot$. Recall that the Khatri-Rao product is a column-wise Kronecker product, i.e. for matrices $\mathbf{A}$ and $\mathbf{B}$ with respective dimensions $M\times N$ and $P\times N$ we have: $\mathbf{A}\odot\mathbf{B}=(\mathbf{A}_{.1}\otimes\mathbf{B}_{.1} \quad \cdots \quad \mathbf{A}_{.N}\otimes\mathbf{B}_{.N})$. We have the following properties:

$$vec(\mathbf{ACB}^T)=(\mathbf{B}\otimes\mathbf{A})vec(\mathbf{C}), \tag{1}$$

$$vec(\mathbf{A}diag(\mathbf{c})\mathbf{B}^T)=(\mathbf{B}\odot\mathbf{A})\mathbf{c}. \tag{2}$$

For a vector $\mathbf{c}=(c_0 \quad \cdots \quad c_{L-1})^T$, we denote by $\mathscr{T}_M(\mathbf{c})$ the Toeplitz matrix of dimensions $(L+M-1)\times M$, with the element $(i,j)$ equal to $c_{i-j}$, where by convention $c_{i,j}=0$ if $i-j<0$ or $i-j>L-1$.

## 2. PARAFAC DECOMPOSITION WITH TOEPLITZ MATRIX FACTORS

### 2.1 The PARAFAC decomposition

Let $\mathbb{X}$ be a third-order tensor, also called a three-way array, with entries $x_{i,j,k}$, $i=1,2,\cdots,I$, $j=1,2,\cdots,J$, and $k=1,2,\cdots,K$. It can always be decomposed as

$$\mathbb{X}=\sum_{m=1}^{M}\mathbf{A}_{.m}\circ\mathbf{B}_{.m}\circ\mathbf{C}_{.m}, \tag{3}$$

where $\mathbf{A}\in\mathbb{C}^{I\times M}$, $\mathbf{B}\in\mathbb{C}^{J\times M}$, and $\mathbf{C}\in\mathbb{C}^{K\times M}$, denote the matrix factors. This canonical decomposition is called PARAFAC decom-

position. Its scalar writing is given by:

$$x_{i,j,k} = \sum_{m=1}^{M} a_{i,m} b_{j,m} c_{k,m}. \quad (4)$$

It is convenient to represent a tensor by means of matrices. Slicing a third-order tensor along each one of its three dimensions, or modes, gives rise to the three following slice matrix representations with respective dimensions $J \times K$, $K \times I$, and $I \times J$

$$\mathbf{X}_{i..} = \begin{pmatrix} x_{i,1,1} & \cdots & x_{i,1,K} \\ \vdots & \ddots & \vdots \\ x_{i,J,1} & \cdots & x_{i,J,K} \end{pmatrix}, \mathbf{X}_{.j.} = \begin{pmatrix} x_{1,j,1} & \cdots & x_{I,j,1} \\ \vdots & \ddots & \vdots \\ x_{1,j,K} & \cdots & x_{I,j,K} \end{pmatrix}$$

$$\mathbf{X}_{..k} = \begin{pmatrix} x_{1,1,k} & \cdots & x_{1,J,k} \\ \vdots & \ddots & \vdots \\ x_{I,1,k} & \cdots & x_{I,J,k} \end{pmatrix}.$$

By stacking these matrix slices, we get three unfolded matrices $\mathbf{X}_1 \in \mathbb{C}^{IJ \times K}$, $\mathbf{X}_2 \in \mathbb{C}^{JK \times I}$, and $\mathbf{X}_3 \in \mathbb{C}^{KI \times J}$ that contain all the data of the tensor $\mathbb{X}$:

$$\mathbf{X}_1 = \begin{pmatrix} \mathbf{X}_{1..} \\ \vdots \\ \mathbf{X}_{I..} \end{pmatrix}, \quad \mathbf{X}_2 = \begin{pmatrix} \mathbf{X}_{.1.} \\ \vdots \\ \mathbf{X}_{.J.} \end{pmatrix}, \quad \mathbf{X}_3 = \begin{pmatrix} \mathbf{X}_{..1} \\ \vdots \\ \mathbf{X}_{..K} \end{pmatrix}.$$

It can be shown that these unfolded matrices are given by:

$$\mathbf{X}_1 = (\mathbf{A} \odot \mathbf{B})\,\mathbf{C}^T,\ \mathbf{X}_2 = (\mathbf{B} \odot \mathbf{C})\,\mathbf{A}^T,\ \mathbf{X}_3 = (\mathbf{C} \odot \mathbf{A})\,\mathbf{B}^T. \quad (5)$$

Computing the PARAFAC decomposition of a tensor $\mathbb{X}$ consists in estimating its matrix factors $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$. That is generally carried out by means of the Alternating Least Squares (ALS) algorithm. The basic idea behind ALS is: at each step of the iterative algorithm, one matrix factor is updated using the Least Squares (LS) algorithm conditioned on the previous estimates of the two other matrix factors, and the algorithm is iterated until convergence. Obviously, only convergence towards a local minimum is guaranteed and the algorithm performance strongly depends on the initialization.

### 2.2 Examples of tensors with Toeplitz matrix factors

*2.2.1 Volterra kernels associated with a Wiener-Hammerstein nonlinear system*

Let us consider a Wiener-Hammerstein nonlinear system composed of a memoryless nonlinear system sandwiched between two linear FIR systems with respective impulse responses $l(.)$ and $g(.)$. If the memoryless nonlinear subsystem is approximated by means of a polynomial $C(.)$, then the overall system admits a Volterra series representation whose kernels are given by [7]:

$$h_p(i_1, \cdots, i_p) = c_p \sum_{i=0}^{M_g-1} g(i) \prod_{k=1}^{p} l(i_k - i),\ i_k = 0, \cdots, M-1$$

where $c_p$ is the $p$-th coefficient of the polynomial $C(.)$, $M = M_l + M_g - 1$, $M_l$ and $M_g$ being respectively the memory of $l(.)$ and $g(.)$. In particular, the third-order kernel is given by

$$h_3(i_1, i_2, i_3) = c_3 \sum_{i=0}^{M_g-1} g(i) l(i_1 - i) l(i_2 - i) l(i_3 - i)$$

or equivalently

$$x_{i,j,k} = h_3(i-1, j-1, k-1) = \sum_{m=1}^{M_g} a_{i,m} b_{j,m} c_{k,m}, \quad (6)$$

with $a_{i,m} = c_3 g(m-1) l(i-m)$, $b_{j,m} = l(j-m)$, and $c_{k,m} = l(k-m)$, $i, j, k = 1, \cdots, M$. We recognize in (6) the scalar writing of the PARAFAC decomposition of the $M \times M \times M$ tensor $\mathbb{X}$ with entries $x_{i,j,k}$. The three matrix factors $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ with respective entries $a_{i,m}$, $b_{j,m}$, and $c_{k,m}$ can be written as:

$$\mathbf{A} = \mathscr{T}_{M_g}(\mathbf{l}) diag(\bar{\mathbf{g}}),\ \mathbf{B} = \mathbf{C} = \mathscr{T}_{M_g}(\mathbf{l}),$$

with $\bar{\mathbf{g}} = c_3 \mathbf{g}$, $\mathbf{g} = (g(0) \ \cdots \ g(M_g - 1))^T$, and $\mathbf{l} = (l(0) \ \cdots \ l(M_l - 1))^T$. As a consequence, we can estimate the parameters of the linear subsystems from the PARAFAC decomposition of the third-order Volterra kernel $\mathbb{X}$.

*2.2.2 Output cumulants of a linear FIR system*

Let us consider a causal single-input single-output linear FIR system with impulse response coefficients $h_l$ and memory $L$. When the input signal is assumed to be stationary, ergodic, independent and identically distributed (i.i.d.) with symmetric distribution, zero-mean and non-zero kurtosis $\gamma_{4,u}$, the fourth-order cumulants of the output signal $y(.)$, for lags $\tau_n = -L+1, -L+2, \cdots, L-1$, $n = 1, 2, 3$, are given by:

$$\begin{aligned} C_{4,y}(\tau_1, \tau_2, \tau_3) &\triangleq Cum[y^*(n), y(n+\tau_1), y^*(n+\tau_2), y(n+\tau_3)] \\ &= \gamma_{4,u} \sum_{l=0}^{L-1} h_l^* h_{l+\tau_1} h_{l+\tau_2}^* h_{l+\tau_3}. \end{aligned}$$

By making the coordinate changes $m = L - l$, $i = \tau_1 + L$, $j = \tau_2 + L$, and $k = \tau_3 + L$, we get:

$$\begin{aligned} x_{i,j,k} = C_{4,y}(i-L, j-L, k-L) &= \gamma_{4,u} \sum_{m=1}^{L} h_{L-m}^* h_{i-m} h_{j-m}^* h_{k-m} \\ &= \sum_{m=1}^{L} a_{i,m} b_{j,m} c_{k,m}, \end{aligned} \quad (7)$$

with $a_{i,m} = \gamma_{4,u} h_{L-m}^* h_{i-m}$, $b_{j,m} = h_{j-m}^*$, and $c_{k,m} = h_{k-m}$, $i, j, k = 1, 2, \cdots, 2L-1$. By comparing (7) with (4), we recognize the scalar writing of the PARAFAC decomposition of the tensor $\mathbb{X}$ with entries $x_{i,j,k}$ and matrix factors $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ given by:

$$\mathbf{A} = \gamma_{4,u} \mathscr{T}_L(\mathbf{h}) diag(\mathbf{J}_L \mathbf{h}^*), \quad \mathbf{B} = \mathscr{T}_L(\mathbf{h}^*), \quad \mathbf{C} = \mathscr{T}_L(\mathbf{h}),$$

the generator $\mathbf{h}$ being the system impulse response vector $\mathbf{h} = (h_0 \ \cdots \ h_{L-1})^T$. Thus, from the matrix factors of the PARAFAC decomposition of $\mathbb{X}$ we can deduce $\mathbf{h}$.

## 3. ESTIMATION OF THE MATRIX FACTORS

In what follows we assume that:

- H1: $\mathbf{A} \odot \mathbf{B} \in \mathbb{C}^{IJ \times M}$ is full column rank.
- H2: $\mathbf{C} = \mathscr{T}_M(\mathbf{c}) \in \mathbb{C}^{K \times M}$ with $\mathbf{c} = (c_0 \ \cdots \ c_{L-1})^T$, $L \geq 1$, $c_0 = 1$, and $c_{L-1} \neq 0$, with $K = L + M - 1$. This assumption H2 implies that $\mathbf{C}$ is full column rank.

Let us consider the singular value decomposition (SVD) of the unfolded matrix $\mathbf{X}_1$ given by (5):

$$\mathbf{X}_1 = (\mathbf{U}_1 \quad \mathbf{U}_2) \begin{pmatrix} \mathbf{\Sigma} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{V}_1^H \\ \mathbf{V}_2^H \end{pmatrix} = \mathbf{U}_1 \mathbf{\Sigma} \mathbf{V}_1^H \quad (8)$$

where $\mathbf{U}_1$ and $\mathbf{V}_1$, with respective dimensions $IJ \times M$ and $K \times M$, contain the left and right singular vectors associated with the $M$ nonzero singular values that form the diagonal matrix $\mathbf{\Sigma}$. From $\mathbf{X}_1 = (\mathbf{A} \odot \mathbf{B})\,\mathbf{C}^T$, we can conclude that $\mathbf{C}$ and $\mathbf{V}_1^*$ span the same column space. As a consequence, we can deduce the existence of a nonsingular matrix $\mathbf{D} \in \mathbb{C}^{M \times M}$ such that:

$$\mathbf{V}_1^* \mathbf{D} = \mathbf{C} \quad (9)$$

$$\mathbf{U}_1 \mathbf{\Sigma} \mathbf{D}^{-T} = \mathbf{A} \odot \mathbf{B} \quad (10)$$

The idea of the proposed algorithm is to determine the nonsingular matrix $\mathbf{D}$ by exploiting the Toeplitz structure of $\mathbf{C}$, and then to deduce $\mathbf{C}$ whereas $\mathbf{A}$ and $\mathbf{B}$ are obtained from the Khatri-Rao product $\mathbf{A} \odot \mathbf{B}$.

### 3.1 Determination of the Toeplitz factor $\mathbf{C}$

We take the Toeplitz structure of $\mathbf{C}$ into account for determining the ambiguity matrix $\mathbf{D}$. Such a calculation is carried out by exploiting the repetition of the same elements in each column of $\mathbf{C}$ that contains $(M-1)$ zeros and one 1, and $\bar{\mathbf{c}} = (c_1, \cdots, c_{L-1})^T$ the vector of the unknown parameters of the Toeplitz matrix factor generator $\mathbf{c}$.

For selecting the zeros and 1 in each column of $\mathbf{C}$, we define the $M$ following row selection matrices, with dimensions $M \times K$:

$$\mathbf{S}_M = \begin{pmatrix} \mathbf{e}_1^{(K)} & \cdots & \mathbf{e}_M^{(K)} \end{pmatrix}^T = \begin{pmatrix} \mathbf{I}_M & \mathbf{0}_{M\times(K-M)} \end{pmatrix}, \quad (11)$$

and for $n = 1, 2, \cdots, M-1$:

$$\mathbf{S}_{M-n} = \begin{pmatrix} \mathbf{e}_{K-n+1}^{(K)} & \mathbf{e}_{K-n+2}^{(K)} & \cdots & \mathbf{e}_K^{(K)} & \mathbf{e}_1^{(K)} & \cdots & \mathbf{e}_{M-n}^{(K)} \end{pmatrix}^T. \quad (12)$$

These row selection matrices can be recursively calculated as follows:

$$\mathbf{S}_{M-n} = \mathbf{S}_{M-n+1}\mathbf{Q}_K^T, \quad n = 1, 2, \cdots, M-1. \quad (13)$$

where $\mathbf{Q}_K$ denotes the following $K \times K$ cyclic permutation matrix

$$\mathbf{Q}_K = \begin{pmatrix} \mathbf{e}_K^{(K)} & \mathbf{e}_1^{(K)} & \cdots & \mathbf{e}_{K-1}^{(K)} \end{pmatrix}.$$

We easily check that $\mathbf{S}_{M-n}\mathbf{C}_{.M-n} = \mathbf{e}_M^{(M)}$, $n = M-1, M-2, \cdots, 0$. Then, by concatenating these equations, we get:

$$\begin{pmatrix} \mathbf{S}_1\mathbf{C}_{.1} \\ \vdots \\ \mathbf{S}_M\mathbf{C}_{.M} \end{pmatrix} = \begin{pmatrix} \mathbf{e}_M^{(M)} \\ \vdots \\ \mathbf{e}_M^{(M)} \end{pmatrix} = \mathbf{1}_M \otimes \mathbf{e}_M^{(M)}, \quad (14)$$

which can be rewritten, using (9), as

$$\begin{pmatrix} \mathbf{S}_1\mathbf{V}_1^*\mathbf{D}_{.1} \\ \vdots \\ \mathbf{S}_M\mathbf{V}_1^*\mathbf{D}_{.M} \end{pmatrix} = \mathbf{1}_M \otimes \mathbf{e}_M^{(M)}$$

or equivalently

$$diag(\mathbf{S}_1\mathbf{V}_1^*, \cdots, \mathbf{S}_M\mathbf{V}_1^*)vec(\mathbf{D}) = \mathbf{1}_M \otimes \mathbf{e}_M^{(M)}. \quad (15)$$

Now, for extracting the vector $\bar{\mathbf{c}}$ from each column of $\mathbf{C}$, we make use of the $(L-1) \times K$ row selection matrices $\bar{\mathbf{S}}_{M-n}$, $n = 0, \cdots, M-1$, defined by the following recursive formula:

$$\begin{aligned} \bar{\mathbf{S}}_M &= \begin{pmatrix} \mathbf{0}_{(L-1)\times M} & \mathbf{I}_{L-1} \end{pmatrix}, \\ \bar{\mathbf{S}}_{M-n} &= \bar{\mathbf{S}}_{M-n+1}\mathbf{Q}_K^T, \quad n = 1, 2, \cdots, M-1. \end{aligned} \quad (16)$$

We have:

$$\bar{\mathbf{S}}_{M-n}\mathbf{C}_{.M-n} = \bar{\mathbf{c}}, \quad n = M-1, M-2, \cdots, 0. \quad (17)$$

By using (9), equation (17) can be rewritten as

$$\bar{\mathbf{S}}_{M-n}\mathbf{V}_1^*\mathbf{D}_{.M-n} = \bar{\mathbf{c}}, \quad n = M-1, M-2, \cdots, 0. \quad (18)$$

We get:

$$\bar{\mathbf{S}}_1\mathbf{V}_1^*\mathbf{D}_{.1} - \bar{\mathbf{S}}_{M-n}\mathbf{V}_1^*\mathbf{D}_{.M-n} = \mathbf{0}_{(L-1)\times 1}, \quad n = M-2, M-3, \cdots, 0.$$

Concatenating these equations yields

$$\mathbf{Z}vec(\mathbf{D}) = \mathbf{0}_{(M-1)(L-1)\times 1}, \quad (19)$$

with $\mathbf{Z}$ a $(L-1)(M-1) \times M^2$ matrix defined by

$$\mathbf{Z} = \begin{pmatrix} \bar{\mathbf{S}}_1\mathbf{V}_1^* & -\bar{\mathbf{S}}_2\mathbf{V}_1^* & \mathbf{0}_{(L-1)\times M} & \cdots & \mathbf{0}_{(L-1)\times M} \\ \bar{\mathbf{S}}_1\mathbf{V}_1^* & \mathbf{0}_{(L-1)\times M} & -\bar{\mathbf{S}}_3\mathbf{V}_1^* & \cdots & \mathbf{0}_{(L-1)\times M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \bar{\mathbf{S}}_1\mathbf{V}_1^* & \mathbf{0}_{(L-1)\times M} & \mathbf{0}_{(L-1)\times M} & \cdots & -\bar{\mathbf{S}}_M\mathbf{V}_1^* \end{pmatrix} \quad (20)$$

From equations (15) and (19), we get:

$$\mathbf{\Phi}vec(\mathbf{D}) = \boldsymbol{\psi}, \quad (21)$$

with

$$\mathbf{\Phi} = \begin{pmatrix} diag(\mathbf{S}_1\mathbf{V}_1^*, \cdots, \mathbf{S}_M\mathbf{V}_1^*) \\ \mathbf{Z} \end{pmatrix} \in \mathbb{C}^{\left(M^2+(L-1)(M-1)\right)\times M^2},$$

$$\boldsymbol{\psi} = \begin{pmatrix} \mathbf{1}_M \otimes \mathbf{e}_M^{(M)} \\ \mathbf{0}_{(M-1)(L-1)\times 1} \end{pmatrix} \in \mathbb{C}^{\left(M^2+(L-1)(M-1)\right)\times 1}.$$

Since $\mathbf{\Phi}$ is a full column rank matrix[1], then $\mathbf{D}$ can be computed as:

$$\mathbf{D} = unvec_{M\times M}\left(\mathbf{\Phi}^\dagger\boldsymbol{\psi}\right) \quad (22)$$

We can then deduce $\mathbf{C}$ using (9). However, due to the presence of noise in the data that constitute the tensor $\mathbb{X}$ and consequently its unfolded matrix representation $\mathbf{X}_1$, we can get a matrix $\mathbf{C}$ that slightly differs from the Toeplitz structure. So, instead of directly using (9), we derive a least squares (LS) solution for the generator $\mathbf{c}$ and then we reconstruct the Toeplitz matrix as $\mathscr{T}_M(\mathbf{c})$.

$\mathbf{C}$ being a Toeplitz matrix and defining $\mathbf{G} = \mathbf{D}^{-1}$, we have the following equalities: $\mathscr{T}_M(\mathbf{c})\mathbf{G}_{.m} = \mathscr{T}_L(\mathbf{G}_{.m})\mathbf{c}$, $m = 1, \cdots, M$, which gives

$$\begin{aligned} \mathbf{V}_1^* = \mathbf{C}\mathbf{G} &= \begin{pmatrix} \mathscr{T}_M(\mathbf{c})\mathbf{G}_{.1} & \cdots & \mathscr{T}_M(\mathbf{c})\mathbf{G}_{.M} \end{pmatrix} \\ &= \begin{pmatrix} \mathscr{T}_L(\mathbf{G}_{.1})\mathbf{c} & \cdots & \mathscr{T}_L(\mathbf{G}_{.M})\mathbf{c} \end{pmatrix}. \end{aligned}$$

We deduce that $vec(\mathbf{V}_1^*) = \mathbf{\Gamma}\mathbf{c}$, with

$$\mathbf{\Gamma} = \begin{pmatrix} \mathscr{T}_L(\mathbf{G}_{.1})^T & \cdots & \mathscr{T}_L(\mathbf{G}_{.M})^T \end{pmatrix}^T. \quad (23)$$

The LS solution for computing $\mathbf{c}$ is given by

$$\hat{\mathbf{c}} = \tilde{\mathbf{c}}/\tilde{c}_1, \quad \tilde{\mathbf{c}} = \mathbf{\Gamma}^\dagger vec(\mathbf{V}_1^*), \quad (24)$$

$\tilde{c}_1$ being the first element of $\tilde{\mathbf{c}}$. Then, we deduce

$$\hat{\mathbf{C}} = \mathscr{T}_M(\hat{\mathbf{c}}). \quad (25)$$

In summary, the computation of the matrix factor in Toeplitz form can be done by applying the following TOMFAC (TOeplitz Matrix FActor Computation) algorithm.

**TOMFAC algorithm**:
Given the unfolded matrix $\mathbf{X}_1$ of the tensor $\mathbb{X}$, the number $M$ of PARAFAC factors and the length $L$ of the Toeplitz matrix factor generator.

1. Compute the SVD of $\mathbf{X}_1$: $\mathbf{X}_1 = \mathbf{U}_1\mathbf{\Sigma}\mathbf{V}_1^H$.
2. Generate the row selection matrices $\mathbf{S}_{M-n}$ and $\bar{\mathbf{S}}_{M-n}$, $n = 0, 1, \cdots, M-1$.
3. Compute the matrix $\mathbf{G}$ as $\mathbf{G} = \left(unvec_{M\times M}\left(\mathbf{\Phi}^\dagger\boldsymbol{\psi}\right)\right)^{-1}$.
4. Form the matrix $\mathbf{\Gamma}$ as $\mathbf{\Gamma} = \left(\mathscr{T}_L(\mathbf{G}_{.1})^T \quad \cdots \quad \mathscr{T}_L(\mathbf{G}_{.M})^T\right)^T$.
5. Compute the Toeplitz matrix factor $\hat{\mathbf{C}}$ as $\hat{\mathbf{C}} = \mathscr{T}_M(\tilde{\mathbf{c}}/\tilde{c}_1)$, $\tilde{\mathbf{c}} = \mathbf{\Gamma}^\dagger vec(\mathbf{V}_1^*)$.

[1] The proof has been omitted due to a lack of space

### 3.2 Determination of the matrix factors A and B

Now, we consider the determination of the two other matrix factors $\mathbf{A}$ and $\mathbf{B}$ of the PARAFAC decomposition. Using the definition $\mathbf{G} = \mathbf{D}^{-1}$, equation (10) can be rewritten as:

$$\mathbf{U}_1 \mathbf{\Sigma} \mathbf{G}^T = \mathbf{A} \odot \mathbf{B} = \mathbf{W}.$$

The matrix $\mathbf{W}$ resulting from the Khatri-Rao product of $\mathbf{A}$ and $\mathbf{B}$, we define the rank-one matrices $\mathbf{F}^{(m)}$, $m = 1, \cdots, M$, as

$$\mathbf{F}^{(m)} = unvec_{J \times I}(\mathbf{W}_{.m}) = unvec_{J \times I}(\mathbf{A}_{.m} \otimes \mathbf{B}_{.m}) = \mathbf{B}_{.m} \mathbf{A}_{.m}^T. \quad (26)$$

Hence, the vectors $\mathbf{A}_{.m}^*$ and $\mathbf{B}_{.m}$ are obtained up to a scaling, as the right and left singular vectors associated with the unique nonzero singular value $\sigma_m$ of $\mathbf{F}^{(m)}$. So, let $\mathbf{v}_m$ and $\mathbf{u}_m$ be respectively the right and left singular vectors associated with $\sigma_m$. The two PARAFAC factors are then estimated as:

$$\hat{\mathbf{A}} = (\mathbf{v}_1^* \quad \cdots \quad \mathbf{v}_M^*), \quad \hat{\mathbf{B}} = (\sigma_1 \mathbf{u}_1 \quad \cdots \quad \sigma_M \mathbf{u}_M). \quad (27)$$

### 3.3 Summary

In summary, the computation of the PARAFAC factors with one factor in Toeplitz form, can be done using the following non-iterative method, called T-PARAFAC-1 algorithm.

**T-PARAFAC-1 algorithm**:
Given the unfolded matrix $\mathbf{X}_1$ of the tensor $\mathbb{X}$, the number $M$ of PARAFAC factors, and the length $L$ of the Toeplitz matrix factor generator.

1. Determine $\mathbf{U}_1$, $\mathbf{\Sigma}$, $\mathbf{G}$ and $\hat{\mathbf{C}}$ using the TOMFAC algorithm.
2. Compute $\mathbf{W} = \mathbf{U}_1 \mathbf{\Sigma} \mathbf{G}^T$.
3. For $m = 1, \cdots, M$, compute the right and left singular vectors $\mathbf{v}_m$ and $\mathbf{u}_m$ associated with the greatest singular value $\sigma_m$ of $unvec_{J \times I}(\mathbf{W}_{.m})$.
4. Compute $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ as follows $\hat{\mathbf{A}} = (\mathbf{v}_1^* \quad \cdots \quad \mathbf{v}_M^*)$, $\hat{\mathbf{B}} = (\sigma_1 \mathbf{u}_1 \quad \cdots \quad \sigma_M \mathbf{u}_M)$.

The T-PARAFAC-1 algorithm requires the computation of the SVD of one $IJ \times K$ matrix and $M$ $J \times I$ matrices, the pseudo-inverse of two matrices with respective dimensions $MK \times L$ and $\left((M-1)(L-1) + M^2\right) \times M^2$, and the inverse of one $M \times M$ matrix, while the standard ALS algorithm needs to compute the pseudo-inverse of $IJ \times M$, $JK \times M$, and $IK \times M$ matrices at each iteration, the number of iterations for convergence being generally very high.

## 4. SIMULATION RESULTS

In this section, we present some simulation results to illustrate the performance of the proposed closed form solution for estimating the parameters of a PARAFAC decomposition with Toeplitz constraints.

We first compare ALS and T-PARAFAC-1 in terms of the Normalized Square Estimation Error (NSEE) calculated in using the reconstructed tensor, as $NSEE = 10 \log \|\mathbb{X}(\mathbf{A}, \mathbf{B}, \mathbf{C}) - \mathbb{X}(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}})\|_F^2 / \|\mathbb{X}(\mathbf{A}, \mathbf{B}, \mathbf{C})\|_F^2$. We considered a noiseless double degeneracy case, i.e. one of the matrix factor exhibits almost proportional columns and another one has more columns than rows. In this case, it is well known that convergence of ALS is very difficult as depicted in Fig. 1, showing that ALS needs hundreds of thousands of iterations for achieving a performance close to that obtained with the T-PARAFAC-1 algorithm. These simulations results illustrate the superiority of the non-iterative T-PARAFAC-1 algorithm on ALS.

Now, we consider the blind channel identification using fourth-order output cumulants, the input signal being a BPSK one. We call TOMFAC-BCI the blind channel identification method based on the TOMFAC algorithm. We compare our method with the ALS-type algorithm recently proposed in the literature [5], the so-called Single Step Least Squares PARAFAC based Blind Channel Identification algorithm (SS-LS PBCI). The performance is evaluated according to the Normalized Mean Square Error (NMSE): $NMSE = 10 \log_{10} \frac{\|\mathbf{h} - \hat{\mathbf{h}}\|^2}{\|\mathbf{h}\|^2}$. The NMSEs depicted in the figures were obtained by averaging the results over 100 independent Monte Carlo runs.

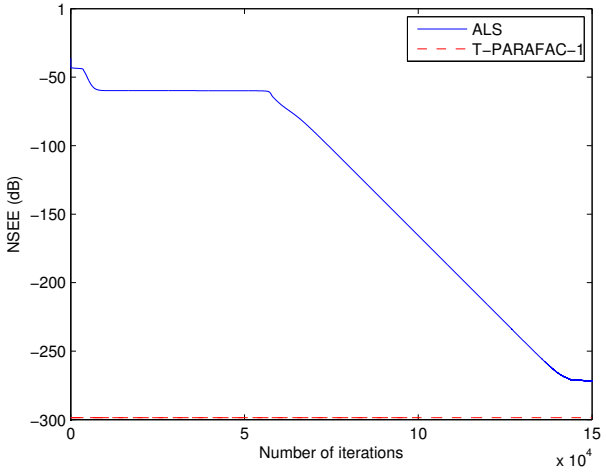


Figure 1: Comparison of T-PARAFAC-1 with ALS in a noiseless double degeneracy case

Like all ALS-type algorithms, the convergence of SS-LS PBCI depends on its initialization. The standard initialization scheme consists in generating a random initial channel parameter vector. Such a scheme can be improved by considering several initializations and then select the best one, in the sense of a minimum square error between the reconstructed tensor and the actual one. Obviously, the second scheme is much more time consuming than the first one (the computational cost is multiplied by the number of used initial vectors). Note that the SS-LS PBCI algorithm is stopped when $\frac{\|\hat{\mathbf{h}}_r - \hat{\mathbf{h}}_{r-1}\|}{\|\hat{\mathbf{h}}_r\|} \leq 10^{-5}$, where $\hat{\mathbf{h}}_r$ denotes the channel estimate at the $r$th iteration.

We consider the case where 100 channels with memory $L$ are randomly generated from a Gaussian distribution. Table 1 gives the mean value of the NMSE, in the noiseless case, for four different channel lengths. As expected, the TOMFAC-BCI algorithm largely outperforms the iterative method.

Table 1: NMSE (dB) mean values in the noiseless case

| L | SS-LS PBCI | | TOMFAC-BCI |
|---|---|---|---|
| | 1 initialization | 10 initializations | |
| 3 | -53.04 | -104.86 | -299.25 |
| 4 | -52.21 | -98.27 | -292.00 |
| 5 | -31.74 | -77.74 | -279.81 |
| 6 | -23.50 | -76.11 | -274.68 |

In Figure 2, we compare the mean CPU time for estimating the channel parameters with the TOMFAC-BCI and SS-LS PBCI algorithms, for different channel memories. This comparison was carried out using a PC with a 3.20 GHz Pentium IV processor and 0.99 GByte of RAM. We note that SS-LS PBCI requires much more computation time than TOMFAC-BCI. For example, for $L = 3$ (resp. $L = 10$), SS-LS PBCI requires 0.021 (resp. 0.608) seconds and 0.059 (resp. 1.849) seconds respectively for a single and ten initializations. In mean, each iteration needs 0.001 (resp. 0.028)

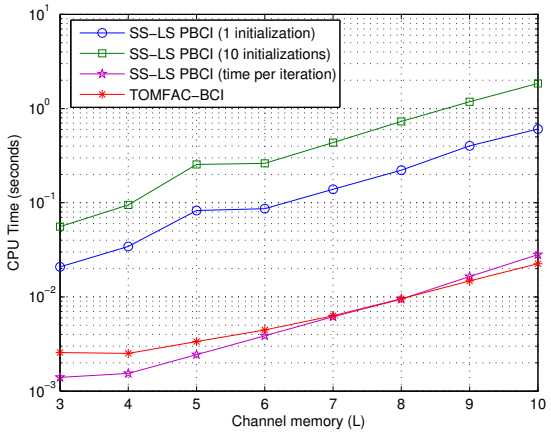


Figure 2: Mean value of the CPU time

seconds. For TOMFAC-BCI, we need 0.0026 and 0.0226 seconds for $L = 3$ and $L = 10$ respectively. By using TOMFAC-BCI instead of SS-LS PBCI, with a single initialization, the computation time, for $L = 3, \cdots, 10$, is reduced from 8 to 27 times.

We now consider the following channel proposed in [11] $\mathbf{h} = (1,\ -0.7+0.2j,\ -0.1+0.2j,\ 0.9)^T$. A complex white Gaussian noise was added to the channel output. For SS-LS PBCI, ten random initializations were considered and then the best one was selected. For different data number and values of the output SNR, Fig. 3 depicts the mean value of the NMSE for SS-LS PBCI and TOMFAC.

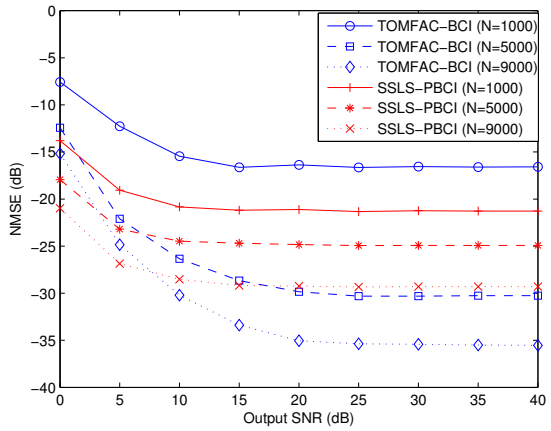


Figure 3: Mean value of the NMSE for different values of the output SNR

We note that for SNR values higher than 20 dB, the NMSE is relatively constant for both algorithms, which means that the effect of the additive noise in the cumulant estimation becomes negligible. By increasing the data number, the cumulants are better estimated. Therefore, the performance of TOMFAC-BCI is significantly improved.

## 5. CONCLUSION

In this paper, we have presented a new non-iterative method for computing the PARAFAC decomposition of a third-order tensor when at least one matrix factor has a Toeplitz structure. This method can be applied when the factor in Toeplitz form and the matrix resulting from the Khatri-Rao product of the two other matrix factors are full column rank. It can also be applied to PARAFAC decompositions with Hankel factors. The efficiency of the proposed closed-form solution has been illustrated in the context of blind channel identification, by means of simulation results. In a future work, robustness to noise, extension to higher-order tensors, and application to block-structured nonlinear system identification will be considered.

## REFERENCES


[1] T. Acar, Y. Yu, and A. Petropulu. Blind MIMO system identification based on PARAFAC decomposition of higher order output tensors. *IEEE Trans. on Signal Processing*, 54(11):4156–4168, Nov. 2006.

[2] J.-F. Cardoso. Eigen-structure of the fourth-order cumulant tensor with application to the blind source separation problem. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2655–2658, Albuquerque, New Mexico, USA, 3-6 April 1990.

[3] A.-L.-F. de Almeida, G. Favier, and J.-C.-M. Mota. PARAFAC-based unified tensor modeling for wireless communication systems with application to blind multiuser equalization. *Signal Processing*, 87(2):337–351, February 2007.

[4] L. De Lathauwer. *Signal processing based on multilinear algebra*. PhD thesis, Katholieke Universiteit Leuven, Sept. 1997.

[5] C.-E.-R. Fernandes, G. Favier, and J.-C.-M. Mota. Blind channel identification algorithms based on the PARAFAC decomposition of cumulant tensors: the single and multiuser cases. *Signal Processing*, 88(6):1382–1401, June 2008.

[6] R. Harshman. Foundation of the PARAFAC procedure: models and conditions for an "explanatory" multimodal factor analysis. *UCLA working papers in phonetics*, 16:1–84, 1970.

[7] A. Kibangou and G. Favier. Wiener-Hammerstein systems modeling using diagonal Volterra kernels coefficients. *IEEE Signal Proc. Letters*, 13(6):381–384, June 2006.

[8] A. Kibangou and G. Favier. Matrix and tensor decompositions for identification of block-structured nonlinear channels in digital transmission systems. In *Proc. IEEE Signal Proc. Advances in Wireless Communications (SPAWC)*, Recife, Brazil, July 2008.

[9] A. Kibangou and G. Favier. Blind equalization of nonlinear channels using tensor decompositions with code/space/time diversities. *Signal Processing*, 89(2):133–143, February 2009.

[10] D. Nion and L. De Lathauwer. An enhanced line search scheme for complex-valued tensor decompositions. Application in DS-CDMA. *Signal Processing*, 88(3):749–755, March 2008.

[11] B. Porat and B. Friedlander. FIR system identification using fourth-order cumulants with application to channel equalization. *IEEE Trans. on Signal Processing*, 38(9):1394–1398, Sept. 1993.

[12] M. Rajih and P. Comon. Enhanced line search: A novel method to accelerate PARAFAC. In *Proc. EUSIPCO*, Antalya, Turkey, Sept. 4-8, 2005.

[13] F. Roemer and M. Haardt. A closed-form solution for parallel factor (PARAFAC) analysis. In *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2365–2368, Las Vegas, Nevada, USA, March 30-April 4 2008.

[14] N. Sidiropoulos, G. Giannakis, and R. Bro. Blind PARAFAC receivers for DS-CDMA systems. *IEEE Trans. on Signal Processing*, 48(3):810–823, March 2000.

[15] G. Tomasi and R. Bro. A comparison of algorithms for fitting the PARAFAC model. *Comp. Stat. Data Anal.*, 50(7):1700–1734, 2006.