algorithms

Review

Tensor-Based Approaches for Nonlinear and Multilinear Systems Modeling and Identification

Gérard Favier [1,*,†] and Alain Kibangou [2,3,†]

1 I3S Laboratory, Côte d'Azur University, CNRS, 06900 Sophia Antipolis, France
2 Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP, GIPSA-Lab, 38000 Grenoble, France; alain.kibangou@univ-grenoble-alpes.fr
3 Faculty of Science (Auckland Park Campus), University of Johannesburg, Johannesburg 2006, South Africa
* Correspondence: favier@i3s.unice.fr
† These authors contributed equally to this work.

Abstract: Nonlinear (NL) and multilinear (ML) systems play a fundamental role in engineering and science. Over the last two decades, active research has been carried out on exploiting the intrinsically multilinear structure of input–output signals and/or models in order to develop more efficient identification algorithms. This has been achieved using the notion of tensors, which are the central objects in multilinear algebra, giving rise to tensor-based approaches. The aim of this paper is to review such approaches for modeling and identifying NL and ML systems using input–output data, with a reminder of the tensor operations and decompositions needed to render the presentation as self-contained as possible. In the case of NL systems, two families of models are considered: the Volterra models and block-oriented ones. Volterra models, frequently used in numerous fields of application, have the drawback to be characterized by a huge number of coefficients contained in the so-called Volterra kernels, making their identification difficult. In order to reduce this parametric complexity, we show how Volterra systems can be represented by expanding high-order kernels using the parallel factor (PARAFAC) decomposition or generalized orthogonal basis (GOB) functions, which leads to the so-called Volterra–PARAFAC, and Volterra–GOB models, respectively. The extended Kalman filter (EKF) is presented to estimate the parameters of a Volterra–PARAFAC model. Another approach to reduce the parametric complexity consists in using block-oriented models such as those of Wiener, Hammerstein and Wiener–Hammerstein. With the purpose of estimating the parameters of such models, we show how the Volterra kernels associated with these models can be written under the form of structured tensor decompositions. In the last part of the paper, the notion of tensor systems is introduced using the Einstein product of tensors. Discrete-time memoryless tensor-input tensor-output (TITO) systems are defined by means of a relation between an Nth-order tensor of input signals and a Pth-order tensor of output signals via a $(P+N)$th-order transfer tensor. Such systems generalize the standard memoryless multi-input multi-output (MIMO) system to the case where input and output data define tensors of order higher than two. The case of a TISO system is then considered assuming the system transfer is a rank-one Nth-order tensor viewed as a global multilinear impulse response (IR) whose parameters are estimated using the weighted least-squares (WLS) method. A closed-form solution is proposed for estimating each individual IR associated with each mode-n subsystem.

Keywords: block-oriented nonlinear systems; multilinear systems; parameter estimation; tensor decompositions; tensor systems; Volterra systems; Wiener–Hammerstein systems

check for updates

Citation: Favier, G.; Kibangou, A. Tensor-Based Approaches for Nonlinear and Multilinear Systems Modeling and Identification. *Algorithms* **2023**, *16*, 443. https://doi.org/10.3390/a16090443

Academic Editors: Alicia Cordero and Juan Ramón Torregrosa Sánchez

Received: 30 July 2023
Revised: 29 August 2023
Accepted: 9 September 2023
Published: 14 September 2023

1. Introduction

The continuous development of mathematical knowledge, together with a constantly renewed and growing need to study, represent and analyze ever more complex physical phenomena and systems, are at the origin of new mathematical objects and models. In particular, the notion of matrices introduced by Gauss in 1810 to solve systems of linear algebraic equations, with the foundations of matrix computation developed during the 19th century by Sylvester (1814–1897) and Cayley (1821–1895), among several other mathematicians, has later given rise to the notion of tensors. Tensors of order higher than two, i.e., mathematical objects indexed by more than two indices, are multidimensional generalizations of vectors and matrices which are tensors of orders one and two, respectively. Such objects are well suited to represent and process multidimensional and multimodal signals and data, like in computer vision [1], pattern recognition [2], array processing [3], machine learning [4], recommender systems [5], ECG applications [6], bioinformatics [7], and wireless communications [8], among many other fields of application. Today, with ever constantly growing big data (texts, images, audio, and videos) to manage in multimedia applications and social networks, tensor tools are well adapted to fuse, classify, analyze and process digital information [9].

The purpose of this paper is to present an overview of tensor-based methods for modeling and identifying nonlinear and multilinear systems using input–output data, as encountered in signal processing applications, with a focus on truncated Volterra models and block-oriented nonlinear ones and an introduction to memoryless input–output tensor systems. With a detailed reminder of the tensor tools useful to make the presentation as self-contained as possible and a review of main nonlinear models and their applications, this paper should be of interest to researchers and engineers concerned with signal processing applications.

First of all, developed as computational and representation tools in physics and geometry, tensors were the subject of mathematical developments related to polyadic decomposition [10], aiming to generalize dyadic decompositions, i.e., matrix decompositions such as the singular value decomposition (SVD), discovered independently by Beltrami (1835–1900) and Jordan (1838–1922) in 1873 and 1874, respectively. Then, tensors were used for the analysis of three-dimensional data generalizing matrix analysis to sets of matrices, seen as arrays of data characterized by three indices, in the fields of psychometrics and chemometrics [11–14]. This explains the other name given to tensors as multiway arrays in the context of data analysis and data mining [15].

Matrix decompositions, such as the SVD, have thus been generalized into tensor decompositions, such as the PARAFAC decomposition [13], also called canonical polyadic decomposition (CPD), and the Tucker decomposition (TD) [12]. Tensor decompositions consist in representing a high-order tensor by means of factor matrices and lower-order tensors, called core tensors. In the context of data analysis, such decompositions make it possible to highlight hidden structures of the data while preserving their multilinear structure, which is not the case when stacking the data in the form of vectors or matrices. Tensor decompositions can be used to reduce data dimensionality [16], merge coupled data tensors [17], handle missing data through the application of tensor completion methods [18,19], and design semi-blind receivers for tensor-based wireless communication systems [8].

In Table 1, we present basic and very useful matrix and third-order tensor decompositions, namely the reduced SVD, also known as the compact SVD, PARAFAC/CPD and TD, in a comparative way. A detailed presentation of PARAFAC and Tucker decompositions is given in Section 4.2. Note that the matrix factors $\mathbf{U}$ and $\mathbf{V}$ which are column-orthonormal, contain the left and right singular vectors, respectively, whereas the diagonal matrix $\mathbf{\Sigma}$ contains the nonzero singular values, and R denotes the rank of the matrix.

A historical review of the theory of matrices and tensors, with basic decompositions and applications, can be found in [20].

Similarly, from the system modeling point of view, linear models of dynamic systems in the form of input–output relationships or state space equations have given rise to nonlinear and multilinear models to take into account nonlinearities inherent in physical systems. This explains why nonlinear models are appropriate in many engineering applications. Consequently, standard parameter estimation and filtering methods for linear systems, such as the least-squares (LS) algorithm and the Kalman filter (KF), first proposed by Legendre in 1805 [21] and Kalman in 1960 [22], respectively, were extended for parameter and state estimation of nonlinear systems. Thus, the alternating least-squares (ALS) algorithm [13] and the extended Kalman filter (EKF) [23] were developed, respectively, for estimating the parameters of a PARAFAC decomposition and applying the KF to nonlinear systems.

Table 1. Reduced SVD, PARAFAC/CPD, and TD.

Matrices	⋮	**Third-Order Tensors**
$\mathbf{X} \in \mathbb{R}^{I\times J}$	⋮	$\mathcal{X} \in \mathbb{R}^{I\times J\times K}$
Reduced SVD	⋮	PARAFAC/CPD
$x_{i,j} = \sum_{r=1}^{R} \sigma_r u_{i,r} v_{j,r} \iff \mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$	⋮	$x_{i,j,k} = \sum_{r=1}^{R} a_{i,r} b_{j,r} c_{k,r}$
$\mathbf{U} \in \mathbb{R}^{I\times R}, \mathbf{V} \in \mathbb{R}^{J\times R}, \boldsymbol{\Sigma} \in \mathbb{R}^{R\times R}$	⋮	$\mathbf{A} \in \mathbb{R}^{I\times R}, \mathbf{B} \in \mathbb{R}^{J\times R}, \mathbf{C} \in \mathbb{R}^{K\times R}$
	⋮	TD
	⋮	$x_{i,j,k} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{s=1}^{S} g_{p,q,s} a_{i,p} b_{j,q} c_{k,s}$
	⋮	$\mathbf{A} \in \mathbb{R}^{I\times P}, \mathbf{B} \in \mathbb{R}^{J\times Q}, \mathbf{C} \in \mathbb{R}^{K\times S}, \mathcal{G} \in \mathbb{R}^{P\times Q\times S}$

In Table 2, we present two examples of standard linear models, namely the single-input single-output (SISO) finite impulse response (FIR) model and the memoryless multi-input multi-output (MIMO) model, often used for modeling a communication channel between n_T transmit antennas and n_R receive antennas, where $h_{i,j}$ is the fading coefficient between the jth transmit antenna and the ith receiver antenna. The FIR model is one of the most used for modeling linear time-invariant (LTI) systems, i.e., systems which satisfy the constraints of linearity and time-invariance, which means that the system output $y(t)$ can be obtained from the input via a convolution $y(t) = (h \star u)(t)$, where $h(.)$ is the system's impulse response (IR), and $\star$ denotes the convolution operator.

The notion of linear dynamical system has been generalized to multilinear dynamical systems in [24] to model tensor time series data, i.e., time series in which input and output data are tensors. In this paper, the multilinear operator is chosen in the form of a Kronecker product of matrices, and the parameters are estimated by means of an expectation-maximization algorithm, with application to various real datasets. Then, the notion of LTI system has been extended to multilinear LTI (MLTI) systems by [25] using the Einstein product of even-order paired tensors, with an extension of the classical stability, reachability, and observability criteria to the case of MLTI systems. In Table 2, four examples of nonlinear (NL) and multilinear (ML) models are introduced, namely the polynomial, truncated Volterra, tensor-input tensor-output (TITO), and multilinear tensor-input single-output (TISO) models, which will be studied in more detail in Sections 5 and 6, as mentioned in Table 2.

System modeling and identification is a fundamental problem in engineering applications. Real-life systems being often nonlinear in nature, NL models are very useful for various application areas. Parameter estimation using measurements of input and output (I/O) signals is at the heart of identification methods. In this paper, two main families of NL models are considered: (i) discrete-time Volterra models, also called truncated Volterra series expansions; (ii) block-oriented (Wiener, Hammerstein, Wiener–Hammerstein) models. In the sequel, we assume that the systems to be modeled are time invariant, i.e., their properties and consequently the parameters of their model do not depend on time.

Table 2. Some examples of linear, nonlinear and multilinear models.

Linear models
SISO FIR model $y(t) = \sum_{i}^{n_u} h_i u(t-i)$
Memoryless MIMO model $y_i(t) = \sum_{j=1}^{n_T} h_{i,j} u_j(t),\ \ i \in [1, n_R]$ $\mathbf{y}(t) = \mathbf{H}\mathbf{u}(t),\ \ \mathbf{y}(t) \in \mathbb{R}^{n_R},\ \ \mathbf{u}(t) \in \mathbb{R}^{n_T}, \mathbf{H} \in \mathbb{R}^{n_R \times n_T}$
Nonlinear models
Polynomial model (Section 5.1) $y(t) = \sum_{p=1}^{P} f_p[u(t), \cdots, u(t-n_u), y(t-1), \cdots, y(t-n_y)]$ $f_p(.) =$ pth-degree polynomial in the system input (u) and output (y) signals
Truncated Volterra model (Section 5.2) $y(t) = h_0 + \sum_{p=1}^{P} \sum_{m_1=1}^{M_p} \cdots \sum_{m_P=1}^{M_P} h^{(p)}_{m_1,\cdots,m_P} \prod_{q=1}^{p} u(t-m_q+1)$ $h^{(p)}_{m_1,\cdots,m_P} =$ pth-order Volterra kernel with memory M_p
Multilinear models
TITO model (Section 6.2) $y_{i_1,\cdots,i_P}(t) = \sum_{j_1=1}^{J_1} \cdots \sum_{j_N=1}^{J_N} h_{i_1,\cdots,i_P,j_1,\cdots,j_N} u_{j_1,\cdots,j_N}(t)$ $\mathcal{U}(t) \in \mathbb{R}^{J_1 \times \cdots \times J_N}, \mathcal{Y}(t) \in \mathbb{R}^{I_1 \times \cdots \times I_P}$
Multilinear TISO model (Sections 6.3 and 6.4) $y(t) = \sum_{j_1=1}^{J_1} \cdots \sum_{j_N=1}^{J_N} \left(\prod_{n=1}^{N} h^{(n)}_{j_n} \right) u_{j_1,\cdots,j_N}(t)$

Volterra models are frequently used due to the fact that they allow approximating any fading memory nonlinear systems with an arbitrary precision, as shown in [26]. They represent a direct nonlinear extension of the very popular FIR linear model, with guaranteed stability in the bounded-input bounded-output (BIBO) sense, and they have the advantage to be linear in their parameters, the kernel coefficients [27]. The nonlinearity of a Pth-order truncated Volterra model is due to products of up to P samples of delayed inputs. Moreover, they are interpretable in terms of multidimensional convolutions which makes the derivation of their z-transform and Fourier transform representations easy[28].

Among the numerous application areas of Volterra models, we can mention chemical and biochemical processes [29], radio-over-fiber (RoF) wireless communication systems (due to optical/electrical (O/E) conversion) [30,31], high-power amplifiers (HPA) in satellite communications [32,33], physiological systems [34], vibrating structures and more generally mechatronic systems like robots [35], and acoustic echo cancellation [36].

The main drawback of Volterra models is their parametric complexity implying the need to estimate a huge number of parameters which exponentially grows with the order and memory of the kernels. So, several complexity reduction approaches for Volterra models have been developed using symmetrization or triangularization of Volterra kernels, or their expansion on orthogonal bases like Laguerre and Kautz ones, or generalized orthogonal bases (GOB). Considering Volterra kernels as tensors, they can also be decomposed using a PARAFAC decomposition or a tensor train (TT). These approaches lead to the so-called Volterra–Laguerre, Volterra–GOB–Tucker, Volterra–PARAFAC and Volterra–TT models [37–42]. In Sections 5.3 and 5.4, we review the Volterra–PARAFAC and Volterra–GOB–Tucker models. Note that a model-pruning approach can also be employed to adjust the complexity reduction in considering only nearly diagonal coefficients of the kernels and removing the other ones which correspond to more delayed input values whose influence decreases when the delay increases [43].

Another approach for ensuring a reduced parametric complexity consists in considering block-oriented NL models, composed of two types of blocks: linear time-invariant

(LTI) dynamic blocks and static NL blocks. The linear blocks may be parametric (transfer functions, FIR models, state-space representations) or nonparametric (impulse responses), whereas the NL blocks may be with memory or memoryless. The different blocks are concatenated in series leading to the so-called Hammerstein (NL-LTI) and Wiener (LTI-NL) models, extended to the Wiener–Hammerstein (LTI-NL-LTI) and Hammerstein–Wiener (NL-LTI-NL) models, abbreviated W-H and H-W, respectively. To extend the modeling potential of block-oriented models, several W-H and H-W models can also be interconnected in parallel. Although such models are simpler but less general than Volterra models, they allow us to represent numerous nonlinear systems. One of the first applications of block-oriented NL models was for modeling biological systems [44]. A lot of papers have been devoted to the identification of block-oriented models and their applications. For more details, the reader is referred to the book [45] and the survey papers [46,47].

In Section 5.5, we show that the Wiener, Hammerstein and W-H models are equivalent to structured Volterra models. This equivalence is at the origin of the structure identification method for block-oriented systems, which will be presented in Section 5.5.4. Tensor-based methods using this equivalence have been developed to estimate the parameters of block-oriented nonlinear systems [48–51]. These methods are generally composed of two steps. In the first one, the Volterra kernel associated with a particular block-oriented system is used to estimate the LTI component(s). Note that there exist closed-form solutions for estimating only the Volterra kernel of interest. Such a solution is proposed in [52,53] for a third-order and fifth-order kernel, respectively. Then, in a second step, the nonlinear block is estimated using the LS method. An example of a tensor-based method for identifying a nonlinear communication channel represented by means of a W-H model was proposed in [54] using the associated third-order Volterra kernel.

On the other hand, multilinear models are useful for modeling coupled dynamical systems in engineering, biology, and physics. Tensor-based approaches have been proposed for solving and identifying multilinear systems [24,55,56]. Using the Einstein product of tensors, we first introduce a new class of systems, the so-called memoryless tensor-input tensor-output (TITO) systems, in which the multidimensional input and output signals define two tensors. The LS method is applied to estimate the tensor transfer of such a system. Then the case of a tensor-input single-output (TISO) system is considered assuming the system transfer is a rank-one Nth-order tensor, which leads to a multilinear system with respect to the impulse responses (IR) of the N subsystems associated with the N modes of the input tensor.

The non-recursive weighted least-squares (WLS) method is used to estimate the multilinear impulse response (MIR) under a vectorized form. A closed-form method is also proposed to estimate the IR of each subsystem from the estimated MIR.

The rest of the paper is structured as follows. In Section 2, we present the notations with the index convention used throughout the paper. In Section 3, we introduce some tensor sets in connection with multilinear forms. In Section 4, we briefly recall basic tensor operations and decompositions. Sections 5 and 6 are devoted to tensor-based approaches for nonlinear and multilinear systems modeling and identification, respectively. Finally, Section 7 concludes the paper, with some perspectives for future work.

Many books and survey papers discuss estimation theory and system identification. In the field of engineering sciences, we can cite the fundamental contributions of [57–63] for linear systems and [27–29,47,64–69] for nonlinear systems. In the case of multilinear systems, the reader is referred to [55,56] for more details.

2. Notation and Index Convention

Scalars, column vectors, matrices, and tensors are denoted by lower-case, boldface lower-case, boldface upper-case, and calligraphic letters, e.g., x, $\mathbf{x}$, $\mathbf{X}$, $\mathcal{X}$, respectively. We denote by $a_{i,r}$ the (i,r) element and by $\mathbf{A}_{.r}$ (resp. $\mathbf{A}_{i.}$) the rth column (resp. ith row) of $\mathbf{A} \in \mathbb{C}^{I\times R}$. $\mathbf{I}_R$ denotes the identity matrix of size $R \times R$.

The transpose, complex conjugate, transconjugate, and Moore–Penrose pseudo-inverse operators are represented by $(.)^T$, $(\cdot)^*$, $(\cdot)^H$ and $(\cdot)^\dagger$, respectively.

The operator $\mathrm{diag}(\cdot)$ forms a diagonal matrix from its vector argument, while $D_i(\mathbf{A})$ stands for a diagonal matrix holding the ith row of $\mathbf{A} \in \mathbb{C}^{I\times R}$ on the diagonal.

The operator $\mathcal{T}_{M+N-1,N}(\cdot)$ forms a $(M+N-1)\times N$ Toeplitz matrix from its vector argument $\mathbf{x} \in \mathbb{C}^M$, whose first column and row are, respectively, $\begin{pmatrix} x_1 & \cdots & x_M & \mathbf{0}_{N-1}^T \end{pmatrix}^T$ and $\begin{pmatrix} x_1 & \mathbf{0}_{N-1}^T \end{pmatrix}$.

Given $\mathbf{Y} \in \mathbb{C}^{I\times J}$, the vec and unvec operators are defined such that: $\mathbf{y} = \mathrm{vec}(\mathbf{Y}) \in \mathbb{C}^{JI} \leftrightarrow \mathbf{Y} = \mathrm{unvec}(\mathbf{y}) \in \mathbb{C}^{I\times J}$, where the order of dimensions in the product JI is linked to the order of variation of the indices, with the column index j varying more slowly than the row index i.

The outer, Kronecker and Khatri–Rao products are denoted by $\circ$, $\otimes$ and $\diamond$, respectively.

Table 3 summarizes the notation used for sets of indices and dimensions [70].

Table 3. Notation for sets of indices and dimensions.

$\underline{\mathbf{i}}_P \triangleq \{i_1, \cdots, i_P\}$; $\underline{\mathbf{j}}_N \triangleq \{j_1, \cdots, j_N\}$
$\underline{\mathbf{I}}_P \triangleq \{I_1, \cdots, I_P\}$; $\underline{\mathbf{J}}_N \triangleq \{J_1, \cdots, J_N\}$
$\underline{I}_P \triangleq I_1 \times \cdots \times I_P$; $\underline{J}_N \triangleq J_1 \times \cdots \times J_N$
$\underline{I}_P \times \underline{J}_N = I_1 \times \cdots \times I_P \times J_1 \times \cdots \times J_N$
$\underline{I}_P \times \underline{I}_P = I_1 \times \cdots \times I_P \times I_1 \times \cdots \times I_P$
$\Pi I_P \triangleq I_1 \cdots I_P = \prod_{p=1}^{P} I_p$

We now introduce the index convention which allows eliminating the summation symbols in formulae involving multi-index variables. For example, $\sum_{i=1}^{I} a_i b_i$ is simply written as $a_i b_i$. Note there are two differences relative to Einstein's summation convention:

- Each index can be repeated more than twice in an expression;
- Ordered index sets are allowed.

The index convention can be interpreted in terms of two types of summation, the first associated with the row indices (superscripts) and the second associated with the column indices (subscripts), with the following rules [70]:

- The order of the column indices is independent of the order of the row indices;
- Consecutive row and column indices (or index sets) can be permuted.

In Table 4, we give some examples of vector and matrix products using index convention, where $\mathbf{e}_{ij} \triangleq \mathbf{e}_i^{(I)} \otimes \mathbf{e}_j^{(J)}, \mathbf{e}_i^j \triangleq \mathbf{e}_i^{(I)} \otimes (\mathbf{e}_j^{(J)})^T, \mathbf{e}_{ik}^j \triangleq \mathbf{e}_i^{(I)} \otimes \mathbf{e}_k^{(K)} \otimes (\mathbf{e}_j^{(J)})^T$.

Table 4. Vector and matrix products using the index convention.

$\mathbf{u} \in \mathbb{K}^I, \mathbf{v} \in \mathbb{K}^J, \mathbf{w} \in \mathbb{K}^K$
$\mathbf{u} \otimes \mathbf{v} = u_i v_j \mathbf{e}_{ij} \in \mathbb{K}^{IJ}$
$\mathbf{u} \otimes \mathbf{v}^T = u_i v_j \mathbf{e}_i^j \in \mathbb{K}^{I\times J}$
$\mathbf{u} \otimes \mathbf{v}^T \otimes \mathbf{w} = u_i v_j w_k \mathbf{e}_{ik}^j \in \mathbb{K}^{IK\times J}$
$\mathbf{A} \in \mathbb{K}^{I\times J}, \mathbf{B} \in \mathbb{K}^{J\times K}, \mathbf{C} \in \mathbb{K}^{K\times J}$
$\mathbf{AB} = \sum_{i=1}^{I} \sum_{k=1}^{K} (\sum_{j=1}^{J} a_{ij} b_{jk}) \mathbf{e}_i^k = a_{ij} b_{jk} \mathbf{e}_i^k \in \mathbb{K}^{I\times K}$
$\mathbf{AC}^T = a_{ij} c_{kj} \mathbf{e}_i^k \in \mathbb{K}^{I\times K}$

Using the index convention, the multiple sum over the indices of $x_{i_1,\cdots,i_P}y_{i_1,\cdots,i_P}$ will be abbreviated to

$$\sum_{i_1=1}^{I_1}\cdots\sum_{i_P=1}^{I_P} x_{i_1,\cdots,i_P}y_{i_1,\cdots,i_P} = \sum_{\underline{\mathbf{i}}_P=\underline{\mathbf{1}}}^{\underline{\mathbf{I}}_P} x_{\underline{\mathbf{i}}_P}y_{\underline{\mathbf{i}}_P} = x_{\underline{\mathbf{i}}_P}y_{\underline{\mathbf{i}}_P}, \tag{1}$$

where $\underline{\mathbf{1}}$ denotes a set of ones whose number is fixed by the index P of the set $\underline{\mathbf{I}}_P$. The notation $\underline{\mathbf{i}}_P$ and $\underline{\mathbf{I}}_P$ allows us to simplify the expression of the multiple sum into a single sum over an index set, which is further simplified by using the index convention.

3. Tensors and Multilinear Forms

In signal processing applications, a tensor $\mathcal{X} \in \mathbb{K}^{I_1\times\cdots\times I_N}$ of order N and size $I_1 \times \cdots \times I_N$ is typically viewed as an array of numbers $[x_{i_1,\cdots,i_N}]$. The order corresponds to the number of indices $(i_1,\cdots,i_N)$ that characterize its elements $x_{i_1,\cdots,i_N} \in \mathbb{K}$, also denoted $x_{i_1\cdots i_N}$ or $(\mathcal{X})_{i_1,\cdots,i_N}$. Each index $i_n \in \langle I_n\rangle \triangleq \{1,\cdots,I_n\}$, for $n \in \langle N\rangle \triangleq \{1,\cdots,N\}$, is associated with a mode, also called a way, and I_n denotes the dimension of the nth mode. The number of elements in $\mathcal{X}$ is equal to $\prod_{n=1}^{N} I_n$. For instance, in a wireless communication system [8], each index of a signal $x_{i_1,\cdots,i_N}$ corresponds to a different form of diversity (in time, space, frequency, code, etc., domains), and the dimensions I_n are the numbers of time samples, receive antennas, subcarriers, the code length, etc.

The tensor $\mathcal{X}$ is said to be real (resp. complex) if its elements are real numbers (resp. complex numbers), which corresponds to $\mathbb{K} = \mathbb{R}$ (resp. $\mathbb{K} = \mathbb{C}$). It is said to have even order (resp. odd order) if N is even (resp. odd). The special cases $N = 2$ and $N = 1$ correspond to the sets of matrices $\mathbf{X} \in \mathbb{K}^{I\times J}$ and column vectors $\mathbf{x} \in \mathbb{K}^{I}$, respectively.

If $I_1 = \cdots = I_N = I$, the Nth-order tensor $\mathcal{X} = [x_{i_1,\cdots,i_N}] \in \mathbb{K}^{I\times I\times\cdots\times I}$ is said to be hypercubic, of dimensions I, with $i_n \in \langle I\rangle$, for $n \in \langle N\rangle$. The number of elements in $\mathcal{X}$ is then equal to I^N. The set of (real or complex) hypercubic tensors of order N and dimensions I will be denoted $\mathbb{K}^{[N;I]}$.

A hypercubic tensor of order N and dimensions I is said to be symmetric if it is invariant under any permutation π of its modes, i.e.,

$$a_{\pi(i_1,i_2,\cdots,i_N)} \triangleq a_{i_{\pi(1)},i_{\pi(2)},\cdots,i_{\pi(N)}} = a_{i_1,i_2,\cdots,i_N}. \tag{2}$$

The identity tensor of order N and dimensions I is denoted $\mathcal{I}_{N,I} = [\delta_{i_1,\cdots,i_N}]$, with $i_n \in \langle I\rangle$, for $n \in \langle N\rangle$, or simply $\mathcal{I}$. It is a hypercubic tensor whose elements are defined using the generalized Kronecker delta

$$\delta_{i_1,\cdots,i_N} = \begin{cases} 1 & \text{if} \quad i_1 = \cdots = i_N \\ 0 & \text{otherwise} \end{cases}.$$

It is a diagonal tensor whose diagonal elements are equal to 1 and other elements to zero, which can be written as the sum of I outer products of N canonical basis vectors $\mathbf{e}_i^{(I)}$ of the space $\mathbb{R}^I$

$$\mathcal{I}_{N,I} = \sum_{i=1}^{I} \underbrace{\mathbf{e}_i^{(I)} \circ \cdots \circ \mathbf{e}_i^{(I)}}_{N \text{ terms}}.$$

where the outer product operation is defined later in Table 9.

A diagonal tensor $\mathcal{X} \in \mathbb{K}^{I\times\cdots\times I}$ of order N, whose diagonal elements are the entries of vector $\mathbf{a} = [a_1,\cdots,a_I]^T$, will be written as

$$x_{i,i_2,\cdots,i_N} = a_i\,\delta_{i,i_2,\cdots,i_N} \Leftrightarrow \mathcal{X} = \sum_{i=1}^{I} a_i \underbrace{\mathbf{e}_i^{(I)} \circ \cdots \circ \mathbf{e}_i^{(I)}}_{N \text{ terms}}.$$

Different matricizations, also called matrix unfoldings, can be defined for a tensor $\mathcal{X} \in \mathbb{K}^{I_1\times\cdots\times I_N}$. Consider a partitioning of the set of modes $\langle N\rangle$ into two disjoint ordered subsets $\mathbb{S}_1$ and $\mathbb{S}_2$, composed of p and $N-p$ modes, respectively, with $p \in \langle N-1\rangle$. A general matrix unfolding formula was given by [71] as follows

$$\mathbf{X}_{\mathbb{S}_1;\mathbb{S}_2} = \sum_{i_1=1}^{I_1}\cdots\sum_{i_N=1}^{I_N} x_{i_1,\cdots,i_N}\left(\underset{n\in\mathbb{S}_1}{\otimes}\mathbf{e}_{i_n}^{(I_n)}\right)\left(\underset{n\in\mathbb{S}_2}{\otimes}\mathbf{e}_{i_n}^{(I_n)}\right)^T \in \mathbb{K}^{J_1\times J_2}, \tag{3}$$

where $\mathbf{e}_{i_n}^{(I_n)}$ is the i_n-th vector of the canonical basis of $\mathbb{R}^{I_n}$, and $J_{n_1} = \prod_{n\in\mathbb{S}_{n_1}} I_n$, for $n_1 = 1$ and 2. We say that $\mathbf{X}_{\mathbb{S}_1;\mathbb{S}_2}$ is a matrix unfolding of $\mathcal{X}$ along the modes of $\mathbb{S}_1$ for the rows and along the modes of $\mathbb{S}_2$ for the columns, with $\mathbb{S}_1\cap\mathbb{S}_2=\emptyset$ and $\mathbb{S}_1\cup\mathbb{S}_2=\langle N\rangle$.

For instance, in the case of a third-order tensor $\mathcal{X}\in\mathbb{K}^{I\times J\times K}$, we have six flat unfoldings and six tall unfoldings. For $\mathbb{S}_1 = 1$ and $\mathbb{S}_2=\{2,3\}$, we have the following mode-1 flat unfolding $\mathbf{X}_{I\times JK} \triangleq \mathbf{X}_{1;\{2,3\}}$, while for $\mathbb{S}_1=\{2,3\}$ and $\mathbb{S}_2 = 1$ we obtain the following mode-1 tall unfolding $\mathbf{X}_{JK\times I} \triangleq \mathbf{X}_{\{2,3\};1} = \mathbf{X}_{I\times JK}^T$.

Vectorized forms of $\mathcal{X}\in\mathbb{K}^{I_1\times\cdots\times I_N}$ are obtained by combining the modes in a given order. Thus, a lexicographical vectorization gives the vector $\mathbf{y} \triangleq \mathbf{x}_{I_1\cdots I_N}$ with element $x_{i_1,\cdots,i_N}$ at the position $m = \overline{i_1 i_2\cdots i_N}$ in $\mathbf{y}$, i.e., $y_m = x_{i_1,\cdots,i_N} \triangleq x_{\underline{\mathbf{i}}_N}$, with [72]

$$\overline{i_1 i_2\cdots i_N} \triangleq i_N + \sum_{n=1}^{N-1}(i_n-1)\prod_{k=n+1}^{N} I_k. \tag{4}$$

By convention, the order of the dimensions in a product $\prod_{n=1}^N I_n \triangleq I_1\cdots I_N$ associated with the index combination $\overline{i_1 i_2\cdots i_N}$ follows the order of variation of the indices $(i_1,\cdots,i_N)$, with i_1 varying more slowly than i_2, which in turn varies more slowly than i_3, etc.

The Frobenius norm of $\mathcal{X}\in\mathbb{K}^{I_1\times\cdots\times I_N}$ is the square root of the inner product of the tensor with itself, i.e.,

$$\|\mathcal{X}\|_F = \sqrt{\langle\mathcal{X},\mathcal{X}\rangle} = \left(\sum_{i_1=1}^{I_1}\cdots\sum_{i_N=1}^{I_N}|x_{i_1,\cdots,i_N}|^2\right)^{1/2}. \tag{5}$$

Table 5 presents various sets of tensors that will be considered in this paper, with the notation introduced in [70].

Table 5. Various sets of tensors.

Order	Size	Sets of tensors
P	$\underline{I}_P = I_1\times\cdots\times I_P$	$\mathbb{K}^{I_1\times\cdots\times I_P} \triangleq \mathbb{K}^{\underline{I}_P}$
P	$\underline{I}_P = I_1\times\cdots\times I_P$ with $I_p = I, \forall p\in\langle P\rangle$	$\mathbb{K}^{[P;I]}$
$P+N$	$\underline{I}_P\times\underline{J}_N = I_1\times\cdots\times I_P\times J_1\times\cdots\times J_N$	$\mathbb{K}^{\underline{I}_P\times\underline{J}_N}$
$P+N$	$\underline{I}_P\times\underline{J}_N = I\times\cdots\times I\times J\times\cdots\times J$ with $I_p = I, \forall p\in\langle P\rangle$ and $J_n = J, \forall n\in\langle N\rangle$	$\mathbb{K}^{[P+N;I,J]}$
$2P$	$\underline{I}_P\times\underline{I}_P$ with $I_p = I, \forall p\in\langle P\rangle$	$\mathbb{K}^{[2P;I]}$

We can make the following remarks about the sets of tensors defined in Table 5:

- For $P=N=1$, the set $\mathbb{K}^{[2;I,J]}$ is the set $\mathbb{K}^{I\times J}$ of (real or complex) matrices of size $I\times J$.
- The set $\mathbb{K}^{[P;I]}$ is also denoted $\mathbb{K}^{I^P}$ or $\mathrm{T}^P(K^I)$ by some authors.

- The set $\mathbb{K}^{\underline{I}_P \times \underline{I}_P}$ is called the set of even-order (or square) tensors of order $2P$ and size $\underline{I}_P \times \underline{I}_P$. The name of square tensor comes from the fact that the index set is divided into two identical subsets of dimension $\underline{I}_P$.
- Analogously to matrices, tensors in the sets $\mathbb{K}^{\underline{I}_P \times \underline{J}_P}$ with $J_p \neq I_p$ and $\mathbb{K}^{\underline{I}_P \times \underline{J}_N}$ are said to be rectangular. The set $\mathbb{K}^{\underline{I}_P \times \underline{J}_N}$ is called the set of rectangular tensors with index blocks of dimensions $\underline{I}_P$ and $\underline{J}_N$.

The various tensor sets introduced above can be associated with scalar real-valued multilinear forms in vector variables and with homogeneous polynomials. Like in the matrix case, we will distinguish between homogeneous polynomials of degree P that depend on the components of P vector variables and those that depend on just one vector variable.

A real-valued multilinear form, also called a P-linear form, is a map f such as

$$\underset{p=1}{\overset{P}{\times}} \mathbb{R}^{I_p} \ni (\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(P)}) \longmapsto f\big(\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(P)}\big) \in \mathbb{R} \tag{6}$$

that is separately linear with respect to each vector variable $\mathbf{x}^{(p)}$ when the other variables $\mathbf{x}^{(q)}$, for $q \neq p$, are fixed. Using the index convention, the multilinear form can be written for $\mathbf{x}^{(p)} \in \mathbb{R}^{I_p}$, $p \in \langle P \rangle$, as

$$f\big(\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(P)}\big) = \sum_{i_1=1}^{I_1} \cdots \sum_{i_P=1}^{I_P} a_{i_1,\cdots,i_P} x_{i_1}^{(1)} \cdots x_{i_P}^{(P)} = a_{\underline{\mathbf{i}}_P} \prod_{p=1}^{P} x_{i_p}^{(p)}. \tag{7}$$

The tensor $\mathcal{A} \in \mathbb{R}^{\underline{I}_P}$ is called the tensor associated with the multilinear form f.

Two multilinear forms are presented in Table 6, which also states the transformation corresponding to each of them, as well as the associated tensor.

Table 6. Multilinear forms and associated tensors.

Multilin. Forms	Transformations	Tensors
real-valued in P vectors	$\underset{p=1}{\overset{P}{\times}} \mathbb{R}^{I_p} \ni (\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(P)}) \longmapsto f\big(\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(P)}\big) \in \mathbb{R}$	$\mathcal{A} \in \mathbb{R}^{\underline{I}_P}$
real-valued in one vector	$\mathbb{R}^I \ni \mathbf{x} \longmapsto f(\underbrace{\mathbf{x}, \cdots, \mathbf{x}}_{P \text{ terms}}) \in \mathbb{R}$	$\mathcal{A} \in \mathbb{R}^{[P;I]}$

Table 7 recalls the definitions of bilinear/quadratic forms using the index convention, then presents the multilinear forms defined in Table 6, as well as the associated tensors from Table 5 and the corresponding homogeneous polynomials.

Table 7. Multilinear forms and associated homogeneous polynomials.

Forms	Matrices/Tensors	Homogeneous Polynomials
Bilinear	$\mathbf{A} \in \mathbb{R}^{I \times J}; \mathbf{y} \in \mathbb{R}^I, \mathbf{x} \in \mathbb{R}^J$	$f(\mathbf{x}, \mathbf{y}) = \mathbf{y}^T \mathbf{A} \mathbf{x} = a_{ij} y_i x_j \ , \ i \in \langle I \rangle \, , \, j \in \langle J \rangle$
Quadratic	$\mathbf{A} \in \mathbb{R}^{I \times I}; \mathbf{x} \in \mathbb{R}^I$	$f(\mathbf{x}) = \mathbf{x}^T \mathbf{A} \mathbf{x} = a_{ij} x_i x_j \ , \ i, j \in \langle I \rangle$
Real multilinear in P vector	$\mathcal{A} \in \mathbb{R}^{\underline{I}_P}; \mathbf{x}^{(p)} \in \mathbb{R}^{I_p}$	$f\big(\mathbf{x}^{(1)}, \cdots \mathbf{x}^{(P)}\big) = a_{\underline{\mathbf{i}}_P} \prod_{p=1}^{P} x_{i_p}^{(p)} \ , \ i_p \in \langle I_p \rangle \ , \ p \in \langle P \rangle$
Real multilinear in one vector	$\mathcal{A} \in \mathbb{R}^{[P;I]}; \mathbf{x} \in \mathbb{R}^I$	$f(\underbrace{\mathbf{x}, \cdots, \mathbf{x}}_{P \text{ terms}}) = a_{\underline{\mathbf{i}}_P} \prod_{p=1}^{P} x_{i_p} \ , \ i_p \in \langle I \rangle \ , \ p \in \langle P \rangle$

We can make the following remarks:

- In the same way that bilinear forms depend on two variables that do not necessarily belong to the same vector space, general real multilinear forms depend on P variables that may belong to different vector spaces: $\mathbf{x}^{(p)} \in \mathbb{R}^{I_p}$.

- Analogously to quadratic forms obtained from bilinear forms by replacing the pair $(\mathbf{x}, \mathbf{y})$ with the vector $\mathbf{x}$, real multilinear forms can be expressed using just one vector $\mathbf{x} \in \mathbb{K}^I$. In the same way symmetric quadratic forms lead to the notion of symmetric matrices, the symmetry of multilinear forms is directly linked to the symmetry of their associated tensors.

4. Tensor Operations and Decompositions

In Section 4.1, we introduce different multiplications with tensors. Then, in Section 4.2, we present the two most used tensor decompositions, namely the PARAFAC (parallel factors) and Tucker decompositions [12,13].

For a more in-depth presentation of tensor tools, the reader is referred to the recent book [70] and review papers [73,74].

4.1. Multiplications with Tensors

In Table 8, we present three types of multiplication with tensors, using the notation of Table 3 and the index convention: mode-p, mode-(p,n), and Einstein products.

Table 8. Different types of multiplication with tensors.

Tensors	Operations	Definitions
$\mathcal{X} \in \mathbb{K}^{\underline{I}_P}, \mathbf{A} \in \mathbb{K}^{J \times I_p}$	$\mathcal{Y} = \mathcal{X} \times_p \mathbf{A}$	$y_{i_1,\cdots,i_{p-1},j,i_{p+1},\cdots,i_P} = \sum_{i_p} a_{j,i_p} x_{\underline{\mathbf{i}}_P} = a_{j,i_p} x_{\underline{\mathbf{i}}_P}$
$\mathcal{X} \in \mathbb{K}^{\underline{I}_P}, \mathbf{u} \in \mathbb{K}^{I_p}$	$\mathcal{Y} = \mathcal{X} \times_p \mathbf{u}^T$	$y_{i_1,\cdots,i_{p-1},i_p,\cdots,i_P} = \sum_{i_p} u_{i_p} x_{\underline{\mathbf{i}}_P} = u_{i_p} x_{\underline{\mathbf{i}}_P}$
$\mathcal{X} \in \mathbb{K}^{\underline{I}_P}, \mathcal{Y} \in \mathbb{K}^{\underline{J}_N}$ with $I_p = J_n = K$	$\mathcal{Z} = \mathcal{X} \times_p^n \mathcal{Y}$	$z_{i_1,\cdots,i_{p-1},i_{p+1},\cdots,i_P,j_1,\cdots,j_{n-1},j_{n+1},\cdots,j_N} = \sum_{k=1}^K a_{i_1,\cdots,i_{p-1},k,i_{p+1},\cdots,i_P} b_{j_1,\cdots,j_{n-1},k,j_{n+1},\cdots,j_N}$
$\mathcal{A} \in \mathbb{K}^{\underline{I}_P \times \underline{J}_N}, \mathcal{X} \in \mathbb{K}^{\underline{J}_N \times \underline{K}_Q}$	$\mathcal{Y} = \mathcal{A} \star_N \mathcal{X}$	$y_{\underline{\mathbf{i}}_P,\underline{\mathbf{k}}_Q} = \sum_{\underline{\mathbf{j}}_N=\underline{\mathbf{1}}}^{\underline{\mathbf{J}}_N} a_{\underline{\mathbf{i}}_P,\underline{\mathbf{j}}_N} x_{\underline{\mathbf{j}}_N,\underline{\mathbf{k}}_Q} = a_{\underline{\mathbf{i}}_P,\underline{\mathbf{j}}_N} x_{\underline{\mathbf{j}}_N,\underline{\mathbf{k}}_Q}$

The multiplication $\times_p$, called mode-p or Tucker product, corresponds to a summation over the index i_p associated with the mode p of the Pth-order tensor $\mathcal{X}$ and the second index of $\mathbf{A}$, giving a tensor of order $P-1$, and size $I_1 \times \cdots \times I_{p-1} \times I_{p+1} \times \cdots \times I_P$.

The mode-(p,n) product, denoted $\times_p^n$, corresponds to a contraction operation performed for two arbitrary modes (p,n), such as: $I_p = J_n = K$. This multiplication gives a tensor of order $P+N-2$ and size $I_1 \times \cdots \times I_{p-1} \times I_{p+1} \times \cdots \times I_P \times J_1 \times \cdots \times J_{n-1} \times J_{n+1} \times \cdots \times J_N$.

The Einstein product, denoted $\mathcal{A} \star_N \mathcal{X}$, of the tensors $\mathcal{A} \in \mathbb{K}^{\underline{I}_P \times \underline{J}_N}$ of order $P+N$ and $\mathcal{X} \in \mathbb{K}^{\underline{J}_N \times \underline{K}_Q}$ of order $N+Q$ corresponds to a contraction along the N shared indices $\underline{\mathbf{j}}_N$, associated with the N last modes of $\mathcal{A}$ and the N first modes of $\mathcal{X}$. The tensor $\mathcal{A}$ can be interpreted as a multilinear operator associated with a multilinear transformation applied to the tensor $\mathcal{X}$. The Einstein product will be used in section 6 for defining multilinear systems.

Table 9 presents a few examples of outer products of vectors, matrices, and tensors, indicating the order and the space to which the tensors resulting from the products belong.

Table 9. Outer products of vectors, matrices, and tensors.

Vectors/Matrices/Tensors	Outer Products	Spaces	Orders
$\mathbf{u}^{(p)} \in \mathbb{K}^{I_p}, p \in \langle P \rangle$	$\underset{p=1}{\overset{P}{\circ}} \mathbf{u}^{(p)}$	$\mathbb{K}^{\underline{I}_P}$	P
$\mathbf{A}^{(p)} \in \mathbb{K}^{I_p \times J_p}, p \in \langle P \rangle$	$\underset{p=1}{\overset{P}{\circ}} \mathbf{A}^{(p)}$	$\mathbb{K}^{I_1 \times J_1 \times \cdots \times I_P \times J_P}$	$2P$
$\mathcal{A} \in \mathbb{K}^{\underline{I}_P}, \mathcal{B} \in \mathbb{K}^{\underline{J}_N}$	$\mathcal{A} \circ \mathcal{B}$	$\mathbb{K}^{\underline{I}_P \times \underline{J}_N}$	$P+N$
$\mathcal{A}^{(p)} \in \mathbb{K}^{\underline{J}_{N_p}}, p \in \langle P \rangle$	$\underset{p=1}{\overset{P}{\circ}} \mathcal{A}^{(p)}$	$\mathbb{K}^{\underline{J}_{N_1} \times \cdots \times \underline{J}_{N_P}}$	$\sum_{p=1}^P N_p$

4.2. PARAFAC and Tucker Decompositions

The PARAFAC decomposition [13] is also called CANDECOMP (canonical decomposition) by [75] and CP for CANDECOMP/PARAFAC by [76] when applied to decompose a data tensor. In the context of system modeling, it is called a PARAFAC model. It amounts to decomposing a tensor into a sum of R polyads, i.e., R rank-one tensors [10]. For an Nth-order tensor $\mathcal{X}$, each polyad corresponds to the outer product of the rth columns of N factor matrices $\mathbf{A}^{(n)} \in \mathbb{K}^{I_n \times R}$, i.e., $\underset{n=1}{\overset{N}{\circ}} \mathbf{A}^{(n)}_{.r}$. When R is minimal, it is called the tensor rank or canonical rank of $\mathcal{X}$. PARAFAC is also called a canonical polyadic decomposition (CPD), and concisely written as $\{\mathbf{A}^{(1)}, \cdots, \mathbf{A}^{(N)}; R\}$. When $R = 1$, $\mathcal{X} \in \mathbb{K}^{\underline{I}_N}$ is a rank-one tensor, also called a separable tensor. Then, it can be written as the outer product of N non-zero vectors $\mathbf{a}^{(n)} \in \mathbb{K}^{I_n}$

$$\mathcal{X} = \underset{n=1}{\overset{N}{\circ}} \mathbf{a}^{(n)} \iff x_{\underline{\mathbf{i}}_N} = \prod_{n=1}^{N} a^{(n)}_{i_n}. \tag{8}$$

In the case of a symmetric rank-one tensor $\mathcal{X} \in \mathbb{K}^{[N,I]}$, all the vectors $\mathbf{a}^{(n)} \in \mathbb{K}^{I}$ are identical [77].

In Table 10, we present different ways of writing a PARAFAC decomposition for a third-order and Nth-order tensor: scalar writing, with mode-p and outer products, and matrix unfoldings as defined in (3).

Table 10. PARAFAC decomposition of a tensor of order three and order N.

Third-Order Tensor		***N*th-Order Tensor**
$\mathcal{X} \in \mathbb{K}^{I \times J \times K}$		$\mathcal{X} \in \mathbb{K}^{\underline{I}_N}$
$\mathbf{A} \in \mathbb{K}^{I \times R}, \mathbf{B} \in \mathbb{K}^{J \times R}, \mathbf{C} \in \mathbb{K}^{K \times R},$		$\mathbf{A}^{(n)} \in \mathbb{K}^{I_n \times R}$
$x_{i,j,k} = \sum\limits_{r=1}^{R} a_{ir} b_{jr} c_{kr}$	Scalar writing	$x_{\underline{i}_N} = \sum\limits_{r=1}^{R} \prod\limits_{n=1}^{N} a^{(n)}_{i_n,r}$
$\mathcal{X} = \mathcal{I}_R \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$	with mode-n products	$\mathcal{X} = \mathcal{I}_R \underset{n=1}{\overset{N}{\times}} \mathbf{A}^{(n)}$
$\mathcal{X} = \sum\limits_{r=1}^{R} \mathbf{A}_{.r} \circ \mathbf{B}_{.r} \circ \mathbf{C}_{.r}$	with outer products	$\mathcal{X} = \sum\limits_{r=1}^{R} \underset{n=1}{\overset{N}{\circ}} \mathbf{A}^{(n)}_{.r}$
$\mathbf{X}_{IJ \times K} = (\mathbf{A} \diamond \mathbf{B})\mathbf{C}^T$ $\mathbf{X}_{JK \times I} = (\mathbf{B} \diamond \mathbf{C})\mathbf{A}^T$ $\mathbf{X}_{KI \times J} = (\mathbf{C} \diamond \mathbf{A})\mathbf{B}^T$	Matrix unfoldings	$\mathcal{X}_{\mathbb{S}_1;\mathbb{S}_2} = \left(\underset{n \in \mathbb{S}_1}{\diamond} \mathbf{A}^{(n)} \right) \left(\underset{n \in \mathbb{S}_2}{\diamond} \mathbf{A}^{(n)} \right)^T$
$x_{IJK} = (\mathbf{A} \diamond \mathbf{B} \diamond \mathbf{C}) 1_R$	Vectorized form	$\mathbf{x}_{I_1 \cdot I_N} = (\mathbf{A}^{(1)} \diamond \mathbf{A}^{(2)} \diamond \cdots \diamond \mathbf{A}^{(N)})\mathbf{1}_R$

PARAFAC models have the following two main features:

1. Essential uniqueness, i.e., uniqueness up to trivial indeterminacies corresponding to permutation and scalar ambiguities of the columns of the factor matrices (see [78,79]);
2. Existence of a simple algorithm, the so-called alternating least-squares (ALS), for estimating the PARAFAC parameters for a tensor of an arbitrary order N.

The Tucker decomposition [12] of a tensor $\mathcal{X} \in \mathbb{K}^{\underline{I}_N}$ can be viewed as a generalization of the PARAFAC decomposition in the sense that such a decomposition allows taking into account all interactions between distinct columns of the factor matrices $\mathbf{A}^{(n)} \in \mathbb{K}^{I_n \times R_n}$, when a PARAFAC model only involves interactions between the same columns $r \in \langle R \rangle$ of the factor matrices $\mathbf{A}^{(n)} \in \mathbb{K}^{I_n \times R}$. In Table 11, we present different ways of writing a Tucker decomposition for a third-order and Nth-order tensor. From the writing with outer products, we can conclude that the Tucker model of a Nth-order tensor consists in a weighted sum of $\prod_{n=1}^{N} R_n$ rank-one tensors, where the coefficients $g_{r_1,\cdots,r_N}$ of the core tensor $\mathcal{G} \in \mathbb{K}^{\underline{R}_N}$, define the weights of the interactions between the columns $\mathbf{A}^{(n)}_{.r_n}$ of the factor matrices.

Note that a Tucker decomposition is generally not essentially unique, unless additional constraints are imposed, such as a perfect knowledge of the core tensor, certain sparseness or structural constraints on the core tensor or the matrix factors [80,81]. Consult [82] for a review of uniqueness results for Tucker models.

Table 11. Tucker decomposition of a tensor of order three and order N.

Third-Order Tensor		Nth-Order Tensor
$\mathcal{X} \in \mathbb{K}^{I\times J\times K}$		$\mathcal{X} \in \mathbb{K}^{\underline{I}_N}$
$\mathcal{G} \in \mathbb{K}^{P\times Q\times S}, \mathbf{A} \in \mathbb{K}^{I\times P}$, $\mathbf{B} \in \mathbb{K}^{J\times Q}, \mathbf{C} \in \mathbb{K}^{K\times S}$		$\mathcal{G} \in \mathbb{K}^{\underline{R}_N}$, $\mathbf{A}^{(n)} \in \mathbb{K}^{I_n\times R_n}, n \in \langle N\rangle$
$x_{ijk} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{s=1}^{S} g_{pqs}a_{ip}b_{jq}c_{ks}$	Scalar writing	$x_{\underline{i}_N} = \sum_{r_1=1}^{R_1}\cdots\sum_{r_N=1}^{R_N} g_{r_1,\cdots,r_N}\prod_{n=1}^{N} a_{i_n,r_n}^{(n)}$
$\mathcal{X} = \mathcal{G}\times_1\mathbf{A}\times_2\mathbf{B}\times_3\mathbf{C}$	with mode-n products	$\mathcal{X} = \mathcal{G}\times_1\mathbf{A}^{(1)}\times_2\mathbf{A}^{(2)}\times_3\cdots\times_N\mathbf{A}^{(N)}$
$\mathcal{X} = \sum_{p=1}^{P}\sum_{q=1}^{Q}\sum_{s=1}^{S} g_{pqs}\,\mathbf{A}_{.p}\circ\mathbf{B}_{.q}\circ\mathbf{C}_{.s}$	with outer products	$\mathcal{X} = \sum_{r_1=1}^{R_1}\cdots\sum_{r_N=1}^{R_N} g_{r_1,\cdots,r_N}\mathop{\circ}_{n=1}^{N}\mathbf{A}_{.r_n}^{(n)}$

5. Tensor-Based Approaches for Nonlinear Systems

The use of tensor-based approaches for nonlinear systems has proved advantageous in three areas: (i) parametric complexity reduction in order to get efficient and computationally fast parameters estimation algorithms, (ii) generating new representations of nonlinear systems thanks to tensor decompositions, and (iii) structural identification of systems which can be represented as combinations of dynamical linear systems with static nonlinear blocks. In Section 5.1, we first describe polynomial models. Then, in Section 5.2, we introduce standard discrete-time Volterra models. To reduce the parametric complexity of such models, in Section 5.3, we present a tensor approach which consists in using a PARAFAC decomposition of Volterra kernels. The expansion of Volterra kernels on orthogonal bases is considered in Section 5.4. Finally, some links between block-oriented models (Hammerstein, Wiener, and Wiener–Hammerstein) and tensor representations via their associated Volterra kernels are established in Section 5.5

5.1. Polynomial Models

Polynomial models are a direct extension of linear models. For a single-input single-output (SISO) system, the output of a recursive polynomial model, at discrete-time instant t, is given by

$$\hat{y}(t) = \sum_{p=1}^{P} f_p[u(t),\cdots,u(t-n_u),y(t-1),\cdots,y(t-n_y)], \tag{9}$$

where $f_p(.)$ is a pth-degree polynomial in the system input (u) and output (y) signals, P is the nonlinearity order, and $M = max(n_u, n_y)$ is the memory of the model. In the sequel, all the signals will be assumed to be real-valued.

The input/output (I/O) relationship (9) is also called a nonlinear autoregressive with exogenous input (NARX) model [83], or a one-step prediction model, i.e., a model whose output $\hat{y}(t)$ at time t depends on past values $y(t-n)$ (for $n \in \langle n_y\rangle$) of the system output, and current and past values $u(t-n)$ (for $n = 0, 1, \cdots, n_u$) of the system input. This model is an extension of the standard autoregressive with exogenous input (ARX) model, frequently used to study discrete time series, due to the presence of nonlinear terms in the input–output signals, which explains its success in many industrial applications.

Equation (9) can also be written as a regression model which is linear in its parameters, namely the polynomial coefficients, and nonlinear in the I/O signals:

$$\hat{y}(t) = \boldsymbol{\varphi}^T(\mathbf{u}(t), \mathbf{y}(t-1))\boldsymbol{\theta}, \tag{10}$$

where $\mathbf{u}(t) = [u(t) \cdots u(t-n_u)]^T$, $\mathbf{y}(t-1) = [y(t-1) \cdots y(t-n_y)]^T$, and $\boldsymbol{\varphi}$ is the nonlinear regressor vector whose components are monomials in (i.e., products of) previous system outputs and previous and current system inputs contained in the vectors $\mathbf{y}(t-1)$ and $\mathbf{u}(t)$, and $\boldsymbol{\theta}$ is the parameter vector containing the polynomial coefficients.

If the previous system outputs $y(t-1), \cdots, y(t-n_y)$ are replaced by previous model outputs $\hat{y}(t-1), \cdots, \hat{y}(t-n_y)$, the polynomial model (9) is then called a simulation or nonlinear output error (NOE) model, defined as

$$\hat{y}(t) = \sum_{p=1}^{P} f_p[u(t), \cdots, u(t-n_u), \hat{y}(t-1), \cdots, \hat{y}(t-n_y)] = \boldsymbol{\varphi}^T(\mathbf{u}(t), \hat{\mathbf{y}}(t-1))\boldsymbol{\theta}, \qquad (11)$$

with $\hat{\mathbf{y}}(t-1) = [\hat{y}(t-1) \cdots \hat{y}(t-n_y)]^T$.

This model is recursive with respect to previous model outputs $\hat{y}(t-n)$ (for $n \in \langle n_y \rangle$), while the one-step prediction model (10) is purely feedforward.

As for linear systems, the advantage of NARX and NOE models with output feedback is to be more parsimonious than without output feedback, which means a reduced parametric complexity in terms of dimension of the parameter vector $\boldsymbol{\theta}$. One drawback of output feedback is that stability is generally not guaranteed. Another drawback of NOE models is that they need to use a nonlinear optimization method for parameter estimation due to the dependence of $\hat{\mathbf{y}}(t-1)$ on $\boldsymbol{\theta}$ in the regression Equation (11) implying a nonlinear dependence of the model output with respect to model parameters. That is not the case for the NARX model that is linear in its parameters, whose estimation can therefore be carried out by means of the standard least-squares (LS) algorithm.

5.2. Truncated Volterra Models

When the polynomial functions $f_p(.)$ in (9) are independent from the output signal, i.e., without output feedback, the polynomial model is called a nonrecursive polynomial model or a discrete-time Volterra model. A Pth-order Volterra model for a causal, stable, finite-memory, time-invariant SISO system is described by the following I/O relationship:

$$\hat{\mathbf{y}}(t) = h_0 + \sum_{p=1}^{P} \sum_{m_1=1}^{M_p} \cdots \sum_{m_P=1}^{M_P} h^{(p)}_{m_1,\cdots,m_P} \prod_{q=1}^{p} u(t-m_q+1) = h_0 + \sum_{p=1}^{P} \hat{y}^{(p)}(t), \qquad (12)$$

where h_0 is the offset, M_p is the memory of the pth-order homogeneous term $\hat{y}^{(p)}(t)$, and $h^{(p)}_{m_1,\cdots,m_P}$ is a coefficient of the pth-order Volterra kernel, assumed to be real-valued.

Note that a truncated Volterra model can be seen as a truncated Taylor series expansion for approximating a given smooth nonlinear function (around 0 by convention).

Equation (12) can also be written as a polynomial regression model linear in its parameters and composed of monomials in previous samples of the input signal

$$\hat{\mathbf{y}}(t) = h_0 + \boldsymbol{\varphi}^T(\mathbf{u}(t))\boldsymbol{\theta}, \qquad (13)$$

where $\mathbf{u}(t) = [u(t), \cdots, u(t-M)]^T$, with $M = max_p(M_p)$, and $\boldsymbol{\theta}$ is the parameter vector containing all the kernel coefficients, and the vector $\boldsymbol{\varphi}$ contains all possible monomials in u up to degree P. In the sequel, we assume that all memories M_p are equal to M. The coefficient $h^{(p)}_{m_1,\cdots,m_P}$ being characterized by p indices can be viewed as an element of a tensor $\mathcal{H}^{(p)} \in \mathbb{R}^{[p,M]}$, of order p, characterized by M^p entries, which is a number growing very fast with the kernel order p. The pth-order homogeneous term $\hat{y}^{(p)}(t)$ can then be written using the Tucker product as

$$\hat{y}^{(p)}(t) = \mathcal{H}^{(p)} \overset{p}{\underset{q=1}{\times}} \mathbf{u}^T(t), \qquad (14)$$

which is a homogeneous polynomial of degree p in the components of the input vector.

Several adaptive and nonadaptive methods have been proposed to identify truncated Volterra models from I/O measurements, both in the time and frequency domains. Frequency methods are based on the use of input signal cumulants, which requires estimating high-order statistics of the input signal, up to order $2P$ for a Pth-order Volterra model. Such an approach is mainly interesting with a Gaussian input signal, since the input cumulants of order higher than two are then zero, which implies a significant simplification of frequency methods. In the time domain, we can distinguish the optimal minimum mean-square error (MMSE) estimator, based on the use of input signal statistics, the nonrecursive least-squares (LS) algorithm, which can be viewed as an approximation of the MMSE solution, and adaptive methods. Note that estimating the parameters of an homogeneous Pth-order Volterra kernel, with memory M, using the MMSE and nonrecursive LS solutions requires inverting an autocorrelation matrix of size $M^P \times M^P$, which is a time consuming and numerically difficult task.

Adaptive methods are often associated with adaptive Volterra filters used for representing NL time-varying signals and systems as encountered in echo cancellation, for instance. Parameter estimation of adaptive Volterra filters is carried out using the well-known least-mean-square (LMS) or recursive LS (RLS) algorithms. See the book [28] and the references therein for an overview of the methods briefly introduced above.

In the next section, we present an approach for identifying reduced complexity Volterra models which is based on a PARAFAC decomposition of symmetrized kernels, leading to the so-called Volterra–PARAFAC models.

5.3. Volterra–PARAFAC Models

As each permutation of the indices $m_1, \cdots, m_p$ corresponds to the same product $\prod_{q=1}^{p} u(t - m_q + 1)$ of delayed inputs, we can sum all the coefficients associated with these permutations to get a symmetric kernel given by

$$h^{(p,sym)}_{m_1,\cdots,m_P} = \frac{1}{p!} \sum_{\pi(.)} h^{(p)}_{m_{\pi(1)},\cdots,m_{\pi(p)}},$$

where $(\pi(1), \cdots, \pi(p))$ denotes a permutation of $(1, \cdots, p)$. So, in the sequel, without loss of generality, the Volterra kernels of order $p \geq 2$ will be considered in symmetric form. Assuming all the kernels have the same memory M, the number of independent coefficients contained in the symmetric pth-order kernel is equal to $C_p^{M+p-1} = \frac{(M+p-1)!}{p!\,(M-1)!}$, showing that this number, and consequently the parametric complexity of the Volterra model, grows quickly with M even for moderate p.

In order to reduce the complexity of Volterra models, a PARAFAC decomposition of symmetrized kernels was exploited in [40,41]. The symmetrized pth-order Volterra kernel can then be decomposed using a symmetric PARAFAC decomposition, with symmetric rank r_p and matrix factor $\mathbf{A}^{(p)} \in \mathbb{R}^{M \times r_p}$, for $p \in \langle P \rangle$, as [77]

$$h^{(p,sym)}_{m_1,\cdots,m_P} = \sum_{r=1}^{r_p} \prod_{q=1}^{p} a^{(p)}_{m_q,r} \ , \ m_q = 1, \cdots M. \tag{15}$$

Remark 1. *Note that a pth-order Volterra kernel is said to be separable if it can be written as the product of p first-order kernels [28], i.e.,*

$$h^{(p)}_{m_1,\cdots,m_P} = \prod_{q=1}^{p} a^{(p)}_{m_q} \ , \ m_q = 1, \cdots M \tag{16}$$

which corresponds to a rank-one PARAFAC decomposition (15).

The kernel decomposition (15) allows rewritting the pth-order homogeneous term as follows:

$$\hat{y}^{(p)}(t) = \sum_{m_1=1}^{M} \cdots \sum_{m_P=1}^{M} h^{(p,sym)}_{m_1,\cdots,m_P} \prod_{q=1}^{p} u(t-m_q+1) \tag{17}$$

$$= \sum_{m_1=1}^{M} \cdots \sum_{m_P=1}^{M} \Big(\sum_{r=1}^{r_p} \prod_{q=1}^{p} a^{(p)}_{m_q,r}\Big) \prod_{q=1}^{p} u(t-m_q+1), \tag{18}$$

or equivalently

$$\hat{y}^{(p)}(t) = \sum_{r=1}^{r_p} \prod_{q=1}^{p} \Big(\sum_{m_q=1}^{M} a^{(p)}_{m_q,r} u(t-m_q+1)\Big) = \sum_{r=1}^{r_p} \big(\mathbf{u}^T(t)\mathbf{A}^{(p)}_{.r}\big)^p. \tag{19}$$

We then obtain a homogeneous polynomial of degree p expressed as a sum of powers of linear forms, which is directly connected to the Waring problem. Note that a Waring's decomposition consists in expressing a homogeneous polynomial of degree p in n variables (i.e., a quantics), associated with a symmetric tensor, as a sum of pth powers of linear forms [84]. This pth-order homogeneous term can therefore be carried out in parallelizing r_p Wiener models. As introduced later (see Section 5.5.2), each Wiener model is composed of a FIR linear filter whose coefficients are the components of a column $\mathbf{A}^{(p)}_{.r} \in \mathbb{R}^M$ of the matrix factor $\mathbf{A}^{(p)}$, in cascade with a static nonlinearity equal to the power $(.)^p$. Consequently, the Volterra model output (12) is obtained as the sum of the offset term h_0, and the outputs of $\sum_{p=1}^{P} r_p$ Wiener models in parallel, as illustrated in Figure 1 for a cubic Volterra–PARAFAC model, where $\mathbf{A}^{(1)}_{.1} = \Big[h^{(1)}_1, \cdots, h^{(1)}_M\Big]$ and $r_1 = 1$.

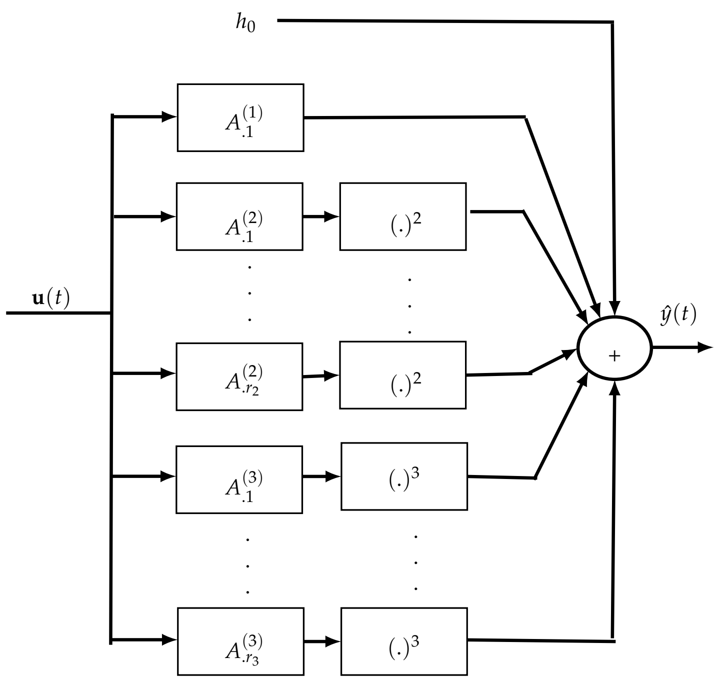

Figure 1. Realization of a third-order Volterra–PARAFAC model as Wiener models in parallel.

It is worth noting that such a Volterra–PARAFAC model provides a very attractive modular and parallel architecture for approximating nonlinear systems with a low computational complexity.

This Volterra–PARAFAC architecture is to be compared with the parallel cascade Wiener (PCW) model composed of P Wiener models in parallel and described by the following equation:

$$\hat{y}(t) = \sum_{p=1}^{P} \mathcal{N}^{(p)}\Big(\sum_{m=1}^{M} h_m^{(p)} u(t-m+1)\Big), \tag{20}$$

where $h_m^{(p)}$ is the mth coefficient of the pth FIR model $\mathbf{h}^{(p)}$, and $\mathcal{N}^{(p)}$ represents a static nonlinearity for the pth path, $p \in \langle P \rangle$. Comparing Equation (20) with Equation (19) allows us to conclude that the Volterra–PARAFAC model is a PCW one whose the FIR filters are the columns of the factor matrices of the PARAFAC representations of the Volterra kernels, and the pth static nonlinearity $\mathcal{N}^{(p)}(.)$ is the power $(.)^p$. In [85], it is shown that any discrete-time, finite-memory nonlinear system can be approximated with an arbitrary accuracy by a PCW model, with a finite number P of paths. A method based on a joint diagonalization of third-order Volterra kernel slices is proposed in [50] for identifying PCW systems.

The extended Kalman filter (EKF) was proposed in [40,41] to estimate the parameters of a Volterra–PARAFAC model, associated with the following state-space representation

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}(t-1) + \mathbf{w}(t) \tag{21}$$

$$y(t) = \sum_{p=1}^{P}\sum_{r=1}^{r_p} \left(\mathbf{u}^T(t)\mathbf{A}_{.r}^{(p)}\right)^p = G(\boldsymbol{\theta}(t), \mathbf{u}(t)), \tag{22}$$

where the state vector is the Volterra–PARAFAC parameters vector $\boldsymbol{\theta}$ defined as

$$\boldsymbol{\theta} \triangleq \Big[[\mathbf{A}_{.1}^{(1)}]^T, [\mathbf{A}_{.1}^{(2)}]^T, \cdots, [\mathbf{A}_{.r_2}^{(2)}]^T, \cdots, [\mathbf{A}_{.1}^{(P)}]^T, \cdots, [\mathbf{A}_{.r_P}^{(P)}]^T\Big]^T \in \mathbb{R}^{\bar{M}} \tag{23}$$

$$= \Big[[\mathbf{A}_{.1}^{(1)}]^T, \mathrm{vec}^T(\mathbf{A}^{(2)}), \cdots, \mathrm{vec}^T(\mathbf{A}^{(P)})\Big]^T \tag{24}$$

$$= \Big[[\boldsymbol{\theta}^{(1)}]^T, [\boldsymbol{\theta}^{(2)}]^T, \cdots, [\boldsymbol{\theta}^{(P)}]^T\Big]^T, \tag{25}$$

with $\boldsymbol{\theta}^{(p)} \triangleq \mathrm{vec}(\mathbf{A}^{(P)}) \triangleq \Big[[\mathbf{A}_{.1}^{(p)}]^T, \cdots, [\mathbf{A}_{.r_p}^{(p)}]^T\Big]^T$, for $p \in \langle P \rangle$, and $\bar{M} = M\Big(1 + \sum_{p=2}^{P} r_p\Big)$.

Equation (21) corresponds to a random walk model for modeling slowly time-varying parameters $\boldsymbol{\theta}$, and $\mathbf{w}(t) \in \mathbb{R}^{\bar{M}}$ is a white Gaussian noise sequence with covariance $\sigma_w^2 \mathbf{I}_{\bar{M}}$.

The EKF algorithm can be used online for updating the estimated parameters as input samples become available, and even for tracking time-varying kernels. It is obtained by applying the Kalman filter after linearization of the nonlinear function $G(\boldsymbol{\theta}, \mathbf{u}(t))$ around the last estimate $\hat{\boldsymbol{\theta}}(t-1)$

$$y(t) \approx G(\hat{\boldsymbol{\theta}}(t-1), \mathbf{u}(t)) + \mathbf{h}^T(t)(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}(t-1)), \tag{26}$$

where $\mathbf{h}(t)$ is the gradient of $G(\boldsymbol{\theta}, \mathbf{u}(t))$ with respect to the parameter vector $\boldsymbol{\theta}$, calculated at the point $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}(t-1)$

$$\mathbf{h}(t) \triangleq \frac{\partial G(\boldsymbol{\theta}, \mathbf{u}(t))}{\partial \boldsymbol{\theta}}\Big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}(t-1)} \in \mathbb{R}^{\bar{M}} \tag{27}$$

$$= \Big[(\mathbf{h}^{(1)}(t))^T, (\mathbf{h}^{(2)}(t))^T, \cdots, (\mathbf{h}^{(P)}(t))^T\Big]^T \tag{28}$$

$$\mathbf{h}^{(1)}(t)) = \mathbf{u}(t) \tag{29}$$

$$\mathbf{h}^{(P)}(t) \triangleq \frac{\partial G(\boldsymbol{\theta}, \mathbf{u}(t))}{\partial \boldsymbol{\theta}^{(p)}}\Big|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}(t-1)}, \text{ for } p \in [2, P]. \tag{30}$$

Let us define the scalar quantity $z_{p,r}(t)$ as

$$z_{p,r}(t) \triangleq \mathbf{u}^T(t)\mathbf{A}_{.r}^{(p)}. \tag{31}$$

The nonlinear function $G(\boldsymbol{\theta}, \mathbf{u}(t))$ defined in (22) can then be written as

$$G(\boldsymbol{\theta}, \mathbf{u}(t)) = \sum_{p=1}^{P} \sum_{r=1}^{r_p} z_{p,r}^{p}(t). \tag{32}$$

By the chain rule, we have

$$\begin{aligned}
\frac{\partial G(\boldsymbol{\theta}, \mathbf{u}(t))}{\partial \mathbf{A}_{.r}^{(p)}} &= p\, z_{p,r}^{p-1}(t)\mathbf{u}(t) \in \mathbb{R}^M \\
&\Downarrow \\
\mathbf{h}^{(P)}(t) &= \left[\left[\frac{\partial G(\boldsymbol{\theta}, \mathbf{u}(t))}{\partial \mathbf{A}_{.1}^{(p)}} \right]^T, \cdots, \left[\frac{\partial G(\boldsymbol{\theta}, \mathbf{u}(t))}{\partial \mathbf{A}_{.r_p}^{(p)}} \right]^T \right]^T_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}(t-1)} \\
&= p \left[\hat{z}_{p,1}^{p-1}(t), \cdots, \hat{z}_{p,r_p}^{p-1}(t) \right]^T \otimes \mathbf{u}(t) \in \mathbb{R}^{Mr_p},
\end{aligned} \tag{33}$$

where $\hat{z}_{p,r}(t) = \mathbf{u}^T(t)\hat{\mathbf{A}}_{.r}^{(p)}$.

The EKF equations are then derived from the Kalman filter associated with the linearized state space equations

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}(t-1) + \mathbf{w}(t) \tag{34}$$

$$y(t) = \mathbf{h}^T(t)\boldsymbol{\theta} + n(t), \tag{35}$$

where $n(t)$ is assumed to be a white Gaussian noise, with variance σ_n^2, including both the measurement noise and the modeling error.

The innovation process associated with the linearized Equation (26) is equal to

$$e(t) = y(t) - G(\hat{\boldsymbol{\theta}}(t-1), \mathbf{u}(t)) \tag{36}$$

$$= \mathbf{h}^T(t)\big(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}(t-1)\big) + n(t), \tag{37}$$

with variance $s(t) = E[e^2(t)] = \mathbf{h}^T(t)\mathbf{P}(t|t-1)\mathbf{h}(t) + \sigma_n^2$, where $\mathbf{P}(t|t-1)$ is the covariance matrix of the prediction error $\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}(t-1)$, and

$$G(\hat{\boldsymbol{\theta}}(t-1), \mathbf{u}(t)) = \sum_{p=1}^{P} \sum_{r=1}^{r_p} \big(\mathbf{u}^T(t)\hat{\mathbf{A}}_{.r}^{(p)}(t-1)\big)^p. \tag{38}$$

The Kalman gain is given by

$$\mathbf{k}(t) = \frac{1}{s(t)}\mathbf{P}(t|t-1)\mathbf{h}(t), \tag{39}$$

and the recursive equation for calculating the parameter vector estimate is

$$\hat{\boldsymbol{\theta}}(t) = \hat{\boldsymbol{\theta}}(t-1) + \mathbf{k}(t)e(t). \tag{40}$$

Finally, the equation for updating the covariance matrix of the one-step prediction error is

$$\mathbf{P}(t+1/t) = \Big(\mathbf{I}_{\bar{M}} - \mathbf{k}(t)\mathbf{h}^T(t)\Big)\mathbf{P}(t|t-1) + \sigma_w^2\mathbf{I}_{\bar{M}}. \tag{41}$$

The EKF algorithm is summarized in Algorithm 1.

Algorithm 1: Extended Kalman filter for parameter estimation of a Volterra–PARAFAC model.

```
Given $\sigma_w^2$ and $\sigma_n^2$:
1.  Initialize $\mathbf{P}(0/-1)$ and $\hat{\boldsymbol{\theta}}(0)$.
2.  For $t = 1$ to $t = T$, compute:
    • The innovation process $e(t)$ using Equations (36) and (38);
    • The gradient $\mathbf{h}(t)$ using Equations (28), (29), (31) and (33);
    • The Kalman gain $\mathbf{k}(t)$ using Equation (39);
    • The recursive parameter estimate with Equation (40);
    • The updated error covariance matrix $\mathbf{P}(t+1/t)$ using Equation (41).
```

Example 1. *In this example, we consider a memory $M = 5$ third-order Volterra model with rank one second-order and third-order kernels. Each kernel acts on a specific bandwidth, making the nonlinear distortion frequency selective. Precisely:*

- *First-order kernel $h_{m_1}^{(1)} = a_{m_1}^{(1)}$, with $a_{m_1}^{(1)}$ the m_1th entry of $\mathbf{A}_{.1}^{(1)} = \begin{bmatrix} 0.0284 & 0.2370 & 0.4692 & 0.2370 & 0.0284 \end{bmatrix}^T$, which represents a low pass FIR filter with normalized cut-off frequency 0.2.*
- *Second-order kernel $h_{m_1,m_2}^{(2)} = a_{m_1}^{(2)} a_{m_2}^{(2)}$ with a_{m_i}, $i = 1,2$ the entries of $\mathbf{A}_{.1}^{(2)} = \begin{bmatrix} -0.0568 & 0.0708 & 0.8698 & 0.0708 & -0.0568 \end{bmatrix}^T$, which stands for a bandpass filter with normalized frequencies 0.3 and 0.5.*
- *Third-order kernel $h_{m_1,m_2,m_3}^{(2)} = a_{m_1}^{(3)} a_{m_2}^{(3)} a_{m_3}^{(3)}$ with a_{m_i}, $i = 1,2,3$ the entries of $\mathbf{A}_{.1}^{(3)} = \begin{bmatrix} 0.0302 & -0.3463 & 0.7471 & -0.3463 & 0.0302 \end{bmatrix}^T$, a high-pass filter with normalized frequency 0.9.*

We first analyze the output reconstruction capability of Volterra–PARAFAC with parameters estimated by an EKF. Then, we evaluate the transient behavior of the algorithm in the noiseless case in comparison with PCWS. Eventually, the steady state results in the noisy case are evaluated. The considered PCWS has three branches, each branch being a Wiener system of order 3 and memory 5. The parameters of PCWS were estimated using an EKF. The simulation results given hereafter were obtained by implementing the algorithms with MATLAB R2018b. The code is provided as Supplementary Material.

Output reconstruction: *We consider the composite signal $u(t) = 0.5sin(0.01\pi t) + 0.5sin(0.9\pi t)$ as input. Figure 2 depicts output reconstruction obtained with the proposed EKF filter from a noisy signal. One can note a very good reconstruction after convergence of the filter.*

Transient behavior evaluation in the noiseless case: *$R = 100$ Monte Carlo runs are considered for this analysis. For each run ρ, the square error $e_\rho(t) = \left(y(t) - \hat{y}_\rho(t)\right)^2$, with $\hat{y}_\rho(t)$ the reconstructed output at the ρ-th run, is computed. Then the median value over the R runs is computed as $\epsilon(t) = \text{median}\{e_\rho(t),\ \rho = 1,2,\cdots,R\}$. This allows discarding outliers due to ill convergence of EKF. Indeed, depending on the initialization, the EKF sometimes failed to converge with the selected number of samples. This is particularly the case for PCWS. Finally, $\epsilon(t)$ is smoothed with a moving average filter: $\epsilon_L(t) = \frac{1}{L}\sum_{\tau=0}^{L-1} \epsilon(t-\tau)$. The obtained results are given in Figure 3 where a comparison between PCWS and Volterra–PARAFAC in terms of the square error $\epsilon_L(t)$, with $L = 100$, is depicted. In general, EKF converges faster with Volterra–PARAFAC than with PCWS in the noiseless case.*

Evaluation in steady state: *To evaluate the steady state performance, the NMSE (normalized mean square error) is calculated as $NMSE = \frac{\bar{\epsilon}^2}{\bar{y}^2}$, with $\bar{\epsilon}^2 = \frac{1}{t_f - t_0}\sum_{t=t_0}^{t_f} \epsilon^2(t)$ and*

$\bar{y}^2 = \frac{1}{t_f - t_0} \sum_{t=t_0}^{t_f} y^2(t)$, *where the interval* $[t_0, t_f]$ *characterizes the steady state. The evaluation was carried out with two types of input signals: the composite sum of sines previously used and a random input. The random input was drawn from a uniform distribution between* -1 *and* 1. *The number of iterations needed for convergence of the EKF with the random input was much less than with sum of sines;* 10,000 *samples were generated for a random input and the steady performance was evaluated from the* 1000 *last samples of the reconstructed output. In the case of sum of sines,* 100,000 *samples were generated and the steady state was evaluated from the* 10,000 *last samples. A white Gaussian noise was added to the output; its variance depends on a specified signal-to-noise ratio (SNR). For different values of SNR, Figures 4 and 5 depict the NMSE in steady state for sum of sines and for the considered random input, respectively. It can be noticed that in steady state, both Volterra–PARAFAC and PCWS give the same performance whatever the input signal. With a lower SNR value, Volterra–PARAFAC is slightly better than PCWS.*

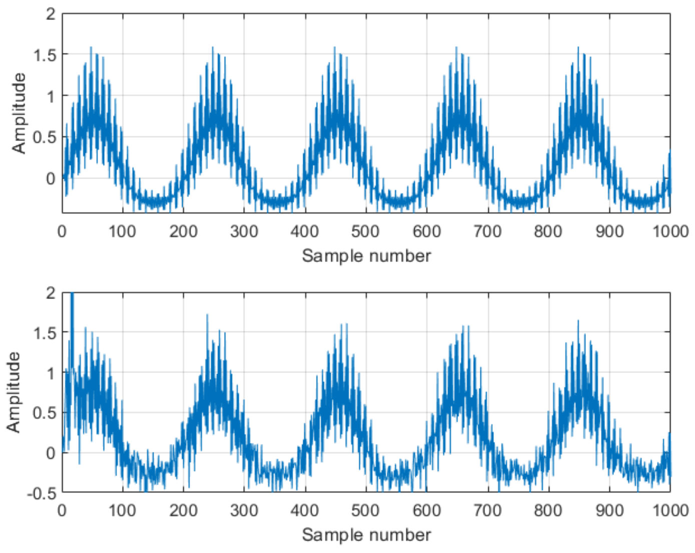

Figure 2. Top: Original output signal. **Bottom**: reconstructed output from noisy measurements (SNR = 30 dB).

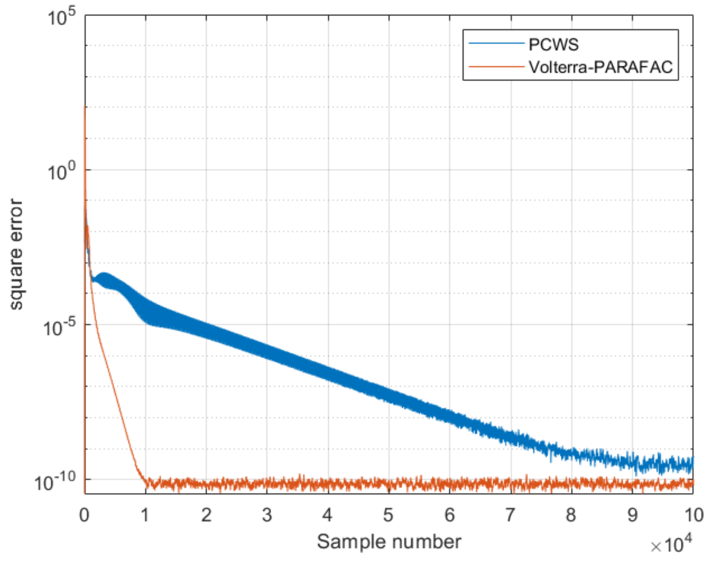

Figure 3. Evolution of the square error $\epsilon_L(t)$, $L = 100$.

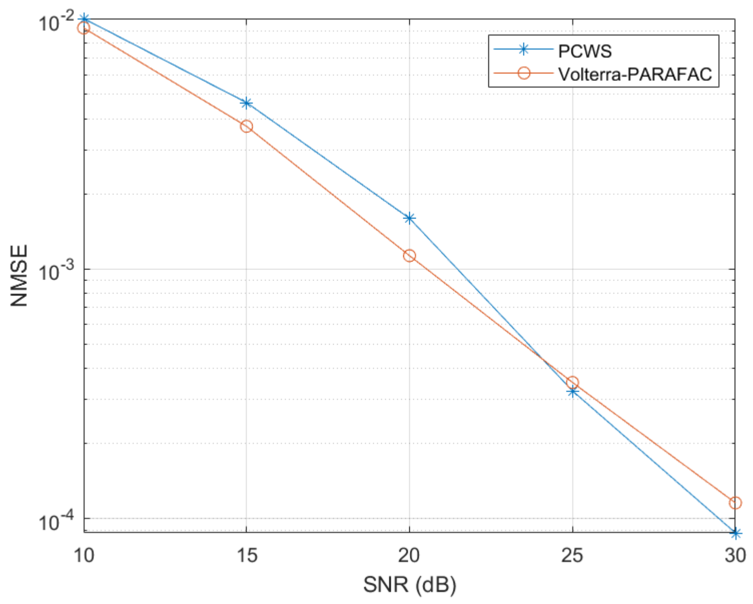

Figure 4. Normalized mean square error in steady state for a sum of sines.

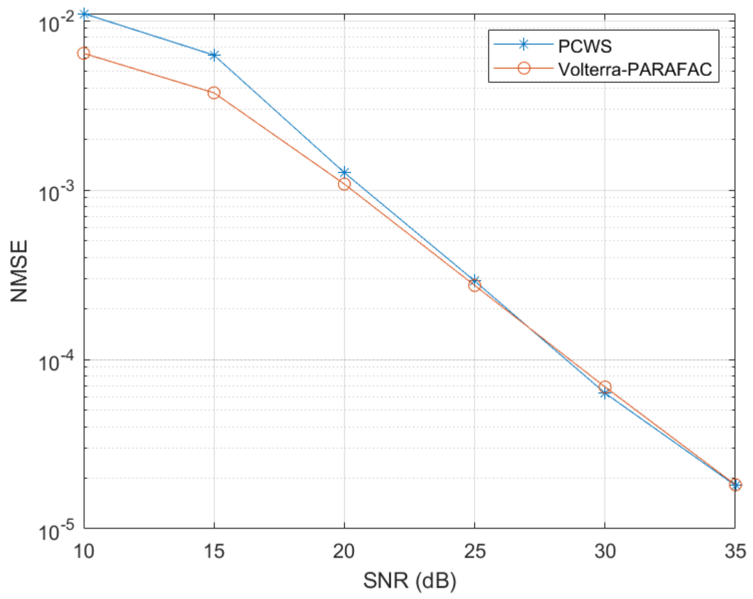

Figure 5. Normalized mean square error in steady state for a random input.

5.4. Volterra–GOB Models

Under stability and causality conditions, a Volterra kernel $\mathcal{H}^{(p)}$ can be expanded on a basis of orthogonal functions [27]. Various functions have been introduced in the literature (Laguerre, Kautz, generalized orthogonal basis functions (GOBF), etc.). Selection of such a basis has been widely studied (see [38] for instance). Defining by $b_{k_j}^{(j,p)}(.)$, $k_j = 1, 2, \cdots$, a set of orthogonal basis functions for expanding the pth-order Volterra kernel along its jth mode, $j \in \langle p \rangle$, the GOB expansion of this Volterra kernel in such a basis is given by

$$h^{(p)}_{m_1,m_2,\cdots,m_p} = \sum_{k_1=1}^{\infty}\sum_{k_2=1}^{\infty}\cdots\sum_{k_p=1}^{\infty} g^{(p)}_{k_1,k_2,\cdots,k_p}\prod_{j=1}^{p} b_{k_j}^{(j,p)}(m_j)\ ,\ m_j \in \langle M_p \rangle\ ,\ j \in \langle p \rangle, \tag{42}$$

where $g^{(p)}_{k_1,k_2,\cdots,k_p}$ are the coefficients of the expansion, also called Fourier coefficients, and the GOB functions $b_{k_j}^{(j,p)}(.)$ in the time domain can be derived from the inverse z-transform of some transfer function [86]. This expansion is often truncated to a given order K_p for practical reasons, leading to the following truncated development

$$h^{(p)}_{m_1,m_2,\cdots,m_p} = \sum_{k_1=1}^{K_p}\sum_{k_2=1}^{K_p}\cdots\sum_{k_p=1}^{K_p} g^{(p)}_{k_1,k_2,\cdots,k_p}\prod_{j=1}^{p} b^{(j,p)}_{k_j}(m_j), \tag{43}$$

where $b^{(j,p)}_{k_j}(m_j)$ is the m_jth entry of the k_jth column $\mathbf{B}^{(j,p)}_{.k_j} \in \mathbb{R}^{M_p}$ of the matrix factor $\mathbf{B}^{(j,p)} \in \mathbb{R}^{M_p\times K_p}$, associated with mode $j \in \langle p\rangle$.

The development (43) of the pth-order Volterra kernel, viewed as a pth-order tensor $\mathcal{H}^{(p)} = \big[h^{(p)}_{m_1,m_2,\cdots,m_p}\big] \in \mathbb{R}^{M_p\times\cdots\times M_p}$, can be interpreted as the following Tucker model:

$$\mathcal{H}^{(p)} = \mathcal{G}^{(p)} \overset{p}{\underset{j=1}{\times}} \mathbf{B}^{(j,p)}, \tag{44}$$

where the core tensor $\mathcal{G}^{(p)} \in \mathbb{R}^{K_p\times\cdots\times K_p}$ contains the Fourier coefficients.

Consider the FIR linear filter $B^{(j,p)}_{k_j}(q^{-1}) = \sum_{m_j=1}^{M_p} b^{(j,p)}_{k_j}(m_j)q^{-m_j}$, where q^{-1} is the unit delay operator. This filter, with memory M_p, is associated with mode j of the tensor $\mathcal{H}^{(p)}$.

Now, let us define the filtered input $s^{(j,p)}_{k_j}(t)$, for $k_j \in \langle K_p\rangle$ and $j \in \langle p\rangle$, as

$$s^{(j,p)}_{k_j}(t) = B^{(j,p)}_{k_j}(q^{-1})u(t) = \sum_{m_j=1}^{M_p} b^{(j,p)}_{k_j}(m_j)u(t-m_j). \tag{45}$$

Using the truncated expansion (43) of the pth Volterra kernel and the filtered inputs (45), the input–output relationship for the pth-order homogeneous Volterra–GOB term can then be written as

$$\hat{y}^{(p)}(t) = \sum_{m_1=1}^{M_p}\sum_{m_2=1}^{M_p}\cdots\sum_{m_p=1}^{M_p} h^{(p)}_{m_1,m_2,\cdots,m_p}\prod_{j=1}^{p} u(t-m_j) \tag{46}$$

$$= \sum_{k_1=1}^{K_p}\sum_{k_2=1}^{K_p}\cdots\sum_{k_p=1}^{K_p} g^{(p)}_{k_1,k_2,\cdots,k_p}\prod_{j=1}^{p} s^{(j,p)}_{k_j}(t). \tag{47}$$

Taking the Tucker model (44) of the pth order kernel tensor $\mathcal{H}^{(p)}$ into account, and defining the vector $\mathbf{s}^{(j,p)}(t) = \Big[s^{(j,p)}_1(t),\cdots,s^{(j,p)}_{K_p}(t)\Big]^T \in \mathbb{R}^{K_p}$, the input–output equation for the Volterra–GOB model then becomes

$$\hat{y}(t) = h_0 + \sum_{p=1}^{P} \mathcal{G}^{(p)} \overset{p}{\underset{j=1}{\times}} [\mathbf{s}^{(j,p)}(t)]^T. \tag{48}$$

Figure 6 illustrates a third-order Volterra–GOB model.

Remark 2. *Note that the truncation order K_p, and as a consequence the parametric complexity of the Volterra–GOB model, is strongly dependent on the choice of the GOB functions , which is a difficult task. Once these functions are fixed, the Volterra–GOB model is linear in its parameters, the Fourier coefficients, which can be estimated using the standard least-squares (LS) method. In comparison, the Volterra–PARAFAC model is nonlinear in its parameters, the PARAFAC coefficients, which requires the use of a nonlinear optimization method like the extended Kalman filter, for their estimation.*

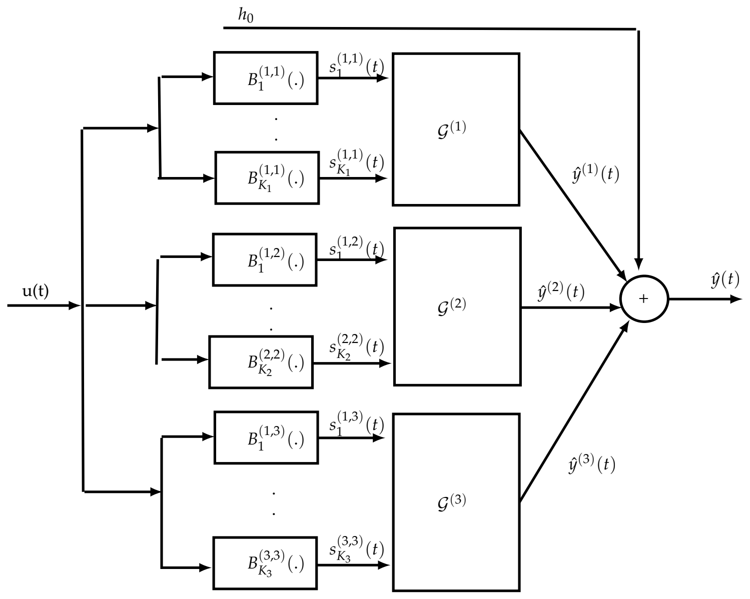

Figure 6. Realization of a third-order Volterra–GOB model.

5.5. Block-Oriented Models

Nonlinear input–output models constituted by a cascade of linear dynamic subsystems with memoryless (static) nonlinearities, also called block-oriented (or block-structured) nonlinear models, have been extensively studied by many authors during the last three decades. They play an important role in many fields of application owing to their low parametric complexity implying a low computational cost for system identification. Moreover, they often reflect the structure of physical systems. We review hereafter the three most common block-oriented models and their tensor representation. According to the Weierstrass theorem, it is assumed that nonlinear blocks are continuous and therefore can be represented with a polynomial of a given degree P: $c(x) = \sum_{p=0}^{P} c_p x^p$.

5.5.1. Hammerstein Model

It is constituted with a nonlinear functional block followed by a FIR linear one $g(.)$, with memory M_g. In control applications, as illustrated on Figure 7, the Hammerstein model is used for representing control systems with nonlinearities in the actuator.

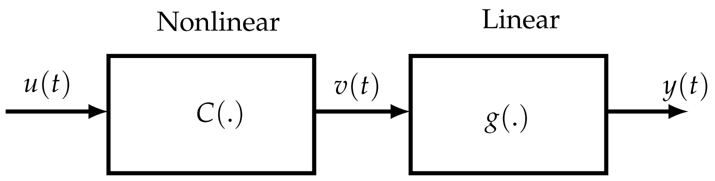

Figure 7. Block diagram of a Hammerstein model.

The output $y(t)$ of the Hammerstein model is given by

$$y(t) = \sum_{i=1}^{M_g} g_i v(t-i) \tag{49}$$

$$= \sum_{i=1}^{M_g} g_i \Big(\sum_{p=0}^{P} c_p u^p(t-i) \Big) \tag{50}$$

$$= \sum_{p=0}^{P} c_p \sum_{i=1}^{M_g} g_i u^p(t-i). \tag{51}$$

This model is therefore equivalent to a Volterra model of order P, with the following pth-order kernel

$$h^{(p)}_{i,i_2,\cdots,i_p} = c_p g_i \delta_{i,i_2,\cdots,i_p} \ , \ i \in \langle M_g \rangle, \tag{52}$$

where $\delta_{i,i_2,\cdots,i_p}$ is the generalized Kronecker delta. The corresponding tensor is diagonal and given by [49]

$$\mathcal{H}^{(p)} = c_p \mathcal{G}^{(p)}, \tag{53}$$

where $\mathcal{G}^{(p)} \in \mathbb{R}^{M_g \times \cdots \times M_g}$ is a diagonal tensor whose diagonal elements are the components of the FIR coefficients vector $\mathbf{g} = [g_1, \cdots, g_{M_g}]^T$.

5.5.2. Wiener Model

It is the dual of the Hammerstein model, i.e., the FIR linear functional block $l(.)$, with memory M_h, comes before the nonlinear one, as illustrated on Figure 8. It allows taking sensor nonlinearities into account for instance.

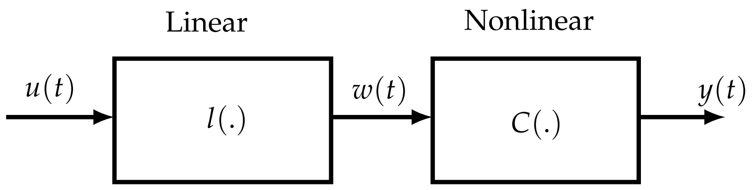

Figure 8. Block diagram of a Wiener model.

For this model, the output $y(t)$ is given by

$$\begin{aligned} y(t) =& c(w(t)) = \sum_{p=0}^{P} c_p w^p(t) && (54)\\ =& \sum_{p=0}^{P} c_p \Big(\sum_{i=1}^{M_h} l_i u(t-i) \Big)^p && (55)\\ =& \sum_{p=0}^{P} c_p \sum_{i_1=1}^{M_h} \cdots \sum_{i_p=1}^{M_h} \prod_{j=1}^{p} l_{i_j} u(t-i_j). && (56) \end{aligned}$$

This model is equivalent to a Volterra model of order P, whose the pth-order kernel is a rank one symmetric tensor defined as

$$h^{(p)}_{i_1,i_2,\cdots,i_p} = c_p \prod_{j=1}^{p} l_{i_j} \ , \ i_j \in \langle M_h \rangle \ , \ j \in \langle p \rangle, \tag{57}$$

or equivalently

$$\mathcal{H}^{(p)} = c_p \overset{P}{\underset{j=1}{\circ}} \mathbf{l}, \tag{58}$$

where $\mathbf{l} = [l_1, \cdots, l_{M_h}]^T$ is the vector of FIR coefficients.

5.5.3. Wiener–Hammerstein Model

The Wiener–Hammerstein model, whose structure is illustrated in Figure 9, is a combination of the Wiener and Hammerstein models described previously. Its output $y(t)$ is given by

$$y(t) = \sum_{i=1}^{M_g} g_i v(t-i) = \sum_{i=1}^{M_g} g_i c(w(t-i)) \tag{59}$$

$$= \sum_{i=1}^{M_g} g_i \sum_{p=0}^{P} c_p w^p(t-i) \tag{60}$$

$$= \sum_{p=0}^{P} c_p \sum_{i=1}^{M_g} g_i \prod_{j=1}^{p} \sum_{m_j=1}^{M_h} l_{m_j} u(t-i-m_j). \tag{61}$$

Defining the changes of variables $i_j = i + m_j$, for $j \in \langle p \rangle$, and reordering the sums lead to the following I/O relationship:

$$y(t) = \sum_{p=0}^{P} c_p \sum_{i_1=2}^{M_v} \cdots \sum_{i_p=2}^{M_v} \sum_{i=1}^{M_g} g_i \prod_{j=1}^{p} l_{i_j-i} u(t-i_j), \tag{62}$$

where $M_v = M_g + M_h$ stands for the memory of the overall system. The Wiener–Hammerstein model is associated with a Volterra model whose pth-order kernel is given by [49]

$$h^{(p)}_{i_1,\cdots,i_P} = c_p \sum_{i=1}^{Mg} g_i \prod_{j=1}^{P} l_{i_j-i} \ , \ i_j = 2,\cdots,M_v \ , \ j \in \langle p \rangle. \tag{63}$$

The corresponding tensor $\mathcal{H}^{(p)} \in \mathbb{R}^{M_v-1\times\cdots\times M_v-1}$ is a rank M_g tensor admitting a PARAFAC decomposition written as

$$\mathcal{H}^{(p)} = c_p \sum_{i=1}^{Mg} g_i \overset{p}{\underset{j=1}{\circ}} \mathbf{a}_i = c_p \mathcal{I}_{p,M_v-1} \overset{p}{\underset{j=1}{\times}} \mathbf{A}^{(j)}, \tag{64}$$

where

$$\mathbf{a}_i = \begin{bmatrix} \mathbf{0}_{i-1} \\ \mathbf{1} \\ \mathbf{0}_{M_g-i} \end{bmatrix} \in \mathbb{R}^{M_v-1} \ , \ i \in \langle M_g \rangle, \tag{65}$$

$$\mathbf{A}^{(j)} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_{M_g} \end{bmatrix} = \mathcal{T}_{M_v-1,M_g}(\mathbf{l}) = \begin{bmatrix} l_1 & 0 & \cdots & 0 \\ \vdots & l_1 & & \vdots \\ l_{M_h} & \vdots & \ddots & 0 \\ 0 & l_{M_h} & & l_1 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & l_{M_h} \end{bmatrix}, \ j \in \langle p-1 \rangle, \tag{66}$$

$$\mathbf{A}^{(p)} = \mathcal{T}_{M_v-1,M_g}(\mathbf{l}) diag(\mathbf{g}) \in \mathbb{R}^{M_v-1\times M_g}. \tag{67}$$

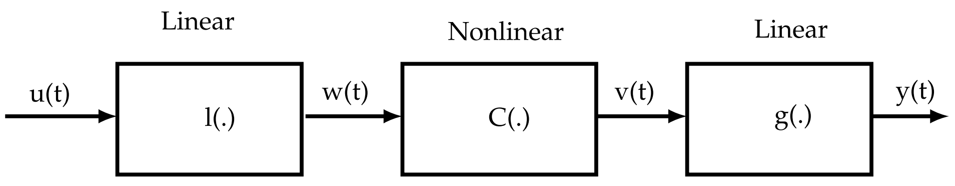

Figure 9. Block diagram of a Wiener–Hammerstein model.

5.5.4. Tensor Rank and Structure Identification

Given the Volterra model associated with a block-oriented system among the three previously considered, it was shown in [51] that the inherent structure of the system can be inferred from analyzing the tensor rank of the pth-order Volterra kernel $\mathcal{H}^{(p)}$. Indeed, from the results established in Sections 5.5.1–5.5.3, we can conclude that the tensor $\mathcal{H}^{(p)}$ has a rank less than or equal to M_g. It is precisely rank 1 for a Wiener model and diagonal for a Hammerstein one. However, the PARAFAC decomposition of $\mathcal{H}^{(p)}$ is not guaranteed to be of minimal rank. The method is based on a filtering of the system output with a FIR filter with nonzero random coefficients. The Volterra kernel of this augmented system is ensured to have minimal rank. For an order M_f FIR filter with an impulse response coefficients vector $\mathbf{f}$, the tensor corresponding to the pth order Volterra kernel of the augmented system is given by

$$\bar{\mathcal{H}}^{(p)} = c_p \mathcal{I} \overset{p}{\underset{i=1}{\times}} \bar{\mathbf{A}}^{(i)}, \tag{68}$$

where $\bar{\mathbf{A}}^{(i)} = \mathcal{T}_{\bar{M}_v, M_g}(\mathbf{l})$, $i \in \langle p-1 \rangle$, $\bar{\mathbf{A}}^{(p)} = \mathcal{T}_{\bar{M}_v, M_g}(\mathbf{l}) diag(\bar{\mathbf{g}})$, $\bar{\mathbf{g}} = \mathcal{T}_{\bar{M}_g, M_g}(\mathbf{f})\mathbf{g}$, $\bar{M}_g = M_f + M_g - 1$, and $\bar{M}_v = M_v + M_f - 1$. The factor matrices are generically full column rank. Therefore, the matrix unfolding of the tensor along the pth dimension is full rank and reflects the tensor rank. This leads to the following decision rule

$rank(\bar{\mathcal{H}}^{(p)}) = \bar{M}_v \Longrightarrow$ Hammerstein structure (Nonlinear–Linear)

$rank(\bar{\mathcal{H}}^{(p)}) = M_f \Longrightarrow$ Wiener structure (Linear–Nonlinear)

$rank(\bar{\mathcal{H}}^{(p)}) \in \left] M_f, \bar{M}_v \right[\Longrightarrow$ Wiener–Hammerstein structure (Linear–Nonlinear–Linear).

Since the factor matrices are full column rank, the tensor rank is precisely given by the rank of the pth unfolding of tensor, hereafter denoted $\bar{\mathbf{H}}_p$. However, in presence of Volterra kernel estimation errors, $\bar{\mathbf{H}}_p$ is often full column rank, which can lead to an erroneous selection of the Hammerstein structure. Thus, it is necessary to check the diagonal structure of the tensor in order to confirm the decision or not. This is performed by checking if the sum of diagonal entries of the tensor is much higher than those of off-diagonal ones. If the matrix is rank deficient, an interesting rule for computing the rank r from the singular values of the matrix is given in [87] as

$$r = \arg\min_i \rho(i), \quad \rho(i) = \frac{\sigma_{i+1}^2}{\sigma_i^2 - 2\sigma_{i+1}^2} \text{ if } \sigma_{i+1}^2 \le \frac{\sigma_i^2}{3} \text{ else } \rho(i) = 1. \tag{69}$$

The algorithm for detecting the structure of a block-oriented nonlinear system is then described in Algorithm 2:

```
Algorithm 2: Structure identification of a block-oriented nonlinear system.
Given the coefficients $h^{(p)}_{i_1,i_2,\cdots,i_p}$ of the $p$th order kernel of the Volterra model
  associated with the block-oriented nonlinear system with memory $M_v$:
1. Generate the impulse response $\mathbf{f}$ of an $M_f \geq M_v$ order FIR filter with random
   coefficients.
2. Form the tensor $\bar{\mathcal{H}}^{(p)}$ of the augmented system by filtering the Volterra kernel as
       $\bar{h}^{(p)}_{i_1,i_2,\cdots,i_p} = \sum_{i=0}^{M_f-1} f_i h^{(p)}_{i_1-i,i_2-i,\cdots,i_p-i}.$
3. Compute the singular values $\sigma_i$ of the matrix unfolding $\bar{\mathbf{H}}_p$.
4. Compute the rank $r$ of $\bar{\mathbf{H}}_p$ as the smallest integer such that
       $\sum_{i=1}^{k-1} \sigma_i < \epsilon \sum_{i=1}^{M_v+M_f-1} \sigma_i \leq \sum_{i=1}^{k} \sigma_i,$
   where $\epsilon$ is a constant close to 1.
5. If $r = \bar{M}_v = M_v + Mf - 1$, test if $\bar{\mathcal{H}}^{(p)}$ is diagonal. If yes then conclude that the
   system has a Hammerstein structure.
6. If $r < M_v + Mf - 1$
   Compute the rank $r$ using (69).
   (a) If $r = M_f$, then the system has a Wiener structure.
   (b) If $M_f < r < M_v + M_f - 1$, then the system has a Wiener–Hammerstein
       structure whose first linear block is of order $M_l = M_v + Mf - r$, while the
       second linear block is of order $M_g = r - M_f + 1$.
```

6. Tensor-Based Approaches for Multilinear Systems

In Sections 6.1 and 6.2, we introduce the notions of tensor system and memoryless discrete-time tensor-input tensor-output (TITO) system, respectively. Then, in Section 6.3, we consider a tensor-input single-output (TISO) system whose system transfer is a rank-one Nth-order tensor, which leads to a multilinear system whose the N vector factors represent IRs of subsystems associated with the N modes of the input tensor. In Section 6.4, we present the weighted least-squares (WLS) algorithm for estimating the system transfer tensor of a multilinear system from input–output (I/O) data. A closed-form solution is also proposed for estimating the individual IR of each subsystem .

6.1. Tensor Systems

In this section, we introduce the notion of tensor system using the Einstein product [55]. In Table 12, we present two examples of tensor systems: one is linear in the unknown tensor variable $\mathcal{X}$, while the other one is bilinear in the unknown tensor variables $(\mathcal{X}, \mathcal{Z})$.

Table 12. Examples of tensor systems.

Forms	Dimensions	Tensor Systems
Linear	$\mathcal{Y} \in \mathbb{R}^{\underline{I}_P}$, $\mathcal{A} \in \mathbb{R}^{\underline{I}_P \times \underline{J}_N}$, $\mathcal{X} \in \mathbb{R}^{\underline{J}_N}$	$\mathcal{Y} = \mathcal{A} \star_N \mathcal{X}$
Bilinear	$\mathcal{Y} \in \mathbb{R}^{\underline{I}_P}$, $\mathcal{A} \in \mathbb{R}^{\underline{I}_P \times \underline{K}_M \times \underline{J}_N}$, $\mathcal{X} \in \mathbb{R}^{\underline{J}_N}$, $\mathcal{Z} \in \mathbb{R}^{\underline{K}_M}$	$\mathcal{Y} = \mathcal{A} \star_N \mathcal{X} \star_M \mathcal{Z}$

Example 2. *To illustrate the notion of tensor system, let us consider the following equation:*

$$\mathbf{Y} = \mathcal{A} \star_2 \mathbf{X}. \tag{70}$$

This equation can be associated with the following map $f : \mathbb{R}^{K\times L} \ni \mathbf{X} \longmapsto f(\mathbf{X}) = \mathbf{Y} \in \mathbb{R}^{I\times J}$ *such as*

$$y_{i,j} = \sum_{k=1}^{K}\sum_{l=1}^{L} a_{i,j,k,l}x_{k,l} \tag{71}$$

with the associated fourth-order tensor $\mathcal{A} \in \mathbb{R}^{I\times J\times K\times L}$.

Equation (70) can be solved with respect to the unknown matrix $\mathbf{X}$ *by minimizing the LS criterion* $\min_{\mathbf{X}} \|\mathbf{Y} - \mathcal{A} \star_2 \mathbf{X}\|_F^2$. *This minimization is carried out after a vectorization of Equation (70) using the unfolding* $\mathbf{A}_{IJ\times KL}$ *of the tensor* $\mathcal{A}$ *and the vectorized forms* $\mathbf{x}_{KL}$ *and* $\mathbf{y}_{IJ}$ *of the matrices* $\mathbf{X}$ *and* $\mathbf{Y}$*, which leads to a standard system of linear equations in matrix form, with a coefficient matrix. The LS criterion then becomes*

$$\min_{\mathbf{x}_{KL}} \|\mathbf{y}_{IJ} - \mathbf{A}_{IJ\times KL}\,\mathbf{x}_{KL}\|_2^2. \tag{72}$$

Minimizing this criterion with respect to the unknown vector $\mathbf{x}_{KL}$ *gives the following normal equations:*

$$(\mathbf{A}_{IJ\times KL}^T\mathbf{A}_{IJ\times KL})\hat{\mathbf{x}}_{KL} = \mathbf{A}_{IJ\times KL}^T\mathbf{y}_{IJ} \tag{73}$$

$$\Downarrow \tag{74}$$

$$\hat{\mathbf{x}}_{KL} = (\mathbf{A}_{IJ\times KL}^T\mathbf{A}_{IJ\times KL})^{-1}\mathbf{A}_{IJ\times KL}^T\mathbf{y}_{IJ} \tag{75}$$

if the matrix $\mathbf{A}_{IJ\times KL}^T\mathbf{A}_{IJ\times KL}$ *is invertible, i.e., if* $\mathbf{A}_{IJ\times KL}$ *has full column rank, which implies the necessary but not sufficient condition that* $IJ \geq KL$.

Remark 3. *Let us consider the inner product of the Nth-order real tensors* $\mathcal{A} \in \mathbb{R}^{\underline{I}_N}$ *and* $\mathcal{X} \in \mathbb{R}^{\underline{I}_N}$

$$\langle \mathcal{A}, \mathcal{X}\rangle = \mathcal{A} \star_N \mathcal{X} = \sum_{i_1=1}^{I_1}\cdots\sum_{i_N=1}^{I_N} a_{i_1,\cdots i_N}x_{i_1,\cdots i_N} = a_{\underline{i}_N}x_{\underline{i}_N}. \tag{76}$$

Assuming $\mathcal{X}$ *has rank-one, i.e.,*

$$\mathcal{X} = \mathop{\circ}_{n=1}^{N} \mathbf{x}^{(n)} \iff x_{\underline{i}_N} = \prod_{n=1}^{N} x_{i_n}^{(n)}, \tag{77}$$

Equation (76) becomes

$$\begin{aligned}\mathcal{A} \star_N \mathcal{X} &= \sum_{i_1=1}^{I_1}\cdots\sum_{i_N=1}^{I_N} a_{i_1,\cdots i_N}\prod_{n=1}^{N} x_{i_n}^{(n)} \\ &= \mathcal{A} \times_1 \mathbf{x}^{(1)} \cdots \times_N \mathbf{x}^{(N)} = \mathcal{A} \mathop{\times}_{n=1}^{N} \mathbf{x}^{(n)},\end{aligned} \tag{78}$$

and we obtain a homogeneous multivariate polynomial of degree N in the components of the N vector factors $\mathbf{x}^{(n)}, n \in \langle N\rangle$.

If we assume that $\mathcal{A}$ *has also rank-one, i.e.,* $\mathcal{A} = \mathop{\circ}_{n=1}^{N} \mathbf{a}^{(n)}$, *Equation (76) can be written as*

$$\begin{aligned}\mathcal{A} \star_N \mathcal{X} &= \sum_{i_1=1}^{I_1}\cdots\sum_{i_N=1}^{I_N}\prod_{n=1}^{N} a_{i_n}^{(n)}x_{i_n}^{(n)} \\ &= \prod_{n=1}^{N}\Big(\sum_{i_n=1}^{I_n} a_{i_n}^{(n)}x_{i_n}^{(n)}\Big) = \prod_{n=1}^{N}(\mathbf{a}^{(n)})^T\mathbf{x}^{(n)}.\end{aligned} \tag{79}$$

In conclusion, when $\mathcal{A}$ has rank-one, the multivariate polynomial (78) is equal to the product of N linear forms, each linear form being an univariate polynomial in the components $x_{i_n}^{(n)}$ of the vector $\mathbf{x}^{(n)}$.

If $\mathcal{A}$ satisfies a rank-R PARAFAC decomposition (see Table 10), i.e., $\mathcal{A} = \sum\limits_{r=1}^{R} \mathop{\circ}\limits_{n=1}^{N} \mathbf{A}_{.r}^{(n)}$, with $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R}$, Equation (78) becomes

$$\mathcal{A} \star_N \mathcal{X} = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} \Big(\sum_{r=1}^{R} \prod_{n=1}^{N} a_{i_n,r}^{(n)} \Big) \prod_{n=1}^{N} x_{i_n}^{(n)} \tag{80}$$

$$= \sum_{r=1}^{R} \Big(\sum_{i_1=1}^{I_1} a_{i_1,r}^{(1)} x_{i_1}^{(1)} \Big) \cdots \Big(\sum_{i_N=1}^{I_N} a_{i_N,r}^{(N)} x_{i_N}^{(N)} \Big)$$

$$= \sum_{r=1}^{R} \big((\mathbf{A}_{.r}^{(1)})^T \mathbf{x}^{(1)} \big) \cdots \big((\mathbf{A}_{.r}^{(N)})^T \mathbf{x}^{(N)} \big). \tag{81}$$

In this case, we obtain a sum of products of N linear forms, to be compared with the Volterra–PARAFAC model (19) which corresponds to the particular case where $\mathbf{x}^{(n)} = \mathbf{u}(t)$ and $\mathbf{A}_{.r}^{(n)} = \mathbf{A}_{.r}^{(p)}$, for $n \in \langle N \rangle$. This last constraint results from the assumption of symmetry of the Volterra kernel.

6.2. Discrete-Time Memoryless Tensor-Input Tensor-Output Systems

Assuming system input and model output data are contained in two tensors $\mathcal{X}(t) \in \mathbb{R}^{\underline{J}_N}$ and $\hat{\mathcal{Y}}(t) \in \mathbb{R}^{\underline{I}_P}$ which depend on time $t \in [1, T]$, we define a discrete-time memoryless tensor-input tensor-output (TITO) model by means of the following I/O relationship:

$$\hat{\mathcal{Y}}(t) = \mathcal{A} \star_N \mathcal{X}(t), \tag{82}$$

where $\mathcal{X}(t)$ and $\hat{\mathcal{Y}}(t)$ are the tensors of system input and model output signals of the TITO system, at the time instant t, and $\mathcal{A} \in \mathbb{R}^{\underline{I}_P \times \underline{J}_N}$ is the system transfer tensor. Using the index convention, Equation (82) can be written in scalar form as

$$\hat{y}_{\underline{i}_P}(t) \triangleq \hat{y}_{i_1,\cdots,i_P}(t) = a_{\underline{i}_P,\underline{j}_N} x_{\underline{j}_N}(t). \tag{83}$$

This equation is associated with the following map:

$$\mathbb{R}^{\underline{J}_N} \ni \mathcal{X}(t) \longmapsto f(\mathcal{X}(t)) = \hat{\mathcal{Y}}(t) \in \mathbb{R}^{\underline{I}_P} \tag{84}$$

with the associated $(P+N)$th-order tensor $\mathcal{A} \in \mathbb{R}^{\underline{I}_P \times \underline{J}_N}$.

Considering measurements of I/O signals during the time interval T, the sets of input and output signals are concatenated along the time mode to form the matrix unfoldings $\mathbf{X}_{\Pi J_N \times T} \in \mathbb{R}^{\Pi J_N \times T}$ and $\hat{\mathbf{Y}}_{\Pi I_P \times T} \in \mathbb{R}^{\Pi I_P \times T}$ of the tensors $\mathcal{X}(T) \in \mathbb{R}^{\underline{J}_N \times T}$ and $\hat{\mathcal{Y}}(T) \in \mathbb{R}^{\underline{I}_P \times T}$, respectively, with $\Pi\, I_P$ and $\Pi\, J_N$ defined as in Table 3. The I/O relationship of the TITO model can then be written in the following matrix form:

$$\hat{\mathbf{Y}}_{\Pi I_P \times T} = \mathbf{A}_{\Pi I_P \times \Pi J_N} \mathbf{X}_{\Pi J_N \times T}. \tag{85}$$

Let us assume the model output (83) is corrupted by a zero-mean additive white Gaussian noise (AWGN) $e_{\underline{i}_P}(t)$ such as the measured noisy output signal is given by

$$y_{\underline{i}_P}(t) = \hat{y}_{\underline{i}_P}(t) + e_{\underline{i}_P}(t) = a_{\underline{i}_P,\underline{j}_N} x_{\underline{j}_N}(t) + e_{\underline{i}_P}(t). \tag{86}$$

Transposing both members of Equation (85) gives

$$\mathbf{Y}_{T \times \Pi I_P} = \mathbf{X}_{T \times \Pi J_N} \mathbf{A}_{\Pi J_N \times \Pi I_P} + \mathbf{E}_{T \times \Pi I_P}. \tag{87}$$

From this equation, it is easy to derive the LS estimate of the matrix unfolding $\mathbf{A}_{\Pi J_N \times \Pi I_P}$ of the system transfer tensor, which minimizes the least mean square error between the model outputs and the noisy system output measurements

$$\min_{\mathbf{A}_{\Pi J_N \times \Pi I_P}} \Big[\|\mathbf{E}_{T\times \Pi I_P}\|_F^2 = \sum_{i_1=1}^{I_1} \cdots \sum_{i_P=1}^{I_P} e^2_{\underline{i}_P}(t) \Big] = \min_{\mathbf{A}_{\Pi J_N \times \Pi I_P}} \|\mathbf{Y}_{T\times \Pi I_P} - \mathbf{X}_{T\times \Pi J_N} \mathbf{A}_{\Pi J_N \times \Pi I_P}\|_F^2$$
$$\Downarrow$$
$$\hat{\mathbf{A}}_{\Pi J_N \times \Pi I_P} = [\mathbf{X}_{T\times \Pi J_N}]^\dagger \mathbf{Y}_{T\times \Pi I_P}. \tag{88}$$

To ensure uniqueness of this LS solution, it is necessary that $\mathbf{X}_{T\times \Pi J_N}$ be full column rank, which implies the necessary condition $T \geq \Pi J_N$; i.e., the number T of input–output samples must be greater or equal to the number of input signal samples at each time instant t.

6.3. Multilinear TISO Systems

In the case of a memoryless tensor-input single-output (TISO) system, let us assume that the system transfer is a rank-one tensor $\mathcal{A} \in \mathbb{R}^{\underline{J}_N}$ written as

$$\mathcal{A} = \mathop{\circ}_{n=1}^{N} \mathbf{h}^{(n)} \iff a_{\underline{j}_N} = \prod_{n=1}^{N} h^{(n)}_{j_n}, \tag{89}$$

with $\mathbf{h}^{(n)} \in \mathbb{R}^{J_n}$, for $n \in \langle N \rangle$. The model output (82) is then given by

$$\begin{aligned}
\hat{y}(t) =& (\mathop{\circ}_{n=1}^{N} \mathbf{h}^{(n)}) \star_N \mathcal{X}(t) \\
=& \sum_{j_1=1}^{J_1} \cdots \sum_{j_N=1}^{J_N} \Big(\prod_{n=1}^{N} h^{(n)}_{j_n}\Big) x_{\underline{j}_N}(t) \\
=& \Big(\sum_{j_1=1}^{J_1} h^{(1)}_{j_1} x_{j_1,\cdots,j_N}(t)\Big) \cdots \Big(\sum_{j_N=1}^{J_N} h^{(N)}_{j_N} x_{j_1,\cdots,j_N}(t)\Big)
\end{aligned} \tag{90}$$

or using the index convention

$$\hat{y}(t) = \prod_{n=1}^{N} \Big(h^{(n)}_{j_n} x_{\underline{j}_N}(t) \Big) \tag{91}$$

or equivalently

$$\begin{aligned}
\hat{y}(t) =& \Big(\mathcal{X}(t) \times_1 (\mathbf{h}^{(1)})^T\Big) \cdots \Big(\mathcal{X}(t) \times_N (\mathbf{h}^{(N)})^T\Big) \\
=& \mathcal{X}(t) \mathop{\times}_{n=1}^{N} (\mathbf{h}^{(n)})^T.
\end{aligned} \tag{92}$$

The resulting system output is multilinear (N-linear) with respect to the vector factors $\mathbf{h}^{(n)}$. Each vector can be interpreted as the impulse response (IR) of length J_n of the subsystem associated to the nth mode of the input tensor $\mathcal{X}(t)$. The output signal $\hat{y}(t)$ is therefore a multilinear form in the N individual IR vectors (see Table 7).

6.4. Estimation of the System Transfer Tensor from I/O Data

Let us define the lexicographical vectorization $\mathbf{u}(t) \triangleq \mathrm{vec}(\mathcal{X}(t)) \in \mathbb{R}^{\Pi J_N}$ such as

$$u_{\overline{j_1 \cdots j_N}}(t) = x_{j_1,\cdots,j_N}(t), \tag{93}$$

and the (multilinear) global impulse response (GIR) $\mathbf{h}$ as the vectorized form of the system transfer tensor $\mathcal{A}$

$$\mathbf{h} = \mathrm{vec}(\overset{N}{\underset{n=1}{\circ}} \mathbf{h}^{(n)}) = \overset{N}{\underset{n=1}{\otimes}} \mathbf{h}^{(n)} = \mathbf{h}^{(1)} \otimes \mathbf{h}^{(2)} \otimes \cdots \otimes \mathbf{h}^{(N)} \in \mathbb{R}^{\Pi J_N}. \tag{94}$$

The output of the multilinear model can then be rewritten as

$$\hat{y}(t) = \mathbf{u}^T(t)\,\mathbf{h}. \tag{95}$$

Considering noisy output measurements on the time interval $[1, T]$, the noisy output vector $\mathbf{y}(T) \in \mathbb{R}^T$ is given by

$$\mathbf{y}(T) \triangleq \begin{bmatrix} y(1) \\ \vdots \\ y(T) \end{bmatrix} = \begin{bmatrix} \mathbf{u}^T(1) \\ \vdots \\ \mathbf{u}^T(T) \end{bmatrix} \mathbf{h} + \begin{bmatrix} e(1) \\ \vdots \\ e(T) \end{bmatrix} \triangleq \mathbf{U}(T)\,\mathbf{h} + \mathbf{e}(T), \tag{96}$$

where $e(t)$ is a zero-mean AWGN, at the time instant t, and $\mathbf{U}(T) \in \mathbb{R}^{T \times \Pi J_N}$.

We now determine the weighted least-squares (WLS) estimate of the vectorized form $\mathbf{h}$ of the GIR tensor, which minimizes the following cost funtion $\min_{\mathbf{h}} \|\mathbf{e}(T)\|^2_{\mathbf{W}}$, with

$$\|\mathbf{e}(T)\|^2_{\mathbf{W}} \triangleq \mathbf{e}^T(T)\mathbf{W}\mathbf{e}(T) = \sum_{t=1}^{T} w_t\, e^2(t) = \|\mathbf{y}(T) - \mathbf{U}(T)\mathbf{h}\|^2_{\mathbf{W}}, \tag{97}$$

where $\mathbf{W} \triangleq \mathrm{diag}(w_1, \cdots, w_T)$ is a diagonal weighting matrix, with $w_t > 0$ for all $t \in [1, T]$. The WLS criterion (97) can be developed as

$$\|\mathbf{e}(T)\|^2_{\mathbf{W}} = \|\mathbf{y}(T)\|^2_{\mathbf{W}} - 2\mathbf{h}^T\mathbf{U}^T(T)\mathbf{W}\mathbf{y}(T) + \mathbf{h}^T\mathbf{U}^T(T)\mathbf{W}\mathbf{U}(T)\mathbf{h}. \tag{98}$$

It is a quadratic cost function with respect to the unknown parameters vector $\mathbf{h}$. The Hessian $(2\mathbf{U}^T(T)\mathbf{W}\mathbf{U}(T))$ being a nonnegative definite matrix, this criterion has a unique global minimum obtained in canceling its gradient with respect to $\mathbf{h}$, which gives

$$\Big(\mathbf{U}^T(T)\mathbf{W}\mathbf{U}(T)\Big)\hat{\mathbf{h}}(T) = \mathbf{U}^T(T)\mathbf{W}\mathbf{y}(T). \tag{99}$$

Assuming the matrix $\mathbf{U}^T(T)\mathbf{W}\mathbf{U}(T)$ is nonsingular, the WLS estimate of $\mathbf{h}$ is given by

$$\hat{\mathbf{h}}(T) = \Big(\mathbf{U}^T(T)\mathbf{W}\mathbf{U}(T)\Big)^{-1}\mathbf{U}^T(T)\mathbf{W}\mathbf{y}(T). \tag{100}$$

As the diagonal weighting matrix $\mathbf{W}$ is positive definite, a condition for ensuring the uniqueness of the WLS estimate is that $\mathbf{U}(T)$ be full column rank, which implies the necessary condition $T \geq \Pi J_N$. When the weighting matrix is chosen as the identity matrix, we obtain the standard LS estimate of the GIR given by

$$\hat{\mathbf{h}}(T) = \mathbf{U}^{\dagger}(T)\mathbf{y}(T). \tag{101}$$

In [56], an iterative Wiener filter and LMS-based algorithms are proposed to identify multilinear systems as described in (91).

Tensorizing the GIR vector estimate $\hat{\mathbf{h}}$ as a Nth-order rank-one tensor $\hat{\mathcal{H}} \in \mathbb{R}^{\underline{J}_N}$, an estimate $\hat{\mathbf{h}}^{(n)}$ of each individual IR $\mathbf{h}^{(n)}$ can be obtained by using the high order singular value decomposition (HOSVD) of $\hat{\mathcal{H}}$, i.e., calculating the left singular vector associated with the largest singular value of the matrix unfolding $\hat{\mathbf{H}}_{J_n \times J_1 \cdots J_{n-1} J_{n+1} \cdots J_N}$. For more details concerning the HOSVD-based estimation of matrix or vector factors of a multiple Kronecker product, the reader is referred to the following references [70,81]. Uniqueness of individual estimates $\hat{\mathbf{h}}^{(n)}$ is ensured assuming the first coefficient $h_1^{(n)} = 1$ for $n \in \langle N \rangle$.

7. Conclusions and Perspectives

The aim of this paper is to outline links between tensors and nonlinear and multilinear systems. In the case of NL systems, a focus has been made on Volterra models, with the objective of parametric complexity reduction using a PARAFAC decomposition of symmetrized kernels or their expansion on generalized orthogonal basis functions. The EKF algorithm has been proposed to estimate the parameters of a Volterra–PARAFAC model. Then, three block-oriented nonlinear systems have been represented by means of associated Volterra models in the form of a structured tensor decomposition. It has been shown how this equivalent tensor representation can be exploited to identify the structure of a block-oriented system. This tensor representation can also be used for parameter estimation of a block-oriented system. As perspectives of these results, it would be interesting to compare the different NL models considered, both in terms of parametric complexity and quality of modeling via parameter estimation for a given benchmark.

For multilinear systems, a new class of systems called tensor-input tensor-output (TITO) systems is introduced using Einstein product of tensors. The case of a TISO system has been studied in more detail assuming that the transfer tensor has rank one. The WLS algorithm has been derived for estimating the multilinear global impulse response (GIR) associated with the vectorized form of the system transfer tensor. A closed-form HOSVD-based solution has been proposed to estimate the individual impulse response of each subsystem from the estimated GIR. Another line of research will be to consider a sparse input data tensor modeled using different tensor models and apply tensor completion methods to reconstruct missing data.

Supplementary Materials: The following are available online at https://www.mdpi.com/article/10.3390/a16090443/s1, Matlab code for simulations results in Figures 2–5.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vasilescu, M.A.O.; Terzopoulos, D. Multilinear analysis of image ensembles: TensorFaces. In Proceedings of the European Conference on Computer Vision (ECCV 2002), Copenhagen, Denmark, 28–31 May 2002; pp. 447–460.
2. Lu, H.; Plataniotis, K.N.; Venetsanopoulos, A.N. MPCA: Multilinear principal component analysis of tensor objects. *IEEE Trans. Neural Netw.* **2008**, *19*, 18–39.
3. Raimondi, F.; Cabral Farias, R.; Michel, O.; Comon, P. Wideband multiple diversity tensor array processing. *IEEE Trans. Signal Process.* **2017**, *65*, 5334–5346. [CrossRef]
4. Ji, Y.; Wang, Q.; Li, X.; Liu, J. A Survey on tensor techniques and applications in machine learning. *IEEE Access* **2019**, *7*, 162950. [CrossRef]
5. Frolov, E.; Oseledets, I. Tensor methods and recommender systems. *WIREs Data Mining Knowl. Discov.* **2017**, *7*, e1201. [CrossRef]
6. Padhy, S.; Goovaerts, G.; Boussé, M.; De Lathauwer, L.; Van Huffel, S. The Power of Tensor-Based Approaches in Cardiac Applications, In *Biomedical Signal Processing. Advances in Theory, Algorithms and Applications*; Naik, G., Ed.; Springer: Singapore, 2019.
7. Wang, R.; Li, S.; Cheng, L.; Wong, M.H.; Leung, K.S. Predicting associations among drugs, targets and diseases by tensor decomposition for drug repositioning. *BMC Bioinform.* **2019**, *26*, 628. [CrossRef] [PubMed]
8. Favier, G.; Sousa Rocha, D. Overview of tensor-based cooperative MIMO communication systems— Part 1: Tensor modeling. *MDPI Entropy* **2023**, *25*, 1181. [CrossRef]
9. Cichocki, A. Era of big data processing: A new approach via tensor networks and tensor decompositions. *arXiv* **2014**, arXiv:1403.2048v4.
10. Hitchcock, F.L. The expression of a tensor or a polyadic as a sum of products. *J. Math. Phys.* **1927**, *6*, 164–189. [CrossRef]
11. Cattell, R. Parallel proportional profiles and other principles for determining the choice of factors by rotation. *Psychometrika* **1944**, *9*, 267–283. [CrossRef]
12. Tucker, L.R. Some mathematical notes on three-mode factor analysis. *Psychometrika* **1966**, *31*, 279–311. [CrossRef] [PubMed]

13. Harshman, R.A. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. *UCLA Work. Pap. Phon.* **1970**, *16*, 1–84.
14. Bro, R. PARAFAC. Tutorial and applications. *Chemom. Intell. Lab. Syst.* **1997**, *38*, 149–171. [CrossRef]
15. Morup, M. Applications of tensor (multiway array) factorizations and decompositions in data mining. *WIREs Data Min. Knowl. Discov.* **2011**, *1*, 20–40. [CrossRef]
16. Cichocki, A.; Lee, N.; Oseledets, I.; Phan, A.H.; Zhao, Q.; Mandic, D.P. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 Low-rank tensor decompositions. *Found. Trends Mach. Learn.* **2016**, *9*, 249–429. [CrossRef]
17. Acar, E.; Bro, R.; Smilde, A. Data fusion in metabolomics using coupled matrix and tensor factorizations. *Proc. IEEE* **2015**, *103*, 1602–1620. [CrossRef]
18. Gandy, S.; Recht, B.; Yamada, I. Tensor completion and low-n-rank tensor recovery via convex optimization. *Inverse Probl.* **2011**, *27*, 025010. [CrossRef]
19. Liu, J.; Musialski, P.; Wonka, P.; Ye, J. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 208–220. [CrossRef]
20. Favier, G. *From Algebraic Structures to Tensors*; Wiley: Hoboken, NJ, USA, 2019; Volume 1.
21. Legendre, A.M. *Appendice: Sur la Méthode des Moindres Quarrés, in "Nouvelles Méthodes Pour la Détermination des Orbites des Comètes"*; Firmin-Didot: Paris, France, 1805; pp. 72–80.
22. Kalman, R.E. A new approach to linear filtering and prediction problems. *Trans. ASME J. Basic Eng.* **1960**, *82D*, 34–45. [CrossRef]
23. Jazwinski, A.H. *Stochastic Processes and Filtering Theory*; Academic Press: Cambridge, MA, USA, 1970.
24. Rogers, M.; Li, L.; Russell, S.J. Multilinear dynamical systems for tensor time series. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; pp. 2634–2642.
25. Chen, C.; Surana, A.; Bloch, A.; Rajapakse, I. Multilinear control systems theory. *SIAM J. Control Optim.* **2021**, *5*, 749–776. [CrossRef]
26. Boyd, S.; Chua, L. Fading memory and the problem of approximating nonlinear operators with Volterra series. *IEEE Trans. Circuits Syst.* **1985**, *32*, 1150–1161. [CrossRef]
27. Schetzen, M. *The Volterra and Wiener Theories of Nonlinear Systems*; John Wiley & Sons: New York, NY, USA, 1980.
28. Mathews, V.; Sicuranza, G. *Polynomial Signal Processing*; John Wiley & Sons: New York, NY, USA, 2000.
29. Doyle III, F.; Pearson, R.; Ogunnaike, B. *Identification and Control Using Volterra Models*; Springer: Berlin/Heidelberg, Germany, 2002.
30. Fernando, X.N.; Sesay, A.B. Adaptive asymmetric linearization of radio over fiber links for wireless access. *IEEE Trans. Veh. Technol.* **2002**, *51*, 1576–1586. [CrossRef]
31. He, J.; Lee, J.; Kandeepan, S.; Wang, K. Machine learning techniques in radio-over-fiber systems and networks. *Photonics* **2020**, *7*, 105. [CrossRef]
32. Benedetto, S.; Biglieri, E.; Daffara, S. Modeling and peformance evaluation of nonlinear satellite links—A Volterra series approach. *IEEE Trans. Aerosp. Electron. Syst.* **1979**, *AES-15*, 494–507. [CrossRef]
33. Cheng, C.H.; Powers, E.J. Optimal Volterra kernel estimation algorithms for a nonlinear communication system for PSK and QAM inputs. *IEEE Trans. Signal Process.* **2001**, *49*, 147–163. [CrossRef]
34. Marmarelis, V. *Nonlinear Dynamic Modeling of Physiological Systems*; Wiley-IEEE Press: Hoboken, NJ, USA, 2004.
35. Kerschen, G.; Worden, K.; Vakakis, A.; Golinval, J. Past, present and future of nonlinear system identification in structural dynamics. *Mech. Syst. Signal Process.* **2006**, *20*, 505–592. [CrossRef]
36. Azpicueta, L.; Zeller, M.; Figueiras-Vidal, A.; Kellerman, W.; Arenas-Garcia, J. Enhanced adaptive Volterra filtering by automatic attenuation of memory regions and its application to acoustic echo cancellation. *IEEE Trans. Signal Process.* **2013**, *61*, 2745–2750. [CrossRef]
37. Campello, R.J.; Favier, G.; Amaral, W.C. Optimal expansions of discrete-time Volterra models using Laguerre functions. *Automatica* **2004**, *42*, 815–822. [CrossRef]
38. Kibangou, A.; Favier, G.; Hassani, M.M. Selection of generalized orthonormal bases for second order Volterra filters. *Signal Process.* **2005**, *85*, 2371–2385. [CrossRef]
39. da Rosa, A.; Campello, R.; Amaral, W. Choice of free parameters in expansions of discrete-time Volterra models using Kautz functions. *Automatica* **2007**, *43*, 1084–1091. [CrossRef]
40. Favier, G.; Bouilloc, T. Identification de modèles de Volterra basée sur la décomposition PARAFAC de leurs noyaux et le filtre de Kalman etendu. *Traitement du Signal* **2010**, *27*, 27–51.
41. Favier, G.; Kibangou, A.; Bouilloc, T. Nonlinear system modeling and identification using Volterra–PARAFAC models. *Int. J. Adapt. Control Signal Process.* **2012**, *26*, 30–53. [CrossRef]
42. Batselier, K.; Chen, Z.; Wong, N. Tensor network alternating linear scheme for MIMO Volterra system identification. *Automatica* **2017**, *84*, 26–35. [CrossRef]
43. Crespo-Cadenas, C.; Reina-Tosina, J.; Madero-Ayora, M.J.; Muñoz-Cruzato, J. A new approach to pruning Volterra models for power amplifiers. *IEEE Trans. Signal Process.* **2010**, *58*, 2113–2120. [CrossRef]
44. Hunter, I.W.; Korenberg, M.J. The identification of nonlinear biological systems: Wiener and Hammerstein cascade models. *Biol. Cybern.* **1986**, *55*, 135–144. [CrossRef]
45. Giri, F.; Bai, E.W. *Block-Oriented Nonlinear System Identification*; LNCIS; Springer: London, UK, 2010; Volume 404.

46. Pearson, R.K.; Pottmann, M. Gray-box identification of block-oriented nonlinear models. *J. Process. Control* **2000**, *10*, 301–315. [CrossRef]
47. Schoukens, J.; Tiels, K. Identification of block-oriented nonlinear systems starting from linear approximations: A survey. *Automatica* **2017**, *85*, 272–292. [CrossRef]
48. Favier, G. Nonlinear system modeling and identification using tensor approaches. In Proceedings of the 10th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA'2009), Hammamet, Tunisia, 20–22 December 2009.
49. Kibangou, A.; Favier, G. Wiener-Hammerstein systems modeling using diagonal Volterra kernels coefficients. *IEEE Signal Process. Lett.* **2006**, *13*, 381–384. [CrossRef]
50. Kibangou, A.; Favier, G. Identification of parallel-cascade Wiener systems using joint diagonalization of third-order Volterra kernel slices. *IEEE Signal Process. Lett.* **2009**, *16*, 188–191. [CrossRef]
51. Kibangou, A.; Favier, G. Tensor analysis-based model structure determination and parameter estimation for block-oriented nonlinear systems. *IEEE J. Sel. Top. Signal Process. Spec. Issue Model Order Sel. Signal Process. Syst.* **2010**, *4*, 514–525. [CrossRef]
52. Tseng, C.; Powers, E. Identification of cubic systems using higher order moments of i.i.d. signals. *IEEE Trans. Signal Process.* **1995**, *43*, 1733–1735. [CrossRef]
53. Kibangou, A.; Favier, G. Identification of fifth-order Volterra systems using i.i.d. inputs. *IET Signal Process.* **2010**, *4*, 30–44. [CrossRef]
54. Kibangou, A.; Favier, G. Matrix and tensor decompositions for identification of block-structured nonlinear channels in digital transmission systems. In Proceedings of the IEEE 9th Worshop on Signal Processing Advances in Wireless Communications (SPAWC), Recife, Brazil, 6–9 July 2008.
55. Brazell, M.; Li, N.; Navasca, C.; Tamon, C. Solving multilinear systems via tensor inversion. *SIAM J. Matrix Anal. Appl.* **2013**, *34*, 542–570. [CrossRef]
56. Dogariu, L.M.; Paleologu, C.; Benesty, J.; Ciochina, S. Identification of Multilinear Systems: A Brief Overview. In *Principal Component Analysis*; IntechOpen: London, UK, 2022.
57. Sage, A.P.; Melsa, J.L. *System Identification*; Academic Press: Cambridge, MA, USA, 1971.
58. Söderström, T.; Stoica, P. *System Identification*; Prentice-Hall, Englewood Cliffs, NJ, USA, 1989.
59. Eykhoff, P. *System Identification. Parameter and State Estimation*; John Wiley & Sons: Hoboken, NJ, USA, 1974.
60. Goodwin, G.; Payne, R. *Dynamic system Identification: Experiment Design and Data Analysis*; Academic Press: Cambridge, MA, USA, 1977.
61. Norton, J. *An introduction to Identification*; Academic Press: Cambridge, MA, USA, 1986.
62. Ljung, L. *System Identification: Theory for the User*; Prentice-Hall: Hoboken, NJ, USA, 1987.
63. Heuberger, P.; Van den Hof, P.; Wahlberg, B. *Modelling and Identification with Rational Orthogonal Basis Functions*; Springer: Berlin/Heidelberg, Germany, 2005.
64. Billings, S.A. Identification of nonlinear systems—A survey. *IEE Proc.* **1980**, *127*, 272–285. [CrossRef]
65. Rugh, W.J. *Nonlinear System Theory. The Volterra-Wiener Approach*; Johns Hopkins University Press: Baltimore, MD, USA, 1981.
66. Haber, R.; Keviczky, L. *Nonlinear System Identification. Input-Ouput Modeling Approach. Vol. 1: Nonlinear System Parameter Identification*; Kluwer Academic Publishers: New York, NY, USA, 1999.
67. Giannakis, G.; Serpedin, E. A bibliography on nonlinear system identification. *Signal Process.* **2001**, *81*, 533–580. [CrossRef]
68. Nelles, O. *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*; Springer: Berlin/Heidelberg, Germany, 2001.
69. Schoukens, J.; Ljung, L. Nonlinear system identification. A user-oriented road map. *IEEE Control Syst. Mag.* **2019**, *39*, 28–99. [CrossRef]
70. Favier, G. *Matrix and Tensor Decompositions in Signal Processing. Vol. 2*; Wiley: Hoboken, NJ, USA, 2022.
71. Favier, G.; de Almeida, A.L.F. Overview of constrained PARAFAC models. *EURASIP J. Adv. Signal Process.* **2014**, *5*, 1–25. [CrossRef]
72. Ragnarsson, S.; Van Loan, C. Block tensors and symmetric embeddings. *Linear Algebra Its Appl.* **2013**, *438*, 853–874. [CrossRef]
73. Cichocki, A.; Mandic, D.; De Lathauwer, L.; Zhou, G.; Zhao, Q.; Caiafa, C. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Trans. Signal Process.* **2015**, *32*, 145–163. [CrossRef]
74. Sidiropoulos, N.D.; de Lathauwer, L.; Fu, X.; Huang, K.; Papalexakis, E.; Faloutsos, C. Tensor decomposition for signal processing and machine learning. *IEEE Trans. Signal Process.* **2017**, *65*, 3551–3582. [CrossRef]
75. Carroll, J.D.; Chang, J. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika* **1970**, *35*, 283–319. [CrossRef]
76. Kiers, H.A.L. Towards a standardized notation and terminology in multiway analysis. *J. Chemom.* **2000**, *14*, 105–122. [CrossRef]
77. Comon, P.; Golub, G.; Lim, L.H.; Mourrain, B. Symmetric tensors and symmetric tensor rank. *SIAM J. Matrix Anal. Appl.* **2008**, *30*, 1254–1279. [CrossRef]
78. Kruskal, J.B. Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra Its Appl.* **1977**, *18*, 95–138. [CrossRef]
79. Sidiropoulos, N.D.; Bro, R. On the uniqueness of multilinear decomposition of N-way arrays. *J. Chemom.* **2000**, *14*, 229–239. [CrossRef]

80. Ten Berge, J.M.F.; Smilde, A.K. Non-triviality and identification of a constrained Tucker3 analysis. *J. Chemom.* **2002**, *16*, 609–612. [CrossRef]
81. De Lathauwer, L.; de Moor, B.; Vandewalle, J. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.* **2000**, *21*, 1253–1278. [CrossRef]
82. Smilde, A.K.; Bro, R.; Geladi, P. *Multi-Way Analysis. Applications in the Chemical Sciences*; Wiley: Chichester, England, 2004.
83. Leontaritis, I.J.; Billings, S.A. Input-output parametric models for non-linear systems. *Int. J. Control* **1985**, *41*, 303–344. [CrossRef]
84. Comon, P.; Mourrain, B. Decomposition of quantics in sums of power of linear forms. *Signal Process.* **1996**, *53*, 93–107. [CrossRef]
85. Korenberg, M. Parallel cascade identification and kernel estimation for nonlinear systems. *Ann. Biomed. Eng.* **1991**, *19*, 429–455. [CrossRef]
86. Ninness, B.; Gustafsson, F. A unifying construction of orthonormal bases for system identification. *IEEE Trans. Autom. Control* **1997**, *42*, 515–521. [CrossRef]
87. Liavas, A.; Regalia, P.; Delmas, J. Blind channel approximation: Effective channel order determination. *IEEE Trans. Signal Process.* **1999**, *47*, 3336–3344. [CrossRef]