Volterra kernels of bilinear systems have tensor train structure

José Henrique de Morais Goulart
IRIT, Université de Toulouse, CNRS, Toulouse INP
Toulouse, France
henrique.goulart@irit.fr

Phillip M. S. Burt
Escola Politécnica, Universidade de São Paulo
São Paulo, Brazil
phillip@lcs.poli.usp.br

2021 29th European Signal Processing Conference (EUSIPCO) | 978-9-0827-9706-0/21/$31.00 ©2021IEEE | DOI: 10.23919/EUSIPCO54536.2021.9616311

***Abstract*—Despite being able to approximate the outputs of a wide class of (weakly) nonlinear dynamical systems, the finite-memory discrete-time Volterra models known as Volterra filters (VF) are notoriously too heavy from a computational point of view, due to the often huge number of parameters needed to fully describe their kernels. This shortcoming has prompted the development of alternative, low-complexity approximate models, among which low-rank tensor-based approaches figure prominently. In this work, we argue that for bilinear (or more generally, linear-analytic) systems, the Volterra kernels in the so-called regular form are naturally structured in the form of a tensor-train decomposition, a property that can be easily exploited for achieving complexity reduction. We compare this proposed approach with other existing tensor-based ones in the case where state-space equations are known but typically hard and/or too costly to realize in discrete-time, which motivates the use of low-complexity discrete-time nonlinear filters. Our numerical results illustrate the benefits of our proposal in an example involving a nonlinear loudspeaker of known state-space equations.**

***Index Terms*—nonlinear system, Volterra model, tensor decomposition.**

I. Introduction

The mathematical modeling of dynamical systems is a cornerstone of modern science and engineering. In many cases, such systems exhibit significantly nonlinear behavior that needs to be accurately modeled in order to meet application-specific goals. To accomplish this task, a practitioner can often choose among several kinds of nonlinear models, each with its own strengths and drawbacks.

Among them, the Volterra model applies to a wide class of mildly nonlinear systems—namely, those with fading memory, as shown by the seminal work of Boyd and Chua [1]. When used for simulating or predicting system outputs, the discrete-time finite-memory Volterra model (also called Volterra filter) is conceptually simple to implement and stable by construction, while still accurate provided that its memory (in samples) is sufficiently long.

The price to pay, however, is the typically high parametric complexity of the model, especially when its degree is moderately high and/or its memory is long [2]. A line of research has thus been dedicated to finding ways of alleviating this shortcoming by proposing alternative, more sophisticated models that are built on the Volterra model and allow trading some precision for a significant reduction in complexity. Examples include the sparsification of Volterra kernels, possibly coupled with an interpolation scheme [3], [4], and the use of low-rank tensor decompositions to approximate the kernels [5], [6].

It was shown in previous works that this tensor-based approach can bring a dramatic reduction of complexity whilst only incurring a small loss of precision [2], [6]. This is in particular achieved by a low-rank approximate canonical polyadic decomposition (CPD) of the symmetric kernels that characterize a nonlinear loudspeaker model, which leads to a realization consisting of a filterbank followed by simple nonlinearities [5]. Although this idea can yield remarkable results, computing an approximate CPD is hard and involves pitfalls such as the possible non-existence of best approximations [7]. As an alternative, the recent work [6] has opted for the tensor train (TT) decomposition [8], whereby kernel elements $h_{n_1,\ldots,n_p}$ are written as a chain of products of the form $\mathbf{a}_1(n_1)^\mathsf{T}\mathbf{A}_2(n_2)\ldots\mathbf{A}_{p-1}(n_{p-1})\mathbf{a}_p(n_p)$, where the dimensions of matrices $\mathbf{A}_i$ (known as TT-ranks) are small, possibly bringing a high reduction in parametric complexity with little loss of precision. Moreover, TT approximation does not suffer from the same intricacies as with the CPD, and can in fact be performed by a sequence of truncated SVDs [8].

Here we take a different direction, choosing to trade some generality (rather than precision) for a reduction of complexity. Specifically, we restrict ourselves to Volterra kernels of bilinear systems (and thus of linear-analytic systems [9], as explained below), and show that in the so-called regular form these kernels already have a natural exact TT structure with the same matrices $\mathbf{A}_i$ for all kernels, and that their TT-ranks are a function of the state dimension. Finally, this TT structure is shown to lead to a cascade realization with much smaller computational complexity than a Volterra filter (VF), as required.

In fact, this approach applies more broadly to linear-analytic system by virtue of the Carleman bilinearization procedure, which allows computing the exact Volterra kernels of any such a system by first constructing a bilinear system having the same kernels up to a chosen degree P [9]. As we will see, the TT-ranks are in this case a function of the dimension of the linear-analytic system's state but also of P. Several important real-world devices, such as loudspeakers [10], RF power transistors [11] and underwater acoustic transducers [12] can be modeled as linear-analytic systems. Often this arises from nonlinear reactances or the reactive coupling of instantaneous non-linearities.

For illustration, we consider a scenario where a discrete-time model for an automotive loudspeaker of known (linear-analytic) state-space equations [10] is required. As a direct discretization is typically hard or too costly for practical use, several alternative models are derived using the Volterra kernels computed from these equations. Our results highlight the benefits of exploiting the natural TT structure described above, as opposed to employing the approaches of [5], [6].

In the following, we assume that the reader is familiar with basic notions pertaining to tensors, as reviewed *e.g.* by [13].

II. The tensor train decomposition

The tensor train (TT) decomposition, whose name alludes to the chain of products that describes its elements, was introduced in [8] with the goal of allowing a low-dimensional representation of a large tensor whose storage would otherwise be prohibitively unwieldy. Furthermore, it can reduce the complexity of performing certain operations involving that tensor. Concretely, the TT decomposition of an $N_1 \times \cdots \times N_p$ tensor $\mathcal{H}$ expresses its elements $h_{n_1,\ldots,n_p}$ as

$$h_{n_1,\ldots,n_p} = \mathbf{a}_1(n_1)^\mathsf{T} \mathbf{A}_2(n_2) \ldots \mathbf{A}_{p-1}(n_{p-1}) \mathbf{a}_p(n_p), \quad (1)$$

where vectors $\mathbf{a}_1(n_1)$ and $\mathbf{a}_p(n_p)$ are respectively the n_1th and n_pth columns of matrices $\mathbf{A}_1 \in \mathbb{R}^{R_1 \times N_1}$ and $\mathbf{A}_p \in \mathbb{R}^{R_{p-1} \times N_p}$, and for $i = 2, \ldots, p-1$ the matrix $\mathbf{A}_i(n_i)$ is the n_ith lateral slice of a tensor $\mathcal{A}_i \in \mathbb{R}^{R_{i-1} \times N_i \times R_i}$. Collectively, these matrices and tensors are known as the (TT) cores of the decomposition (1), and their "internal" dimensions $R_1, \ldots, R_{p-1}$ are called TT-ranks.

In order to take advantage of the aforementioned complexity-reducing properties of the TT decomposition, one usually seeks to compute TT cores describing a target tensor. This description, however, is typically only approximate, and thus the form (1) is effectively used to model some data in tensor form.

As we will see, this expedient has been used by [6] to reduce the cost of computing the output of a Volterra model, which is given by a polynomial in a vector $\mathbf{u} \in \mathbb{R}^N$ comprising input samples, and can be written as a p-fold contraction of a tensor $\mathcal{H} \in \mathbb{R}^{N \times \cdots \times N}$ (containing Volterra kernels) with $\mathbf{u}$:

$$\mathcal{H}(\mathbf{u}) := \sum_{n_1=1}^{N} \cdots \sum_{n_p=1}^{N} h_{n_1,\ldots,n_p} \prod_{i=1}^{p} u_{n_i}. \quad (2)$$

This computation has cost $\mathcal{O}(N^p)$. But if $h_{n_1,\ldots,n_p}$ approximately decomposes as in (1), then by plugging this expression into (2) one can decouple the summations, obtaining $\mathcal{H}(\mathbf{u}) \approx \sum_n u_n \mathbf{a}_1(n)^\mathsf{T} \sum_n u_n \mathbf{A}_2(n) \ldots \sum_n u_n \mathbf{A}_{p-1}(n) \sum_n u_n \mathbf{a}_p(n)$, which now costs $\mathcal{O}(N(R_1 + \sum_{i=2}^{p-1} R_{i-1} R_i + R_p))$, thus being cheaper when the TT-ranks are sufficiently small.

As we will explain later, our work takes a different route, leveraging the TT decomposition of Volterra kernels in the so-called *regular* form, whose TT cores follow directly from the state-space description of a linear-analytic system. Although for such regular kernels (2) is replaced by a sequential "contraction-like" computation, the decomposition (1) can still be exploited for efficiency, leading to a cascade realization of the system.

III. Volterra series representation of bilinear and linear-analytic systems

A. The Volterra series and regular kernels

The Volterra series of a causal (single-input, single-output and time-invariant) nonlinear system expresses its output signal $y(t)$ in terms of multiple convolution integrals of products of the input signal $u(t)$ with certain functions v_p called Volterra kernels: $y(t) = \sum_{p=1}^{\infty} y_p(t)$, with

$$y_p(t) = \int_{\mathbb{R}_+} \cdots \int_{\mathbb{R}_+} v_p(\tau_1, \ldots, \tau_p) \prod_{i=1}^{p} u(t - \tau_i) d\tau_1 \ldots d\tau_p.$$

Each component y_p is known as the pth-degree homogeneous subsystem output, as it gets scaled by a factor α^p if u is multiplied by α. Several results on the convergence of the series have been derived, see e.g. [14].

A system that admits a (convergent) Volterra series is thus completely described by its kernels, just like the impulse response completely describes a linear time-invariant system. However, the symmetry of $\prod_{i=1}^{p} u(t-\tau_i)$ implies such kernels are not unique, and one usually adds some constraint to restore uniqueness: commonly assumed (unique) kernel forms are the symmetric form, which is invariant with respect to a permutation of its arguments, and the triangular form, which is null outside the domain $\tau_1 \leq \cdots \leq \tau_p$.

As shown ahead, the less common form known as "regular"[1] [9] naturally possesses TT structure or, in other words, has an exact TT decomposition. It is obtained from the triangular form by the change of variables $\theta_p = \tau_1$ and $\theta_i = \tau_{p-i+1} - \tau_{p-i}$ for $i = 1, \ldots, p-1$. Hence, denoting such regular kernels by h_p and defining $\bar{\theta}_i = \sum_{j=1}^{p} \theta_j$, gives

$$y_p(t) = \int_{\mathbb{R}_+} \cdots \int_{\mathbb{R}_+} h_p(\theta_1, \ldots, \theta_p) \prod_{i=1}^{p} u(t - \bar{\theta}_i) d\theta_1 \ldots d\theta_p. \quad (3)$$

B. Kernels of bilinear and linear-analytic systems

Consider the state-space description of a nonlinear system

$$\tfrac{d}{dt}\, \mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t)) u(t),\ y(t) = \rho(\mathbf{x}(t)), \quad (4)$$

where $\mathbf{x}(t) \in \mathbb{R}^m$ is the state at time t, u is the input, y is the output and $\mathbf{f}, \mathbf{g} : \mathbb{R}^m \to \mathbb{R}^m$ and $\rho : \mathbb{R}^m \to \mathbb{R}$ are analytic. Such systems are called linear-analytic [15], since u enters linearly in the state equation, and admit convergent Volterra series for inputs of sufficient small L^∞ norm:

Theorem 1 ([15]): Suppose the solution of a linear-analytic system for $u(t) \equiv 0$ on the interval $t \in [0, T]$ exists. Then, there exists $\varepsilon > 0$ such that the system admits a uniformly convergent Volterra series representation (3) on $[0, T]$ for all inputs satisfying $\sup_{t \in [0,T]} |u(t)| < \varepsilon$.

Bilinear systems, in particular, amount to a simple special case of (4) where $\mathbf{f}$, $\mathbf{g}$ and ρ are replaced by linear maps:

$$\tfrac{d}{dt}\, \mathbf{z}(t) = \mathbf{F}\mathbf{z}(t) + \mathbf{G}\mathbf{z}(t) u(t) + \mathbf{b} u(t), \quad y(t) = \mathbf{c}^\mathsf{T} \mathbf{z}(t). \quad (5)$$

[1] We adopt this terminology for historical reasons, though it is not related to regularity in the usual sense.

For this particular class, Theorem 1 can be strengthened by only requiring u to be bounded on $[0, T]$. It can also be shown that the regular Volterra kernels of a bilinear system satisfy

$$h_p(\tau_1, \ldots, \tau_p) = \mathbf{c}^\mathsf{T} e^{\mathbf{F}\tau_p} \mathbf{G} e^{\mathbf{F}\tau_{p-1}} \mathbf{G} \ldots \mathbf{G} e^{\mathbf{F}\tau_1} \mathbf{b}, \quad (6)$$

for $\tau_i \geq 0$, $i = 1, \ldots, p$.

The importance of bilinear systems stems in no small part from them being able to approximate linear-analytic systems to an arbitrary degree. More precisely, from [9, Chapter 3] follows the constructive result:

Proposition 1: Given any linear-analytic system (4), there exists a bilinear system of dimension $M = \binom{m+P}{P} - 1$ whose first P homogeneous subsystem outputs coincide with those of the linear-analytic system for all bounded inputs u. The vectors $\mathbf{b}, \mathbf{c}$ and matrices $\mathbf{F}, \mathbf{G}$ are explicitly given by the *Carleman bilinearization* of (4): $\mathbf{b} = \begin{bmatrix}\mathbf{b}_0^\mathsf{T} & \mathbf{0}^\mathsf{T}\end{bmatrix}^\mathsf{T}$, $\mathbf{c} = \begin{bmatrix}\mathbf{c}_0^\mathsf{T} & \mathbf{0}^\mathsf{T}\end{bmatrix}^\mathsf{T}$,

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{1,1} & \mathbf{F}_{1,2} & \cdots & \mathbf{F}_{1,P} \\ \mathbf{O}_{2,1} & \mathbf{F}_{2,2} & \cdots & \mathbf{F}_{2,P} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O}_{P,1} & \mathbf{O}_{P,2} & \cdots & \mathbf{F}_{P,P} \end{bmatrix}, \mathbf{G} = \begin{bmatrix} \mathbf{G}_{1,1} & \mathbf{G}_{1,2} & \cdots & \mathbf{G}_{1,P-1} & \mathbf{O}_{1,P} \\ \mathbf{G}_{2,1} & \mathbf{G}_{2,2} & \cdots & \mathbf{G}_{2,P-1} & \mathbf{O}_{2,P} \\ \mathbf{O}_{P,1} & \mathbf{G}_{3,2} & \cdots & \mathbf{G}_{3,P-1} & \mathbf{O}_{3,P} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{O}_{P,1} & \mathbf{O}_{P,2} & \cdots & \mathbf{G}_{P,P-1} & \mathbf{O}_{P,P} \end{bmatrix};$$

where $\mathbf{F}$ contains the Taylor expansion coefficients of $\mathbf{f}$ up to degree P, whereas those of $\mathbf{g}$ appear in $\mathbf{b}_0 \in \mathbb{R}^m$ and in $\mathbf{G}$, and those of ρ appear in $\mathbf{c}_0$. The blocks $\mathbf{O}_{i,j}$ contain only zeros and $\mathbf{O}_{i,j}, \mathbf{F}_{i,j}, \mathbf{G}_{i,j} \in \mathbb{R}^{M_i \times M_j}$, where $M_i = \binom{m+i-1}{i}$.

From this follows a key observation, exploited ahead:

Corollary 1: For any $P \in \mathbb{N}$, the first P (regular) Volterra kernels of a linear-analytic system of dimension m satisfy (6), where $\mathbf{F}, \mathbf{G}, \mathbf{b}$ and $\mathbf{c}$ are as described by Proposition 1.

C. Discrete-time kernels and Volterra filters

For digital signal processing, we consider a discrete-time version of (3), with $\bar{n}_i = \sum_{j=i}^{p} n_j$ and truncated in memory:

$$y_p(n) = T_s^p \sum_{n_p=0}^{N-1} \cdots \sum_{n_1=0}^{N-1} h_p(n_1, \ldots, n_p) \prod_{i=1}^{p} u(n - \bar{n}_i), \quad (7)$$

where the scaling factor T_s^p is discussed below. Truncating now the degree gives $y(n) = \sum_{p=1}^{P} y_p(n)$. We refer to this polynomial model as a regular Volterra filter (VF). It is BIBO stable by construction and conceptually simple.

Unfortunately, though, the amount of multiplications required to compute $y(n)$ via the above expressions grows as $\mathcal{O}(PN^P)$, hence quite swiftly in P and N. This well-known shortcoming of the VF precludes its use for systems of even moderately long memory (in samples) when relatively high orders (say, $P > 3$) are required for precise modeling.

If a sampling period T_s is sufficiently small so that the kernels and input in (3) are approximately constant in an interval of width T_s in all time arguments, then the sampled output $y_p(n) := y_p(t)|_{t=nT_s}$ is approximated by (7).[2] This follows directly from segmenting the integrals in (3) in intervals of width T_s and will be assumed throughout the paper. For simplicity, we will actually absorb the factor T_s into the discrete-time signal $u(n)$, so that T_s^p will not explicitly appear as it does in (7). For linear-analytic systems in particular, Corollary 1 implies that the discretized regular kernels $h_p(n_1, \ldots, n_p)$ of orders 1 up to P have the form $h_p(n_1, \ldots, n_p) = \mathbf{c}^\mathsf{T} e^{\mathbf{F}T_s n_p} \mathbf{G} \ldots \mathbf{G} e^{\mathbf{F}T_s n_1} \mathbf{b}$. We show next how this property can be exploited to obtain a much more efficient finite-memory model than a VF. In a upcoming paper, we will drop the finite memory constraint and derive low-rank infinite-memory realizations of bilinear kernels [16].

[2]For simplicity, we opt to use, with some abuse of notation, the same symbol for discrete-time and continuous-time variables.

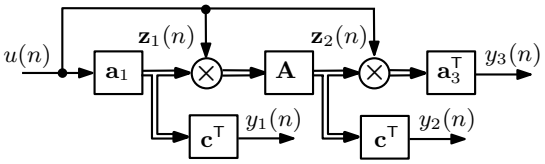

Fig. 1. Cascade-TT realization of a Volterra model of order $P = 3$.

IV. Tensor-train Volterra model of linear-analytic systems

A. Tensor-train structure of linear-analytic system kernels

From the above discussion and (1) it follows that the discretized kernels of a linear-analytic system can be seen as tensors with TT structure, namely $h_p(n_1, \ldots, n_p) = \mathbf{a}_p^\mathsf{T}(n_p) \mathbf{A}(n_{p-1}) \ldots \mathbf{A}(n_2) \mathbf{a}_1(n_1)$, with

$$\mathbf{a}_p^\mathsf{T}(n_p) = \mathbf{c}^\mathsf{T} e^{\mathbf{F}T_s n_p} \mathbf{G}, \ \mathbf{a}_1(n_1) = e^{\mathbf{F}T_s n_1} \mathbf{b}, \quad (8)$$

$$\mathbf{A}(n_i) = e^{\mathbf{F}T_s n_i} \mathbf{G}, \quad i = 2, \ldots, p-1. \quad (9)$$

The TT-ranks of h_p seen as a TT-structured tensor are thus all equal to $M = \binom{m+P}{P} - 1$. Furthermore, these cores are completely characterized by two $M \times M$ matrices $\mathbf{F}, \mathbf{G}$ and two M-dimensional vectors $\mathbf{c}, \mathbf{b}$. We will see next how taking this property into account can also bring a reduction of the computational complexity of a VF realization.

B. Cascade-TT (CaTT) realization of regular kernels

Using P (the model degree) in place of p and replacing the kernels by their TT expressions with cores (8)–(9), we can write (7) as a cascade of P filters, of which $P-2$ have vector inputs and vector outputs. It turns out that the (finite-memory) impulse response of each of those filters is one of the three possible cores defined by (8)–(9), giving the outputs:

$$\mathbf{z}_1(n) = \sum_{n_1=0}^{N-1} \mathbf{a}_1(k) u(n - n_1), \quad (10)$$

$$\mathbf{z}_p(n) = \sum_{n_i=0}^{N-1} \mathbf{A}(n_i) \mathbf{z}_{p-1}(n - n_p) u(n - n_p), \quad (11)$$

$$y_P(n) = \sum_{n_P=0}^{N-1} \mathbf{a}_P^\mathsf{T}(n_P) \mathbf{z}_{P-1}(n - n_P) u(n - n_P), \quad (12)$$

where $p = 2, \ldots, P-1$. Furthermore, since these filters are the same for all kernels of orders 1 up to P, their outputs can be reused to compute all subsystem outputs, $y_p(n) = \mathbf{c}^\mathsf{T} \mathbf{z}_p(n)$, $p = 1, \ldots, P-1$. A depiction of this scheme is in Fig. 1 for $P = 3$, where the single and double arrows indicate scalar- and vector-valued signals, respectively.

Let us now calculate the computational complexity of this realization, in terms of the number of (scalar) performed

multiplications per output sample. Since our focus is on the case where cores are given by (8)–(9), where $\mathbf{F}$, $\mathbf{G}$, $\mathbf{b}$ and $\mathbf{c}$ all come from the Carleman bilinearization of a linear-analytic system, we will take into account the sparse structure of these matrices and vectors, as described by Proposition 1. This implies, in particular that $\mathbf{A}(n_i)$ has a block structure identical to that of $\mathbf{G}$ described in Proposition 1, with blocks $\mathbf{A}_{j,k}(n_i) \in \mathbb{R}^{M_j \times M_k}$, where M_i is as defined in Proposition 1. We will also assume that the matrices $\mathbf{A}(n)$, as well as the vectors $\mathbf{a}_1(n)$, have been pre-computed for $n = 0, \ldots, N-1$ from their definition (8)–(9).

First, from the structure of $\mathbf{F}$ and $\mathbf{b}$ it follows that $\mathbf{a}_1(n_1)$ can only have its first M_1 components nonzero, and hence it takes NM_1 multiplications to compute $\mathbf{z}_1(n)$. By induction and from the block structure of $\mathbf{A}(n_p)$, it can be shown then for $p = 2, \ldots, P-1$ that $\mathbf{z}_p(n)$ can only have $\sum_{k=1}^{p} M_k$ nonzero components. From this follows that its computation takes $S_{1,p-1} + N[(M_1+M_2+1)S_{1,p-1} + \sum_{k=3}^{p} M_k S_{k-1,p-1}]$ multiplications, where $S_{j,k} = \sum_{\ell=j}^{k} M_\ell$ and the second term between brackets is absent when $p = 2$. It also follows that $y_P(n)$ takes $(N+1)S_{1,P-1}$ multiplications. Finally, since only the first M_1 elements of $\mathbf{c}$ can be nonzero, $y_p(n) = \mathbf{c}^{\mathsf{T}}\mathbf{z}_p(n)$, $p = 1, \ldots, P-1$, takes $(P-1)M_1$ multiplications. Adding up all these multiplications, we get then for $P > 1$ a total of

$$\mathcal{C}_{\mathsf{CaTT}} = NM_1 + (N+1)S_{1,P-1} + (P-1)M_1 + \sum_{p=2}^{P-1} S_{1,p-1} + N[(M_1+M_2+1)S_{1,p-1} + \sum_{k=3}^{p} M_k S_{k-1,p-1}].$$

For fixed m and P, this formula grows linearly in the memory N, while the cost of a VF grows as $\mathcal{O}(N^P)$.

Remark 1: An additional reduction in computational complexity can be achieved from low-rank (smaller than M) approximate TT decompositions of the regular kernels. In this paper, however, we will only focus on the exact decomposition (8)-(9).

C. Comparison with other tensor-based approaches

Other approaches usually relate to a conventional VF with triangular or symmetric kernels, with $y(n) = \sum_{p=1}^{P} y_p(n)$ and

$$y_p(n) = \sum_{n_p=0}^{N-1} \cdots \sum_{n_1=0}^{N-1} v_p(n_1, \ldots, n_p) \prod_{i=1}^{p} u(n-n_i). \qquad (13)$$

In the triangular case and reusing input products across all orders up to $P > 1$, it takes $\mathcal{C}_{\mathsf{VF}} = \binom{N+P}{P} - 1 + \sum_{p=2}^{P} \binom{N+p-1}{p}$ multiplications to calculate $y(n)$ using (13).

1) Contraction-TT (CoTT): The approach closest to ours is that of [6]. It is based on assembling all P kernels of a conventional VF into a Pth-order tensor $\mathcal{V}$ of dimensions $N+1 \times \cdots \times N+1$ in such a way that its output can be written as a p-fold contraction $y(n) = \mathcal{V}(\mathbf{u}(n))$, where $\mathbf{u}(n) = [1 \quad u(n) \quad \ldots \quad u(n-N+1)]^{\mathsf{T}}$. This identity holds for symmetric and triangular kernels, but not for regular ones. Then, as follows from the discussion in Section II, an approximate TT decomposition of $\mathcal{V}$ can be exploited to bring the computational cost of the contraction down to $\mathcal{C}_{\mathsf{CoTT}} = (N+1)(R_1 + R_{P-1}) + (N+2)\sum_{p=2}^{P-1} R_{p-1}R_p + \min(R_1, \ldots, R_{P-1})$ multiplications. Now, even though this approach is aimed at the case where the kernels are unknown and must be identified from data, it can also be applied to known (triangular or symmetric) kernels by constructing $\mathcal{V}$ and then feeding it into a TT approximation algorithm, such as the TT-SVD procedure [8]. However, unlike our proposal, this strategy does not exploit the exact TT structure of regular kernels, and one generally has to approximate $\mathcal{V}$ in order to obtain cores with low TT-ranks.

2) Volterra-Parafac (VP): The VP structure proposed in [5] approximates each discrete-time symmetric kernel v_p by a low-rank symmetric CPD, that is, $v_p(n_1, \ldots, n_p) \approx \sum_{q=1}^{Q_p} \prod_{i=1}^{p} b_q^{(p)}(n_i)$, where $n_i \in \{0, \ldots, N-1\}$ and Q_p is the (symmetric) rank. This allows approximating $y_p(n)$ by a sum of powers of outputs of Q_p linear systems with impulse responses $b_q^{(p)}$,

$$y_p(n) \approx \sum_{q=1}^{Q_p} \left(\sum_{k=0}^{N-1} b_q^{(p)}(k) u(n-k) \right)^p,$$

which costs $Q_p(N+p-1)$ multiplications, leading to an overall $\mathcal{C}_{\mathsf{VP}} = N + \sum_{p=2}^{P} Q_p(N+p-1)$ complexity. While this cost can be smaller than or comparable to $\mathcal{C}_{\mathsf{CaTT}}$ when the ranks Q_p are small, choosing such ranks and computing the filters $b_q^{(p)}$ from the kernels is a difficult task. In particular, there is no algorithm for directly computing the filters from a state-space description of a linear-analytic system. Instead, one has to first compute the kernels and then find a rank-Q_p approximation for each v_p [17], which is computationally heavy for high p and involves pitfalls such as the non-existence of best approximations[3] [7]. By contrast, the cores of the CaTT follow directly from the state-space equations by Carleman bilinearization, without the need of computing kernels.

V. Numerical results

We now apply the discussed structures to a bass loudspeaker model, adapted [2] from [10], of dimension $m = 3$. As the model is linear-analytic, each homogeneous subsystem admits an exact CaTT model (up to memory truncation), whose outputs are thus used as reference for comparison with other structures. A pre-processing block in nonlinear acoustic echo cancellation is a possible application. Responses $y(n) = \sum_{p=1}^{P} y_p(n)$ are compared for 50 realizations of a low-pass, $\pi/4$ bandwidth, Gaussian input $u(n)$.

From the loudspeaker model, we follow [2] to calculate the triangular kernels for a sampling frequency of $1/T_s = 5$ kHz, chosen to reduce the aliasing due to nonlinearity. Based on their main diagonals plotted in Fig. 2 up to $p = 4$, a memory of $N = 120$ is chosen for the (conventional) VF. The TT-SVD procedure [8] is then applied to approximate the Pth-order tensor $\mathcal{V}$ containing the corresponding symmetric kernels, for different choices of the SVD truncation threshold ϵ. We use the label CoTT H for the higher $\epsilon = 0.1$ cases, which have

[3]Constraining the CPD factors (e.g., by imposing orthogonality) can ensure existence of solutions, but there are no physical grounds for doing so.

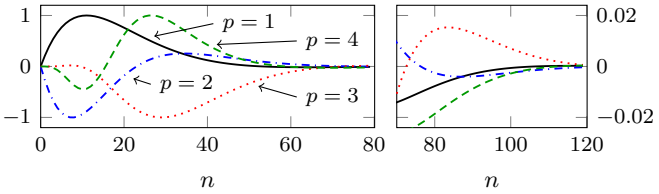

Fig. 2. Diagonals $v_p(n, \ldots, n)$ of triangular kernels (normalized to peak).

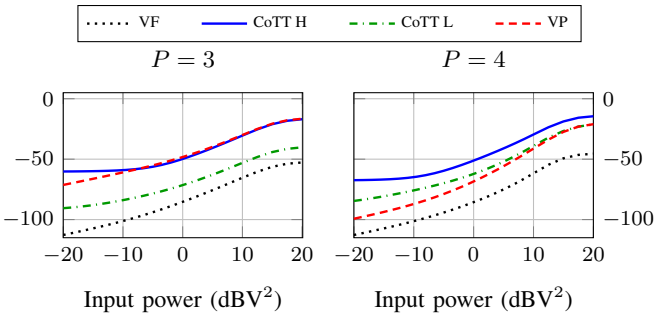

Fig. 3. NMSE (in dB) of evaluated structures, defined with respect to the output of CaTT: NMSE $= \sum_n (y(n) - y_{\mathsf{CaTT}}(n))^2 / \sum_n (y_{\mathsf{CaTT}}(n))^2$.

TT-ranks $(40, 36)$ for $P = 3$ and $(40, 63, 33)$ for $P = 4$, and use the label CoTT L for the lower $\epsilon = 0.01$ ($P = 3$) and $\epsilon = 0.05$ ($P = 4$) cases, which have ranks $(65, 63)$ and $(53, 117, 50)$, respectively. To compensate for their different scales, we normalize all kernels by their respective ℓ_2 norms before constructing $\mathcal{V}$. To derive the VP structure, we compute a CPD of each symmetric kernel with all ranks $Q_p = 19$ when $P = 3$, and $Q_p = 63$ when $P = 4$. As $m = 3$, the TT-ranks of the CaTT structure all equal $M = 19$ for $P = 3$ and $M = 34$ for $P = 4$. The choice $N = 120$ is also kept for this structure, its effective memory due to $\bar{n}_i = \sum_{j=i}^{p} n_j$ in (7) being though larger than N, better approximating then the infinite-duration kernels. Its output is thus used to define normalized mean-squared errors (NMSEs) $\sum_n (y(n) - y_{\mathsf{CaTT}}(n))^2 / \sum_n (y_{\mathsf{CaTT}}(n))^2$ for the outputs $y(n)$ of the other structures. As the ranks of the CoTT and VP structures grow, so do their complexities, while their NMSEs approach that of the VF.

Fig. 3 shows the (ensemble-average) NMSE of the VF, CoTT and VP structures. The NMSEs increase with the input power since this also increases the importance of the higher-order kernel approximation errors. The corresponding complexities, seen in Section IV, are given in Table I, where the notation $X\mathrm{e}E$ stands for $X \times 10^E$. It can be seen that the proposed structure CaTT is, together with VP, among the least costly ones (in particular, with a cost more than one order of magnitude lower than that of CoTT), whilst it involves no approximation (other than due to memory truncation), offering the most precise realization among all alternatives.

VI. Conclusion

We have established and exploited a connection between the tensor train decomposition and the discrete-time regular Volterra kernels of linear-analytic systems, which had been hitherto unexplored. Thanks to the properties of this decomposition, such a connection leads to a discrete-time model whose realization bears a low complexity in terms of the number of multiplications per sample output. One can thus use this discrete-time model for efficiently predicting the target system's output. Our numerical results show how this strategy can be more attractive than other tensor-based alternatives by means of a nonlinear loudspeaker example.

Future work should study identification algorithms for the proposed structure. A paper in preparation will investigate low-rank infinite-memory realizations of bilinear kernels [16].

TABLE I
Per output sample complexities of the compared structures, in number of multiplications

	VF	CaTT	CoTT H	CoTT L	VP
P=3	6.1e5	4.7e3	1.8e5	5.2e5	4.7e3
P=4	1.9e7	2.3e4	3.2e5	7.8e5	2.3e4

References

[1] S. Boyd and L. Chua, "Fading memory and the problem of approximating nonlinear operators with Volterra series," *IEEE Trans. Circuits Syst.*, vol. CAS-32, no. 11, pp. 1150–1161, Nov. 1985.

[2] P. M. S. Burt and J. H. de M. Goulart, "Efficient computation of bilinear approximations and Volterra models of nonlinear systems," *IEEE Trans. Signal Process.*, vol. 66, no. 3, pp. 804–816, Feb 2018.

[3] E. L. O. Batista and R. Seara, "A fully LMS/NLMS adaptive scheme applied to sparse-interpolated Volterra filters with removed boundary effect," *Signal Process.*, vol. 92, no. 10, pp. 2381 – 2393, 2012.

[4] M. Zeller and W. Kellermann, "Fast and robust adaptation of DFT-domain Volterra filters in diagonal coordinates using iterated coefficient updates," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1589–1604, 2010.

[5] G. Favier, A. Y. Kibangou, and T. Bouilloc, "Nonlinear system modeling and identification using Volterra-PARAFAC models," *Int. J. Adaptive Control Signal Process.*, vol. 26, no. 1, pp. 30–53, Jan. 2012.

[6] K. Batselier, Z. Chen, and N. Wong, "Tensor network alternating linear scheme for MIMO Volterra system identification," *Automatica*, vol. 84, pp. 26–35, 2017.

[7] V. de Silva and L.-H. Lim, "Tensor rank and the ill-posedness of the best low-rank approximation problem," *SIAM J. Matrix Anal. Appl.*, vol. 30, no. 3, pp. 1084–1127, 2008.

[8] I. V. Oseledets, "Tensor-train decomposition," *SIAM J. Scientific Comput.*, vol. 33, no. 5, pp. 2295–2317, 2011.

[9] W. J. Rugh, *Nonlinear System Theory: The Volterra/Wiener Approach*. Baltimore, MD: Johns Hopkins University Press, 1981.

[10] D. Franken, K. Meerkotter, and J. Waßmuth, "Passive parametric modeling of dynamic loudspeakers," *IEEE Trans. Speech Audio Process.*, vol. 9, no. 8, pp. 885–891, nov. 2002.

[11] A. Prasad, M. Thorsell, H. Zirath, and C. Fager, "Accurate modeling of GaN HEMT RF behavior using an effective trapping potential," *IEEE Trans. Microw. Theory Techn.*, vol. 66, no. 2, pp. 845–857, Feb 2018.

[12] C. H. Sherman and J. L. Butler, *Transducers and arrays for underwater sound*. Springer, 2007, vol. 4.

[13] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3551–3582, 2017.

[14] T. Hélie and B. Laroche, "Computation of convergence bounds for Volterra series of linear-analytic single-input systems," *IEEE Trans. Autom. Control*, vol. 56, no. 9, pp. 2062–2072, 2011.

[15] R. Brockett, "Volterra series and geometric control theory," *Automatica*, vol. 12, no. 2, pp. 167–176, mar. 1976.

[16] P. M. S. Burt and J. H. de M. Goulart, "Infinite-memory low-rank realization of bilinear Volterra kernels," (in preparation).

[17] ——, "Evaluating the potential of Volterra-PARAFAC IIR models," in *IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2013, pp. 5745–5749.