Model Reduction and Simulation of Nonlinear Circuits via Tensor Decomposition

Haotian Liu, *Student Member, IEEE,* Luca Daniel, *Member, IEEE,* and Ngai Wong, *Member, IEEE*

***Abstract*—Model order reduction of nonlinear circuits (especially highly nonlinear circuits), has always been a theoretically and numerically challenging task. In this paper we utilize tensors (namely, a higher order generalization of matrices) to develop a tensor-based nonlinear model order reduction (TNMOR) algorithm for the efficient simulation of nonlinear circuits. Unlike existing nonlinear model order reduction methods, in TNMOR high-order nonlinearities are captured using tensors, followed by decomposition and reduction to a compact tensor-based reduced-order model. Therefore, TNMOR completely avoids the dense reduced-order system matrices, which in turn allows faster simulation and a smaller memory requirement if relatively low-rank approximations of these tensors exist. Numerical experiments on transient and periodic steady-state analyses confirm the superior accuracy and efficiency of TNMOR, particularly in highly nonlinear scenarios.**

***Keywords*—Tensor, nonlinear model order reduction, reduced-order model**

I. Introduction

THE complexity and reliability of modern VLSI chips rely heavily on the effective simulation and verification of circuits during the design phase. In particular, mixed-signal and radio-frequency (RF) modules are critical and often hard to analyze due to their intrinsic nonlinearities and their large problem sizes. Consequently, nonlinear model order reduction (NMOR) becomes necessary in the electronic design automation (EDA) flow. The goal of NMOR is to find a reduced-order model (ROM) that simulates fast and yet still captures the input-output behavior of the original system accurately.

Compared to the mature model order reduction (MOR) methods in linear time-invariant (LTI) systems [1]–[4], NMOR is much more challenging. Several projection-based NMOR methods have been developed in the last decade. In [5], [6], the nonlinear system is expanded into a series of cascaded linear subsystems, whereby the outputs from the low-order subsystems serve as inputs to the higher-order ones. Then, existing projection-based linear MOR methods, e.g., [1], [3], can be applied to these linear subsystems recursively. We refer to the method in [5], [6] as the "standard projection" approach. Nonetheless, in this method, the dimension of the resulting ROM grows exponentially with respect to the orders of the subsystems. Moreover, to reduce the size of Arnoldi starting vectors, lower order projection subspaces are used to approximate the column spaces of the higher order system matrices. Consequently, approximation errors in lower order subspaces can easily propagate and accumulate in higher order subsystems.

To tackle this accuracy issue, a more compact NMOR scheme, called NORM, is proposed in [7] where each explicit moment of high-order Volterra transfer functions is matched. For weakly nonlinear circuits, NORM exhibits an extraordinary improvement in accuracy over the standard projection approach since lower order approximations are completely skipped. The resulting ROM tends to be more compact as the sizes of lower order reduced subsystems will not carry forward to higher order ones. However, this approach still needs to build the reduced but dense system matrices whose dimensions grow exponentially as the order increases. This limits the practicality of NORM as simulation of small but dense problems is sometimes even slower than simulating the large but sparse original system.

To overcome the curse of dimensionality, rather than treating the exponentially growing system matrices as 2-dimensional matrices, their nature should be recognized. To this end, tensors, as high dimensional generalization of matrices, can be utilized. In recent years, there has been a strong trend toward the investigation of tensors and their low-rank approximation [8]–[13], due to their high dimensional nature ideal for complex problem characterization and their efficient compact representation ideal for large scale data analyses. Therefore, it is natural to characterize circuit nonlinearities by tensors whereby the tensor structure can be exploited to reduce the original nonlinear system.

In this paper, we propose a tensor-based nonlinear model order reduction (TNMOR) scheme for typical circuits with a few nonlinear components. The work is a variation of the Volterra series-based projection methods [5]–[7]. The nonlinear system is modeled by a truncated Volterra series up to a certain high order. However, in the proposed method, the higher order system matrices are modeled by high-order tensors, so that the high dimensional data can be approximated by the sum of only a few outer products of vectors via the canonical tensor decomposition [8], [12], [14]. Next, the projection spaces are generated by matching the moments of each subsystem, in terms of those decomposed vectors. Finally, the ROM is represented in the canonical tensor decomposition form, where the sparsity of the high dimensional system matrices is preserved after TNMOR.

H. Liu and N. Wong are with the Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong SAR (email: {htliu, nwong}@eee.hku.hk).

L. Daniel is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, USA (email: luca@mit.edu).

The main contribution of this work is that unlike previous approaches, simulation of the TNMOR-produced ROM completely avoids the overhead of solving high-order dense system matrices. This truly allows the simulation to exploit the acceleration brought about by NMOR. We remark that the utilization of TNMOR depends on the existence of low-rank approximations of these high-order tensors, which are generally available for circuits with a few nonlinear components. Moreover, the size of the ROM depends only on the tensor rank and the order of moments being matched for each system matrix. In other words, it will not grow exponentially as the order of subsystems increases, which enables NMOR of highly nonlinear circuits not amenable before.

The paper is organized as follows. Section II reviews the backgrounds of Volterra series, existing NMOR approaches, as well as tensors and tensor decomposition. After that, Section III presents the tensor-based modeling of nonlinear systems. The proposed TNMOR is described in Section IV and simulation of the TNMOR-reduced ROM is discussed in Section V. Numerical examples are given in Section VI. Finally, Section VII draws the conclusion.

II. Background and Related Work

A. Volterra subsystems

We consider a nonlinear multi-input multi-output (MIMO) time-invariant circuit modeled by the differential-algebraic equation (DAE)

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[q\left(x(t)\right)\right] + f\left(x(t)\right) = Bu(t), \quad y(t) = L^T x(t), \qquad (1)$$

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^l$ are the state and input vectors, respectively; $q(\cdot)$ and $f(\cdot)$ are the nonlinear capacitance and conductance functions extracted from the modified nodal analysis (MNA); B and L are the input and output matrices, respectively. The nonlinear system can be expanded under a perturbation around its equilibrium point x_0 by the Taylor expansion

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x + C_2(x\otimes x) + C_3(x\otimes x\otimes x) + \cdots\right] + G_1 x + G_2(x\otimes x) + G_3(x\otimes x\otimes x) + \cdots = Bu, \qquad (2)$$

where $\otimes$ denotes the Kronecker product and we will use the shorthand $x^{③} = x \otimes x \otimes x$ etc. for the Kronecker powers throughout the paper. The conductance and capacitance matrices are given by

$$G_i = \frac{1}{i!}\frac{\partial^i f}{\partial x^i}\bigg|_{x=x_0} \in \mathbb{R}^{n\times n^i}, C_i = \frac{1}{i!}\frac{\partial^i q}{\partial x^i}\bigg|_{x=x_0} \in \mathbb{R}^{n\times n^i}. \qquad (3)$$

By Volterra theory and variational analysis [15], [16], the solution x to (2) is approximated with the Volterra series $x(t) = x_1(t) + x_2(t) + x_3(t) + \cdots$, where $x_i(t)$ is the response to each of the following Volterra subsystems

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_1\right] + G_1 x_1 = Bu, \qquad (4a)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_2\right] + G_1 x_2 = -\frac{\mathrm{d}}{\mathrm{d}t}\left[C_2 x_1^{②}\right] - G_2 x_1^{②}, \qquad (4b)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_3\right] + G_1 x_3 = -\frac{\mathrm{d}}{\mathrm{d}t}\left[C_3 x_1^{③} + C_2(\overline{x_{i_1}\otimes x_{i_2}})_3\right] - G_3 x_1^{③} - G_2(\overline{x_{i_1}\otimes x_{i_2}})_3, \qquad (4c)$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_4\right] + G_1 x_4 = -\frac{\mathrm{d}}{\mathrm{d}t}\Big[C_4 x_1^{④} + C_3(\overline{x_{i_1}\otimes x_{i_2}\otimes x_{i_3}})_4 + C_2(\overline{x_{i_1}\otimes x_{i_2}})_4\Big] - G_4 x_1^{④} - G_3(\overline{x_{i_1}\otimes x_{i_2}\otimes x_{i_3}})_4 - G_2(\overline{x_{i_1}\otimes x_{i_2}})_4, \qquad (4d)$$

and so on, where $(\overline{x_{i_1}\otimes x_{i_2}})_3 = x_1\otimes x_2 + x_2 \otimes x_1$, $(\overline{x_{i_1}\otimes x_{i_2}\otimes x_{i_3}})_4 = x_1\otimes x_1\otimes x_2 + x_1\otimes x_2\otimes x_1 + x_2\otimes x_1\otimes x_1$ and more generally $(\overline{x_{i_1}\otimes\cdots\otimes x_{i_n}})_k = \sum_{i_1+\cdots+i_n=k} x_{i_1}\otimes\cdots\otimes x_{i_n}$, $i_1,\ldots,i_n \in \mathbb{Z}^+$, where $\mathbb{Z}^+$ denotes the set of positive integers.

B. Existing projection-based NMOR methods

To reduce the original system (2), the standard projection approach [5], [6] treats (4) as a series of MIMO linear systems, with the right hand side of each equation serving as its actual "input". Then, the projection-based linear MOR approach, e.g., [1], is applied.

Suppose up to k_1th-order (viz. from 0th to k_1th) moments of x_1 in (4a) are matched by a Krylov subspace projector $x_1 \approx V_{k_1}\tilde{x}_1$, the number of columns of V_{k_1} is $(k_1+1)l$. After that, (4b) is recast into a concatenated, stacked descriptor system and V_{k_1} is used to approximate its input by assuming $x_1^{②} \approx (V_{k_1}\tilde{x}_1)^{②} = V_{k_1}^{②}\tilde{x}_1^{②}$,

$$\begin{bmatrix} C_1 & -I \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \dot{x}_2 \\ \dot{x}_{2e} \end{bmatrix} + \begin{bmatrix} G_1 & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} x_2 \\ x_{2e} \end{bmatrix} = -\begin{bmatrix} G_2 \\ C_2 \end{bmatrix} x_1^{②} \approx -\begin{bmatrix} G_2 V_{k_1}^{②} \\ C_2 V_{k_1}^{②} \end{bmatrix}\tilde{x}_1^{②}. \qquad (5)$$

Consequently, the multiple Krylov starting vectors of (5) become $-\begin{bmatrix} G_2 V_{k_1}^{②} \\ C_2 V_{k_1}^{②} \end{bmatrix}$ instead of $-\begin{bmatrix} G_2 \\ C_2 \end{bmatrix}$, therefore the dimensionality of the input is reduced to $(k_1+1)^2 l^2$ from n^2. If up to k_2th order moments of x_2 are preserved, (4b) can be reduced by another projection $x_2 \approx V_{k_2}\tilde{x}_2$ to a smaller linear system with $(k_2+1)(k_1+1)^2 l^2$ inputs. Similarly, higher order projectors V_{k_3}, V_{k_4}, etc. are obtained by iteratively reducing the remaining subsystems in (4).

Suppose we have N linear subsystems in (4) and $k = k_1 = k_2 = \cdots = k_N$ order of moments are matched in each subsystem, the standard projection approach will result in a ROM with size $O(k^{2N-1} l^N)$. Thus, in practical circuit reduction examples, k_1, k_2, etc. should be relatively small (otherwise the size of the reduced system may even exceed n quickly), which could hamper the accuracy of the ROM.

Instead of regarding (4) as a set of linear equations, NORM [7] derives frequency-domain high-order nonlinear Volterra transfer functions $H_2(s_1,s_2)$, $H_3(s_1,s_2,s_3)$, etc. associated to the subsystems in (4). These transfer functions are expanded into multivariate polynomials of $s_1, s_2, \ldots$ such

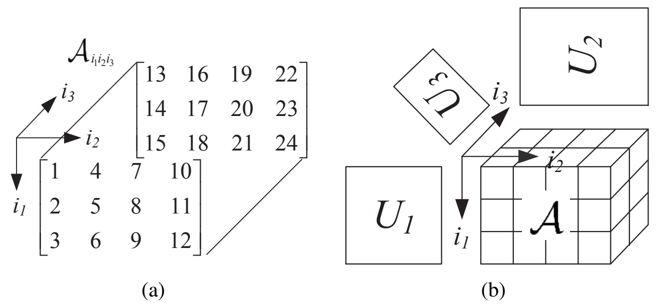

Fig. 1. (a) A tensor $\mathcal{A} \in \mathbb{R}^{3\times4\times2}$. (b) Illustration of $\mathcal{A} \times_1 U_1 \times_2 U_2 \times_3 U_3$.

that the coefficients (moments) can be explicitly matched. In NORM, the size of an Nth-order ROM is in $O(k^{N+1}l^N)$ if $k = k_1 = \cdots = k_N$[1].

In both aforementioned methods, the final step consists of replacing the original nonlinear system (2) by a smaller system via the transformations

$$\begin{aligned}\tilde{x} = V^T x, \quad \tilde{B} = V^T B, \quad \tilde{L} = V^T L,\\ \tilde{G_i} = V^T G_i(V^{\textcircled{i}}), \quad \tilde{C_i} = V^T C_i(V^{\textcircled{i}}),\end{aligned} \tag{6}$$

where $i = 1, \ldots, N$ and $V = \text{orth}[V_{k_1}, V_{k_2}, V_{k_3}, \ldots]$ is the orthogonal projector. Suppose q is the size of the reduced state, $\tilde{G_i}$ and $\tilde{C_i}$ will be dense matrices with $O(q^{i+1})$ entries, despite the sparsity of G_i and C_i. To store these dense matrices, the memory space required grows exponentially.

C. Tensors and tensor decomposition

Some tensor basics are reviewed here, while more tensor properties and decompositions can be found in [9].

1) Tensors: A dth-order tensor is a d-way array defined by[2]

$$\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_d}. \tag{7}$$

For example, Fig. 1(a) depicts a 3rd-order $3 \times 4 \times 2$ tensor. In particular, scalars, vectors and matrices can be regarded as 0th-order, 1st-order and 2nd-order tensors, respectively.

Matricization is a process that unfolds or flattens a tensor into a 2nd-order matrix. The k-mode matricization is aligning each kth-direction "vector fiber" to be the columns of the matrix. For example a 3rd-order $n_1 \times n_2 \times n_3$ tensor $\mathcal{A}$ can be 1-mode matricized into an $n_1 \times n_2n_3$ matrix $A^{(1)}$ as illustrated in Fig. 2.

2) Tensor-matrix product: The k-mode product of a tensor $\mathcal{A} \in \mathbb{R}^{n_1\times\cdots\times n_k\times\cdots\times n_d}$ with a matrix $U \in \mathbb{R}^{p_k\times n_k}$ results in a new tensor $\mathcal{A} \times_k U \in \mathbb{R}^{n_1\times\cdots\times n_{k-1}\times p_k\times n_{k+1}\times\cdots\times n_d}$ given by

$$(\mathcal{A} \times_k U)_{j_1\cdots j_{k-1}m_kj_{k+1}\cdots j_d} = \sum_{j_k=1}^{n_k} \mathcal{A}_{j_1\cdots j_k\cdots j_d} U_{m_kj_k}. \tag{8}$$

A conceptual explanation of k-mode product is to multiply each kth-direction "vector fiber" in $\mathcal{A}$ by the matrix U. An

[1]The complexity refers to the single-point expansion algorithm of NORM. The multi-point version of NORM would have a lower complexity.

[2]We denote tensors by calligraphic letters, e.g., $\mathcal{A}$ and $\mathcal{G}$.

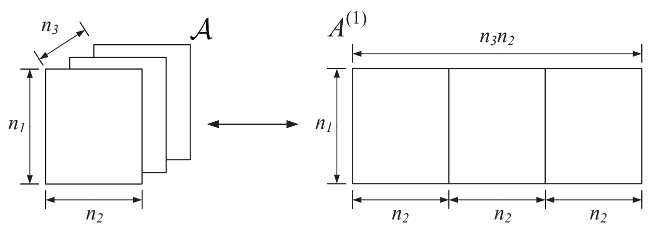

Fig. 2. 1-mode matricization of a 3rd-order tensor.

illustration of the multiplication to a 3rd-order tensor is shown in Fig. 1(b).

The "Khatri-Rao product" is the "matching columnwise" Kronecker product. The Khatri-Rao product of matrices $A = [a_1, a_2, \ldots, a_k] \in \mathbb{R}^{n_1\times k}$ and $B = [b_1, b_2, \ldots, b_k] \in \mathbb{R}^{n_2\times k}$ is defined by $A \odot B = [a_1 \otimes b_1, a_2 \otimes b_2, \ldots, a_k \otimes b_k] \in \mathbb{R}^{n_1n_2\times k}$. If A and B are column vectors, $A \odot B = A \otimes B$. And if A and B are row vectors, $A \odot B$ becomes the Hadamard product (viz. element-by-element product) of the two rows.

3) Rank-1 tensors and canonical decomposition: A rank-1 tensor of dimension d can be written as the outer product of d vectors

$$\mathcal{A} = a^{(1)} \circ a^{(2)} \circ \cdots \circ a^{(d)}, \; a^{(k)} \in \mathbb{R}^{n_k}, \tag{9}$$

where $\circ$ denotes the outer product. Its element $\mathcal{A}_{i_1i_2\cdots i_d} = a^{(1)}_{i_1} a^{(2)}_{i_2} \cdots a^{(d)}_{i_d}$, where $a^{(k)}_{i_k}$ is the i_kth entry of vector $a^{(k)}$.

The CANDECOMP/PARAFAC (CP) decomposition[3] [8], [9], [14], [17] approximates a tensor $\mathcal{A}$ by a finite sum of rank-1 tensors, which can be written by

$$\mathcal{A} \approx \sum_{r=1}^{R} a^{(1)}_r \circ a^{(2)}_r \circ \cdots \circ a^{(d)}_r, \; a^{(k)}_r \in \mathbb{R}^{n_k}, \tag{10}$$

where $R \in \mathbb{Z}^+$. Concisely, using the factor matrices $\mathbf{A}^{(k)} \triangleq [a^{(k)}_1, a^{(k)}_2, \ldots, a^{(k)}_R] \in \mathbb{R}^{n_k\times R}$, the right-hand side of the CP (10) can be expressed by the notation $\mathcal{A} \approx [\![\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(d)}]\!]$. Moreover, it is worth mentioning that the k-mode matricization of $\mathcal{A}$ could be reconstructed by these factor matrices

$$A(k) \approx \mathbf{A}^{(k)}(\mathbf{A}^{(d)} \odot \cdots \odot \mathbf{A}^{(k+1)} \odot \mathbf{A}^{(k-1)} \odot \cdots \odot \mathbf{A}^{(1)})^T. \tag{11}$$

The rank of the tensor $\mathcal{A}$, rank($\mathcal{A}$), is the minimum value of R in the exact decomposition (10). A rank-R approximation of a 3rd-order tensor is shown in Fig. 3.

Several methods have been developed to compute the CP decomposition, for example, the alternating least squares (ALS) [8], [17] (as well as many of its derivatives) or the optimization methods such as CPOPT [11]. Most of them solve the optimization problem of minimizing the Frobenius norm

[3]CANDECOMP (canonical decomposition) by Carroll and Chang [8] and PARAFAC (parallel factors) by Harshman [17]. They are found independently in history, but the underlying algorithms are the same.

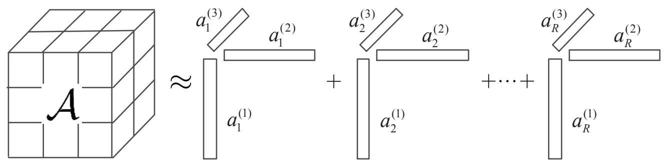

Fig. 3. A CP decomposition of a 3rd-order tensor.

of the difference between the original tensor and its rank-R approximation

$$\min f(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(d)}) \triangleq \frac{1}{2}\left\|\mathcal{A}-[\![\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(d)}]\!]\right\|_F^2. \tag{12}$$

The ALS algorithm iteratively optimizes one factor matrix $\mathbf{A}^{(i)}$ at a time, by holding all other factors fixed and solving the linear least square problem

$$\min_{\mathbf{A}^{(i)}} f(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(d)}) \tag{13}$$

for the updated $\mathbf{A}^{(i)}$. Alternatively, CPOPT calculates the gradient of the objective function f in (12) and uses the generic nonlinear conjugate gradient method to optimize (12). For both ALS and CPOPT, the rank R should be prescribed and is fixed during the computation. It is reported in [11] that the computational complexities for both ALS and CPOPT to approximate an Nth-order tensor $\mathcal{A}\in\mathbb{R}^{n_1\times\cdots\times n_N}$ are $O(NQR)$ per iteration, where $Q=\prod_{i=1}^{N} n_i$. It is also mentioned in [11] that ALS is several times faster than CPOPT in general. However CPOPT shows an "essentially perfect" accuracy compared with ALS. A review of different CP methods also can be found in [9].

III. Tensor-form modeling of nonlinear systems

To begin with, we give an equivalent tensor-based modeling of the nonlinear system (2). Recall the definitions of G_i and C_i in (3), it is readily found that these coefficient matrices are respectively 1-mode matricizations of $(i+1)$th-order tensors $\mathcal{G}_i$ and $\mathcal{C}_i$,

$$\mathcal{G}_i,\ \mathcal{C}_i \in \mathbb{R}^{\overbrace{n\times\cdots\times n}^{i+1}}, \tag{14}$$

where the elements $(\mathcal{G}_i)_{j_0j_1\cdots j_i}$ and $(\mathcal{C}_i)_{j_0j_1\cdots j_i}$ are coefficients of the $\Pi_{k=1}^{i}x_{j_k}$ term in G_i and C_i, respectively. For instance, G_2 is an $n\times n^2$ matrix while $\mathcal{G}_2$ is a 3rd-order $n\times n\times n$ tensor, i.e., G_2 is the 1-mode matricization of $\mathcal{G}_2$.

According to Proposition 3.7 in [13], the Kronecker matrix products in (2) can be represented by the tensor mode multiplication via $G_i(x^{\textcircled{i}})=\mathcal{G}_i\times_2 x^T\times_3 x^T\cdots\times_i x^T\times_{i+1}x^T$ and $C_i(x^{\textcircled{i}})=\mathcal{C}_i\times_2 x^T\times_3 x^T\cdots\times_i x^T\times_{i+1}x^T$. Therefore, (2) is equivalent to

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\left[\mathcal{C}_1\times_2 x^T+\mathcal{C}_2\times_2 x^T\times_3 x^T+\mathcal{C}_3\times_2 x^T\times_3 x^T\times_4 x^T+\cdots\right]\\&+\mathcal{G}_1\times_2 x^T+\mathcal{G}_2\times_2 x^T\times_3 x^T+\mathcal{G}_3\times_2 x^T\times_3 x^T\times_4 x^T+\cdots=Bu.\end{aligned} \tag{15}$$

The key to the tensor-form modeling is to pre-decompose these high dimensional tensors via CP. In practical circuit systems, in spite of the growing dimensionality, high-order nonlinear coefficients $\mathcal{G}_i$ and $\mathcal{C}_i$ (G_i and C_i) are almost always sparse. Therefore it is advantageous to make a rank-R CP approximation of $\mathcal{G}_i$ or $\mathcal{C}_i$ for a relatively small R. In other words, we can use a few rank-1 tensors to express $\mathcal{G}_i$ and $\mathcal{C}_i$ by

$$\begin{aligned}\mathcal{G}_i &\approx [\![\mathbf{G}_i^{(1)},\ldots,\mathbf{G}_i^{(i+1)}]\!]=\sum_{r=1}^{r_{g,i}} g_{i,r}^{(1)}\circ\cdots\circ g_{i,r}^{(i+1)},\\ \mathcal{C}_i &\approx [\![\mathbf{C}_i^{(1)},\ldots,\mathbf{C}_i^{(i+1)}]\!]=\sum_{r=1}^{r_{c,i}} c_{i,r}^{(1)}\circ\cdots\circ c_{i,r}^{(i+1)},\end{aligned} \tag{16}$$

where $i=2,\ldots,N$, $r_{g,i}$ and $r_{c,i}$ are the tensor ranks of $\mathcal{G}_i$ and $\mathcal{C}_i$, respectively, $g_{i,r}^{(k)},c_{i,r}^{(k)}\in\mathbb{R}^n$, $\mathbf{G}_i^{(k)}=[g_{i,1}^{(k)},\ldots,g_{i,r_{g,i}}^{(k)}]\in\mathbb{R}^{n\times r_{g,i}}$ and $\mathbf{C}_i^{(k)}=[c_{i,1}^{(k)},\ldots,c_{i,r_{c,i}}^{(k)}]\in\mathbb{R}^{n\times r_{c,i}}$, for $k=1,\ldots,i+1$. It should be noticed that different permutations of indices can result in the same polynomial term. For example, term x_1x_2 can be represented by any combination of $\alpha x_1x_2+(1-\alpha)x_2x_1$. Therefore, the high-order tensors are not unique for a specific nonlinear system and the consequent low-rank approximations (16) could be very different. Nonetheless, we use the tensors with minimum nonzero entries in our implementation, as they tend to be sparser such that lower rank approximations are usually available.

Using the CP structure (16), the original nonlinear system (15) can be approximated by absorbing tensor products of x into the factor matrices

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\Big[\mathcal{C}_1\times_2 x^T+[\![\mathbf{C}_2^{(1)},x^T\mathbf{C}_2^{(2)},x^T\mathbf{C}_2^{(3)}]\!]\\&+[\![\mathbf{C}_3^{(1)},x^T\mathbf{C}_3^{(2)},x^T\mathbf{C}_3^{(3)},x^T\mathbf{C}_3^{(4)}]\!]+\cdots\Big]+\mathcal{G}_1\times_2 x^T\\&+[\![\mathbf{G}_2^{(1)},x^T\mathbf{G}_2^{(2)},x^T\mathbf{G}_2^{(3)}]\!]+[\![\mathbf{G}_3^{(1)},x^T\mathbf{G}_3^{(2)},x^T\mathbf{G}_3^{(3)},x^T\mathbf{G}_3^{(4)}]\!]+\cdots\\&=Bu.\end{aligned} \tag{17}$$

Applying (11), the 1-mode matricization of (17) is simply

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}\Big[C_1x+\mathbf{C}_2^{(1)}\left(x^T\mathbf{C}_2^{(3)}\odot x^T\mathbf{C}_2^{(2)}\right)^T+\mathbf{C}_3^{(1)}\Big(x^T\mathbf{C}_3^{(4)}\odot x^T\mathbf{C}_3^{(3)}\\&\odot x^T\mathbf{C}_3^{(2)}\Big)^T+\cdots\Big]+G_1x+\mathbf{G}_2^{(1)}\left(x^T\mathbf{G}_2^{(3)}\odot x^T\mathbf{G}_2^{(2)}\right)^T\\&+\mathbf{G}_3^{(1)}\left(x^T\mathbf{G}_3^{(4)}\odot x^T\mathbf{G}_3^{(3)}\odot x^T\mathbf{G}_3^{(2)}\right)^T+\cdots=Bu,\end{aligned} \tag{18}$$

and its corresponding 1-mode matricized Volterra subsystems are given by

$$\frac{\mathrm{d}}{\mathrm{d}t}[C_1x_1]+G_1x_1=Bu, \tag{19a}$$

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}[C_1x_2]+G_1x_2=-\frac{\mathrm{d}}{\mathrm{d}t}\left[\mathbf{C}_2^{(1)}\left(x_1^T\mathbf{C}_2^{(3)}\odot x_1^T\mathbf{C}_2^{(2)}\right)^T\right]\\&-\mathbf{G}_2^{(1)}\left(x_1^T\mathbf{G}_2^{(3)}\odot x_1^T\mathbf{G}_2^{(2)}\right)^T,\end{aligned} \tag{19b}$$

$$\begin{aligned}&\frac{\mathrm{d}}{\mathrm{d}t}[C_1x_3]+G_1x_3=-\frac{\mathrm{d}}{\mathrm{d}t}\Big[\mathbf{C}_3^{(1)}\left(x_1^T\mathbf{C}_3^{(4)}\odot x_1^T\mathbf{C}_3^{(3)}\odot x_1^T\mathbf{C}_3^{(2)}\right)^T\\&+\mathbf{C}_2^{(1)}(\overline{x_{i_1}^T\mathbf{C}_2^{(3)}\odot x_{i_2}^T\mathbf{C}_2^{(2)}})_3^T\Big]-\mathbf{G}_3^{(1)}\left(x_1^T\mathbf{G}_3^{(4)}\odot x_1^T\mathbf{G}_3^{(3)}\odot x_1^T\mathbf{G}_3^{(2)}\right)^T\\&-\mathbf{G}_2^{(1)}(\overline{x_{i_1}^T\mathbf{G}_2^{(3)}\odot x_{i_2}^T\mathbf{G}_2^{(2)}})_3^T,\end{aligned} \tag{19c}$$

and so on.

IV. TNMOR

Without loss of generality, we start from (18) and (19) to derive TNMOR. It can be observed that (19) is a series of linear systems where x_1 is solved by the first equation with input u, x_2 is the solution to the second linear system with the input dependent on x_1, and similarly x_3 is solved in the third system with its input dependent on x_1 and x_2.

Similar to [1], [5]–[7], the frequency-domain transfer function of (19a) is given by

$$H_1(s) = (sG_1^{-1}C_1 + I)^{-1}G_1^{-1}B \triangleq (-sA_1 + I)^{-1}B_1. \quad (20)$$

To match up to k_1th-order moments of (20), its projection space V_{k_1} is the Krylov subspace of $\mathcal{K}_{k_1+1}(A_1, B_1)$ if $H_1(s)$ is expanded around the origin, where $\mathcal{K}_m(A, p) = \text{span}\{p, Ap, \ldots, A^{m-1}p\}$. The Krylov subspace can be efficiently calculated by the block Arnoldi iteration [1].

The 2nd-order subsystem (19b) can be recast into

$$\begin{aligned}&\begin{bmatrix} C_1 & -I \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \dot{x}_2 \\ \dot{x}_{2e} \end{bmatrix} + \begin{bmatrix} G_1 & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} x_2 \\ x_{2e} \end{bmatrix} \\ &= -\begin{bmatrix} \mathbf{G}_2^{(1)} & 0 \\ 0 & \mathbf{C}_2^{(1)} \end{bmatrix}\begin{bmatrix} \left(x_1^T\mathbf{G}_2^{(3)} \odot x_1^T\mathbf{G}_2^{(2)}\right)^T \\ \left(x_1^T\mathbf{C}_2^{(3)} \odot x_1^T\mathbf{C}_2^{(2)}\right)^T \end{bmatrix}, \end{aligned} \quad (21)$$

Consequently, (21) could be treated as a linear system with $r_{g,2} + r_{c,2}$ inputs and its transfer function reads

$$\begin{aligned} H_2(s) &= \left(s\begin{bmatrix} G_1^{-1}C_1 & -G_1^{-1} \\ 0 & 0 \end{bmatrix} + I\right)^{-1}\begin{bmatrix} -G_1^{-1}\mathbf{G}_2^{(1)} & 0 \\ 0 & -\mathbf{C}_2^{(1)} \end{bmatrix} \\ &\triangleq (-sA_2 + I)^{-1}B_2. \end{aligned} \quad (22)$$

Suppose k_2th-order moments of $H_2(s)$ are matched, its Krylov subspace is obtained by $\mathcal{K}_{k_2+1}(A_2, B_2)$. Thus, we have V_{k_2} to be the first n rows of $\mathcal{K}_{k_2+1}(A_2, B_2)$ (noticing $H_2(s)$ is a $2n \times 1$ vector).

For the 3rd-order subsystem (19c), similarly, it could be represented by another linear system

$$\begin{aligned}&\begin{bmatrix} C_1 & -I \\ 0 & 0 \end{bmatrix}\begin{bmatrix} \dot{x}_3 \\ \dot{x}_{3e} \end{bmatrix} + \begin{bmatrix} G_1 & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} x_3 \\ x_{3e} \end{bmatrix} \\ &= -\begin{bmatrix} \mathbf{G}_3^{(1)} & 0 & \mathbf{G}_2^{(1)} & 0 \\ 0 & \mathbf{C}_3^{(1)} & 0 & \mathbf{C}_2^{(1)} \end{bmatrix}\begin{bmatrix} \left(x_1^T\mathbf{G}_3^{(4)} \odot x_1^T\mathbf{G}_3^{(3)} \odot x_1^T\mathbf{G}_3^{(2)}\right)^T \\ \left(x_1^T\mathbf{C}_3^{(4)} \odot x_1^T\mathbf{C}_3^{(3)} \odot x_1^T\mathbf{C}_3^{(2)}\right)^T \\ \overline{\left(x_{i_1}^T\mathbf{G}_2^{(3)} \odot x_{i_2}^T\mathbf{G}_2^{(2)}\right)_3^T} \\ \overline{\left(x_{i_1}^T\mathbf{C}_2^{(3)} \odot x_{i_2}^T\mathbf{C}_2^{(2)}\right)_3^T} \end{bmatrix} \end{aligned} \quad (23)$$

with $r_{g,3} + r_{c,3} + r_{g,2} + r_{c,2}$ inputs. Moreover, the 3rd-order transfer function is given by

$$\begin{aligned} H_3(s) &= \left(s\begin{bmatrix} G_1^{-1}C_1 & -G_1^{-1} \\ 0 & 0 \end{bmatrix} + I\right)^{-1} \\ &\cdot \begin{bmatrix} -G_1^{-1}\mathbf{G}_3^{(1)} & 0 & -G_1^{-1}\mathbf{G}_2^{(1)} & 0 \\ 0 & -\mathbf{C}_3^{(1)} & 0 & -\mathbf{C}_2^{(1)} \end{bmatrix} \\ &\triangleq (-sA_2 + I)^{-1}\left[\, B_3 \quad B_2 \,\right]. \end{aligned} \quad (24)$$

Consequently, the Krylov subspace of the 3rd-order subsystem should be $\mathcal{K}_{k_3+1}(A_2, [B_3 \;\; B_2]) = \mathcal{K}_{k_3+1}(A_2, B_3) \cup \mathcal{K}_{k_3+1}(A_2, B_2)$, if the number of moments being matched is k_3. However, it is readily seen that if $k_3 \leq k_2$, we have $\mathcal{K}_{k_3+1}(A_2, B_2) \subseteq \mathcal{K}_{k_2+1}(A_2, B_2)$. Since $\mathcal{K}_{k_2+1}(A_2, B_2)$ has already been obtained from $H_2(s)$, $\mathcal{K}_{k_3+1}(A_2, B_2)$ does not need to be recomputed. Therefore, only $\mathcal{K}_{k_3+1}(A_2, B_3)$ should be counted at this stage and we denote its first n rows to be V_{k_3}.

Higher order linear transfer functions can be obtained in a similar way. The ith-order projector V_{k_i} is the first n rows of $\mathcal{K}_{k_i+1}(A_2, B_i)$, where $B_i = \begin{bmatrix} -G_1^{-1}\mathbf{G}_i^{(1)} & 0 \\ 0 & -\mathbf{C}_i^{(1)} \end{bmatrix}$.

The reducing projector for the nonlinear system is the orthogonal basis of all V_{k_i}s, denoted by $V = \text{orth}([V_{k_1}, V_{k_2}, \ldots])$. The size of an Nth-order ROM is in $O(k_1 l + k_2(r_{g,2} + r_{c,2}) + k_3(r_{g,3} + r_{c,3}) + \cdots + k_N(r_{g,N} + r_{c,N})) = O(Nkr)$, where $k = \max\{k_1, \ldots, k_N\}$ and $r = \max\{l, r_{g,2} + r_{c,2}, \ldots, r_{g,N} + r_{c,N}\}$. Comparing with $O(k^{2N-1}l^N)$ in the standard projection approach and $O(k^{N+1}l^N)$ in NORM, a slimmer ROM can be achieved if low-rank CP of the tensors are available.

Finally, the tensor-based ROM is given by the following projection

$$\begin{aligned} &\tilde{x} = V^T x, \quad \tilde{B} = V^T B, \quad \tilde{L} = V^T L, \\ &\tilde{\mathcal{G}}_i = [\![\tilde{\mathbf{G}}_i^{(1)}, \ldots, \tilde{\mathbf{G}}_i^{(i+1)}]\!] = [\![V^T\mathbf{G}_i^{(1)}, \ldots, V^T\mathbf{G}_i^{(i+1)}]\!], \\ &\tilde{\mathcal{C}}_i = [\![\tilde{\mathbf{C}}_i^{(1)}, \ldots, \tilde{\mathbf{C}}_i^{(i+1)}]\!] = [\![V^T\mathbf{C}_i^{(1)}, \ldots, V^T\mathbf{C}_i^{(i+1)}]\!], \\ &\tilde{G}_1 = V^T G_1 V, \quad \tilde{C}_1 = V^T C_1 V, \quad i = 2, \ldots, N, \end{aligned} \quad (25)$$

which looks similar to (6) except for the tensor CP structure. By utilizing this structure, only the $\tilde{\mathbf{G}}_i^{(k)}$ and $\tilde{\mathbf{C}}_i^{(k)}$ matrices need to be stored, and simulation of the ROM can be significantly speeded up as will be seen in the next section, as long as low-rank CP approximations of $\mathcal{G}_i$ and $\mathcal{C}_i$ are available. Algorithm 1 summarizes the TNMOR assuming a DC expansion point.

The computational complexity of TNMOR is dominated by the CP decompositions of $\mathcal{C}_i$ and $\mathcal{G}_i$. Any CP method can be used to extract the rank-1 factors. We use CPOPT to compute the CP in TNMOR. Although CPOPT is not as remarkably fast as ALS, we find in practice that CPOPT can always achieve a better accuracy under the same rank. The computational cost for CPOPT to optimize a rank-$r_{g,N}$ approximation of $\mathcal{G}_N$ is in $O(Nn^{N+1}r_{g,N})$ per iteration. By contrast, the costs for NORM and the standard projection method to calculate the Krylov starting vectors for an Nth-order subsystem are in $O(k^N n^{2N})$ and $O(k^{3N^2}l^{N^2}n^N)$, respectively. It should be noticed that the CP decompositions only need to be done once and the resulting reduced CP structure can be used in different on-going simulations.

Another key issue is how to heuristically prescribe/estimate the rank for each tensor. It is readily found that accurate low-rank CP approximations of the high dimensional tensors is critical to the proposed method. Although it will be shown in Section V that the size of the ROM and the time complexity for its simulation are proportional to the tensor ranks r_Gs and r_Cs, the efficiency of the proposed model will be

Algorithm 1 TNMOR Algorithm

```
Input: $N, G_i, C_i, B, L, k_i, k_N \leq k_{N-1} \leq \cdots \leq k_1$
Output: $G_1, C_1, \tilde{\mathcal{G}}_i, \tilde{\mathcal{C}}_i, \tilde{B}, \tilde{L}, i = 2, \ldots, N$
1: for $i = 2$ to $N$ do
2:     $\tilde{\mathcal{G}}_i = G_i$; $\tilde{\mathcal{C}}_i = C_i$;
3:     $[\![\mathbf{G}_i^{(1)}, \ldots, \mathbf{G}_i^{(i+1)}]\!] = \text{CP}(\tilde{\mathcal{G}}_i)$;
4:     $[\![\mathbf{C}_i^{(1)}, \ldots, \mathbf{G}_i^{(i+1)}]\!] = \text{CP}(\tilde{\mathcal{C}}_i)$;
5: end for
6: $A_1 = -G_1^{-1}C_1$; $B_1 = G_1^{-1}B$;
7: $V_{k_1} = \mathcal{K}_{k_1+1}(A_1, B_1)$;
8: for $i = 2$ to $N$ do
9:     $A_2 = \begin{bmatrix} G_1^{-1}C_1 & -G_1^{-1} \\ 0 & 0 \end{bmatrix}$; $B_i = \begin{bmatrix} -G_1^{-1}\mathbf{G}_i^{(1)} & 0 \\ 0 & -\mathbf{C}_i^{(1)} \end{bmatrix}$;
10:    $V_{k_i} = \mathcal{K}_{k_i+1}(A_2, B_i)$;
11: end for
12: $V = \text{orth}([V_{k_1}, V_{k_2}, \ldots, V_{k_N}])$;
13: $\tilde{G}_1 = V^T G_1 V$; $\tilde{C}_1 = V^T C_1 V$; $\tilde{B} = V^T B$; $\tilde{L} = V^T L$;
14: for $i = 2$ to $N$ do
15:     $\tilde{\mathcal{G}}_i = [\![V^T\mathbf{G}_i^{(1)}, \ldots, V^T\mathbf{G}_i^{(i+1)}]\!]$;
16:     $\tilde{\mathcal{C}}_i = [\![V^T\mathbf{C}_i^{(1)}, \ldots, V^T\mathbf{C}_i^{(i+1)}]\!]$;
17: end for
```

compromised if large ranks are required to approximate the tensors. Unfortunately, unlike matrices, there are no feasible algorithms to determine the rank of a specific tensor; actually it is an NP-hard problem [18]. Nonetheless, a loose upper bound on the maximum rank of a sparse tensor is the number of its nonzero entries. Furthermore, in practical circuit examples, we find that $\frac{m}{2}$ to $\frac{m}{3}$ would be a possible rank to approximate a sparse tensor with m nonzero elements. As shown by examples in Section VI, this empirical rank works well for systems with fewer than $O(n)$ nonlinear components. Meanwhile, there is a significant amount of analog and RF circuits with a few (usually in $O(1)$) transistors, for instance, amplifiers and mixers, which would potentially admit low-rank tensor approximations. For systems with more than $O(n)$ nonlinearities, such as ring oscillators or most discretized nonlinear partial differential equations, however, no reduction can be achieved by TNMOR if any of the tensor ranks exceeds n. It should be remarked that such systems may still be reduced by NORM, if relatively low order of moments is matched.

V. ROM SIMULATION

Here we show that whenever low-rank tensor approximations of $\mathcal{G}_i$ and $\mathcal{C}_i$ exist, TNMOR can help to accelerate simulation and avoids the exponential growth of the memory requirement, versus the reduced but dense models generated by the standard projection method or NORM. We describe the time complexity by means of the two key processes in circuit simulation, namely, function evaluation and calculation of the Jacobian matrices. For the ease of illustration, we assume the conductance matrix $C_1 = I$ and all higher order $C_i = \mathbf{0}$, $i = 2, \ldots, N$. Extension to general cases is straightforward.

A. Function evaluation

Rewrite (2) for the ROM

$$f(\tilde{x}) \triangleq \dot{\tilde{x}} = -\tilde{G}_1\tilde{x} - \tilde{G}_2\tilde{x}^{\textcircled{2}} - \tilde{G}_3\tilde{x}^{\textcircled{3}} - \cdots + \tilde{B}u, \tag{26}$$

where $f(\tilde{x})$ is the function to be evaluated in the simulation. Using the ROM in (25), the equivalent 1-mode matricization of (26) is

$$\begin{aligned} f(\tilde{x}) = &-\tilde{G}_1\tilde{x} - \tilde{\mathbf{G}}_2^{(1)}\left(\tilde{x}^T\tilde{\mathbf{G}}_2^{(3)} \odot \tilde{x}^T\tilde{\mathbf{G}}_2^{(2)}\right)^T \\ &- \tilde{\mathbf{G}}_3^{(1)}\left(\tilde{x}^T\tilde{\mathbf{G}}_3^{(4)} \odot \tilde{x}^T\tilde{\mathbf{G}}_3^{(3)} \odot \tilde{x}^T\tilde{\mathbf{G}}_3^{(2)}\right)^T - \cdots + \tilde{B}u. \end{aligned} \tag{27}$$

It should be noticed that all $\tilde{x}^T\tilde{\mathbf{G}}_i^{(k)}$ and $\tilde{x}^T\tilde{\mathbf{C}}_i^{(k)}$ are row vectors, therefore $\odot$ corresponds to the element-by-element multiplication between matrices. If up to the Nth-order nonlinearity is included and the size of the ROM is q, the time complexity for evaluating (27) is in $O(N^2qr_g)$ where $r_g = \max\{r_{g,2}, \ldots, r_{g,N}\}$, while it is in $O(q^{N+2})$ for the model (6) used in the standard projection approach and NORM.

B. Jacobian matrix

Consider the Jacobian matrix of $f(\tilde{x})$ which is often used in time-domain simulation

$$\begin{aligned} J_f(\tilde{x}) \triangleq &-\tilde{G}_1 - \tilde{G}_2(\tilde{x} \otimes I + I \otimes \tilde{x}) \\ &- \tilde{G}_3(\tilde{x} \otimes \tilde{x} \otimes I + \tilde{x} \otimes I \otimes \tilde{x} + I \otimes \tilde{x} \otimes \tilde{x}) - \cdots . \end{aligned} \tag{28}$$

Its equivalent 1-mode matricization is given by

$$\begin{aligned} J_f(\tilde{x}) = &-\tilde{G}_1 - \tilde{\mathbf{G}}_2^{(1)}\left(\tilde{x}^T\tilde{\mathbf{G}}_2^{(3)} \odot \tilde{\mathbf{G}}_2^{(2)} + \tilde{\mathbf{G}}_2^{(3)} \odot \tilde{x}^T\tilde{\mathbf{G}}_2^{(2)}\right)^T \\ &- \tilde{\mathbf{G}}_3^{(1)}\left(\tilde{x}^T\tilde{\mathbf{G}}_3^{(4)} \odot \tilde{x}^T\tilde{\mathbf{G}}_3^{(3)} \odot \tilde{\mathbf{G}}_3^{(2)} + \tilde{x}^T\tilde{\mathbf{G}}_3^{(4)} \odot \tilde{\mathbf{G}}_3^{(3)} \odot \tilde{x}^T\tilde{\mathbf{G}}_3^{(2)}\right. \\ &\left. + \tilde{\mathbf{G}}_3^{(4)} \odot \tilde{x}^T\tilde{\mathbf{G}}_3^{(3)} \odot \tilde{x}^T\tilde{\mathbf{G}}_3^{(2)}\right)^T - \cdots . \end{aligned} \tag{29}$$

The complexity for the standard projection method or NORM to calculate (28) is in $O(Nq^{N+2})$. For the tensor formulation, the complexity for evaluating (29) is further reduced to $O(Nq(N+q)r_g)$.

C. Space complexity

In our proposed method, the amount of memory space for the ROM is determined by the factor matrices in (25), which should be in $O(Nqr)$, where $r = \max\{r_{g,2} + r_{c,2}, \ldots, r_{g,N} + r_{c,N}\}$. For the existing standard projection approach and NORM, the storage consumption is dominated by the matrices $\tilde{G}_N$ and $\tilde{C}_N$ whose numbers of elements are in $O(q^{N+1})$.

VI. NUMERICAL EXAMPLES

In this section, TNMOR is demonstrated and compared with the standard projection approach and NORM. All experiments are implemented in MATLAB on a desktop with an Intel i5 2500@3.3GHz CPU and 16GB RAM. To fairly present the results, all time-domain transient analyses are solved by the trapezoid discretization with fixed step sizes. In the simulations of the original system, the ROMs of the standard projection and NORM approaches, (26) and (28) are used to evaluate the

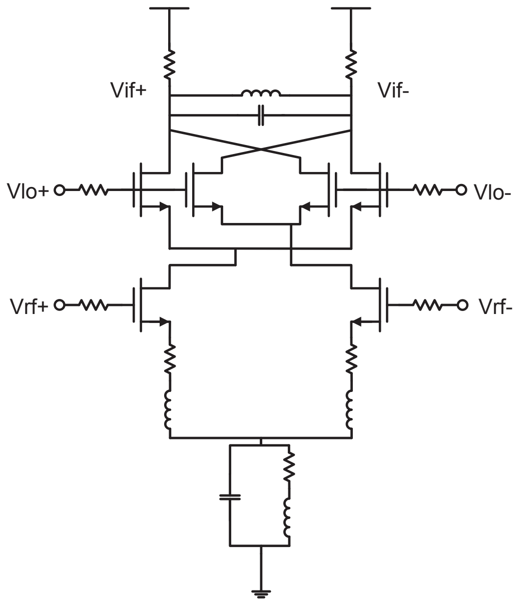

Fig. 4. A double-balanced mixer.

TABLE I. ROM SIZES AND CPU TIMES OF MOR FOR THE MIXER (SMALL SIGNAL)

Method	k_1	k_2	k_3	CPU time	size of ROM
standard projection [5], [6]	1	1	—	0.09s	52
NORM [7]	2	2	—	0.56s	30
TNMOR	2	2	0	4.5s	39

TABLE II. CPU TIMES AND ERRORS OF TRANSIENT SIMULATIONS FOR THE MIXER (SMALL SIGNAL)

Transient	full model	standard projection [5], [6]	NORM [7]	TNMOR
size	93	52	30	39
CPU time	39s	2200s	270s	17s
speedup	—	—	—	3.2x
error	—	3.98%	3.24%	1.24%

TABLE III. CPU TIMES AND ERRORS OF PERIODIC STEADY-STATE SIMULATIONS FOR THE MIXER (SMALL SIGNAL)

Periodic steady state	full model	standard projection [5], [6]	NORM [7]	TNMOR
size	93	52	30	39
CPU time	67s	4900s	410s	21s
speedup	—	—	—	3.2x
error	—	0.93%	0.71%	0.38%

functions and Jacobian matrices, while (27) and (29) are used for the tensor-based model. The CP in TNMOR is computed by the CPOPT algorithm provided in the MATLAB Tensor Toolbox [11], [12], [19].

A. A double-balanced mixer

First, we study a double-balanced mixer circuit in Fig. 4 [20], where $V_{\text{rf}}(= V_{\text{rf+}} - V_{\text{rf-}})$ and $V_{\text{lo}}(= V_{\text{lo+}} - V_{\text{lo-}})$ are the RF and local oscillator (LO) inputs, respectively. We assume V_{rf} and V_{lo} are both sinusoidal and their frequencies are 2GHz and 200MHz, respectively. $V_{\text{if+}}$ and $V_{\text{if-}}$ are the intermediate-frequency (IF) outputs. The size n of the original system is 93. Firstly, we assume a relatively small V_{lo} swing so that the nonlinear system can be approximated by its 3rd-order Taylor expansion

$$\frac{\mathrm{d}}{\mathrm{d}t}[C_1 x] + G_1 x + G_2 x^{②} + G_3 x^{③} = Bu, \tag{30}$$

where the numbers of nonzero elements in $\mathcal{G}_2$ and $\mathcal{G}_3$ are 16 and 30, respectively.

Then, the standard projection approach, NORM and TNMOR are applied to (30). It should be noticed that all methods are expanded at the frequencies 2GHz and 200MHz. The size of the ROM, the order of the moments k_i matched in each subsystem, and the CPU times for each MOR method are listed in Table I. It can be noticed that due to the curse of dimensionality, the standard projection approach generates a larger (and denser) ROM but fewer orders of moments are persevered. TNMOR takes 1.3s to optimize a best rank-6 ($r_{g,2} = 6$) approximation of $\mathcal{G}_2$ with the error $\frac{\|\tilde{\mathcal{G}}_2 - \mathcal{G}_2\|_F}{\|\mathcal{G}_2\|_F} = 4.2 \times 10^{-4}$, and 3.1s for a best rank-9 ($r_{g,3} = 9$) approximation of $\mathcal{G}_3$ with an error 1.2×10^{-4}.

A transient simulation from 0 to $T = 25$ns with a $\Delta t = 5$ps step size is performed on each ROM. The runtimes and overall errors of different methods are summarized in Table II. The overall error is defined as the relative error between the vector of $V_{\text{if+}}$ at subsequent timesteps $V = [V_{\text{if+}}(0), V_{\text{if+}}(\Delta t), \ldots, V_{\text{if+}}(T)]^T$ of the ROM and the corresponding vector computed with the full model, i.e., $\frac{\|V - V_{full}\|_2}{\|V_{full}\|_2}$. The first 7.5ns of the transient waveforms of $V_{\text{if+}}$ and their relative errors are plotted in Figs. 5(a) and 5(b), respectively.

Next, the periodic steady-state analyses of different models are achieved by a shooting Newton method-based periodic steady-state simulator. The CPU times and the overall errors of the frequency responses between 0 to 4GHz are listed in Table III. The relative errors of the frequency responses are plotted in Fig. 5(c).

It is shown in Tables II and III that although TNMOR generates a larger ROM than NORM (39 versus 30), its transient and periodic steady-state analyses are faster due to the efficient algorithm of function and Jacobian evaluations. In contrast, simulations of the dense ROMs generated by NORM and the standard projection approach are much slower than the original large but sparse system, indicating that these methods are impractical for strongly nonlinear systems (3rd-order nonlinearity in this example). This is mainly because though both (26) and (28) have been used in the simulations, the Kronecker powers in (26) and (28) of the original system never have to be explicitly computed due to the sparsity of the original G_i and C_i matrices. Moreover, comparing to NORM, the proposed method provides a competitive accuracy as can be seen in Fig. 5.

Practically, mixers are often modeled by periodic time-varying systems due to the existence of large V_{lo} signals [5]–[7], [21]. Fortunately, TNMOR can be easily extended to periodic time-varying systems as well. Following [5]–[7],

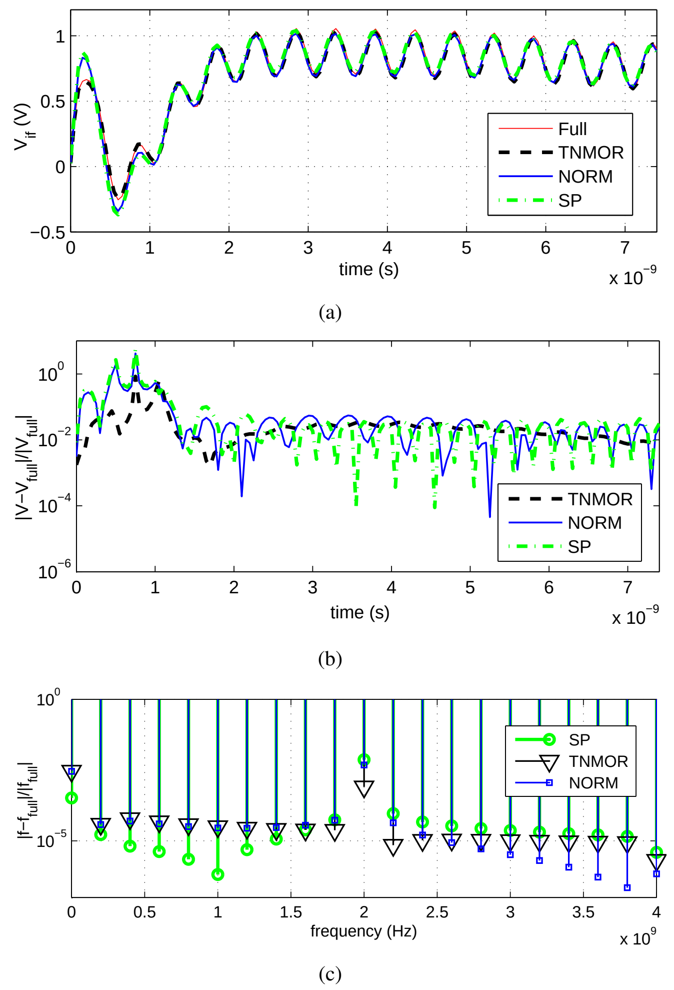

Fig. 5. (a) Transient waveforms of the full model and ROMs. (b) Relative errors of the transient simulations. (c) Relative errors of the periodic steady-state simulations. (SP is the acronym for “standard projection” [5], [6].)

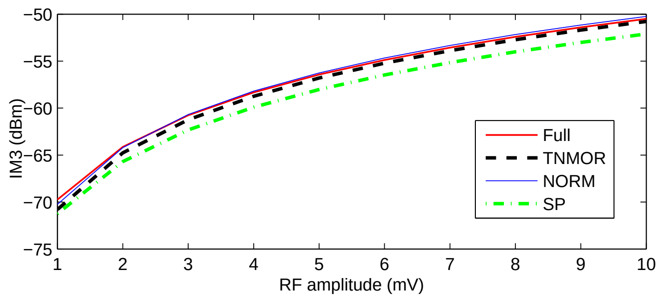

Fig. 6. IM3 of the full model and ROMs. (SP is the acronym for “standard projection” [5], [6].)

TABLE IV. ROM SIZES AND CPU TIMES OF MOR FOR THE MIXER (LARGE SIGNAL)

Method	k_1	k_2	k_3	CPU time	size of ROM
standard projection [5], [6]	1	1	—	0.53s	46
NORM-mp [7]	2	1	0	1.2s	28
TNMOR	2	2	0	28s	78

TABLE V. CPU TIMES OF IM3 TESTS FOR THE MIXER (LARGE SIGNAL)

Transient	full model	standard projection [5], [6]	NORM [7]	TNMOR
size	930	46	28	78
CPU time	146 ± 8s	120 ± 12s	22 ± 2s	15 ± 2s
speedup	—	1.2x	6.6x	9.7x

suppose all higher order $C_i = \mathbf{0}$, all the system matrices in (19) become T_{lo}-periodic where T_{lo} is the period of V_{lo}. Next, a uniform backward-Euler discretization over sampling points $[t_1, t_2, \ldots, t_M]$ where $t_i = \frac{i}{M} T_{\text{lo}}$ is applied on each linear time-varying system in (19), resulting a set of LTV systems with transfer functions $\hat{H}_i(s)$

$$[\hat{J}_1 + s\hat{C}_1]\hat{H}_i(s) = \hat{B}_i, \tag{31}$$

where

$$\begin{aligned}\hat{H}_i(s) &= \left[H_i^T(t_1,s)\ H_i^T(t_2,s)\ \ldots\ H_i^T(t_M,s)\right]^T,\\ \hat{B}_1 &= \left[B^T(t_1)\ B^T(t_2)\ \ldots\ B^T(t_M)\right]^T,\\ \hat{G}_1 &= \begin{bmatrix} G_1(t_1) & & & \\ & G_1(t_2) & & \\ & & \ddots & \\ & & & G_1(t_M)\end{bmatrix},\end{aligned} \tag{32}$$

$$\begin{aligned}\hat{C}_1 &= \begin{bmatrix} C_1(t_1) & & & \\ & C_1(t_2) & & \\ & & \ddots & \\ & & & C_1(t_M)\end{bmatrix},\\ \Delta &= \frac{M}{T_{\text{lo}}}\begin{bmatrix} I & & & -I \\ -I & I & & \\ & \ddots & \ddots & \\ & & -I & I\end{bmatrix},\\ \hat{J}_1 &= \hat{G}_1 + \Delta\hat{C}_1\end{aligned} \tag{33}$$

and

$$\hat{B}_i = -\hat{\mathbf{G}}_i^{(1)} = -\begin{bmatrix} \mathbf{G}_i^{(1)}(t_1) & & & \\ & \mathbf{G}_i^{(1)}(t_2) & & \\ & & \ddots & \\ & & & \mathbf{G}_i^{(1)}(t_M)\end{bmatrix}, \quad i \geq 2. \tag{34}$$

We omit the detailed derivation as it is straightforward. Similarly, the projector is obtained by collocating the moments of each order subsystem in (31). It should be noticed that the factor matrices at each sampling point can be computed individually via CP decompositions. Therefore the computational complexity is proportional to M.

Next, we reformulate the mixer by a 3rd-order nonlinear time-varying system under a large signal V_{lo}. The 930-variable system is expanded around $M = 10$ operating points. This

TABLE VI. ROM SIZES AND CPU TIMES OF MOR FOR THE BIOCHEMICAL SYSTEM

Method	k_1	k_2	k_3	CPU time	size of ROM
standard projection [5], [6]	1	0	—	0.02s	39
NORM [7]	2	1	0	1.2s	45
TNMOR	2	2	0	11s	39

time, TNMOR, multi-point NORM (NORM-mp) and the standard projection method are expanded at 2GHz. The sizes of ROMs and CPU times are listed in Table IV. The CPU time of TNMOR is mainly spent on sequential CP decompositions, which could be further reduced if multi-core parallelization is enabled.

These ROMs are simulated for 3rd-order intermodulation tests. The LO and RF frequencies are fixed while we sweep the amplitude of the sinusoidal RF input from 1mV to 10mV. Fig. 6 shows the 3rd-order intermodulation product (IM3) results of the original system and ROMs. The CPU times are summarized in Table V, where they are written as $a \pm b$ where a is the average value and b is the sample standard deviation. From Fig. 6, good agreement can be observed for the ROMs generated by NORM-mp and TNMOR. NORM-mp achieves a smaller size because it only preserves the values of nonlinear transfer functions at specific points, which is particularly useful for matching high-order distortions. Meanwhile, TNMOR still demonstrates comparable accuracy and better efficiency due to the benefit of the tensor framework.

B. A biochemical reaction system

The second example is a sparse biochemical reaction system model adapted from [22]. The system is generated by a random 2nd-order polynomial system with a $\frac{1}{1+x}$ function

$$\frac{\mathrm{d}}{\mathrm{d}t}x + G_1 x + G_2 x^{\textcircled{2}} + e_1 \frac{10}{1+x_1} = Bu, \tag{35}$$

where $G_1 \in \mathbb{R}^{200\times 200}$, $G_2 \in \mathbb{R}^{200\times 200^2}$, $B \in \mathbb{R}^{200\times 3}$ and $e_1 \in \mathbb{R}^{200\times 1} = [1, 0, \ldots, 0]^T$. It should be noticed that both G_1 and B are dense random matrices and the eigenvalues of G_1 are randomly distributed on $(0, 3]$ so that the nonlinear system is stable at the origin. G_2 is a sparse random matrix with 48 nonzero entries. $\frac{1}{1+x_1}$ is expended by the Taylor series $\frac{1}{1+x_1} \approx 1 - x_1 + x_1^2 - x_1^3$ and we control the inputs to guarantee $|x| < 1$ during the simulations.

The three NMOR approaches are applied on the 3rd-order polynomial system to generate the ROMs. We match the moments at the origin in all approaches. The sizes of the ROMs, the orders of the moments in each subsystem and the CPU times for the MOR have been listed in Table VI. TNMOR optimizes a rank-20 ($r_{g,2} = 20$) approximation of $\mathcal{G}_2$ in 11s with a relative error 0.03. The unique nonzero element in $\mathcal{G}_3$ is $-10x_1^3$, therefore its CP can be obtained immediately with the rank $r_{g,3} = 1$.

We feed these ROMs by 10 sets of sinusoidal inputs for transient simulations with the same time period and the same step size. These sinusoidal inputs are under different frequencies. The CPU times and errors have been summarized

TABLE VII. CPU TIMES AND ERRORS OF TRANSIENT SIMULATIONS FOR THE BIOCHEMICAL SYSTEM

Transient	full model	standard projection [5], [6]	NORM [7]	TNMOR
size	200	39	45	39
CPU time	280 ± 51s	144 ± 22s	247 ± 69s	3.1 ± 0.9s
speedup	—	1.9x	1.1x	90x
error (%)	—	18 ± 12	3.2 ± 1.0	3.1 ± 1.2

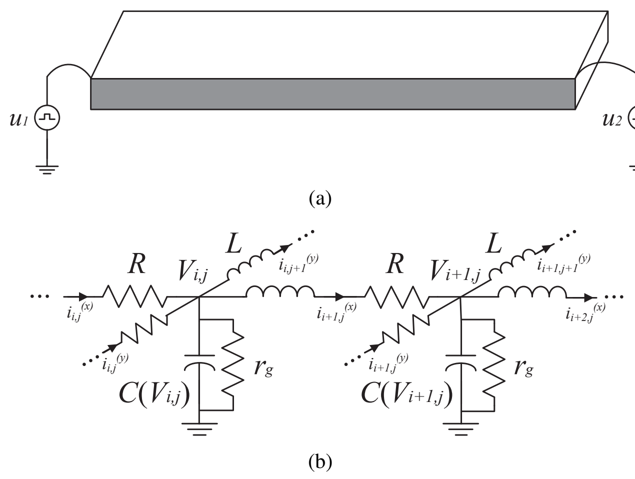

Fig. 7. (a) A nonlinear transmission line. (b) Model of the nonlinear transmission line.

in Table VII. The CPU times further confirm that for sparse systems, TNMOR can utilize the sparsity while the structure cannot be kept in NORM or the standard projection approach.

C. A 2-D pulse-narrowing transmission line

It is reported in [23] that linear lossy transmission line would cause the wave dispersion effect of the input pulse which could be avoided if certain nonlinear capacitors are introduced. We consider a nonlinear pulse-narrowing transmission line shown in Fig. 7(a) [23], [24]. There are two pulse inputs injected at the two corners of the transmission line. We are interested in the voltage at the center of the shaded edge. The length and width of the transmission line are 20cm and 10cm, respectively. It is uniformly partitioned into a 20×10 grid, with each node shown by the lumped circuit in Fig. 7(b). The nodes at the shaded edge of the transmission line are characterized by the nonlinear state equation $C_0 \dot{V}_{i,j} = (i_{i,j}^{(x)} + i_{i,j}^{(y)} - i_{i+1,j}^{(x)} - i_{i,j+1}^{(y)} - V_{i,j}/r_g)(1 + \sum_{N=1}^{\infty}(bV_{i,j})^N)$ where $C_0 = 1\mu$F, $r_g = 10K\Omega$ and $b = -0.9$, while the nonlinear coefficient $b = 0$ elsewhere. In Fig. 7(b), $L = 1\mu$H and $R = 0.1\Omega$.

Using MNA, we end up with a system with 570 state variables. First, we approximate the original system by the 3rd-order Taylor expansion, i.e., up to G_3. There are 78 nonzero elements in each order $\mathcal{G}_i, i = 1, 2, \ldots$ In this case, all the moments are matched at the origin as well. Again, the sizes of the ROMs, the orders of the moments in each subsystem and the CPU times for the MOR are listed in Table VIII. Due to the exponential growth of the size of the projector, to get a ROM fewer than 60 states, only the 0th-order moments

TABLE VIII. SIZES OF ROM AND CPU TIMES OF MOR FOR THE TRANSMISSION LINE

Method	k_1	k_2	k_3	CPU time	size of ROM
standard projection [5], [6]	2	0	—	2.6s	40
NORM [7]	4	2	2	139s	42
TNMOR	4	2	2	7.6s	46

TABLE IX. CPU TIMES AND ERRORS OF TRANSIENT SIMULATIONS FOR THE TRANSMISSION LINE

Transient	full model	standard projection [5], [6]	NORM [7]	TNMOR
size	570	40	42	46
CPU time	58s	440s	149s	8.5s
speedup	—	—	—	6.8x
error	—	45.3%	0.15%	0.012%

(DC term) of the 2nd-order subsystem and none of the 3rd-order could be matched when using the standard projection approach. TNMOR takes 2.8s and 3.5s to optimize the best rank-16 ($r_{g,2} = r_{g,3} = 16$) matches of $\mathcal{G}_2$ and $\mathcal{G}_3$, respectively. In this case, the CP decompositions are exact and the errors of CP are 0.

Then, a transient analysis with the pulse inputs shown in Fig. 8(a) is tested on each ROM. The step size Δt is chosen to be 1ms. The CPU times for ROMs and the overall errors are summarized in Table IX with the resulting waveforms and the relative errors plotted in Figs. 8(b) and 8(c), respectively. It can be seen that to achieve a good accuracy (error $< 0.1\%$), the standard projection approach and NORM may result in a ROM which is even slower than the original one. On the other hand, TNMOR can reduce the order while still preserving the sparsity.

Finally, we raise the order of the Taylor expansion to $\mathcal{C}_5$ and $\mathcal{G}_5$ to match the highly nonlinear behavior. The standard projection approach and NORM would result in a dense matrix $\tilde{\mathcal{G}}_5$ with q^6 elements where q is the size of the reduced system. The ROM is usually impractical to use if an acceptable accuracy is desired. MATLAB failed to construct the matrices $\tilde{\mathcal{G}}_4$ and $\tilde{\mathcal{G}}_5$ even if the projector up to the 3rd-order subsystem is used while the moments of the 4th- and 5th-order subsystems are ignored. Meanwhile, it takes 16s ($r_{g,4} = r_{g,5} = 16$) in total for TNMOR to obtain a 52-state ROM and another 21s for the transient simulation of the ROM. The transient waveforms of the 5th-order full model and the ROM are shown in Fig. 8(d), contrasting with the one of the 3rd-order full model.

VII. CONCLUSION

In this paper, a tensor-based nonlinear model order reduction scheme called TNMOR is proposed. The high-order nonlinearities in the system are characterized by high dimensional tensors such that these nonlinearities can be represented by just a few vectors obtained from the canonical tensor decomposition. Projection-based MOR is employed on these vectors to generate a compact reduced-order model (ROM) under the tensor framework. The key feature of TNMOR is that it preserves the inherent sparsity through low-rank tensors, thereby easing memory requirement and speeding up computation via exploitation of data structures. Examples have been shown to demonstrate the efficiency of TNMOR over existing nonlinear MOR algorithms.

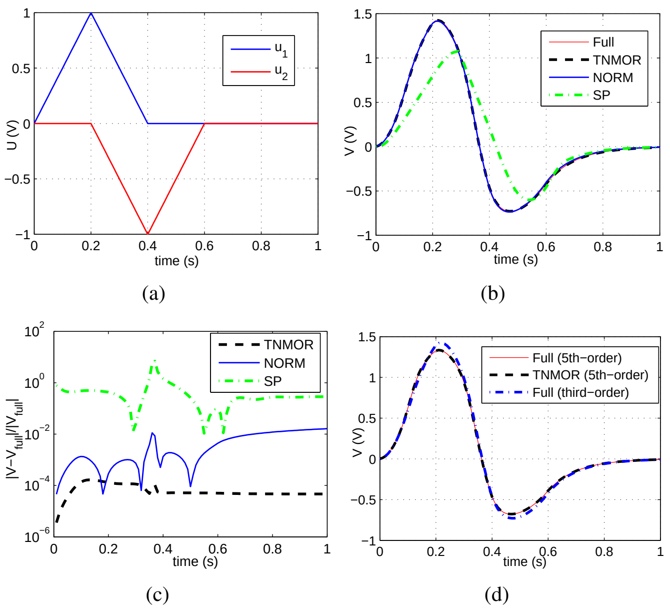

Fig. 8. (a) Pulse inputs to the transmission line. (b) Transient waveforms of the full model and ROMs. (c) Relative errors of the transient simulations. (d) Transient waveforms of the 5th-order model and the 3rd-order model. (SP is the acronym for “standard projection” [5], [6].)

ACKNOWLEDGEMENT

This work was supported in part by the Hong Kong Research Grants Council under General Research Fund (GRF) Projects 718213E and 17208514, the University Research Committee of The University of Hong Kong, and the MIT-China MISTI program. The authors are also thankful to Prof. Ivan Oseledets of Skoltech and Mr. Zheng Zhang of MIT for their valuable comments and advice.

REFERENCES

[1] A. Odabasioglu, M. Celik, and L. Pileggi, “Prima: passive reduced-order interconnect macromodeling algorithm,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 17, no. 8, pp. 645–654, Aug 1998.

[2] J. Phillips, L. Daniel, and L. Silveira, “Guaranteed passive balancing transformations for model order reduction,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 22, no. 8, pp. 1027–1041, Aug 2003.

[3] P. Feldmann and R. Freund, “Efficient linear circuit analysis by pade approximation via the lanczos process,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 14, no. 5, pp. 639–649, May 1995.

[4] B. Bond and L. Daniel, “Guaranteed stable projection-based model reduction for indefinite and unstable linear systems,” in *Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on*, Nov 2008, pp. 728–735.

[5] J. Phillips, “Automated extraction of nonlinear circuit macromodels,” in *Custom Integrated Circuits Conference, 2000. CICC. Proceedings of the IEEE 2000*, 2000, pp. 451–454.

[6] J. Roychowdhury, “Reduced-order modeling of time-varying systems,” *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 46, no. 10, pp. 1273–1288, Oct 1999.

[7] P. Li and L. Pileggi, “Compact reduced-order modeling of weakly nonlinear analog and RF circuits,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 2, pp. 184–203, Feb. 2005.

[8] J. Carroll and J.-J. Chang, “Analysis of individual differences in multidimensional scaling via an n-way generalization of “Eckart-Young” decomposition,” *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[9] T. Kolda and B. Bader, “Tensor decompositions and applications,” *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[10] I. Oseledets, “Tensor-train decomposition,” *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[11] E. Acar, D. M. Dunlavy, and T. G. Kolda, “A scalable optimization approach for fitting canonical tensor decompositions,” *Journal of Chemometrics*, vol. 25, no. 2, pp. 67–86, February 2011.

[12] B. W. Bader and T. G. Kolda, “Efficient MATLAB computations with sparse and factored tensors,” *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 205–231, December 2007.

[13] T. G. Kolda, “Multilinear operators for higher-order decompositions,” Sandia National Laboratories, Tech. Rep. SAND2006-2081, April 2006. [Online]. Available: http://www.osti.gov/scitech/biblio/923081

[14] B. W. Bader and T. G. Kolda, “Algorithm 862: MATLAB tensor classes for fast algorithm prototyping,” *ACM Transactions on Mathematical Software*, vol. 32, no. 4, pp. 635–653, December 2006.

[15] W. Rugh, *Nonlinear System Theory – The Volterra-Wiener Approach*. Baltimore, MD: Johns Hopkins Univ. Press, 1981.

[16] E. Bedrosian and S. O. Rice, “The output properties of Volterra systems (nonlinear systems with memory) driven by harmonic and Gaussian inputs,” *Proc. IEE*, vol. 59, no. 12, pp. 1688–1707, Dec. 1971.

[17] R. A. Harshman, “Foundations of the PARAFAC procedure: Models and conditions for an “explanatory” multi-modal factor analysis,” *UCLA Working Papers in Phonetics*, vol. 16, no. 1, p. 84, 1970.

[18] J. Håstad, “Tensor rank is NP-complete,” *J. Algorithms*, vol. 11, no. 4, pp. 644–654, Dec. 1990.

[19] B. W. Bader, T. G. Kolda *et al.*, “MATLAB Tensor Toolbox Version 2.5,” Available online, January 2012. [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox/

[20] W. Dong and P. Li, “A parallel harmonic-balance approach to steady-state and envelope-following simulation of driven and autonomous circuits,” *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 28, no. 4, pp. 490–501, April 2009.

[21] R. Telichevesky, K. Kundert, and J. White, “Efficient AC and noise analysis of two-tone RF circuits,” in *Design Automation Conference Proceedings 1996, 33rd*, Jun 1996, pp. 292–297.

[22] C. Gu, “QLMOR: a projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems,” *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 9, pp. 1307–1320, Sep. 2011.

[23] E. Afshari and A. Hajimiri, “Nonlinear transmission lines for pulse shaping in silicon,” *Solid-State Circuits, IEEE Journal of*, vol. 40, no. 3, pp. 744–752, 2005.

[24] B. N. Bond, “Stability-preserving model reduction for linear and nonlinear systems arising in analog circuit applications,” Ph.D. dissertation, Massachusetts Institute of Technology, Feb. 2010.

Haotian Liu (S’11) received the B.S. degree in microelectronic engineering from Tsinghua University, Beijing, China, in 2010, and the Ph.D. degree in electronic engineering from the University of Hong Kong, Hong Kong, in 2010. He is currently a Postdoctoral Fellow with the Department of Electrical and Electronic Engineering, the University of Hong Kong.

In 2010, he was a visiting scholar with the Massachusetts Institute of Technology (MIT), Cambridge, MA. His research interests include numerical simulation methods for very-large-scale integration (VLSI) circuits, model order reduction, parallel computation and control theory.

Luca Daniel (S’98–M’03) received the Ph.D. degree in electrical engineering from the University of California, Berkeley, in 2003. He is currently a Full Professor in the Electrical Engineering and Computer Science Department of the Massachusetts Institute of Technology (MIT). Industry experiences include HP Research Labs, Palo Alto (1998) and Cadence Berkeley Labs (2001). His current research interests include integral equation solvers, uncertainty quantification and parameterized model order reduction, applied to RF circuits, silicon photonics, MEMs, Magnetic Resonance Imaging scanners, and the human cardiovascular system.

Prof. Daniel was the recipient of the 1999 IEEE Trans. on Power Electronics best paper award; the 2003 best PhD thesis awards from the Electrical Engineering and the Applied Math departments at UC Berkeley; the 2003 ACM Outstanding Ph.D. Dissertation Award in Electronic Design Automation; the 2009 IBM Corporation Faculty Award; the 2010 IEEE Early Career Award in Electronic Design Automation; the 2014 IEEE Trans. On Computer Aided Design best paper award; and seven best paper awards in conferences.

Ngai Wong (S’98–M’02) received his B.Eng. and Ph.D. degrees in electrical and electronic engineering from the University of Hong Kong, Hong Kong, in 1999 and 2003, respectively.

He was a visiting scholar with Purdue University, West Lafayette, IN, in 2003. He is currently an Associate Professor with the Department of Electrical and Electronic Engineering, the University of Hong Kong. His current research interests include linear and nonlinear circuit modeling and simulation, model order reduction, passivity test and enforcement, and tensor-based numerical algorithms in electronic design automation.