STORM: A Nonlinear Model Order Reduction Method via Symmetric Tensor Decomposition

***Abstract*—Nonlinear model order reduction has always been a challenging but important task in various science and engineering fields. In this paper, a novel symmetric tensor-based order-reduction method (STORM) is presented for simulating large-scale nonlinear systems. The multidimensional data structure of symmetric tensors, as the higher order generalization of symmetric matrices, is utilized for the effective capture of high-order nonlinearities and efficient generation of compact models. Compared to the recent tensor-based nonlinear model order reduction (TNMOR) algorithm [1], STORM shows advantages in two aspects. First, STORM avoids the assumption of the existence of a low-rank tensor approximation. Second, with the use of the symmetric tensor decomposition, STORM allows significantly faster computation and less storage complexity than TNMOR. Numerical experiments demonstrate the superior computational efficiency and accuracy of STORM against existing nonlinear model order reduction methods.**

I. Introduction

To facilitate efficient modeling and simulation of large-scale dynamical systems, model order reduction has been intensively studied in the past decades [2]. The principle of model order reduction is to extract a compact reduced-order model (ROM) which approximates the input-output relationship of the original system accurately, so that efficient simulation and reliable system-level verification can be achieved. The effectiveness and robustness of model order reduction have been proved in various domains, e.g., circuit design [3]–[5], mechanical systems [6] and chemical engineering [7].

Since model order reduction for linear time-invariant (LTI) systems is by now mature, large amounts of research effort are being devoted to nonlinear cases, which are also more practical for real-world problems [8]. For instance, simulations of radio-frequency integrated circuits (RFIC) are highly time-consuming due to their intrinsic nonlinearities and large problem sizes [9]. However, nonlinear model order reduction is much more challenging: the behaviors of nonlinear dynamical systems are much more complex than linear systems, and there is a lack of a universal and efficient way to extract the nonlinear ROM [10].

Several projection-based nonlinear model order reduction methods [9], [11], [12] have been derived from linear model order reduction approaches. In [9], a method called NORM is proposed to match the moments of high-order transfer functions explicitly. NORM shows a great advantage in comparison with the methods in [11], [12], since the lower order approximation is skipped. However, NORM builds the ROM whose matrix form tends to be very dense and whose size grows exponentially as the order increases. This fact limits the practicality of NORM for large sparse systems as well as strongly nonlinear systems.

In fact, existing high-order nonlinear model order reduction problems are always doomed by an explosion of matrix dimensions. To overcome such curse of dimensionality, the data structure of tensors [13], as higher order generalizations of matrices, is introduced to effectively capture the high-order nonlinearities and generate highly compact models. The recent work [1] proposed a tensor-based nonlinear model order reduction method named TNMOR. It relies on the CANDECOMP/PARAFAC (CP) decomposition [14], [15] to find a low-rank rank approximation of the original system. Based on the assumption of the existence of a low-rank approximation, TNMOR allows faster simulation and smaller memory requirement than previous nonlinear model order reduction methods.

In this paper, a symmetric tensor-based order-reduction method called STORM is presented. Comparing to TNMOR, STORM avoids the premise that a low-rank approximation of the original system exists, which is a limitation of TNMOR. Meanwhile, via the symmetric tensor structure and the proposed decomposition algorithm, STORM provides significant improvements of computational performance and storage requirement, for both the order-reduction stage and the ROM simulation stage.

In the rest of this paper, Section II reviews the theoretical backgrounds of STORM, including Volterra series, symmetric tensor decomposition, as well as the existing projection-based nonlinear model order reduction methods. The details of STORM are discussed in Section III. In Section IV, numerical examples are given to verify the validity and advantages of STORM. Finally, Section V concludes this paper.

II. Background

The theory on Volterra systems, symmetric tensors and their symmetric rank-1 decomposition are first briefly reviewed. Then, we show the principles and limitations of the existing projection-based nonlinear model order reduction methods such as NORM and TNMOR.

A. Volterra System

Volterra series is an important mathematical tool for nonlinear analysis [16]. It decomposes the original nonlinear system into a series of homogeneous nonlinear systems. In this work, we consider a nonlinear multi-input multi-output (MIMO)

time-invariant system modeled by the differential-algebraic equation (DAE)

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

where $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^l$ are the state and input vectors, respectively; B and D are the input and output matrices, respectively; $q(\cdot)$ and $f(\cdot)$ are smooth nonlinear vector functions of the state vector x.

According to the Volterra theorem and variational analysis [17], the system response x can be written as a summation of the responses x_i of the ith-order homogeneous nonlinear system, namely $x(t) = \sum_{i=1}^{\infty} x_i(t)$. Therefore, the response of (1) to the inputs $\alpha u(t)$ becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}[q(x(t))] + f\left(x(t)\right) = B\alpha u(t),\ x(t) = \sum_{i=1}^{\infty} \alpha^i x_i(t). \tag{2}$$

The Taylor series expansion of the nonlinear system (1) can be obtained around its equilibrium point x_0, that is

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x + C_2(x\otimes x) + C_3(x\otimes x\otimes x) + \cdots\right] + G_1 x \\ + G_2(x\otimes x) + G_3(x\otimes x\otimes x) + \cdots = Bu,\end{aligned} \tag{3}$$

where $G_k = \frac{1}{k!}\frac{\partial^k f}{\partial x^k}\Big|_{x=x_0}, C_k = \frac{1}{k!}\frac{\partial^k q}{\partial x^k}\Big|_{x=x_0} \in \mathbb{R}^{n\times n^k}$, and $\otimes$ denotes the Kronecker product, here we use the shorthand $x^{\textcircled{d}} = x\otimes x\otimes\cdots\otimes x$ for d-time repeated Kronecker product $\otimes$. Merging (3) and (2), we get

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}[\alpha C_1 x_1 + \alpha^2(C_1 x_2 + C_2 x_1^{\textcircled{2}}) + \cdots] + \\ [\alpha G_1 x_1 + \alpha^2(G_1 x_2 + G_2 x_1^{\textcircled{2}}) + \cdots] = \alpha B u.\end{aligned} \tag{4}$$

By matching the coefficients of the monomials α^i, we have the following Volterra subsystems

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_1\right] + G_1 x_1 = Bu, \tag{5a}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_2\right] + G_1 x_2 = -\frac{\mathrm{d}}{\mathrm{d}t}\left[C_2 x_1^{\textcircled{2}}\right] - G_2 x_1^{\textcircled{2}}, \tag{5b}$$

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x_3\right] + G_1 x_3 = -\frac{\mathrm{d}}{\mathrm{d}t}\left[C_3 x_1^{\textcircled{3}} + C_2(\overline{x_{i_1}\otimes x_{i_2}})_3\right] - G_3 x_1^{\textcircled{3}} \\ - G_2(\overline{x_{i_1}\otimes x_{i_2}})_3,\end{aligned} \tag{5c}$$

and so on, where $(\overline{x_{i_1}\otimes x_{i_2}})_3 = x_1\otimes x_2 + x_2\otimes x_1$, or generally $(\overline{x_{i_1}\otimes\cdots\otimes x_{i_n}})_k = \sum_{i_1+\cdots+i_n=k} x_{i_1}\otimes\cdots\otimes x_{i_n}$, $i_1,\ldots,i_n \in \mathbb{Z}^+$.

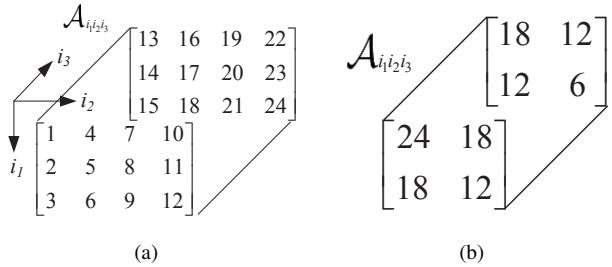
B. Symmetric tensor decomposition

Tensors are high-order generalizations of matrices. Following the conventional terminology used by the tensor community, a dth-order tensor is a d-way array defined as[1]

$$\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_d}. \tag{6}$$

For example, Fig. 1(a) shows 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. A high-order tensor can be unfolded into a 2nd-order matrix by the "matricization" process. The k-mode matricization is aligning each kth-direction "vector fiber" to be the columns of the matrix.

[1]Tensors are denoted by calligraphic letters, e.g., $\mathcal{A}$ and $\mathcal{G}$.



(a) (b)

Fig. 1. (a) A tensor $\mathcal{A} \in \mathbb{R}^{3\times4\times2}$. (b) A symmetric tensor $\mathcal{A} \in \mathbb{R}^{2\times2\times2}$.

1) Symmetric tensor: A tensor $\mathcal{X}$ is called cubical if every mode has the same size, for example, $\mathcal{X} \in \mathbb{R}^{I\times I\times\cdots\times I}$. $\mathcal{A} \in \mathbb{R}^{n_1\times\cdots\times n_d}$ is symmetric if $\mathcal{A}_{i_1\cdots i_d} = \mathcal{A}_{\pi(i_1\cdots i_d)}$ where $\pi(i_1\cdots i_d)$ is any permutations of the indices $i_1\cdots i_d$ [18]. Therefore, a symmetric tensor is also a cubical tensor, which means $n_1 = \cdots = n_d = I$. An example of a 3rd-order symmetric tensor $\mathcal{A}$ is shown Fig. 1(b). In addition, tensors can be partially symmetric in two or more modes. For example, a 3rd-order tensor $\mathcal{X} \in \mathbb{R}^{I\times I\times K}$ is partially symmetric in the first and second mode if its frontal slices are all symmetric, namely $X_k = X_k^T (k = 1,\ldots,K)$.

2) Symmetric tensor decomposition: A rank-1 tensor of order d can be written as the outer product of d vectors

$$\mathcal{A} = a^{(1)}\circ a^{(2)}\circ\cdots\circ a^{(d)},\ a^{(k)} \in \mathbb{R}^{n_k\times1}, \tag{7}$$

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 $a^{(k)}$. When $a^{(1)} = a^{(2)} = \cdots = a^{(d)}$, $\mathcal{A}$ is a symmetric rank-1 tensor. A symmetric rank-1 tensor decomposition expresses a symmetric tensor $\mathcal{A}$ as a linear combination of symmetric rank-1 tensors

$$\mathcal{A} = \sum_{r=1}^{R} \lambda_r w_r\circ w_r\circ\cdots\circ w_r, w_r \in \mathbb{R}^{n\times1}, \tag{8}$$

where $\lambda_r \in \mathbb{R}$ and $R \in \mathbb{Z}^+$. The CP decomposition drops the symmetry requirement of the rank-1 terms. Here we introduce $\mathrm{vec}(\mathcal{A}) \in \mathbb{R}^{n^d\times1}$ to denote the vectorization of a d-way symmetric tensor $\mathcal{A}$ by aligning all entries of $\mathcal{A}$ into a single vector. Using $\mathrm{vec}(\mathcal{A})$, (8) can be rewritten as

$$\mathrm{vec}(\mathcal{A}) = WL, \tag{9}$$

where $W = [w_1^{\textcircled{d}},\ldots,w_R^{\textcircled{d}}]$ and $L = [\lambda_1,\ldots,\lambda_R]^T \in \mathbb{R}^{R\times1}$. The inverse vectorization operation is defined as $\mathcal{A} = \mathrm{unvec}(a)$, which reshapes a vector $a \in \mathbb{R}^{n^d}$ into a tensor $\mathcal{A} \in \mathbb{R}^{n\times\cdots\times n}$.

The rank of a symmetric tensor $\mathcal{A}$, $\mathrm{rank}(\mathcal{A})$, is the minimum value of R needed for the approximation (8). A symmetric tensor decomposition, or a rank-R approximation of a 3rd-order symmetric tensor is shown in Fig. 2. From now on, all tensors will be assumed symmetric unless otherwise stated.

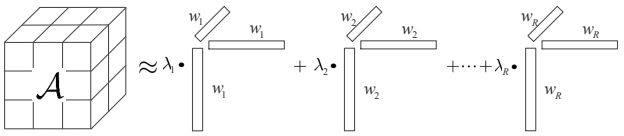


Fig. 2. A symmetric tensor decomposition of a 3rd-order symmetric tensor.

C. Existing projection-based nonlinear model order reduction methods

In this section we briefly introduce both the NORM and TNMOR methods. NORM [9] first 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 with the subsystems in (5). Then, these high-order transfer functions are expanded into multivariate polynomials of s_1, s_2, etc. via Taylor expansion, such that the coefficients (moments) can be explicitly matched. In NORM, the size of an Nth-order ROM is $O(k^{N+1}l^N)$ supposing up to kth-order moments of each Volterra transfer function are matched.

The final step in NORM is the replacement of the original nonlinear system (3) by a smaller system (the ROM) via the transformations

$$\begin{aligned} \tilde{x} = V^T x, \quad \tilde{B} = V^T B, \quad \tilde{D} = V^T D, \\ \tilde{G}_i = V^T G_i(V^{\textcircled{1}}), \quad \tilde{C}_i = V^T C_i(V^{\textcircled{1}}), \end{aligned} \tag{10}$$

where $i = 1, \ldots, N$ and $V = \text{orth}[V_{k_1}, V_{k_2}, V_{k_3}, \ldots]$ is the orthogonal projection matrix of the ROM. 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.

TNMOR [1] gives the equivalent tensor models $\mathcal{G}_i$ and $\mathcal{C}_i$ for G_i and C_i respectively, and applies a CP decomposition [15] to find low-rank approximations

$$\begin{aligned} \mathcal{G}_i &\approx [\![\mathbf{G}_i^{(1)}, \ldots, \mathbf{G}_i^{(i+1)}]\!] = \sum_{r=1}^{r_{g,i}} g_{i,r}^{(1)} \circ \cdots \circ g_{i,r}^{(i+1)}, \\ \mathcal{C}_i &\approx [\![\mathbf{C}_i^{(1)}, \ldots, \mathbf{C}_i^{(i+1)}]\!] = \sum_{r=1}^{r_{c,i}} c_{i,r}^{(1)} \circ \cdots \circ c_{i,r}^{(i+1)}, \end{aligned} \tag{11}$$

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 $k = 1, \ldots, i+1$. Then it follows the same routine in [11], [12] to derive the projection matrix V. Finally, the ROM of TNMOR is given by

$$\begin{aligned} &\tilde{x} = V^T x, \quad \tilde{B} = V^T B, \quad \tilde{D} = V^T D, \\ &\tilde{G}_1 = V^T G_1 V, \quad \tilde{C}_1 = V^T C_1 V, \\ &\tilde{\mathcal{G}}_i = [\![\tilde{\mathbf{G}}_i^{(1)}, \ldots, \tilde{\mathbf{G}}_i^{(i+1)}]\!] = [\![V^T \mathbf{G}_i^{(1)}, \ldots, V^T \mathbf{G}_i^{(i+1)}]\!], \\ &\tilde{\mathcal{C}}_i = [\![\tilde{\mathbf{C}}_i^{(1)}, \ldots, \tilde{\mathbf{C}}_i^{(i+1)}]\!] = [\![V^T \mathbf{C}_i^{(1)}, \ldots, V^T \mathbf{C}_i^{(i+1)}]\!], \\ &i = 2, \ldots, N. \end{aligned} \tag{12}$$

The size of an Nth-order ROM of TNMOR 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 $r = \max\{l, r_{g,2} + r_{c,2}, \ldots, r_{g,N} + r_{c,N}\}$. Therefore, the advantage of TNMOR depends explicitly on the existence of low-rank approximations of the tensor-based models.

III. STORM

In this section, the STORM algorithm is developed. We first explain how we use symmetric tensors to represent multivariate polynomial systems. Then, we briefly explain the STEROID algorithm [19] to compute a symmetric rank-1 decomposition of a symmetric tensor as given in (8). Finally, we describe the projection-based steps in STORM for extracting the ROM from the original system.

A. Symmetric tensor model for multivariate polynomial systems

Given the definitions of G_i and C_i in (3), these coefficient matrices can be treated respectively as the 1-mode matricizations of $(i+1)$th-order tensors $\mathcal{G}_i$ and $\mathcal{C}_i$,

$$\mathcal{G}_i,\ \mathcal{C}_i \in \mathbb{R}^{\overbrace{n \times \cdots \times n}^{i+1}}, \tag{13}$$

where the elements $(\mathcal{G}_i)_{j_0 j_1 \cdots j_i}$ and $(\mathcal{C}_i)_{j_0 j_1 \cdots j_i}$ 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 $\mathcal{G}_2$.

Each row vector $G_i(j,:) \in \mathbb{R}^{1 \times n^i}$ of G_i can be taken as the transpose of the vectorization of the ith-order symmetric tensor $\mathcal{G}_{i,j}$, namely $G_i(j,:) = \text{vec}(\mathcal{G}_{i,j})^T$. It also works for C_i, while $C_i(j,:) = \text{vec}(\mathcal{C}_{i,j})^T$. Based on (9), for $i > 1$, G_i and C_i can be transformed into

$$\begin{aligned} G_i &= \text{blkdiag}(L_{\mathcal{G}_{i,1}}^T, \ldots, L_{\mathcal{G}_{i,n}}^T)[W_{\mathcal{G}_{i,1}}, \ldots, W_{\mathcal{G}_{i,n}}]^T, \\ C_i &= \text{blkdiag}(L_{\mathcal{C}_{i,1}}^T, \ldots, L_{\mathcal{C}_{i,n}}^T)[W_{\mathcal{C}_{i,1}}, \ldots, W_{\mathcal{C}_{i,n}}]^T. \end{aligned} \tag{14}$$

where $\text{blkdiag}(\cdot)$ outputs a block diagonal matrix.

Let $X_i = x^{\textcircled{i}}$. Then the ith degree terms of the jth multivariate polynomial is $\text{vec}(\mathcal{G}_{i,j})^T X_i = L_{\mathcal{G}_{i,j}}^T W_{\mathcal{G}_{i,j}}^T X_i$, where from (9) we have that $W_{\mathcal{G}_{i,j}} = [w_{\mathcal{G}_{i,j},1}^{\textcircled{i}}, \ldots, w_{\mathcal{G}_{i,j},R_{\mathcal{G}_{i,j}}}^{\textcircled{i}}]$. We can prove by induction, that $W_{\mathcal{G}_{i,j}}^T X_i = [(w_{\mathcal{G}_{i,j},1}^T x)^i, \ldots, (w_{\mathcal{G}_{i,j},R_{\mathcal{G}_{i,j}}}^T x)^i]^T$. In order to reduce the computational complexity of computing $W_{\mathcal{G}_{i,j}}^T X_i$ we introduce $U_{\mathcal{G}_{i,j}} \triangleq [w_{\mathcal{G}_{i,j},1}, \ldots, w_{\mathcal{G}_{i,j},R_{\mathcal{G}_{i,j}}}]$, then $W_{\mathcal{G}_{i,j}}^T X_i = \odot^i (U_{\mathcal{G}_{i,j}}^T x)$. The operator $\odot^i$ stands for the i times' repeated Hadamard product $\odot$. Let $L_{G_i} \triangleq \text{blkdiag}(L_{\mathcal{G}_{i,1}}^T, \ldots, L_{\mathcal{G}_{i,n}}^T)$, $M_{G_i}(x) \triangleq [(\odot^i((U_{\mathcal{G}_{i,1}}^T x)^T), \ldots, (\odot^i((U_{\mathcal{G}_{i,n}}^T x)^T)]^T$. The same goes for C_i when $i > 1$. Then the system (3) can be transformed into

$$\begin{aligned} &\frac{\mathrm{d}}{\mathrm{d}t}\left[C_1 x + L_{C_2} M_{C_2}(x) + L_{C_3} M_{C_3}(x) + \cdots\right] \\ &+ G_1 x + L_{G_2} M_{G_2}(x) + L_{G_3} M_{G_3}(x) + \cdots = Bu. \end{aligned} \tag{15}$$

The computational complexity and storage cost of (15) are much smaller than (3), because we can utilize the form

of $M_{G_i}(x)$ and $M_{C_i}(x)$ in (15) to accurately compute the high-order nonlinearities. It is worth mentioning that $\mathcal{G}_{i,j} = \text{unvec}(G_i(j,:))$ and $\mathcal{C}_{i,j} = \text{unvec}(C_i(j,:))$ may not necessarily be symmetric. However, we can always symmetrize $\mathcal{G}_{i,j}$ and $\mathcal{C}_{i,j}$ before we apply the symmetric tensor decomposition, which does not change the inner product of $\text{vec}(\mathcal{G}_{i,j})^T X_i$.

B. Symmetric rank-1 tensor decomposition algorithm

We demonstrate a symmetric tensor eigen-rank-one iterative decomposition (STEROID) algorithm [19] to obtain (9). In each iteration, the first step is a reshaping of a symmetric tensor into a square symmetric matrix. Given a symmetric dth-order symmetric tensor $\mathcal{A}$, we can reshape it into a symmetric matrix $A \in \mathbb{R}^{n^{\frac{d}{2}} \times n^{\frac{d}{2}}}$ when d is an even number. Otherwise we need to embed $\mathcal{A}$ into a $(d+1)$th-order symmetric tensor $\mathcal{B}$ before the reshaping. The details of the embedding step is given in [19]. After the reshaping, the eigenvalue decomposition of A can be computed as $A = \sum_{i=1}^{n^{\frac{d}{2}}} \lambda_i w_i w_i^T, w_i \in \mathbb{R}^{n^{\frac{d}{2}} \times 1}$. It is shown in [19] that each eigenvectors w_i can be reshaped into another $\frac{d}{2}$-way symmetric tensor $\mathcal{W}_i$. Therefore the above iteration will continue until the dimension of w_i becomes $\mathbb{R}^{n \times 1}$.

The last step of STEROID is to collect all w_i into the matrix W, and get the vector L in (9) by solving the following least-square problem

$$\underset{L}{\text{argmin}} \; || \,\text{vec}\,(\mathcal{A}) - W\, L\, ||_F. \tag{16}$$

C. ROM construction

According to [16], we can obtain the matrix transfer function for (3). For example, the first three transfer functions are

$$H_1(s) = (sC_1 + G_1)^{-1} B, \tag{17a}$$

$$H_2(s_1, s_2) = -\frac{1}{2}(\bar{s}C_1 + G_1)^{-1}(\bar{s}C_2 + G_2)(\overline{H_1(s_1) \otimes H_1(s_2)}), \tag{17b}$$

$$H_3(s_1, s_2, s_3) = -\frac{1}{6}(\tilde{s}C_1 + G_1)^{-1}\,[(\bar{s}C_3 + G_3) \cdot (\overline{H_1(s_1) \otimes H_1(s_2) \otimes H_1(s_3)}) + (\bar{s}C_2 + G_2)(\overline{H_1(s_1) \otimes H_2(s_2, s_3)})], \tag{17c}$$

where

$$\begin{aligned}
&\bar{s} = s_1 + s_2, \tilde{s} = s_1 + s_2 + s_3,\\
&\overline{H_1(s_1) \otimes H_1(s_2)} = (H_1(s_1) \otimes H_1(s_2) + H_1(s_2) \otimes H_1(s_1)),\\
&\overline{H_1(s_1) \otimes H_1(s_2) \otimes H_1(s_3)} =\\
&\sum\nolimits_{(i_1,i_2,i_3)=\pi(1,2,3)} H_1(s_{i_1}) \otimes H_1(s_{i_2}) \otimes H_1(s_{i_3}),\\
&\overline{H_1(s_1) \otimes H_2(s_2, s_3)} =\\
&\sum_{(i_1,i_2,i_3)=\pi(1,2,3)} (H_1(s_{i_1}) \otimes H_2(s_{i_2}, s_{i_3}) + H_2(s_{i_2}, s_{i_3}) \otimes H_1(s_{i_1})).
\end{aligned} \tag{18}$$

If we define

$$\begin{aligned}
&A = -G_1^{-1}C_1, R_1 = G_1^{-1}B, R_2 = R_1 \otimes R_1, R_3 = R_1^{\textcircled{3}},\\
&A^{l \otimes m \otimes \cdots \otimes n} = A^l \otimes A^m \otimes \cdots \otimes A^n,\\
&\overline{A^{l \otimes m}} = \frac{1}{2}(A^{l \otimes m} + A^{m \otimes n}),
\end{aligned} \tag{19}$$

then based on (17), the Taylor expansion of these transfer functions can be derived as

$$H_1(s) = \sum_{k=0}^{\infty} M_{1,k} s^k = \sum_{k=0}^{\infty} A^k R_1 s^k, \tag{20a}$$

$$H_2(s_1, s_2) = \sum_{k=0}^{\infty} \sum_{l=0}^{k} s_1^l s_2^{k-l} M_{2,k,l}, \tag{20b}$$

$$H_3(s_1, s_2, s_3) = \sum_{k=0}^{\infty} \sum_{l=0}^{k} \sum_{m=0}^{k-l} s_1^l s_2^m s_3^{k-l-m} M_{3,k,l,m}. \tag{20c}$$

From (20a), it is seen that the kth-order moment $M_{1,k}$ of the first-order transfer function $H_1(s)$ is equal to $A^k R_1$. Therefore, we can compute a Krylov subspace to match up to k_1th-order moments of H_1, which is

$$V_1 = [v_{1,0}, v_{1,1}, \ldots, v_{1,k_1}] = \text{orth}(\mathcal{K}_{k+1}(A, R_1)). \tag{21}$$

where $\mathcal{K}_{k+1}(A, r)$ returns the first $k+1$ vectors of the Krylov subspace spanned by A.

By substituting (20a) into (17b), the kth-order moment $M_{2,k,l}$ of the second-order transfer function $H_2(s)$ can be expressed as

$$\begin{aligned}
M_{2,k,l} = &-\sum_{p=1}^{k} A^{p-1} \sum_{\substack{q=0, q \le l \\ p-q \le k-l}}^{p} \binom{p}{q} G_1^{-1} C_2 \cdot \overline{A^{(l-q) \otimes (k-l-p+q)}} R_2\\
&-\sum_{p=0}^{k} A^{p} \sum_{\substack{q=0, q \le l \\ p-q \le k-l}}^{p} \binom{p}{q} G_1^{-1} G_2 \cdot \overline{A^{(l-q) \otimes (k-l-p+q)}} R_2
\end{aligned} \tag{22}$$

Here we avoid the explicit computation of $A^i, i \in \mathbb{Z}^+$ when deriving the starting vectors for each Krylov subspace, which can lead to numerical stability problems [8]. Instead, we replace $\overline{A^{m \otimes n}} R_2$ by $(v_{1,m} \otimes v_{1,n})$, where $v_{1,m}$ and $v_{1,n}$ are column vectors from V_1 in (21). And it can be proved by induction that this replacement will not change the subspace. If we define $N_{G_i} \triangleq [U_{\mathcal{G}_{i,1}}, \ldots, U_{\mathcal{G}_{i,n}}]^T$, then we utilize the symmetric tensor model of C_2 and G_2, namely $C_2(v_{1,m} \otimes v_{1,n}) = L_{C_2}((N_{C_2} v_{1,m}) \odot (N_{C_2} v_{1,n}))$ and $G_2(v_{1,m} \otimes v_{1,n}) = L_{G_2}((N_{G_2} v_{1,m}) \odot (N_{G_2} v_{1,n}))$, to avoid the complexity to compute the Kronecker products. As a result, in the computation of the starting vectors of the Krylov subspaces, the computation of both A^i and the Kronecker products $\overline{A^{m \otimes n}} R_2$ are avoided. To match up to k_2th-order moments of H_2, we get the projection matrix V_2 as

$$\begin{aligned}
V_2 = \text{orth}(\{&[\mathcal{K}_{k_2-m-n}(A, G_1^{-1} L_{C_2}((N_{C_2} v_{1,m}) \odot (N_{C_2} v_{1,n}))),\\
&\mathcal{K}_{k_2-m-n+1}(A, G_1^{-1} L_{G_2}((N_{G_2} v_{1,m}) \odot (N_{G_2} v_{1,n})))],\\
&m \ge 0, n \ge 0, m \le n, m+n \le k\}).
\end{aligned} \tag{23}$$

Similar procedure can be applied to generate V_j when $j > 2$, which matches up to the kth-order moments of the jth-order transfer function H_j. In the end, the projection matrix for reducing the nonlinear system is denoted by $V_p = \text{orth}([V_1, V_2, \ldots])$. A general procedure to obtain the projection matrix V_p for an expansion around the DC operating points is given in Algorithm 1. The proof of the validity of the V_p can be found in [9].

Algorithm 1 Generation of projection matrix for STORM

Input: $N, G_i, C_i, B, k_i, k_N \leq k_{N-1} \leq \cdots \leq k_1$

Output: the projection matrix V_p

1: $A = -G_1^{-1}C_1, R_1 = G_1^{-1}B$
2: Generate matrix $V_1 = [v_{1,0}, v_{1,1}, \ldots, v_{1,k_1}]$ for $\mathcal{K}_{k_1+1}(A, R_1)$
3: **for** $i = 2$ **to** N **do**
4: $G_i = \text{blkdiag}(L_{\mathcal{G}_{i,1}}^T, \ldots, L_{\mathcal{G}_{i,n}}^T)[W_{\mathcal{G}_{i,1}}, \ldots, W_{\mathcal{G}_{i,n}}]^T$
5: $C_i = \text{blkdiag}(L_{\mathcal{C}_{i,1}}^T, \ldots, L_{\mathcal{C}_{i,n}}^T)[W_{\mathcal{C}_{i,1}}, \ldots, W_{\mathcal{C}_{i,n}}]^T$
6: Build the starting vector set v_{s_i} based on the moment expression and the symmetric tensor model
7: Get the projection matrix $V_i = \text{orth}(\mathcal{K}_{k_i+1}(A, v_{s_i}))$ which matches up to k_ith-order moments of H_i
8: **end for**
9: $V_p = \text{orth}([V_1, V_2, \ldots, V_N])$;

TABLE I
ROM SIZES AND CPU TIMES OF MODEL ORDER REDUCTION FOR THE MIXER

Method	k_1	k_2	k_3	CPU time (s)	size of ROM
NORM	2	2	—	35.33	48
TNMOR	2	2	—	9.53	36
STORM	2	2	2	16.31	49

The size of an Nth-order ROM is $O(k_1 l + k_2 l + k_3 l + \cdots + k_N l) = O(Nkl)$. 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. Moreover, it avoids the limitation of the existence of a low-rank tensor decomposition in TNMOR. Finally, the symmetric tensor-based ROM is given by the following projection

$$\begin{aligned} &\tilde{x} = V_p^T x, \quad \tilde{B} = V_p^T B, \quad \tilde{D} = V_p^T D, \\ &\tilde{L_{C_i}} = V_p^T L_{C_i}, \tilde{L_{G_i}} = V_p^T L_{G_i}, \\ &\tilde{N_{C_i}} = N_{C_i} * V_p, \tilde{N_{G_i}} = N_{G_i} * V_p, \\ &\tilde{C}_i = \tilde{L_{C_i}}(*^i(\tilde{N_{C_i}}^T))^T, \tilde{G}_i = \tilde{L_{G_i}}(*^i(\tilde{N_{G_i}}^T))^T, \end{aligned} \tag{24}$$

where $*^i$ denotes the i times' repeated Khatri-Rao product[2]. By utilizing this structure, only the $\tilde{L_{C_i}}, \tilde{L_{G_i}}, \tilde{N_{C_i}}$ and $\tilde{N_{G_i}}$ matrices need to be stored. Thus in the simulation stage, significant speedup is achieved as shown in the next section.

IV. NUMERICAL EXAMPLES

In this section, the proposed STORM method is demonstrated and compared with two existing projection-based nonlinear model order reduction methods, NORM and TNMOR. All three approaches are implemented in Matlab [20], and the time-domain transient analysis is solved by the built-in ordinary differential equation solver in Matlab. Two numerical experiments are performed on an Intel Core I7 desktop PC with 2.6GHz CPU and 8GB RAM.

The first example is a double-balanced mixer circuit in Fig. 3(a), where $V_{\text{rf}}(= V_{\text{rf+}} - V_{\text{rf-}})$ and $V_{\text{lo}}(= V_{\text{lo+}} - V_{\text{lo-}})$ are

[2]Khatri-Rao product is a column-wise Kronecker Product of two matrix.

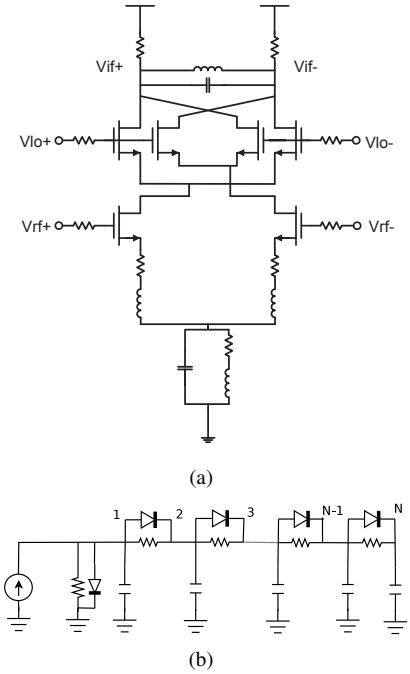


Fig. 3. (a) A double-balanced mixer. (b) A nonlinear transmission line.

TABLE II
CPU TIMES AND ERRORS OF TRANSIENT SIMULATIONS FOR THE MIXER

Transient	full model	NORM	TNMOR	STORM
size	93	48	30	49
CPU time (s)	991.51	172.73	16.86	3.23
speedup	—	6x	60x	300x
error	—	5.05%	3.24%	4.06%

the RF and local oscillator (LO) inputs, respectively. Three transient simulations of the ROMs generated by different nonlinear model order reduction methods are performed from $T = 0$ns to $T = 40$ns with the time step $\Delta T = 1$ps. The reduction time and generated ROM size are recorded in Table I. The transient simulation time and relative error of different methods are listed in Table II. The error is calculated by the output difference between ROM and full model, $(\|V_{\text{ROM}} - V_{\text{full}}\|_2)/\|V_{\text{full}}\|_2$. According to the result, we find that STORM shows a better computational efficiency of transient analysis comparing to NORM and TNMOR. Moreover, the reduction scheme of STORM can match higher order moments while the other two methods cannot.

The second example is a nonlinear transmission line circuit shown in Fig. 3(b). It is a common benchmark for testing nonlinear model order reduction methods. The transient simulation is executed to $T = 400$ns with a step size $\Delta T = 1$ps. TNMOR is not suitable for this case because the low-rank approximation does not exist. However, STORM still works and outperforms NORM in terms of the simulation time. Moreover, the NORM fails when the the size of the original system exceed 1000 because of shortage of memory, while STORM still works well.

TABLE III
ROM SIZES AND CPU TIMES OF MODEL ORDER REDUCTION FOR THE NONLINEAR TRANSMISSION LINE

Method	k_1	k_2	CPU time	size of ROM
NORM	2	4	0.5s	15
STORM	2	4	0.4s	15

TABLE IV
CPU TIMES AND ERRORS OF TRANSIENT SIMULATIONS FOR THE NONLINEAR TRANSMISSION LINE

Transient	full model	NORM	STORM
size	80	15	15
CPU time (s)	33.61	16.38	7.18
speedup	—	2x	4x
error	—	0.03%	0.03%

V. CONCLUSION

This paper presented a symmetric tensor based order-reduction method called STORM. The key feature of STORM is to build symmetric tensor models for high-order polynomial nonlinear systems. By utilizing the symmetric tensor decomposition and the new projection-based reduction scheme under the symmetric tensor framework, faster ways of model order reduction and transient analysis are available. The advantage of STORM has been proved by comparing with existing nonlinear model order reduction methods on some benchmark circuits. STORM has shown the potential of the symmetric tensor structure for efficient simulation of high-order nonlinear large-scale systems.

REFERENCES

[1] H. Liu, L. Daniel, and N. Wong, "Model reduction and simulation of nonlinear circuits via tensor decomposition," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 34, no. 7, pp. 1059–1069, July 2015.

[2] U. Baur, P. Benner, and L. Feng, "Model order reduction for linear and nonlinear systems: a system-theoretic perspective," *Archives of Computational Methods in Engineering*, vol. 21, no. 4, pp. 331–358, 2014.

[3] A. Odabasioglu, M. Celik, and L. T. Pileggi, "PRIMA: passive reduced-order interconnect macromodeling algorithm," in *Proceedings of the 1997 IEEE/ACM international conference on Computer-aided design*. IEEE Computer Society, 1997, pp. 58–65.

[4] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Padé approximation via the Lanczos process," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 14, no. 5, pp. 639–649, 1995.

[5] B. N. Bond and L. Daniel, "Guaranteed stable projection-based model reduction for indefinite and unstable linear systems," in *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2008, pp. 728–735.

[6] J. S. Han, E. B. Rudnyi, and J. G. Korvink, "Efficient optimization of transient dynamic problems in MEMS devices using model order reduction," *Journal of Micromechanics and Microengineering*, vol. 15, no. 4, p. 822, 2005.

[7] N. Vora and P. Daoutidis, "Nonlinear model reduction of chemical reaction systems," *AIChE Journal*, vol. 47, no. 10, pp. 2320–2332, 2001.

[8] C. Gu, "QLMOR: a projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 30, no. 9, pp. 1307–1320, 2011.

[9] P. Li and L. T. Pileggi, "Compact reduced-order modeling of weakly nonlinear analog and RF circuits," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 24, no. 2, pp. 184–203, 2005.

[10] J. M. T. Thompson and H. B. Stewart, *Nonlinear dynamics and chaos*. John Wiley & Sons, 2002.

[11] J. R. Phillips, "Automated extraction of nonlinear circuit macromodels," *IEEE Custom Integrated Circuits Conference*, pp. 451–454, 2000.

[12] 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, 1999.

[13] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

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

[15] R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis," 1970.

[16] W. J. Rugh, *Nonlinear system theory*. Johns Hopkins University Press Baltimore, 1981.

[17] E. Bedrosian and S. O. Rice, "The output properties of Volterra systems (nonlinear systems with memory) driven by harmonic and gaussian inputs," *Proceedings of the IEEE*, vol. 59, no. 12, pp. 1688–1707, 1971.

[18] P. Comon, G. Golub, L.-H. Lim, and B. Mourrain, "Symmetric tensors and symmetric tensor rank," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1254–1279, 2008.

[19] K. Batselier and N. Wong, "Symmetric tensor decomposition by an iterative eigendecomposition algorithm," *arXiv preprint arXiv:1409.4926*, 2014.

[20] MATLAB, *version 7.14 (R2012a)*. Natick, Massachusetts: The MathWorks Inc., 2012.