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

Jian Deng, Haotian Liu, Kim Batselier,
Yu-Kwong Kwok, **Ngai Wong**

ASPDAC 2016

Outline

- Nonlinear model order reduction (NMOR)
- Tensor: what's it?
- Symmetric tensor decomposition
- Symmetric Tensor-based Order Reduction Method (STORM)
- Numerical Examples
- Conclusion

One Origin of Nonlinearity: Products

- Captured by the Kronecker product notation
- Illustration, a 2x1 state vector $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$$x \otimes x = \begin{bmatrix} x_1 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ x_2 \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}$$

Nonlinear Ordinary Differential Eqn (ODE)/Differential Algebraic Eqn (DAE)

- We use a simpler nonlinear ordinary differential equation (ODE), instead of differential algebraic equation (DAE), for ease of illustration. Same idea.
- Nonlinear ODE state space with Kronecker product and a single scalar input u

[redacted] $G_2\left(x \otimes x\right) + G_3\left(x \otimes x \otimes x\right) + bu$

$$x \in \mathbb{R}^{n\times 1}, G_1 \in \mathbb{R}^{n\times n}, G_2 \in \mathbb{R}^{n\times n^2}, G_3 \in \mathbb{R}^{n\times n^3}, b \in \mathbb{R}^{n\times 1}, u \in \mathbb{R}$$

Volterra Series (Time Domain)

$$\ldots G_2\left(x\otimes x\right)+G_3\left(x\otimes x\otimes x\right)+bu$$

Volterra series approximate $x(t)$ globally with different orders of convolutions, called Kernels

$$x(t)=x_1(t)+x_2(t)+x_3(t)+\ldots$$

$$x_1(t)=\int_{-\infty}^{\infty}h_1(\tau)u(t-\tau)d\tau$$

$$x_2(t)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}h_2(\tau_1,\tau_2)u(t-\tau_1)u(t-\tau_2)d\tau_1 d\tau_2$$

$$x_3(t)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}h_3(\tau_1,\tau_2,\tau_3)u(t-\tau_1)u(t-\tau_2)u(t-\tau_3)d\tau_1 d\tau_2 d\tau_3$$

Similar in concept to Taylor series for local behavior…

Time- & Frequency-Domain "Looks"

[redacted] $G_2\left(x\otimes x\right)+G_3\left(x\otimes x\otimes x\right)+bu$

- Time domain

[redacted] bu

[redacted] $G_2\left(x_1\otimes x_1\right)$

[redacted] $G_2\left(x_1\otimes x_2+x_2\otimes x_1\right)+G_3\left(x_1^{\otimes 3}\right)$

[redacted] $G_2\left(x_1\otimes x_3+x_3\otimes x_1+x_2^{\otimes 2}\right)+G_3\left(x_1^{\otimes 2}\otimes x_2+x_2\otimes x_1^{\otimes 2}+x_1\otimes x_2\otimes x_1\right)$

- Frequency domain

$$H_1(s)=\left(sI-G_1\right)^{-1}b$$

$$H_2(s_1,s_2)=\frac{1}{2}\left(\left(s_1+s_2\right)I-G_1\right)^{-1}\left\{G_2\left[H_1(s_1)\otimes H_1(s_2)+H_1(s_2)\otimes H_1(s_1)\right]\right\}$$

$H_3(s_1,s_2,s_3)=\frac{1}{6}\left(\left(s_1+s_2+s_3\right)I-G_1\right)^{-1}\left\{G_2\left[2H_1(s_1)\otimes H_2(s_2,s_3)+\right.\right.$ [redacted]

$H_4(s_1,s_2,s_3,s_4)=$ [redacted]

Nonlinear Model Order Reduction (NMOR)

$$y = h(x,u)$$

$$\hat{y} = \hat{h}(\hat{x},u)$$

- Find a projection matrix V:

$$\begin{bmatrix} x \end{bmatrix} \approx \begin{bmatrix} V \end{bmatrix} \begin{bmatrix} \hat{x} \end{bmatrix}$$

NMOR in Action (Cont'd)

- Kronecker product property: $(AB)\otimes(CD)=\left(A\otimes C\right)\left(B\otimes D\right)$

$$x\otimes x\approx(V\hat{x})\otimes(V\hat{x})=(V\otimes V)(\hat{x}\otimes\hat{x})$$

$$G_2\left(x\otimes x\right)+G_3\left(x\otimes x\otimes x\right)+bu$$

$$V\dot{\hat{x}}=G_1V\hat{x}+G_2(V\otimes V)\left(\hat{x}\otimes\hat{x}\right)+G_3(V\otimes V\otimes V)\left(\hat{x}\otimes\hat{x}\otimes\hat{x}\right)+bu$$

$$\Rightarrow\dot{\hat{x}}=\left(V^TG_1V\right)\hat{x}+\left(V^TG_2(V\otimes V)\right)\left(\hat{x}\otimes\hat{x}\right)$$

$$+\left(V^TG_2(V\otimes V\otimes V)\right)\left(\hat{x}\otimes\hat{x}\otimes\hat{x}\right)+\left(V^Tb\right)u$$

$$\Leftrightarrow\dot{\hat{x}}=G_1'\hat{x}+G_2'\left(\hat{x}\otimes\hat{x}\right)+G_3'\left(\hat{x}\otimes\hat{x}\otimes\hat{x}\right)+b'u$$

NMOR by Moment Matching (@ s_i's)

NORM Algorithm [*Li & Pileggi, DAC03,TCAD05*]

$$H_1(s) = (sI - G_1)^{-1} b = -\left(I - sG_1^{-1}\right)^{-1} G_1^{-1} b = -G_1^{-1} b - G_1^{-2} bs - G_1^{-3} bs^2 - \cdots$$

$$H_2(s_1, s_2) = \frac{1}{2}\left((s_1 + s_2)I - G_1\right)^{-1} \left\{ G_2 \left[H_1(s_1) \otimes H_1(s_2) + H_1(s_2) \otimes H_1(s_1) \right] \right\}$$

$$H_3(s_1, s_2, s_3) = \frac{1}{6}\left((s_1 + s_2 + s_3)I - G_1\right)^{-1} \left\{ G_2 \left[2H_1(s_1) \otimes H_2(s_2, s_3) + \right.\right.$$

1. Use Taylor expansion to expand each transfer function (TF)
2. Match moments with Krylov subspaces

TF	**Moments to be matched for 2nd order accuracy**
$H_1(s)$	$1, s, s^2$
$H_2(s_1,s_2)$	$1, s_1, s_2, s_1^2, s_2^2, s_1 s_2$
$H_3(s_1,s_2,s_3)$	$1, s_1, s_2, s_3, s_1^2, s_2^2, s_3^2, s_1 s_2, s_1 s_3, s_2 s_3$

More columns, SVD etc.

Pros and Cons

Pros

- Automatic macromodel extraction
- Fast and accurate simulation and verification

Cons

- Dense reduced system matrices
- Large memory requirement
- Large computational cost

Curse of Dimensionality

- Any way out?
- We need the right tool(s)!

Tensors in a Nutshell

- Tensor is just a REPRESENTATION for a *d*-way "matrix"

$$\mathcal{A}_{i_1 i_2 \cdots i_d} \in R^{n_1 \times n_2 \times \cdots \times n_d}$$

- It includes

…

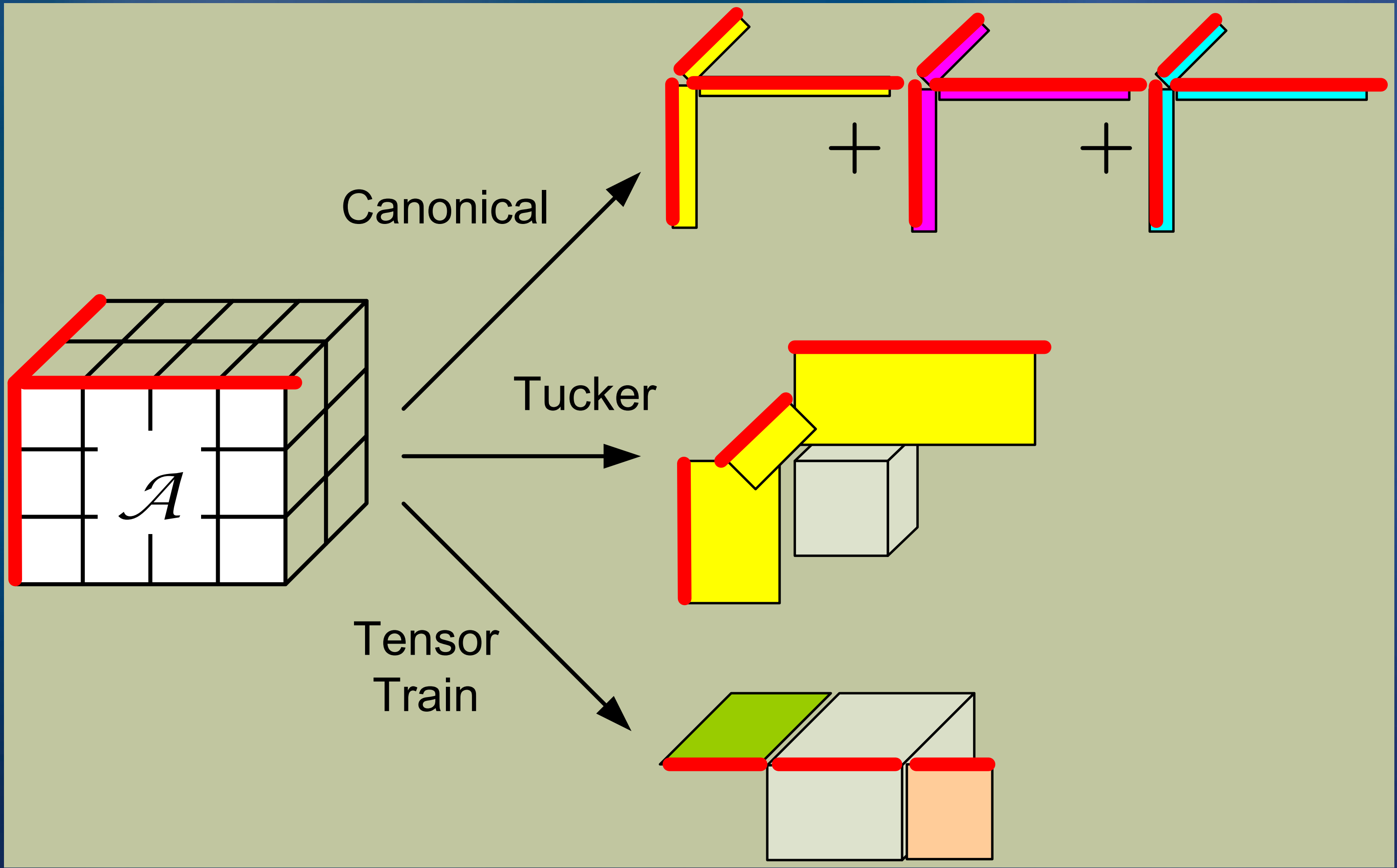
Usage of Tensors

- Numerous!
  - Big data
  - Signal processing
  - Chemometrics
  - Model reduction
  - Nonlinear system modeling
  - ……

- A matter of data representation

Various Decompositions

Canonical, Tucker, Tensor Train

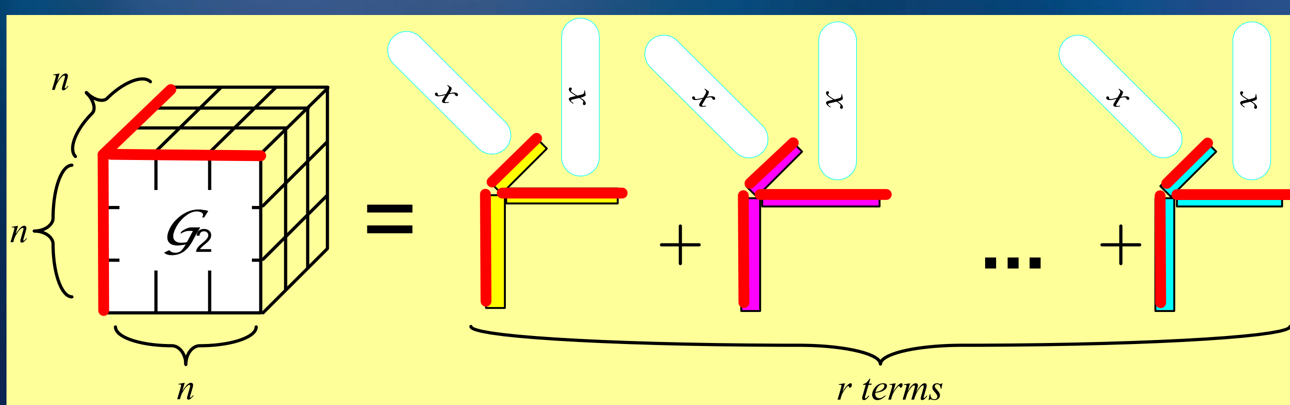


Central Idea of Tensor NMOR

TNMOR: H. Liu, L. Daniel and N. Wong "*Model reduction and simulation of nonlinear circuits via tensor decomposition*"

$$\dots G_2(x\otimes x)+G_3(x\otimes x\otimes x)+bu$$

$$n\{\underbrace{\begin{bmatrix} G_2 \end{bmatrix}}_{n^2}\begin{bmatrix} x\otimes x \end{bmatrix}\}n^2 \qquad O(nn^2)=O(n^3)$$



$$O(2nr+nr)$$

Trick: Exploiting Symmetry

- TNMOR is limited by availability of low-rank decomposition, and NO exploitation of structure such as tensor symmetry.
- Let's try to zoom in a toy case with n=2 in G_2

$$G_2(x \otimes x) = \begin{bmatrix} 10 & 9 & 19 & 20 \\ 1 & 1 & 3 & 5 \end{bmatrix} \left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \otimes \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 10 & 9 & 19 & 20 \\ 1 & 1 & 3 & 5 \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}$$

Get 1st row

Toy G_2 (Cont'd)

- Equivalent to a symmetric matrix and quadratic form

$$G_2(1,:)(x \otimes x) = \begin{bmatrix} 10 & 9 & 19 & 20 \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix}$$

$$= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 10 & 19 \\ 9 & 20 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 10 & 14 \\ 14 & 20 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \left(\begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 1 & 2 \end{bmatrix} + \begin{bmatrix} 3 \\ 4 \end{bmatrix} \begin{bmatrix} 3 & 4 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$= \left(\begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)^2 + \left(\begin{bmatrix} 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)^2 = \begin{bmatrix} 1 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)$$

- ‘ ∘ ’ stands for Hadamard (element-wise) product

Toy G_3 (Cont'd)

- Now let's do $G_3(1,:)$

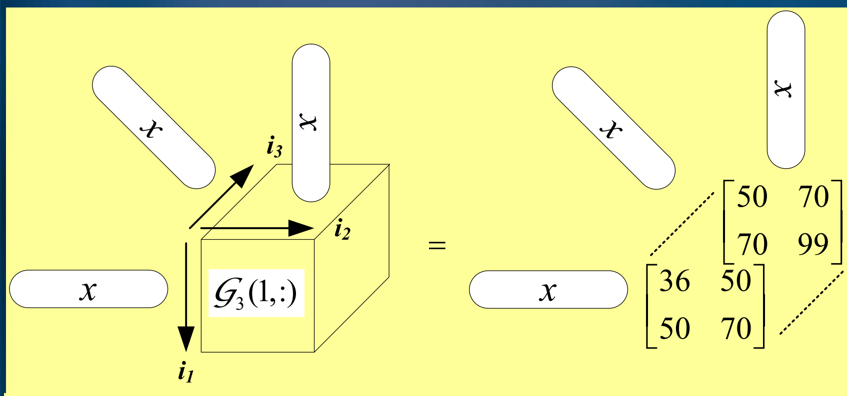
$$G_3(1,:)(x \otimes x \otimes x) = \begin{bmatrix} 36 & 30 & 20 & 90 & 100 & 110 & 10 & 99 \end{bmatrix} \begin{bmatrix} x_1^3 \\ x_1^2 x_2 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_1 x_2^2 \\ x_2^3 \end{bmatrix}$$

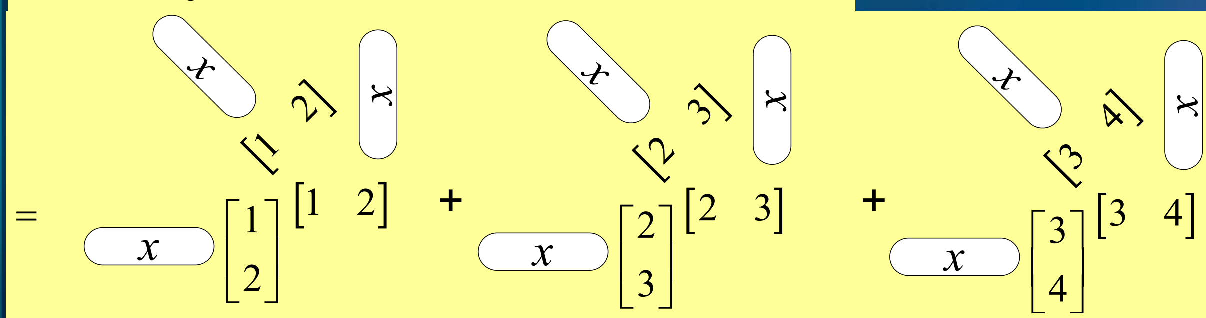
Symmetrizing

$$= \begin{bmatrix} 36 & 50 & 50 & 70 & 50 & 70 & 70 & 99 \end{bmatrix} \begin{bmatrix} x_1^3 \\ x_1^2 x_2 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_1^2 x_2 \\ x_1 x_2^2 \\ x_1 x_2^2 \\ x_2^3 \end{bmatrix}$$

Toy G_3 (Cont'd)

- Now let's do $G_3(1,:)$





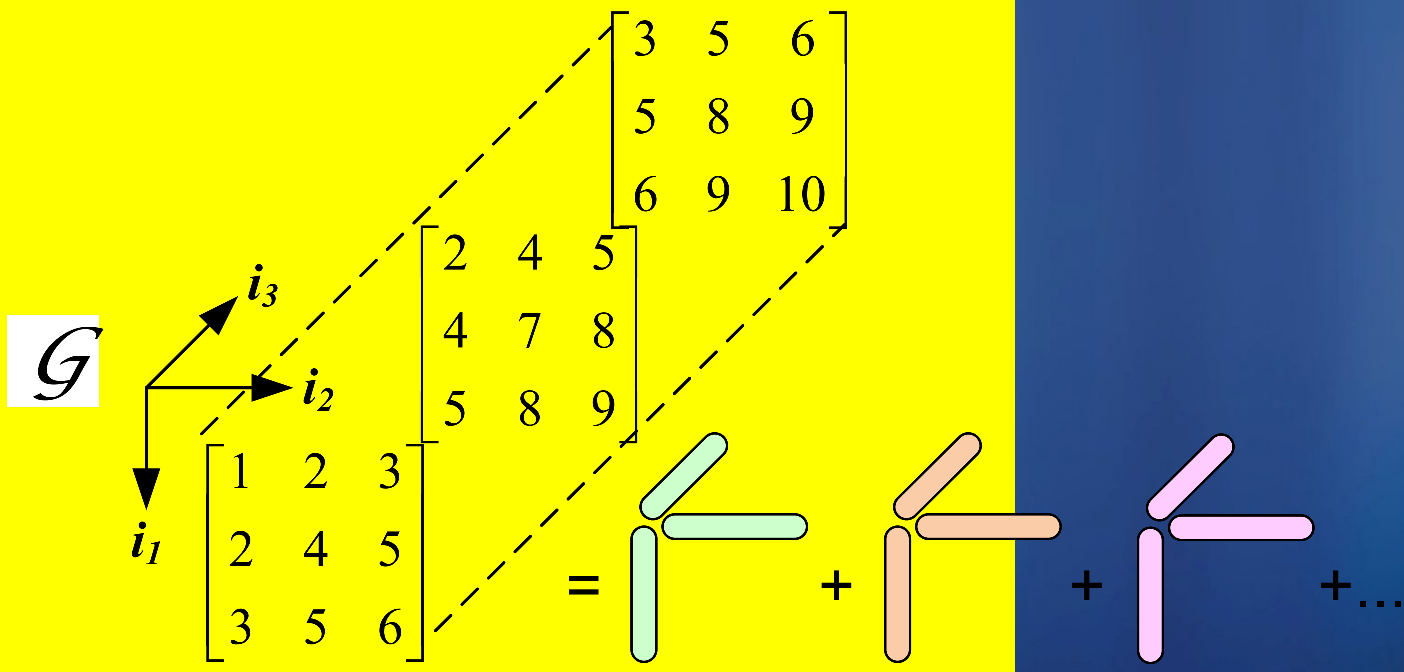
'∘' stands for Hadamard (element-wise) product

$$
=\left(\begin{bmatrix}1 & 2\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}\right)^3+\left(\begin{bmatrix}2 & 3\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}\right)^3+\left(\begin{bmatrix}3 & 4\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}\right)^3=\begin{bmatrix}1 & 1 & 1\end{bmatrix}\left(\begin{bmatrix}1 & 2\\2 & 3\\3 & 4\end{bmatrix}\begin{bmatrix}x_1\\x_2\end{bmatrix}\right)^{\circ}
$$

Symmetric Tensors

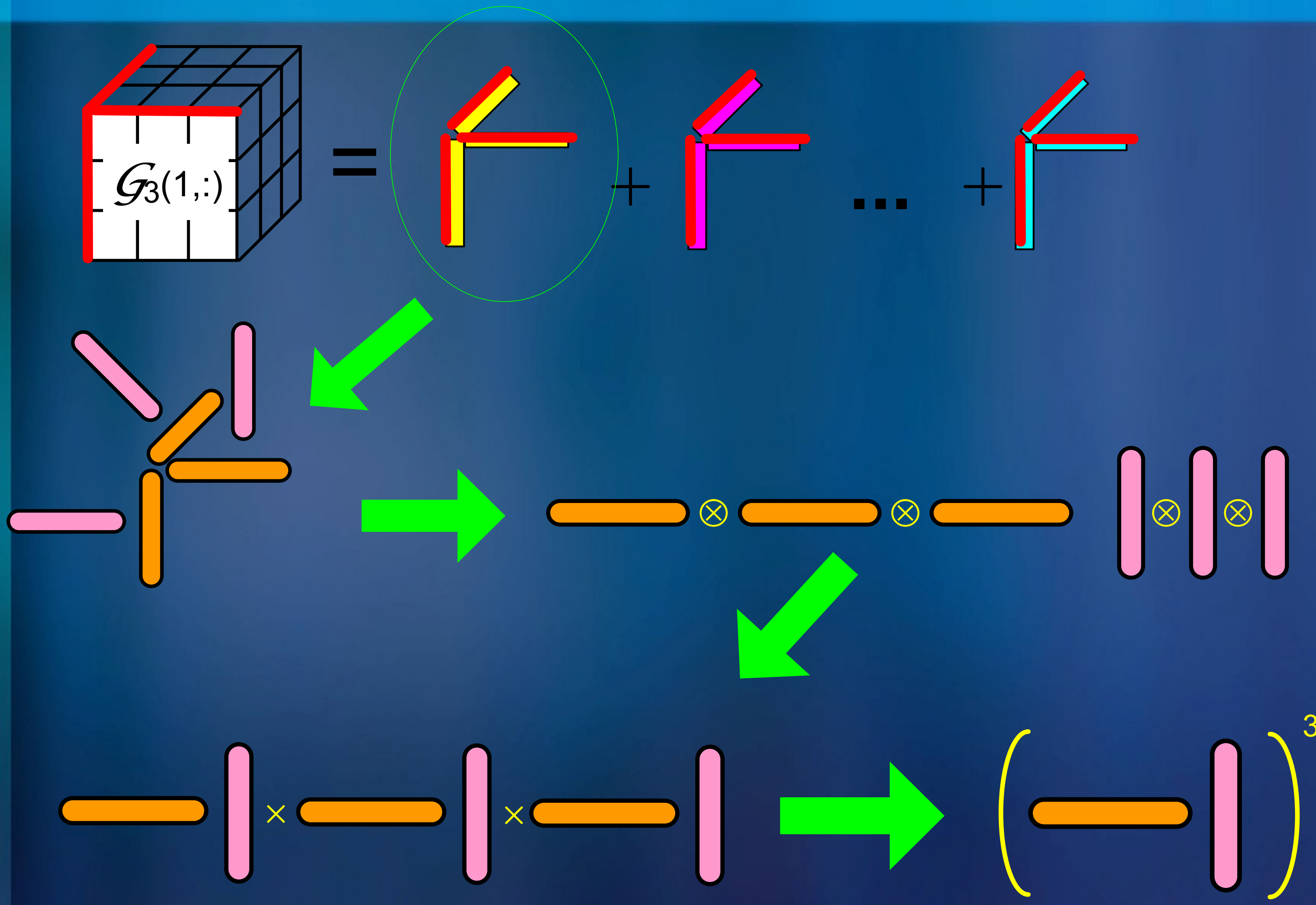
Subset of general arbitrary tensors

$$\mathcal{G}_{i_1 i_2 i_3} = \mathcal{G}_{i_1 i_3 i_2} = \mathcal{G}_{i_2 i_1 i_3} = \mathcal{G}_{i_2 i_3 i_1} = \mathcal{G}_{i_3 i_1 i_2} = \mathcal{G}_{i_3 i_2 i_1}$$



STEROID (Symmetric Tensor Eigen-Rank-One Iterative Decomposition) decomposes a symmetric tensor into symmetric outer products, see K. Batselier and N. Wong, "*Symmetric tensor decomposition by an iterative eigendecomposition algorithm*"

Key: Symmetric Tensor Decomp



STORM

- Symmetric tensor can always be decomposed into a finite sum of symmetric outer products
- Symmetric Tensor-based Order Reduction Method
- Same trick applies to computing Krylov subspaces
- Recalling

$$H_1(s)=\left(sI-G_1\right)^{-1}b=-\left(I-sG_1^{-1}\right)^{-1}G_1^{-1}b=-G_1^{-1}b-G_1^{-2}bs-G_1^{-3}bs^2-\cdots$$

$$H_2(s_1,s_2)=\frac{1}{2}\left(\left(s_1+s_2\right)I-G_1\right)^{-1}\left\{G_2\left[H_1(s_1)\otimes H_1(s_2)+H_1(s_2)\otimes H_1(s_1)\right]\right\}$$

$$H_3(s_1,s_2,s_3)=\frac{1}{6}\left(\left(s_1+s_2+s_3\right)I-G_1\right)^{-1}\left\{G_2\left[2H_1(s_1)\otimes H_2(s_2,s_3)+\right.\right.$$

STORM (Cont'd)

- Reusing Krylov subspace vectors in higher order TF's:

$$H_1(s)=(sI-G_1)^{-1}b=-(I-sG_1^{-1})^{-1}G_1^{-1}b=-\underbrace{G_1^{-1}b}_{v_1}-\underbrace{G_1^{-2}b}_{v_2\left(=G_1^{-1}v_1\right)}s-\underbrace{G_1^{-3}b}_{v_3\left(=G_1^{-1}v_2\right)}s^2-\cdots$$

$$\in \text{span}\underbrace{[v_1 \quad v_2 \quad v_3 \quad \cdots]}_{V_1}$$

Commonly known as the Krylov subspace

$$H_2(s_1,s_2)=\frac{1}{2}\left((s_1+s_2)I-G_1\right)^{-1}\left\{G_2\left[H_1(s_1)\otimes H_1(s_2)+H_1(s_2)\otimes H_1(s_1)\right]\right\}$$

$$=-\frac{1}{2}\left(I-(s_1+s_2)G_1^{-1}\right)^{-1}\left\{G_1^{-1}G_2\left[H_1(s_1)\otimes H_1(s_2)+H_1(s_2)\otimes H_1(s_1)\right]\right\}$$

$$\in \text{span}\underbrace{\left\{G_1^{-1}G_2V_1\otimes V_1\right\}}_{V_2}$$

Same structure as seen before, same trick applies!

$$H_3(s_1,s_2,s_3)=\frac{1}{6}\left((s_1+s_2+s_3)I-G_1\right)^{-1}\left\{G_2\left[2H_1(s_1)\otimes H_2(s_2,s_3)+\right.\right.$$

STORM NMOR Flow

Nonlinear dynamical system in DAE or ODE

↓ STEROID tensor decomp

Symmetric tensor reformulation

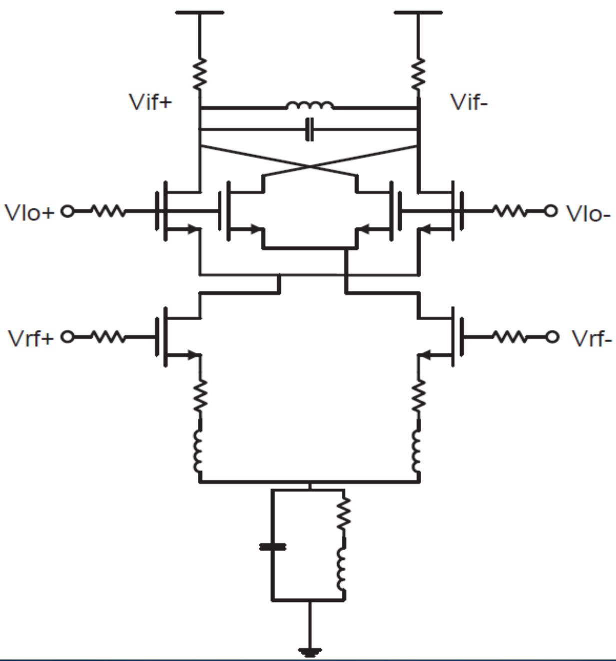
↓ STORM NMOR projection

Reduced-order dynamical system in symmetric tensor model

Example 1: Double Balanced Mixer

Double-balanced mixer
[W. Dong 09] (n=93)

$f_{lo} = 200\text{MHz}, f_{rf} = 2\text{GHz}$



ROM size and CPU time of MOR

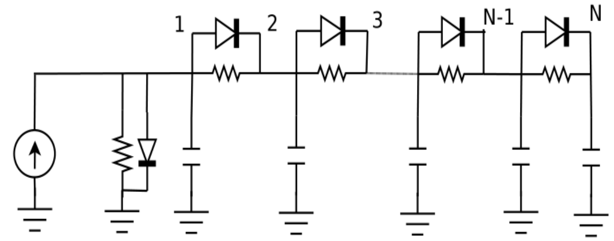
Method	k_1	k_2	k_3	CPU time (s)	size of ROM
NORM	2	2	—	35.33	48
TNMOR	2	2	—	9.53	36
STORM	2	2	2	16.31	49

CPU times and errors of transient simulations

Transient	full model	NORM	TNMOR	STORM
size	93	48	36	49
CPU time (s)	991.51	172.73	16.86	3.23
speedup	—	6x	60x	300x
error	—	5.05%	3.24%	4.06%

Example 2: Nonlinear Transmission Line

Nonlinear transmission line [E. Afshari 05]



ROM size and CPU time of MOR

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

CPU times and errors of transient simulations

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%

Conclusions

STORM: Symmetric tensor-based NMOR

- Utilizes structure of a symmetric tensor to accurately and efficiently model the high-order nonlinear dynamic system
- Represents the symmetric tensor model via symmetric tensor decomposition, reduces the computational and storage cost
- Achieves better speedup in transient simulation compared to other NMOR methods
- Avoids the low-rank limitation of previous tensor-based methods such as TNMOR

Thank You