2016 IEEE 55th Conference on Decision and Control (CDC)
ARIA Resort & Casino
December 12-14, 2016, Las Vegas, USA

Dynamic Tensor Time Series Modeling and Analysis

Amit Surana, Geoff Patterson and Indika Rajapakse

Abstract— **In this paper we propose a model reduction and identification approach for multilinear dynamical system (MLDS) driven by noise. Compared to standard linear dynamical system based approaches which fit vector or matrix models to tensor time series, MLDS provides more natural, compact and accurate representation of tensorial data with fewer model parameters. The proposed algorithm for identifying MLDS parameters employs techniques from multilinear subspace learning: mulilinear Principal Component Analysis and multilinear regression. In addition compact array normal distribution is used to represent and estimate model error and output noise. We illustrate the benefits of the proposed approach on some real world datasets.**

I. Introduction

For many problems arising in domains such as biology, social networks, and signal processing, it is essential to capture simultaneously the function, structure, and dynamics in order to characterize the underlying phenomena. This necessitates analysis of multidimensional or tensor time series data. For instance, the organization of the interphase nucleus reflects a dynamical interaction between 3D genome spatial structure and function, and its relationship to phenotype, a concept known as the 4D Nucleome (4DN) [1]. 4DN research requires a comprehensive view of spatial structure, gene expression, the proteome, and phenotype, and thus is naturally suited for a tensorial representation.

Often in practice vector or matrix models are fitted to tensors so that standard system identification methods can be applied. More recently multilinear subspace learning (MSL) approaches [2] have been developed which preserve the structure of the original data by operating on the natural tensorial representation. MSL methods are higher-order generalizations of linear subspace learning methods such as principal component analysis (PCA), linear discriminant analysis (LDA) and canonical correlation analysis (CCA). It has been shown that MSL approaches can learn more compact representations, have smaller number of parameters than its "vectorized" counterpart, and are more robust to small sample size. However, unlike vector based approaches (e.g. standard PCA), the computational techniques associated with MSL are iterative in nature and can get stuck into local minima, thus posing challenge in terms of initialization and convergence.

A. Surana is with the United Technologies Research Center, 411 Silver Lane, East Hartford, CT 06108, suranaa@utrc.utc.com

G. Patterson is with the Department of Computational Medicine & Bioinformatics, Medical School, University of Michigan, gpatter@umich.edu

I. Rajapakse is with the Department of Computational Medicine & Bioinformatics, Medical School and Department of Mathematics, University of Michigan, indikar@med.umich.edu

While MSL enables static tensor data analysis, only limited work exists which accounts for dynamics and noise in tensor time series analysis, for example see [3]. In this paper we propose a multilinear dynamical system (MLDS) driven by noise as a generative model for tensor time series, and outline a model reduction and identification approach which uses techniques from MSL. The MLDS representation we use is similar to that proposed in [3] which naturally generalizes the notion of standard vector based linear dynamical system (LDS) driven by noise. In the proposed MLDS the state and outputs are tensors instead of vectors, and the state transition and output matrices are replaced by multilinear operator. These multilinear operators are assumed to be factorizable which significantly reduces the number of model parameters (compared to LDS), yet preserve the tensorial structure of the data leading to more accurate models [3]. An Expectation Maximization (EM) method which generalizes the standard EM approach for LDS was proposed in [3] for identifying the MLDS parameters from tensor time series data. This approach does not consider any model reduction during the process of identification which can be important for high dimensional problems, e.g. arising in biology.

Motivated from PCA based approximation methods [4], [5] for identifying reduced order LDS, we propose an alternative model identification approach which allows for model reduction during MLDS identification. The approach is iterative and consists of two key steps: i) applying multilinear PCA [6] on output tensor time series to infer corresponding low dimensional hidden state, and output multlinear operator; and ii) applying multilinear regression [7] to low dimensional hidden tensor state time series to compute the state transition multlinear operator. The dimension of the reduced hidden state tensor along each mode can be determined automatically using techniques proposed in [6]. In addition we use a compact array normal distribution [8] (which is a tensor generalization of normal distribution with factorizable covariance structure) to represent model error and output noise. The estimation of error/noise model parameters are incorporated in the two steps leading to a complete MLDS model reduction/identification algorithm.

The paper is organized in four sections. We begin with basics of tensor algebra followed by multilinear PCA and regression framework in Section II. The MLDS is introduced in Section III, and an algorithm for MLDS identification is discussed. Numerical examples are presented in Section IV. The paper is concluded in Section V with directions for future research.

978-1-5090-1837-6/16/$31.00 ©2016 IEEE

II. Preliminaries on Tensors

In this section we review basics of tensor algebra, see [9] for details. We shall denote tensors by italicized capital letters, matrices by bold capital letters, and vectors by bold small letters. $\otimes$ will denote the Kronercker product, $\circ$ be the outer product and $'$ as the matrix/vector transpose. Identity matrix of size d will be represented by $\mathbf{I}_d$, vector of ones of size d by $\mathbf{1}_d$ and vector of zeros of size d by $\mathbf{0}_d$.

A. Tensor Algebra

Let $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$ be an $N-$the order tensor which assumes values in a real tensor space $\mathbb{R}^{I_1} \otimes \cdots \mathbb{R}^{I_N}$, and I_n is the dimension of n-th mode of the tensor. Tensor inner product for two tensors of same dimension is defined as:

$$< \mathcal{T}_1, \mathcal{T}_2 >= \sum_{i_1, \cdots, i_N} \mathcal{T}_1(i_1, \cdots, i_N)\mathcal{T}_2(i_1, \cdots, i_N), \quad (1)$$

leading to tensor norm as $||\mathcal{T}||_F^2 =< \mathcal{T}, \mathcal{T} >$.

For a matrix $\mathbf{M} \in \mathbb{R}^{J \times I_n}$, matrix tensor multiplication $\mathcal{T} \times_n \mathbf{M}$ along mode n results in a tensor $\mathcal{S} \in \mathbb{R}^{I_1 \times \cdots J \cdots \times I_N}$ of same order

$$\begin{aligned} &\mathcal{S}(i_1, \cdots, i_{n-1}, j, i_{n+1}, \cdots, i_N) = \mathcal{T} \times_n \mathbf{M} \\ = &\sum_{i_n=1}^{I_n} \mathcal{T}(i_1, \cdots, i_n, \cdots, i_N)\mathbf{M}(j, i_n). \end{aligned} \quad (2)$$

Matrix tensor multiplication can be generalized to

$$\mathcal{S} = \mathcal{T} \times_1 \mathbf{M}_1 \times_2 \mathbf{M}_2 \cdots \times_N \mathbf{M}_N, \quad (3)$$

where, $\mathbf{M}_i \in \mathbb{R}^{P_i \times I_i}$, and results in a tensor $\mathcal{S} \in \mathbb{R}^{P_1 \times P_2 \cdots P_N}$. This transformation is known as the *Tucker Product* of the tensor $\mathcal{T}$ with the list of matrices $\mathbf{M}_1, \cdots, \mathbf{M}_N$ which we will write for brevity as

$$\mathcal{S} = \mathcal{T} \times \{\mathbf{M}_1, \mathbf{M}_2 \cdots, \mathbf{M}_N\}. \quad (4)$$

An important class of operations related to the Tucker product are the *matricizations* which reshape a tensor into matrices of various dimensions. The mode-n matricization or $n-$th unfolding of a $I_1 \times \cdots, I_N$-dimensional tensor $\mathcal{T}$ is a $I_n \times (\prod_{j:j \neq n} I_j)$ matrix denoted by $\mathbf{T}_{(n)}$. Vectorization of $\mathcal{T}$ will be denoted by corresponding small letter i.e. $\mathbf{t}$. We follow the standard ordering as proposed in [9] for indexing the matricization and vectorization.

The matricization operation facilitates both understanding and computations of the Tucker product via the following set of equivalences:

$$\begin{aligned} \mathcal{S} &= \mathcal{T} \times \{\mathbf{M}_1, \mathbf{M}_2 \cdots, \mathbf{M}_N\} && (5) \\ \mathbf{s} &= (\mathbf{M}_N \otimes \cdots \otimes \mathbf{M}_1)\mathbf{t} && (6) \\ \mathbf{S}_{(n)} &= \mathbf{M}_n \mathbf{T}_{(n)} \mathbf{M}'_{-n} && (7) \end{aligned}$$

where,

$$\mathbf{M}_{-n} = \mathbf{M}_N \otimes \cdots \otimes \mathbf{M}_{n+1} \otimes \mathbf{M}_{n-1} \otimes \cdots \otimes \mathbf{M}_1 \quad (8)$$

Given as set of tensors $\{\mathcal{Y}_i\}_{i=1}^T$ of same dimension, so $\mathcal{Y}_i \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, the mean tensor $\overline{\mathcal{Y}}$ is given as:

$$\overline{\mathcal{Y}} = \frac{1}{T} \sum_{i=1}^{T} \mathcal{Y}_i, \quad (9)$$

and the total scatter $\Psi_{\mathcal{Y}}$ is defined as

$$\Psi_{\mathcal{Y}} = \sum_{i=1}^{T} ||\mathcal{Y}_i - \overline{\mathcal{Y}}||_F^2. \quad (10)$$

We can stack $\{\mathcal{Y}_i\}_{i=1}^T$ to form a $N+1$ tensor $\mathcal{Y}_e \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N \times T}$ whose $(N+1)$-st mode indexes the tensors $\mathcal{Y}_i$.

B. Multilinear PCA

Given a set of tensor samples, $\{\mathcal{Y}_i\}_{i=1}^T$ where, $\mathcal{Y}_i \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, the mulitlinear PCA (MPCA) seeks a transformation $\mathbf{U}_i \in \mathbb{R}^{I_i \times J_i}$ (with $J_i < I_i$ $i = 1, \cdots, N$),

$$\mathcal{X}_i = \mathcal{Y}_i \times \{\mathbf{U}'_1, \cdots, \mathbf{U}'_N\}, \quad (11)$$

such that the scatter $\Psi_{\mathcal{X}}$ (see definition (10)) of low dimensional tensors $\mathcal{X}_i \in \mathbb{R}^{J_1 \times J_2 \cdots \times J_N}$ is minimized. There are no known optimal solutions which allows for the simultaneous optimization of the N projection matrices $\mathbf{U}_i$, and one has to resort to iterative approaches. Algorithm (1) describes a MPCA algorithm [6], and is based on the fact that given projection matrices $\mathbf{U}_1, \cdots, \mathbf{U}_{n-1}, \mathbf{U}_n, \cdots, \mathbf{U}_N$, the matrix $\mathbf{U}_n$ consists of the J_n eigenvectors corresponding to the largest J_n eigenvalue of the matrix

$$\Phi_n = \sum_{i=1}^{T} (\mathbf{Y}_{i(n)} - \overline{\mathbf{Y}}_{(n)})\mathbf{U}_{-n}\mathbf{U}'_{-n}(\mathbf{Y}_{i(n)} - \overline{\mathbf{Y}}_{(n)})' \quad (12)$$

Note that the algorithm terminates after a preselected K iterations or if the condition in step 10 is satisfied. In the algorithm, the dimensionality $J_n, n = 1, \cdots, N$ for each mode is assumed to be known or predetermined. Techniques for adaptive determination of J_n are discussed in [6].

C. Multilinear Regression

Given a set of predictors and outcomes pairs $\{(\mathcal{X}_i, \mathcal{Y}_i)\}_{i=1}^T$ where, $\mathcal{X}_i \in \mathbb{R}^{J_1 \times J_2 \cdots \times J_N}$ and $\mathcal{Y}_i \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, Tucker product can be used to construct a multilinear tensor regression model of the form

$$\mathcal{Y}_i = \mathcal{X}_i \times \{\mathbf{M}_1 \cdots, \mathbf{M}_N\} + \mathcal{E}_i \quad (14)$$

where, $\mathbf{M}_i \in \mathbb{R}^{I_i \times J_i}$, $i = 1, \cdots, T$ and $\mathcal{E}_i$ is a mean zero tensor of same dimension as $\mathcal{Y}_i$. By stacking the tensors $\{\mathcal{X}_i\}_{i=1}^T$, $\{\mathcal{Y}_i\}_{i=1}^T$ and $\{\mathcal{E}_i\}_{i=1}^T$, the relationships (14) can be more compactly represented as:

$$\mathcal{Y}_e = \mathcal{X}_e \times \{\mathbf{M}_1 \cdots, \mathbf{M}_N, \mathbf{I}_T\} + \mathcal{E}_e. \quad (15)$$

Estimation of matrices $\{\mathbf{M}_n\}_{n=1}^T$ can be facilitated by using the form (5), so that

$$\mathbf{Y}_{e(n)} = \mathbf{M}_n \tilde{\mathbf{X}}_{e(n)} + \mathbf{E}_{e(n)}, \quad (16)$$

where, $\tilde{\mathbf{X}}_{e(n)} = \mathbf{X}_{e(n)}(\mathbf{I}_T \otimes \mathbf{M}_{-n})'$. Given $\mathbf{M}_{-n}$ i.e. $\mathbf{M}_1 \cdots, \mathbf{M}_{n-1}, \mathbf{M}_{n+1}, \cdots, \mathbf{M}_N$, for $\mathbf{M}_n$ this is simply a

Algorithm 1 Pseudocode implementation of Multilinear PCA.

Input: A set of tensor samples $\{\mathcal{Y}_i\}_{i=1}^T$ where $\mathcal{Y}_i \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$

Output: Low-dimensional representation $\mathcal{X}_i \in \mathbb{R}^{J_1 \times J_2 \cdots \times J_N}$, $i = 1, \cdots, T$ of the input tensor samples and matrices $\mathbf{U}_n \in \mathbb{R}^{J_i \times I_i}$, $n = 1, \cdots, N$

Algorithm:

1: **Step 1 (Preprocessing):** Center the input samples as $\{\tilde{\mathcal{Y}}_i = \mathcal{Y}_i - \overline{\mathcal{Y}}\}_{i=1}^T$, where $\overline{\mathcal{Y}}$ is the sample mean computed via (9).

2: **Step 2 (Initialization):** Calculate the eigen-decomposition of

$$\Phi_n^0 = \sum_{i=1}^{T} \tilde{\mathbf{Y}}_{i(n)} \tilde{\mathbf{Y}}'_{i(n)}, \tag{13}$$

and set $\mathbf{U}_n$ to consist of the eigenvectors corresponding to the most significant J_n eigenvalues, $n = 1, \cdots, N$

3: **Step 3 (Local optimization):**

4: Calculate $\{\mathcal{X}_i^0 = \tilde{\mathcal{Y}}_i \times \{\mathbf{U}'_1, \cdots, \mathbf{U}'_N\}\}_{i=1}^T$

5: Calculate $\Psi_{\mathcal{X}^0}$ using (10)

6: **for** $k = 1 : K$ **do** (K is $\#$ iterations)

7: **for** $n = 1 : N$ **do**

8: Set $\mathbf{U}_n$ to consist of the J_n eigenvectors of the matrix Φ_n as defined in (12) corresponding to the largest J_n eigenvalues

9: Calculate $\{\tilde{\mathcal{X}}_i^k\}_{i=1}^T$ and $\Psi_{\mathcal{X}^k}$

10: If $\Psi_{\mathcal{X}^k} - \Psi_{\mathcal{X}^{k-1}} < \eta$ break and go to **Step 4**

11: **end for**

12: **end for**

13: **Step 4 (Projection):** The low dimensional tensors are given by $\{\mathcal{X}_i = \mathcal{Y}_i \times \{\mathbf{U}'_1, \cdots, \mathbf{U}'_N\}\}_{i=1}^T$ along with transformation matrices $\mathbf{U}_n$, $n = 1, \cdots, N$.

multivariate linear regression problem, and the least square solution for $||\mathbf{Y}_{e(n)} - \mathbf{M}_n \tilde{\mathbf{X}}_{e(n)}||_F^2$ can be applied to obtain an estimate of $\mathbf{M}_n$. It follows that the estimates of $\{\mathbf{M}_1, \cdots, \mathbf{M}_N\}$ can be obtained through an iterative procedure (see [7]) as described in Algo. 2.

One can model the residuals $\{\mathcal{E}_i\}_{i=1}^T$ such that their vectorization $\mathbf{e}_1, \cdots, \mathbf{e}_T \sim$ i.i.d. $\mathcal{N}_I(\mathbf{0}, \mathbf{\Sigma})$, where $\mathcal{N}$ is the standard normal distribution of dimension $I = \prod_{n=1}^N I_n$, and $\mathbf{\Sigma}$ is the $I \times I$ covariance matrix whose MLE estimate can be obtained via $\mathbf{\Sigma} = \frac{1}{T} \sum_{i=1}^T \mathbf{e}_i \mathbf{e}_i'$. The limitation of this approach is that the sample size T will generally be too small to reliably estimate $\mathbf{\Sigma}$ without making some restrictions on its form. A flexible, reduced-parameter covariance model that retains the tensor structure of the data is the array normal model which assumes a separable covariance matrix [8].

Array Normal Distribution: $\mathcal{E}$ is said to be a array normal distribution, denoted by $\mathcal{E} \sim \mathcal{N}_{I_1 \times \cdots I_N}(\mathcal{M}, \mathbf{\Sigma}_1, \cdots, \mathbf{\Sigma}_N)$, if the distribution of its vectorization $\mathbf{e}$ is given by $\mathbf{e} \sim \mathcal{N}_I(\mathcal{M}, \mathbf{\Sigma}_I \otimes \cdots \otimes \mathbf{\Sigma}_1)$, where $\mathbf{\Sigma}_k$ is a positive definite $I_n \times I_n$ matrix for each $n = 1, \cdots, N$. Note that $\mathbb{E}[\mathbf{E}_{(n)} \mathbf{E}'_{(n)}] \propto \mathbf{\Sigma}_n$, and $\mathbf{\Sigma}_n$ can be interpreted as covariance along $n-$th mode. Given $i = 1, 1 \cdots, T$ samples $\mathcal{E}_i \sim \mathcal{N}_{I_1 \times \cdots I_N}(\mathcal{M}, \mathbf{\Sigma}_1, \cdots, \mathbf{\Sigma}_N)$, Maximum Likelihood Estimate (MLE) of $\mathbf{m}, \mathbf{\Sigma}_1, \cdots, \mathbf{\Sigma}_N$ (where note $\mathbf{m}$ is vectorization of the mean tensor $\mathcal{M}$) can be obtained iteratively using steps described in Algo. 3, see [8] for details.

Algorithm 2 Local Optimization Step in Multilinear Regression.

1: **for** $n = 1 : N$ **do**

2: Compute $\tilde{\mathcal{X}}_e = \mathcal{X}_e \times \{\mathbf{M}_1 \cdots, \mathbf{M}_{n-1}, \mathbf{I}_{J_n}, \mathbf{M}_{n+1}, \cdots, \mathbf{M}_N, \mathbf{I}_T\}$

3: Form $\tilde{\mathbf{X}}_{e(n)}$, the mode-$n$ unfolding of $\tilde{\mathcal{X}}_e$ and solve the least square problem $||\tilde{\mathbf{Y}}_{e(n)} - \mathbf{M}_n \tilde{\mathbf{X}}_{e(n)}||$ to obtain updated $\mathbf{M}_n$

4: **end for**

Algorithm 3 MLE of Array Normal Distribution parameters.

Input: A set of tensor samples $\{\mathcal{E}_i, \}_{i=1}^T$, where $\mathcal{E}_i \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$

Output: $\mathcal{M}$ and $\mathbf{\Sigma}_1, \cdots, \mathbf{\Sigma}_N$

Algorithm:

1: Initialize $\mathbf{\Sigma}_n, n = 1, \cdots, N$

2: Compute $\mathcal{M} = \frac{1}{T} \sum_{i=1}^T \mathcal{E}_i$ and let $\tilde{\mathcal{E}}_e = \mathcal{E}_e - \mathcal{M} \circ \mathbf{1}_T$

3: **for** $k = 1 : K$ **do** (K is $\#$ iterations)

4: **for** $n = 1 : N$ **do**

5: Compute $\mathbf{P}_n = \hat{\mathbf{E}}_{e(n)} \hat{\mathbf{E}}'_{e(n)}$, where $\hat{\mathbf{E}}_{e(n)}$ is the mode-n unfolding of

$$\hat{\mathcal{E}}_e = \tilde{\mathcal{E}}_e \times \{\mathbf{\Sigma}_1^{-1/2}, \cdots, \mathbf{\Sigma}_{n-1}^{-1/2}, \mathbf{I}_{I_n}, \mathbf{\Sigma}_n^{-1/2}, \cdots, \mathbf{\Sigma}_N^{-1/2}, \mathbf{I}_T\} \tag{17}$$

6: Set $\mathbf{\Sigma}_n = \mathbf{P}_n / n_l$, where $n_l = T \times \prod_{n \neq l} I_n$

7: **end for**

8: **end for**

As before the matrix form of the model can be used to obtain iterative algorithms for parameter estimation. As proposed in [7], multiplying the terms in (16) on the right by $\mathbf{\Sigma}_{-n}^{-1/2}$ allows us to express the model as

$$\mathbf{Y}_{e(n)} = \mathbf{M}_n \tilde{\mathbf{X}}_{e(n)} + \tilde{\mathbf{E}}_{e(n)} \tag{18}$$

where, now

$$\mathbf{Y}_{e(n)} \leftarrow \mathbf{Y}_{e(n)} \mathbf{\Sigma}_{-n}^{-1/2} \tag{19}$$

$$\tilde{\mathbf{X}}_{e(n)} \leftarrow \tilde{\mathbf{X}}_{e(n)} \mathbf{\Sigma}_{-n}^{-1/2} \tag{20}$$

and $\tilde{\mathbf{E}}_{e(n)} \sim \mathcal{N}_{I_n \times I_{-n}}(\mathbf{0}, \mathbf{\Sigma}_n, \mathbf{I}_{I_{-n}})$ with $I_{-n} = \prod_{k:k \neq n} I_k$. Given the parameters other than $\mathbf{M}_n$ this a multivariate linear regression with dependent errors, which can solved using generalized least square as described earlier. Analogously, given the current values of the $\mathbf{M}_n$'s, the likelihood can be minimized in the $\mathbf{\Sigma}_n$ by applying the iterative steps (4-7) as discussed in Algo. 3. The multilinear regression approach with the array normal model is summarized in Algo. 4.

Algorithm 4 Pseudocode implementation of Multilinear Regression with Array Normal Model.

Input: A set of tensor sample pairs $\{(\mathcal{X}_i, \mathcal{Y}_i)\}_{i=1}^T$, where $\mathcal{X}_i \in \mathbb{R}^{J_1 \times J_2 \cdots \times J_N}$ and $\mathcal{Y}_i \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$

Output: Matrices $\mathbf{M}_n \in \mathbb{R}^{I_i \times J_i}$, $\mathbf{\Sigma}_n \in \mathbb{R}^{I_i \times I_i}$ and $n = 1, \cdots, N$

Algorithm:

1: **Step 1 (Initialization):** Initialize $\mathbf{M}_n, \mathbf{\Sigma}_n, n = 1, \cdots, N$

2: **Step 2 (Local optimization):**

3: **for** $k = 1 : K$ **do** (K is # iterations)

4: Given current $\mathbf{\Sigma}_n, n = 1 \cdots, N$ apply steps 1-4 in Algo. 2 to the rescaled input samples via Eqns. (19-20) and obtain updated $\mathbf{M}_n, n = 1 \cdots, N$

5: Compute $\mathcal{E}_i = \mathcal{Y}_i - \mathcal{X}_i \times \{\mathbf{M}_1, \cdots, \mathbf{M}_N\}$ and apply steps 4-7 with zero mean i.e. $\mathbf{M} \equiv 0$ to obtain updated $\mathbf{\Sigma}_n, n = 1 \cdots, N$

6: **end for**

III. Tensor Time Series Modeling

In this section we consider dynamic modeling of a tensor time series $\{\mathcal{Y}_t\}_{t=1}^T$, where $\mathcal{Y}_t \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$. We will assume that a discrete time linear dynamical system (LDS) is a generative model with $\{\mathcal{Y}_t\}_{t=1}^T$ as the outputs. Often in practice vector or matrix models are fitted to tensors so that standard system identification methods can be applied. There are several drawbacks of such an approach.

- Firstly, such an approach does not preserve the structure inherent in the tensor representation. For example, for dynamic representation such as LDS it is less clear how the latent vector space of the state space representation relates to the various tensor modes.
- Secondly, the number of parameters needed to describe the tensor samples can be significantly higher in vectorized form, compared to approaches which retain tensor structure. Thus, in problems with sparse data, the vectorized methods are more prone to over fitting.

To alleviate these shortcomings, we propose a multilinear dynamical system (MLDS) representation as a generative model of the tensor time series $\{\mathcal{Y}_t\}_{t=1}^T$, and develop an associated system identification method. To put things in perspective, we first review standard system identification approaches based on vectorization.

A. LDS Model Identification Using Vectorized Form

Consider the vectorized form $\{\mathbf{y}_t\}_{t=1}^T$ of tensor time series $\{\mathcal{Y}_t\}_{t=1}^T$, where $\mathbf{y}_t \in \mathbb{R}^p$ with dimension $p = \prod_{n=1}^N I_n$. Assuming the generative model of $\{\mathbf{y}_t\}_{t=1}^T$ is a LDS, we have

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A}\mathbf{x}_{t-1} + \mathbf{v}_t \\ \mathbf{y}_t &= \mathbf{C}\mathbf{x}_t + \mathbf{w}_t \end{aligned} \tag{21}$$

where, $\mathbf{x}_t \in \mathbb{R}^d$ is a *hidden state variable*, d is state space dimension, $\mathbf{A} \in \mathbb{R}^{d \times d}$ state transition matrix, $\mathbf{C} \in \mathbb{R}^{p \times d}$ is output matrix, $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ is a zero mean normally distributed process noise with covariance $\mathbf{Q}$ which is a $d \times d$ positive definite matrix, and $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$ is the zero mean normally distributed output noise with a positive definite covariance matrix $\mathbf{R} > 0$ of dimension $p \times p$. For brevity we will refer to vector based LDS (21) as vLDS.

The problem of system identification (sysID) consists of estimating parameters $\mathbf{A}, \mathbf{C}, \mathbf{Q}, \mathbf{R}$ from the time series $\{\mathbf{y}\}_{t=1}^T$. There are a number of methods for sysID, including maximum-likelihood methods [10], non-iterative subspace methods [11] or a suboptimal, but computationally efficient procedure based on PCA [12]. Since, $\mathbf{y}_t$ is high dimensional in our application, we seek the method based on PCA which is summarized in Algo. 5.

Algorithm 5 Pseudocode implementation of PCA based standard vector LDS identification

Inputs: Vector time series $\{\mathbf{y}_t\}_{t=1}^T$, state space dimension n

Outputs: $(\mathbf{A}, \mathbf{C}, \mathbf{Q}, \mathbf{R})$

Algorithm:

1: Center the input samples $\{\tilde{\mathbf{y}}_t = \mathbf{y}_t - \overline{\mathbf{y}}\}$, where $\overline{\mathbf{y}} = \frac{1}{T}\sum_{t=1}^T \mathbf{y}_t$ is the sample mean

2: Compute SVD of $\mathbf{Y} = [\tilde{\mathbf{y}}_1, \cdots, \tilde{\mathbf{y}}_T]$, i.e. $\mathbf{Y} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$

3: Let $\mathbf{C} = \mathbf{U}$ and $[\mathbf{x}_1, \cdots, \mathbf{x}_T] = \mathbf{\Sigma}\mathbf{V}^*$, where only first n principal component vectors are retained in $\mathbf{U}, \mathcal{V}$, and corresponding singular values in $\mathbf{\Sigma}$

4: Form the matrices $\mathbf{X}^- = [\mathbf{x}_2, \cdots, \mathbf{x}_T]$ and $\mathbf{X}^+ = [\mathbf{x}_1, \cdots, \mathbf{x}_{T-1}]$ and compute the least square solution of $||\mathbf{X}^+ - \mathbf{A}\mathbf{X}^-||_F^2$ as $\mathbf{A} = \mathbf{X}^+(\mathbf{X}^-)^\dagger$ where, $\dagger$ denotes the pseudo inverse

5: Compute $\hat{\mathbf{v}}_t = \mathbf{x}_{t+1} - \mathbf{A}\mathbf{x}_t, \quad t = 1, \cdots, T-1$ and

$$\mathbf{Q} = \frac{1}{T-1}\sum_{t=1}^{T-1} \hat{\mathbf{v}}_t \hat{\mathbf{v}}_t' \tag{22}$$

6: Obtain noise statistics by using $\hat{\mathbf{w}}_t = \mathbf{y}_t - \mathbf{C}\mathbf{x}_t, t = 1 \cdots, T$

$$\mathbf{R} = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{w}}_t \hat{\mathbf{w}}_t' \tag{23}$$

B. Multilinear Dynamical System Identification

In this section we introduce a MLDS as a generative model of the tensor time series, which takes into account the tensor structure. We propose a factored MLDS as in [3]:

$$\mathcal{X}_{t+1} = \mathcal{X}_t \times \{\mathbf{A}_1, \cdots, \mathbf{A}_N\} + \mathcal{V}_t, \tag{24}$$

$$\mathcal{Y}_t = \mathcal{X}_t \times \{\mathbf{C}_1, \cdots, \mathbf{C}_N\} + \mathcal{W}_t, \tag{25}$$

where, $\mathcal{X}_t \in \mathbb{R}^{J_1 \times J_2 \cdots \times J_N}$ is the latent state with typically $J_n < I_n, n = 1 \cdots, N$, $\mathbf{C}_n \in \mathbb{R}^{I_n \times J_n}$ and $\mathbf{A}_n \in \mathbb{R}^{J_n \times J_n}$ for $n = 1, \cdots, N$. We will assume that $\mathcal{V}_t, \mathcal{W}_t$ are zero mean array normal distributions as discussed in the Section II-C, so that $\mathcal{V}_t \sim \mathcal{N}_{J_1, \cdots, J_N}(\mathbf{0}, \mathbf{Q}_1, \cdots, \mathbf{Q}_N)$, and $\mathcal{V}_t \sim \mathcal{N}_{I_1 \times \cdots I_N}(\mathbf{0}, \mathbf{R}_1, \cdots, \mathbf{R}_N)$, where $\mathbf{Q}_n \in \mathbb{R}^{J_n \times J_n}$ and $\mathbf{R}_n \in \mathbb{R}^{I_n \times I_n}$ are $n-$ mode covariances for $n = 1, \cdots, N$. As for

LDS, we will refer to $\mathcal{V}_t$ as the process noise tensor, and to $\mathcal{W}_t$ as the output noise tensor.

Algorithm (6) describes an approach to estimating MLDS parameters. Note that this algorithm is very similar to Algo. (5), with vector based PCA and least squares replaced by their tensor counterpart, i.e. multilinear PCA and mulitlinear regression, respectively.

Algorithm 6 Pseudocode implementation of MLDS Identification

Inputs: Tensor time series $\{\mathcal{Y}_t\}_{t=1}^T$
Outputs: MLDS parameters: $\{\mathbf{A}_1, \cdots, \mathbf{A}_N\}$, $\{\mathbf{C}_1, \cdots, \mathbf{C}_N\}$, $\{\mathbf{Q}_1, \cdots, \mathbf{Q}_N\}$ and $\{\mathbf{R}_1, \cdots, \mathbf{R}_N\}$
Algorithm:

1: Compute multilinear PCA of $\{\mathcal{Y}_t\}_{t=1}^T$ via Algo. 1 to obtain latent state variables $\{\mathcal{X}_t\}_{t=1}^T$, and $\{\mathbf{C}_1, \cdots, \mathbf{C}_N\}$
2: Apply multlinear regression Algo. 4 to $\{(\mathcal{X}_t, \mathcal{X}_{t+1})\}_{t=1}^{T-1}$ to obtain $\{\mathbf{A}_1, \cdots, \mathbf{A}_N\}$, and the model noise covariance parameters $\{\mathbf{Q}_1, \cdots, \mathbf{Q}_N\}$
3: Compute the residuals $\mathcal{E}_t = \mathcal{Y}_t - \hat{\mathcal{Y}}_t, t = 1, \cdots, T$, where

$$\hat{\mathcal{Y}}_t = \mathcal{X}_t \times \{\mathbf{C}_1, \cdots, \mathbf{C}_N\}, \tag{26}$$

and, use Algo. 3 to obtain the output noise covariance parameters $\{\mathbf{R}_1, \cdots, \mathbf{R}_N\}$.

C. Comparison of Number of Parameters in vLDS and MLDS Representations

Here we discuss how the number of parameters for vLDS and MLDS scale with the order of the tensor. For discussion we consider the input samples $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$. The number of parameters for the vLDS is

$$N_p^v = n \prod_{n=1}^N I_n + \frac{1}{2} \prod_{n=1}^N I_n (\prod_{n=1}^N I_n + 1) + n^2 + \frac{1}{2} n(n+1) \tag{27}$$

where, latent state vector $\mathbf{x}$ is of dimension n, while that for MLDS is

$$N_p^T = \sum_{n=1}^N I_n J_n + \frac{1}{2} \sum_{n=1}^N I_n (I_n + 1) + \sum_{n=1}^N J_n^2 + \frac{1}{2} \sum_{n=1}^N J_n (J_n + 1) \tag{28}$$

where, the latent state tensor $\mathcal{X}$ has the size $J_1 \times J_2 \cdots \times J_N$. The terms involving $\prod_{n=1}^N I_n$ dominate the expression (27), and thus the number of parameters in vector based LDS grow exponentially with order N of input tensor $\mathcal{Y}$. On other hand for MLDS this growth is only linear in N. As pointed earlier MLDS not only preserves inherent tensorial structure of the data, it has exponentially less number of parameters compared to vLDS, and thus provide significant advantage when number of samples T is small.

IV. Numerical Examples

In this section we compare the performance of vLDS and MLDS on tensor time series datasets. We use model fit error and prediction error to compare the accuracy. The model fit error for each time point is defined as the relative error:

$$E_t^m = \frac{||\mathcal{Y}_t - \hat{\mathcal{Y}}_t||_F}{||\mathcal{Y}_t||_F}, \tag{29}$$

where, $\hat{\mathcal{Y}}_t, t = 1, \cdots, T$ is the mean output

$$\begin{aligned} \hat{\mathcal{X}}_{t+1} &= \hat{\mathcal{X}}_t \times \{\mathbf{A}_1, \cdots, \mathbf{A}_N\}, \\ \hat{\mathcal{Y}}_t &= \hat{\mathcal{X}}_t \times \{\mathbf{C}_1, \cdots, \mathbf{C}_N\}, \end{aligned} \tag{30}$$

generated from the MLDS model fitted to $\{\mathcal{Y}_t\}_{t=1}^T$. The prediction error at each time step is similarly defined as,

$$E_t^p = \frac{||\mathcal{Y}_t - \hat{\mathcal{Y}}_t^p||_F}{||\mathcal{Y}_t||_F}, \tag{31}$$

where, $\{\mathcal{Y}_t\}_{t=T}^{T+T_p}$ is future time series starting at $\mathcal{Y}_T$, and $\{\hat{\mathcal{Y}}_t^p\}_{t=T}^{T+T_p}$ are the mean predictions generated from (30) starting at $t = T$. One can similarly define errors for the vLDS representation. To make the comparison between vLDS and MLDS fair, we impose the constraint

$$\frac{n}{T} = \sqrt[N]{\frac{\prod_{n=1}^N J_n}{\prod_{n=1}^N I_n}}, \tag{32}$$

which, implies that the "order" of dimensionality reduction in vLDS is same as that in MLDS.

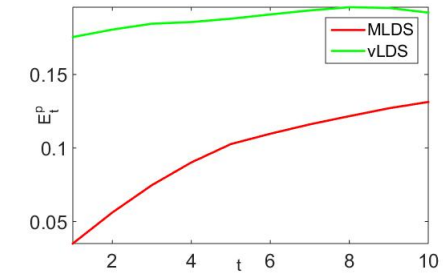
(a) Traffic (b) Waterfall (c) Flame

Fig. 1. Snapshots of three DT videos used in our numerical study.

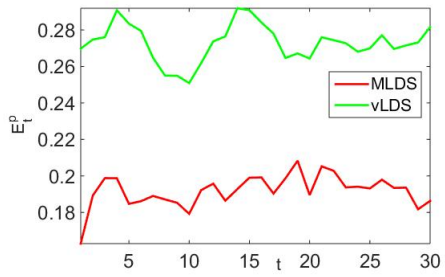
A. Video Dataset

We first consider modeling of dynamic texture (DT) videos which motivated the development PCA based sysID technique discussed in Section III-A. DT are video sequences with spatially repetitive and temporally varying visual patterns that exhibit certain stationary properties, e.g. flames, waterfall, escalators, fire and smoke, flowing water etc., see Figure 1. Each video has three color channels (i.e. Red, Blue, Green), and therefore is a third order tensor. Figure 2 shows comparison of prediction errors for three videos (whose snapshots are shown in Fig. 1): a) traffic video is 48×64 pixel resolution with $T = 42$ time points used for model identification, and a prediction horizon $T_p = 10$; b) waterfall video is 44×53 with $T = 120$ and $T_p = 30$; and c) flame video is 52×66 with $T = 160$, and $T_p = 40$.

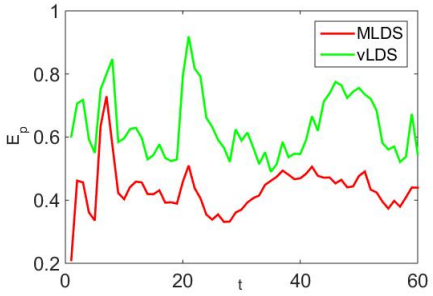
All the above results are without model reduction so that $J_n = I_n, n = 1, 2, 3$, and therefore $n = T$ by Eqn (32). Also, we found that it is better to fit MLDS model to mean zero tensor time series data and then add the mean during



(a) Traffic Video



(b) Waterfall Video



(c) Flame Video

Fig. 2. Prediction error comparison between vLDS and MLDS for DT videos.

reconstruction in Eqn (30). All the initializations required in Algo. 1 and 4 were randomized. For each of above videos, MLDS achieves higher prediction accuracy compared to vLDS. Also, note that the accuracy is much better for traffic and waterfall videos compared to the flame video, as the former two videos have much more correlation structure which is better captured by tensor structure preserving MLDS.

B. Biological Dataset

The organization of the interphase nucleus reflects a dynamical interaction between 3D genome structure and function and its relationship to phenotype, a concept known as the 4D Nucleome (4DN) [1]. 4DN research requires a comprehensive view of architecture, gene expression, the proteome, and phenotype. The data is naturally suited for a tensorial representation. In our preliminary study in this paper we consider chromosomal conformation (Hi-C) time series data which is constructed as coupled correlation matrices which change over time [1]. Hence, Hi-C data is a second order tensor. We applied our framework to a 513×513 Hi-C tensor with $T = 7$ time points used for model identification, and a prediction horizon of one time step, i.e. $T_p = 1$. We consider different orders of model reduction such that $\frac{n}{T} = 1, 0.5, 0.25$ (see Eqn (32)). The error comparison is summarized in table below,

$\frac{n}{T}$	1	0.5	0.25
MLDS E^p	0.1606	0.1675	0.1758
vLDS E^p	0.1686	0.2974	0.1357

With the limited data, it is not clear whether MLDS is better than vLDS. In future we will consider longer time series data, and combine gene expression (RNA-seq) with Hi-C data to study function and structure evolution together.

V. Conclusions

In this paper we proposed MLDS as a generative model for tensor time series, and outlined a system identification approach which uses multilinear subspace learning, including multilinear PCA and multilinear regression. Preliminary numerical validation suggests the benefits of the MLDS modeling approach. In future it will be worthwhile to extend the notion of MLDS with tensorial inputs, and generalize notions of observability and controllability gramians for purposes of estimation and control.

References

[1] H. Chen, J. Chen, L. Muir, S. Ronquist, W. Meixner, M. Ljungman, T. Ried, S. Smale, and I. Rajapakse, "Functional organization of the human 4d nucleome," *PNAS*, vol. 112(26), pp. 8002–7, 2015.

[2] K. N. P. Haiping Lu and A. N. Venetsanopoulos, *Multilinear Subspace Learning: Dimensionality Reduction of Multidimensional Data*. Chapman and Hall/CRC Press, Machine Learning and Pattern Recognition Series, Taylor and Francis, 2013.

[3] M. Rogers, L. Li, and S. J. Russell, "Multilinear dynamical systems for tensor time series," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 2013, pp. 2634–2642.

[4] G. Doretto, D. Cremers, P. Favaro, and S. Soatto, "Dynamic Texture Segmentation," in *ICCV*, 2003, pp. 1236–1242.

[5] R. Vidal, S. Soatto, and A. Chiuso, "Applications of hybrid system identification in computer vision," in *Proceedings of the European Control Conference, Kos, Greece*, 2007.

[6] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Mpca: Multilinear principal component analysis of tensor objects," *IEEE Trans. Neural Netw.*, vol. 19, no. 1, pp. 18–39, 2008.

[7] P. Hoff, "Multilinear tensor regression for longitudinal relational data," *Ann. Appl. Stat.*, vol. 9, no. 3, p. 11691193, 2015.

[8] ——, "Separable covariance arrays via the Tucker product, with applications to multivariate relational data," *Bayesian Analysis*, vol. 6, pp. 179–196, 2011.

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

[10] R. H. Shumway and D. S. Stoffer, "An approach to time series smoothing and forecasting using the EM algorithm," *Journal of Time Series Analysis*, vol. 3, no. 4, pp. 253–264, 1982.

[11] P. V. Overschee and B. D. Moor, "N4SID: Subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, no. 1, pp. 75–93, 1994.

[12] G. Doretto, A. Chiuso, Y. Wu, and S. Soatto, "Dynamic Textures," *Int. J. of Comp. Vision*, vol. 51, no. 2, pp. 91–109, 2003.