Learning Higher-Order Topologies with Time Delays from Nodal Observations

Ruben Wijnands, Borbála Hunyadi, Kim Batselier, Geert Leus
Faculty of EEMCS, Delft University of Technology, Delft, The Netherlands

***Abstract*—This paper proposes...**

***Index Terms*—Higher-order interactions, graph Volterra models, tensor decompositions, structural equation models**

I. Introduction

Recently, explicit polynomial feature maps have been constructed using tensors for polynomial regression. Learning and inference of such models becomes efficient through modeling the model parameters as a tensor decomposition. However, these models do not scale well in the number of variables, which can grow quickly when considering multiple different features or past observations per variable. We propose to remove this complexity by collapsing the additional feature or time dimension through taking a linear combination. Furthermore, we provide two efficient alternating least-squares algorithms for estimating the model parameters.

Notation. Scalars are denoted by y, vectors by $\mathbf{y}$, and matrices by $\mathbf{Y}$. Sets or tensors are denoted by $\mathcal{Y}$, which will be clear from the context. We use $\text{vec}(\cdot)$ for the vectorization operator such that the indices of $\text{vec}(\mathcal{Y})$ follow reverse lexicographic ordering [1]. The Kronecker product is denoted by $\otimes$, the Khatri-Rao product (a column-wise Kronecker product) by $\odot$, the outer product by $\circ$, and the Hadamard product by $\circledast$. The repeated Kronecker product of vector $\mathbf{y}$, is given by $\mathbf{y}^{\otimes D} = \mathbf{y} \otimes \cdots \otimes \mathbf{y}$ (with D factors). We use similar notation for the repeated outer and Hadamard products. Finally, the Frobenius inner product between two tensors of the same size $\mathcal{Y}$ and $\mathcal{W}$ is given by $\langle \mathcal{Y}, \mathcal{W} \rangle_{\text{F}} = \text{vec}(\mathcal{Y})^{\mathsf{T}} \text{vec}(\mathcal{W})$.

II. Problem formulation

Until now, we have only considered spatial relationships in the data. Now, we consider also temporal dependencies by incorporating M additional past observations to the CPVM model. The most straight-forward way of integrating such past observations is through adding additional nodes with those past observations as node signals. However, this will significantly grow the model complexity, as an observation $\mathbf{z}_t$ would have $N + (N-1)M$ entries (ignoring the past observations of the target node). In such case, we amongst others consider thus products between past observations, similar to the temporal Volterra model. Instead, we limit ourselves to node-specific linear combinations of signals and maintain the non-linear spatial filters. Consider now filter coefficients $\mathbf{c}_n$ acting on nodal observations of for node n, which we collect in $\mathbf{m}_{n,t} = [\tilde{x}_{n,t}, ..., \tilde{x}_{n,t-M}]^{\mathsf{T}}$, and $\mathbf{M}_t = [\mathbf{1}, \mathbf{m}_1, ..., \mathbf{m}_{N-1}]^{\mathsf{T}} \in \mathbb{R}^{N \times (M+1)}$. For now, consider each observation to be a linear combination of past observations:

$$\begin{aligned}
\mathbf{z}_t &= \begin{bmatrix} 1 \\ \mathbf{m}_{1,t}^{\mathsf{T}} \mathbf{c}_1 \\ \vdots \\ \mathbf{m}_{N-1,t}^{\mathsf{T}} \mathbf{c}_{N-1} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{1}^{\mathsf{T}} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{c}_1^{\mathsf{T}} & & \vdots \\ \vdots & & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{c}_{N-1}^{\mathsf{T}} \end{bmatrix} \begin{bmatrix} \frac{1}{M+1}\mathbf{1} \\ \mathbf{m}_{1,t} \\ \vdots \\ \mathbf{m}_{N-1,t} \end{bmatrix} \\
&= (\mathbf{I} \bullet \mathbf{C}) \text{vec}(\mathbf{M}_t^{\mathsf{T}}) \\
&= (\mathbf{I} \bullet \mathbf{C}) \mathbf{m}_t
\end{aligned} \tag{1}$$

where $\mathbf{C} = [\mathbf{1}, \mathbf{c}_1, ..., \mathbf{c}_{N-1}]^{\mathsf{T}}$ and $\bullet$ denotes the face-splitting product. Note that if $M = 0$, we obtain our model without temporal memory. As before, we assume that the model weights that are associated to non-linear variable interactions in

$$\mathcal{Z}_t = \mathbf{z}_t \circ \ldots \circ \mathbf{z}_t \tag{2}$$

are structured as a rank-R canonical polyadic decomposition (CPD):

$$\mathcal{W} = \sum_{r=1}^{R} \mathbf{w}_r^{(1)} \circ \ldots \circ \mathbf{w}_r^{(D)}. \tag{3}$$

This leads to the following tensor regression model with scalar output y:

$$\begin{aligned}
y &= \sum_{r=1}^{R} (\mathbf{w}_r^{(D)} \otimes \cdots \otimes \mathbf{w}_r^{(1)})^{\mathsf{T}} \mathbf{z}_t^{\otimes D} \\
&= \mathbf{1}^{\mathsf{T}} \left(\mathbf{W}^{(1)\mathsf{T}} (\mathbf{I} \bullet \mathbf{C}) \mathbf{m}_t \circledast \cdots \circledast \mathbf{W}^{(D)\mathsf{T}} (\mathbf{I} \bullet \mathbf{C}) \mathbf{m}_t \right) \mathbf{1} \\
&= \mathbf{1}^{\mathsf{T}} \left(\mathbf{U}^{(1)\mathsf{T}} \mathbf{m}_t \circledast \cdots \circledast \mathbf{U}^{(D)\mathsf{T}} \mathbf{m}_t \right) \mathbf{1} \\
&= \langle \mathcal{U}, \mathcal{M}_t \rangle_{\text{F}}
\end{aligned} \tag{4}$$

Note that the factor matrices $\mathbf{U}^d$ of tensor $\mathcal{U}$ contain vertically stacked rank-1 blocks:

This work is part of the GraSPA project (project 19497 within the TTW OTP programme), which is financed by the Netherlands Organization for Scientific Research (NWO). E-mails: {R.Wijnands}@tudelft.nl. Code available at: https://github.com/rubenwijnands999/Higher-order-interaction-with-time-delays

$$\mathbf{U}^{(d)} = \begin{bmatrix} \mathbf{1}[w_{1,1}^{(d)}, ..., w_{1,R}^{(d)}] \\ \mathbf{c}_1[w_{2,1}^{(d)}, ..., w_{2,R}^{(d)}] \\ \vdots \\ \mathbf{c}_{N-1}[w_{N,1}^{(d)}, ..., w_{N,R}^{(d)}] \end{bmatrix} = \begin{bmatrix} \mathbf{U}_1^{(d)} \\ \mathbf{U}_2^{(d)} \\ \vdots \\ \mathbf{U}_N^{(d)} \end{bmatrix} \tag{5}$$

The challenge is to estimate both the model weights $\mathbf{W}^{(d)}\ \forall d$ and the filter coefficients $\mathbf{C}$.

III. Methods

To start with, we now consider multiple observations of the target node $\mathbf{y} \in \mathbb{R}^T$ and multiple observations $\{\mathbf{m}_t\}_{t=0}^{T-1}$. Let us then consider the following optimization problem:

$$\begin{aligned} \min_{\mathcal{W},\mathbf{C}} \quad & \sum_{t=0}^{T-1} (y_t - \langle \mathcal{U}, \mathcal{M}_t \rangle_{\mathrm{F}})^2 + \lambda \mathcal{R}(\mathcal{U}) \\ \text{s.t.} \quad & \text{CPD-rank}(\mathcal{U}) = R, \\ & \text{rank}(\mathbf{U}_n^{(d)}) = 1\ \forall d, n \end{aligned} \tag{6}$$

where $\mathcal{R}(\mathcal{U})$ is an optional regularization term. The challenging part of the optimization problem is the non-convexity. By taking a block-coordinate descent approach, the problem can be made convex in one factor matrix $\mathbf{U}^{(d)}$, while fixing the other factor matrices. However, the problem remains non-convex in the coefficients $\mathbf{C}$, as these are shared between each $\mathbf{U}^{(d)}$. In the sections below, we propose two algorithms that tackle this problem using different approaches. The first algorithm, ALS-SVD, includes projections onto the set of matrices that satisfy the last constraint of (6). The second algorithm, ALS-LR, will relax the last constraint and directly estimate the model weights and coefficients.

A. ALS-SVD

In this algorithm, in each iteration, we naively take a block-coordinate descent approach by solving (6) for each d-th factor matrix $\mathbf{U}^{(d)} \in \mathbb{R}^{N \times R}$ separately, while keeping the other factor matrices fixed, and ignoring the last constraint. Please refers to our previous paper for the steps involved in an iteration []. Subsequently, we exploit that the factor matrices have the special rank-1 structure. Namely, we can stack matrices into $\mathbf{U} = [\mathbf{U}^{(1)}, ..., \mathbf{U}^{(D)}]$, which allows us to cleverly update both the coefficients $\mathbf{C}$ and all factor matrices simultaneously through computing the SVD on each block and selecting the component as a rank-1 approximation.

$$\mathbf{U} = [\mathbf{U}^{(1)}, ..., \mathbf{U}^{(D)}] = \begin{bmatrix} \mathbf{U}_1^{(1)} & \dots & \mathbf{U}_1^{(D)} \\ \vdots & \ddots & \vdots \\ \mathbf{U}_N^{(1)} & \dots & \mathbf{U}_N^{(D)} \end{bmatrix} \tag{7}$$

Note that for the first block $\mathbf{U}_1^{(d)}\ \forall d$, the coefficients are known and the least-squares solution of the corresponding concatenated factor rows is given by the column-wise mean of the first block $\mathbf{U}_1$. For the remaining blocks, we replace the current estimate with the rank-1 approximation obtained from the SVD. For example, we decompose $[\mathbf{U}_2^{(1)}, \mathbf{U}_2^{(2)}, ..., \mathbf{U}_2^{(D)}] = \mathbf{DEF}^{\mathsf{T}}$ and set $\mathbf{c}_1 = \mathbf{d}_1$, i.e. the first singular vector, and update the rows of the factor matrices accordingly using $e_{1,1}\mathbf{f}_1$. The complexity of this algorithm is

$$\mathcal{O}\left(DT(NMR)^2 + D(NMR)^3) + \min(DR(NM)^2), NM(DR)^2)\right) \tag{8}$$

B. Rank-1 constrained optimization

In this algorithm, we assume different coefficients $\mathbf{C}^{(d)}$ per dimension. Then, we can rewrite the model as:

$$x = \mathbf{1}^{\mathsf{T}}\left(\begin{bmatrix} \mathbf{1}\overline{\mathbf{w}}_1^{(d)\mathsf{T}} \\ \mathbf{c}_2^{(d)}\overline{\mathbf{w}}_2^{(d)\mathsf{T}} \\ \vdots \\ \mathbf{c}_{N-1}^{(d)}\overline{\mathbf{w}}_{N-1}^{(d)\mathsf{T}} \end{bmatrix} \circledast \begin{bmatrix} \tilde{\mathbf{m}}_1\mathbf{p}^{\mathsf{T}} \\ \vdots \\ \tilde{\mathbf{m}}_{N-1}\mathbf{p}^{\mathsf{T}} \end{bmatrix}\right)\mathbf{1}, \tag{9}$$

where $\mathbf{p} = (\mathbf{U}^{(1)\mathsf{T}}\mathbf{m}_t \circledast \cdots \circledast \mathbf{U}^{(D)\mathsf{T}}\mathbf{m}_t) \in \mathbb{R}^R$ excludes the d-th term. Solving for $\mathbf{c}^{(d)} \in \mathbb{R}^{NM}$ while fixing $\mathbf{W}^{(d)}$ can be written as

$$x = \mathrm{vec}(\mathbf{C}^{(d)\mathsf{T}})^{\mathsf{T}} \begin{bmatrix} \tilde{\mathbf{m}}_1\mathbf{p}^{\mathsf{T}}\overline{\mathbf{w}}_1^{(d)} \\ \vdots \\ \tilde{\mathbf{m}}_{N-1}\mathbf{p}^{\mathsf{T}}\overline{\mathbf{w}}_{N-1}^{(d)} \end{bmatrix} \tag{10}$$

and for multiple observations, we get

$$\mathbf{m} = \begin{bmatrix} (\overline{\mathbf{w}}_1^{(d)\mathsf{T}}\mathbf{p}_1)\tilde{\mathbf{m}}_{1,1}^{\mathsf{T}} & \dots & (\overline{\mathbf{w}}_{N-1}^{(d)\mathsf{T}}\mathbf{p}_1)\tilde{\mathbf{m}}_{N-1,1}^{\mathsf{T}} \\ \vdots & \ddots & \vdots \\ (\overline{\mathbf{w}}_1^{(d)\mathsf{T}}\mathbf{p}_T)\tilde{\mathbf{m}}_{1,T}^{\mathsf{T}} & \dots & (\overline{\mathbf{w}}_{N-1}^{(d)\mathsf{T}}\mathbf{p}_T)\tilde{\mathbf{m}}_{N-1,T}^{\mathsf{T}} \end{bmatrix} \mathrm{vec}(\mathbf{C}^{(d)\mathsf{T}}) \tag{11}$$

Regarding regularization, we can write

$$\begin{aligned} \langle \mathcal{U}, \mathcal{U} \rangle_F &= \mathbf{1}^{\mathsf{T}}\left((\mathbf{U}^{(d)\mathsf{T}}\mathbf{U}^{(d)}) \circledast \mathbf{H}\right)\mathbf{1} \\ &= \mathbf{1}^{\mathsf{T}}\left((\sum_{i=1}^{N} \mathbf{c}_i^{(d)\mathsf{T}}\mathbf{c}_i^{(d)}(\overline{\mathbf{w}}_i\overline{\mathbf{w}}_i^{\mathsf{T}})) \circledast \mathbf{H}\right)\mathbf{1} \\ &= \sum_{i=1}^{N} \mathbf{c}_i^{(d)\mathsf{T}}\mathbf{c}_i^{(d)}(\overline{\mathbf{w}}_i^{\mathsf{T}}\mathbf{H}\overline{\mathbf{w}}_i) \\ &= \mathrm{vec}(\mathbf{C}^{(d)\mathsf{T}})^{\mathsf{T}}\left(\sum_{i=1}^{N} \mathbf{S}_i^{\mathsf{T}}\mathbf{S}_i(\overline{\mathbf{w}}_i^{\mathsf{T}}\mathbf{H}\overline{\mathbf{w}}_i)\right)\mathrm{vec}(\mathbf{C}^{(d)\mathsf{T}}) \\ &= \mathrm{vec}(\mathbf{C}^{(d)\mathsf{T}})^{\mathsf{T}}\left(\mathrm{diag}((\overline{\mathbf{w}}_1^{\mathsf{T}}\mathbf{H}\overline{\mathbf{w}}_1)\mathbf{I}_M, ..., (\overline{\mathbf{w}}_T^{\mathsf{T}}\mathbf{H}\overline{\mathbf{w}}_T)\mathbf{I}_M\right)\mathrm{vec}(\mathbf{C}^{(d)} \end{aligned} \tag{12}$$

where $\mathbf{H} = \mathbf{U}^{(1)\mathsf{T}}\mathbf{U}^{(1)} \circledast \cdots \circledast \mathbf{U}^{(D)\mathsf{T}}\mathbf{U}^{(D)}$ excludes the d-th term. Also, $\mathbf{S}_i \in \mathbb{R}^{N \times NM}$ selects a $\mathbf{c}_i$ from $\mathbf{C}$.
Writing the model in terms of $\mathbf{W}^{(d)}$ results in

$$x = \begin{bmatrix} \mathbf{c}_1^{(d)\mathsf{T}}\tilde{\mathbf{m}}_1\mathbf{p}^{\mathsf{T}} & \dots & \mathbf{c}_{N-1}^{(d)\mathsf{T}}\tilde{\mathbf{m}}_{N-1}\mathbf{p}^{\mathsf{T}} \end{bmatrix} \mathrm{vec}(\mathbf{W}^{(d)\mathsf{T}}) \tag{13}$$

For multiple observations, we get

$$\mathbf{m} = \begin{bmatrix} \mathbf{c}_1^{(d)\mathsf{T}} \tilde{\mathbf{m}}_{1,1} \mathbf{p}_1^{\mathsf{T}} & \dots & \mathbf{c}_{N-1}^{(d)\mathsf{T}} \tilde{\mathbf{m}}_{N-1,1} \mathbf{p}_1^{\mathsf{T}} \\ \vdots & \ddots & \vdots \\ \mathbf{c}_1^{(d)\mathsf{T}} \tilde{\mathbf{m}}_{1,T} \mathbf{p}_T^{\mathsf{T}} & \dots & \mathbf{c}_{N-1}^{(d)\mathsf{T}} \tilde{\mathbf{m}}_{N-1,T} \mathbf{p}_T^{\mathsf{T}} \end{bmatrix} \mathrm{vec}(\mathbf{W}^{(d)\mathsf{T}}) \tag{14}$$

Regarding regularization, we have

$$\begin{aligned} \langle \mathcal{U}, \mathcal{U} \rangle_F &= \mathbf{1}^{\mathsf{T}} \left((\mathbf{U}^{(d)\mathsf{T}} \mathbf{U}^{(d)}) \circledast \mathbf{H} \right) \mathbf{1} \\ &= \mathbf{1}^{\mathsf{T}} \left((\sum_{i=1}^{N} \mathbf{c}_i^{(d)\mathsf{T}} \mathbf{c}_i^{(d)} (\overline{\mathbf{w}}_i \overline{\mathbf{w}}_i^{\mathsf{T}})) \circledast \mathbf{H} \right) \mathbf{1} \\ &= \sum_{i=1}^{N} \mathbf{c}_i^{(d)\mathsf{T}} \mathbf{c}_i^{(d)} (\overline{\mathbf{w}}_i^{\mathsf{T}} \mathbf{H} \overline{\mathbf{w}}_i) \\ &= \mathrm{vec}(\mathbf{W}^{(d)\mathsf{T}})^{\mathsf{T}} \left(\mathrm{diag}(\mathbf{c}_1^{(d)\mathsf{T}} \mathbf{c}_1^{(d)}, ..., \mathbf{c}_N^{(d)\mathsf{T}} \mathbf{c}_N^{(d)}) \otimes \mathbf{H} \right) \mathrm{vec}(\mathbf{W}^{(d)\mathsf{T}}) \end{aligned} \tag{15}$$

REFERENCES

[1] A. Cichocki, N. Lee, I. Oseledets, A. Phan, Q. Zhao *et al.*, "Tensor Networks for Dimensionality Reduction and Large-scale Optimization: Part 1 Low-Rank Tensor Decompositions," *Found. Trends Mach. Learn.*, vol. 9, no. 4-5, pp. 249–429, 2016.