Forecasting Multi-Dimensional Processes Over Graphs

Natali, Alberto; Isufi, Elvin; Leus, Geert

DOI
10.1109/ICASSP40776.2020.9053522

Publication date
2020

Document Version
Final published version

Published in
ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

FORECASTING MULTI-DIMENSIONAL PROCESSES OVER GRAPHS

Alberto Natali, Elvin Isufi and Geert Leus

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology, Delft, The Netherlands
E-mails: {a.natali; e.isufi-1; g.j.t.leus}@tudelft.nl

ABSTRACT

The forecasting of multi-variate time processes through graph-based techniques has recently been addressed under the graph signal processing framework. However, problems in the representation and the processing arise when each time series carries a vector of quantities rather than a scalar one. To tackle this issue, we devise a new framework and propose new methodologies based on the graph vector autoregressive model. More explicitly, we leverage product graphs to model the high-dimensional graph data and develop multi-dimensional graph-based vector autoregressive models to forecast future trends with a number of parameters that is independent of the number of time series and a linear computational complexity. Numerical results demonstrating the prediction of moving point clouds corroborate our findings.

Index Terms— Forecasting, graph signal processing, product graphs, time series, vector autoregressive model.

1. INTRODUCTION

Forecasting time processes is crucial in biology, finance, and position estimation, where historical information is exploited to predict future trends and subsequently take optimal decisions [1] [2] [3]. When the process is multi-dimensional, forecasting methods should also account for the structure hidden in the data to improve accuracy. The canonical tool to exploit these relations is the vector autoregressive (VAR) recursion [4]. Conventional VAR models are nevertheless overparameterized; hence, requiring a large amount of training data. One way to reduce the number of parameters in VAR models is to express the process relations through a graph [5] [6]. In that case, the VAR model parameter matrices are substituted with graph filters leading to a number of trainable parameters that is independent of the process dimension as well as to a lower implementation complexity. As such, the so-called graph VAR model in [6] has led to a superior prediction accuracy compared to the classical VAR model.

The graph VAR models each time series as a scalar signal evolving over the nodes of a graph. However, it does not account for situations where multiple time series are related to a single node. In a position estimation problem, where each node is a point in the 3D space, we have three-dimensional processes evolving over each node. A similar problem is also encountered when predicting related quantities in weather monitoring sensor networks, e.g., temperature, humidity, air quality, and chemical concentrations. While the graph VAR model from [6] can be adopted here to predict the evolution of each time series separately, it will not account for the additional hidden relations in the multi-dimensional process at each node.

To account for this additional information and improve the prediction accuracy, we devise a product graph VAR model that at one hand captures the inter-relations between the processes of different nodes, while on the other hand captures the intra-relations in the multi-dimensional process at a single node. Product graphs enjoy a wide popularity in modeling structured information in large structured data sets [7]. They have been used to aid sampling strategies for graph data [8] [9] [10], build graph wavelets on circulant graphs [11], represent a graph process as a time-invariant graph signal on a larger graph [12] [13], and to build graph neural networks [17]. In this work, we use product graphs to aid forecasting of multi-dimensional processes on graphs. One of the components in the product graph will model the inter-relations between nodes, while the other will capture the intra-relations between features of a node. We subsequently build on the concept of product graph filters [19] to put forth a general parametric VAR model that can forecast multi-dimensional processes with a number of parameters that is independent of the graph and feature dimensions. The latter is also extended to a novel approach that parameterizes also the type of product graph and allows learning an ad-hoc structure for forecasting purposes. Numerical results for position prediction of a 3D point cloud corroborate the proposed method and its potential to improve accuracy compared with the single-feature graph models [6].

The rest of this paper proceeds as follows. Section 2 lays down some preliminary material and the graph VAR model of [6]. Section 3 contains the proposed product graph VAR model. Section 4 contains the numerical results, while Section 5 concludes the paper.

2. GRAPH VAR MODEL

Consider an $N-$dimensional time series $\mathbf{x}_t \in \mathbb{R}^N$, in which each entry $x_t(i)$ is a time-varying signal for a quantity of interest; $x_t(i)$ can be a temperature recording by a sensor i at time t, while $\mathbf{x}_t$ collects the recordings of N such sensors. We can model the temporal evolution of $\mathbf{x}_t$ through the vector autoregressive model:

$$\mathbf{x}_t = -\sum_{p=1}^{P} \mathbf{A}_p \mathbf{x}_{t-p} + \boldsymbol{\varepsilon}_t, \tag{1}$$

which expresses the current value $\mathbf{x}_t$ as the linear combination of P past realizations $\mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-P}$. The $N \times N$ matrices $\mathbf{A}_p$ contain the N^2 parameters of this model and express the influence of the different entries of $\mathbf{x}_{t-p}$ into $\mathbf{x}_t$; vector $\boldsymbol{\varepsilon}_t \in \mathbb{R}^N$ collects the model error also labeled as the *innovation* term [4]. Estimating the PN^2 parameters in (1) is challenging, especially when N is large. As such, parametric models for $\mathbf{A}_p$ are necessary and popular choices include factor models [14] [15] or low-rank data representations [4].

When the relations between the different time series $x_t(i)$ in $\mathbf{x}_t$ can be captured by a network, we can exploit this structure to reduce the PN^2 parameters of (1) in an efficient way that does not hurt prediction accuracy. To be more precise, let graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{S})$

This work was supported in parts by the KAUST-MIT-TUD consortium grant OSR-2015-Sensors-2700.

model the network, where $\mathcal{V} = \{1, \ldots, N\}$ is the set of nodes (or vertices), $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges, and $\mathbf{S}$ is an $N \times N$ matrix to represent this graph structure. We refer to the matrix $\mathbf{S}$ as the graph shift operator matrix [16] —e.g., the weighted adjacency matrix $\mathbf{W}$ (directed graphs) or the graph Laplacian $\mathbf{L}$ (undirected graphs)— with non-zero entries $[\mathbf{S}]_{ij}$ only if $(i, j) \in \mathcal{E}$ or $i = j$. For a fixed t, $\mathbf{x}_t$ is a graph signal in which entry $x_t(i)$ resides on node i [16].

We can process *instantly* the graph signal $\mathbf{x}_t$ over the graph by using so-called graph filters [18]; by setting $\mathbf{x}_t$ as the filter input, we get the filtered signal:

$$\mathbf{y}_t = \sum_{k=0}^{K} h_k \mathbf{S}^k \mathbf{x}_t := \mathbf{H}(\mathbf{S})\mathbf{x}_t, \tag{2}$$

where the polynomial matrix $\mathbf{H}(\mathbf{S}) = \sum_{k=0}^{K} h_k \mathbf{S}^k$ is the graph filter with scalar coefficients $h_0, \ldots, h_K$. The expression in (2) builds the instantaneous output entry $y_t(i)$ of $\mathbf{y}_t$ at node i as a linear combination of $K + 1$ terms: the first term is the signal value $x_t(i)$ of node i; the other K terms are contributions of signal values $x_t(j)$ from K-hop neighbors of node i. Since $\mathbf{S}$ respects the sparsity of the graph, the operation $\mathbf{S}\mathbf{x}_t$ aggregates at node i the signal values of its one-hop neighbors with a complexity $\mathcal{O}(|\mathcal{E}|)$, where $|\mathcal{E}|$ is the number of edges. Likewise, $\mathbf{S}^K\mathbf{x}_t = \mathbf{S}(\mathbf{S}^{K-1}\mathbf{x}_t)$ aggregates at node i the signal values of its K-hop neighbors with a complexity $\mathcal{O}(K|\mathcal{E}|)$, which is also the complexity of the operation in (2).

The graph filtering operation in (2) represents a powerful way to instantly process $\mathbf{x}_t$ by accounting for the underlying structure. Graph filters have been used in [6] to substitute the P matrices $\mathbf{A}_p$ in (1) with P different graph filters for modeling $\mathbf{x}_t$ as:

$$\mathbf{x}_t = -\sum_{p=1}^{P} \mathbf{H}_p(\mathbf{S})\mathbf{x}_{t-p} + \boldsymbol{\varepsilon}_t = -\sum_{p=1}^{P}\sum_{k=0}^{K} h_{kp}\mathbf{S}^k\mathbf{x}_{t-p} + \boldsymbol{\varepsilon}_t. \tag{3}$$

In other words, each of the past realizations $\mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-P}$ is treated as a different graph signal, which is further processed with a different graph filter $\mathbf{H}_p(\mathbf{S})$. As such, the process value $x_t(i)$ at node i depends now on the process values with a temporal window of P and a graph window of K, i.e., $x_t(i)$ depends on the past P values of the nodes that are within K hops from node i. The graph VAR (G-VAR) model in (3) inherits from the graph filters a number of parameters that is $P(K + 1)$ and an implementation complexity of $\mathcal{O}(PK|\mathcal{E}|)$. Both these quantities are orders smaller, and independent on the graph process dimensions, than the respective PN^2 and $\mathcal{O}(PN^2)$ of the classical VAR in (1); hence, requiring less training data to estimate the model parameters.

Despite the above benefits, the G-VAR model cannot handle cases where on each node we have a vector of feature values rather than a scalar $x_t(i)$. Treating each feature separately and predicting its evolution with (3) is certainly an option but it will not account for the dependencies between features. To account efficiently for these dependencies and improve the prediction accuracy, we introduce next the notion of multi-dimensional graph processes and leverage product graphs to develop generalized G-VAR models for forecasting the temporal evolution of such processes.

3. PRODUCT GRAPH VAR MODEL

When on top of each node we have a vector of F feature values rather than a scalar, we talk about a *multi-dimensional* graph signal. We store the feature values of node $i \in \mathcal{V}$ in the $F \times 1$ vector $\mathbf{x}_t(i)$ and refer to it as the *node signal* for node i. The overall *F-dimensional graph signal* at time t is the $NF \times 1$ vector $\mathbf{x}_t = [\mathbf{x}_t^\top(1), \ldots, \mathbf{x}_t^\top(N)]^\top$, i.e., the vector that concatenates all node signals[1]. Likewise, we can also consider the fth feature of all nodes $\{x_t^{(f)}(i)\}_{i\in\mathcal{V}}$ and store them in the $N \times 1$ vector $\mathbf{x}_t^{(f)} = [x_t^{(f)}(1), \ldots, x_t^{(f)}(N)]^\top$, which we refer to as the *feature signal*. With the terminology of Section 2, each feature signal is a graph signal and the F-dimensional graph signal is a collection of F (one-dimensional) graph signals. Fig. 1a illustrates a three-dimensional graph signal.

Our goal now is to forecast the temporal evolution of the $F-$dimensional graph process $\mathbf{x}_t$. We want to develop a graph VAR model as in (3) to forecast now the evolution of an $NF \times 1$ vector $\mathbf{x}_t$. A naive way doing this is to ignore the underlying graph structure $\mathcal{G}$ and build an alternative graph (using topology identification based on a certain metric [20]) of NF nodes and use the G-VAR model (3) to forecast the process. This strategy has two main disadvantages: first, it destroys the known underlying structure present between nodes (e.g., friendships in a social network) and builds a feature-based similarity graph; second, it leads to a shift operator of larger dimensions without any further structure, hence with a larger storage and computational complexity. To tackle both these issues, we use the concept of product graphs to propose a product graph VAR model for forecasting F-dimensional graph processes.

3.1. Product Graph Representation

We assume that the features of each node signal $\mathbf{x}_t(i) \in \mathbb{R}^F$ have a hidden relation which we represent with a *feature graph* $\mathcal{G}_\mathcal{F} = (\mathcal{V}_\mathcal{F}, \mathcal{E}_\mathcal{F}, \mathbf{S}_\mathcal{F})$, where each vertex in $\mathcal{V}_\mathcal{F} = \{1, \ldots, F\}$ is one feature, $\mathcal{E}_\mathcal{F} \subseteq \mathcal{V}_\mathcal{F} \times \mathcal{V}_\mathcal{F}$ is the edge set connecting different features, and $\mathbf{S}_\mathcal{F}$ is the $F \times F$ graph shift operator matrix of $\mathcal{G}_\mathcal{F}$. Observe that all nodes have the same feature graph $\mathcal{G}_\mathcal{F}$. A simple way to build $\mathcal{G}_\mathcal{F}$ is through a feature similarity metric, e.g., Pearson correlation or Gaussian weighting kernels [20]. In other words, we consider each feature graph $\mathcal{G}_\mathcal{F}$ now residing on the nodes $\mathcal{V}$ of the underlying graph $\mathcal{G}$. Fig. 1b illustrates possible feature graphs between the features of each node.

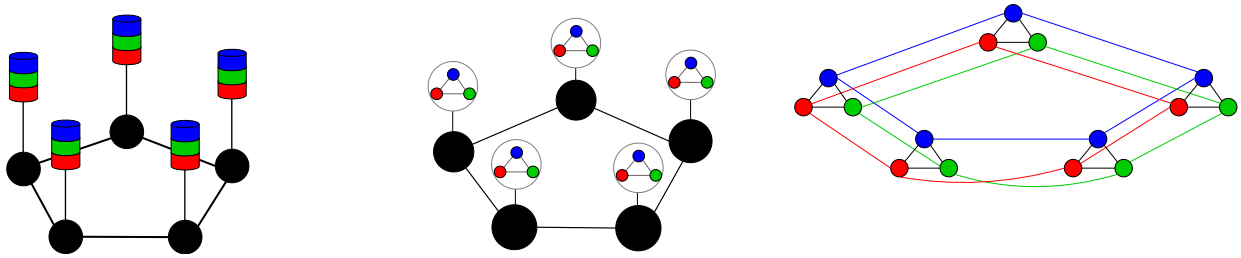
While the graph $\mathcal{G}_\mathcal{F}$ influences the intra-connections between the features of a node i, the graph $\mathcal{G}$ influences the inter-connections between different nodes $i, j \in \mathcal{V}$. In particular, if nodes i and j share an edge in $\mathcal{G}$, this edge should be reflected in the inter-connections between the features of the vectors $\mathbf{x}_t(i)$ and $\mathbf{x}_t(j)$ and, vice-versa, if i and j are not connected in $\mathcal{G}$, there should not be inter-connectivities between the features of the vectors $\mathbf{x}_t(i)$ and $\mathbf{x}_t(j)$. We can formally capture both the intra- and inter-connectivities between features with product graphs. The product graph of the underlying graph $\mathcal{G}$ and the feature graph $\mathcal{G}_\mathcal{F}$ is a new graph:

$$\mathcal{G}_\diamond = \mathcal{G} \diamond \mathcal{G}_\mathcal{F} = (\mathcal{V}_\diamond, \mathcal{E}_\diamond, \mathbf{S}_\diamond), \tag{4}$$

with node set $\mathcal{V}_\diamond$ of cardinality $|\mathcal{V}_\diamond| = NF$, and where the edge set $\mathcal{E}_\diamond \subseteq \mathcal{V}_\diamond \times \mathcal{V}_\diamond$ and the $NF \times NF$ graph shift operator $\mathbf{S}_\diamond$ are dictated by the type of product graph. Popular product graphs are the Kronecker, Cartesian, and strong product [7]. For the Cartesian product, for instance, the graph shift operator is $\mathbf{S}_\diamond = \mathbf{S} \otimes \mathbf{I}_F + \mathbf{I}_N \otimes \mathbf{S}_\mathcal{F}$ with $\mathbf{I}_F$ ($\mathbf{I}_N$) the $F \times F$ ($N \times N$) identity matrix. The edge set cardinality for this graph is $|\mathcal{E}_\diamond| = F|\mathcal{E}| + N|\mathcal{E}_\mathcal{F}|$. Fig. 1c illustrates the Cartesian product graph for the three-dimensional graph signal example.

The F-dimensional graph signal $\mathbf{x}_t$ on $\mathcal{G}$ is now a (one-dimensional) graph signal on $\mathcal{G}_\diamond$. Therefore, we can develop a

[1] Note that the F-dimensional graph signal is denoted with the same symbol $\mathbf{x}_t$, as the one-dimensional graph signal in Section 2. We chose this slight abuse of notation to construct a generalized G-VAR model that follows a similar recursion as in (3).



(a) Three-dimensional graph signal.

(b) Feature graph over each node.

(c) Multi-dimensional graph signal on product graph.

Fig. 1: Multi-dimensional graph signal represented through a product graph. (a) A three-dimensional graph signal. Each color represents a different feature. We can see the three-dimensional graph signal in two ways: (i) each node has a node signal of dimension three composed by the three different colors; (ii) the collection of each feature (e.g., blue) for all nodes forms a feature signal of dimension five (i.e., the number of nodes). (b) A feature graph formed between the features of each node. Each colored circle (red-green-blue) is now a node in the feature graph and the edges connecting them are the edges of the feature graph. (c) Cartesian product graph illustration of the results in (b). The feature graph in (b) influences the intra-connectivities between the features of each node (shown in black); the underlying graph in (a) influences the inter-connectivities between the features of different nodes (shown by the respective feature color to ease visualisation).

G-VAR model as in (3) w.r.t. the product graph shift operator $\mathbf{S}_\diamond$. We will refer to the latter as a product graph VAR model, while we will also develop a more generalized version that goes beyond the direct application of (3) to a product graph. The benefits of this product graph strategy are that the storage complexity of $\mathbf{S}_\diamond$ is that of storing $\mathbf{S}$ and $\mathbf{S}_\mathcal{F}$ separately and the computational complexity is of the combined sparsity orders of $\mathbf{S}$ and $\mathbf{S}_\mathcal{F}$; both in general lower than building a graph for the $NF \times 1$ signal $\mathbf{x}_t$.

Product Graph VAR. As discussed above, we thus model the temporal evolution of the F-dimensional graph process $\mathbf{x}_t \in \mathbb{R}^{NF}$ with the product graph VAR (PG-VAR) model:

$$\mathbf{x}_t = -\sum_{p=1}^{P}\sum_{k=0}^{K} h_{kp}\mathbf{S}_\diamond^k \mathbf{x}_{t-p} + \boldsymbol{\varepsilon}_t. \tag{5}$$

Model (5) expresses, through $\mathbf{S}_\diamond$, the influence that the past F-dimensional graph process realizations $\mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-P}$ have on each node signal at time t, i.e., $\mathbf{x}_t(i)$ for $i \in \mathcal{V}$. The order K implies now that the fth feature $x_t^{(f)}(i)$ in $\mathbf{x}_t(i)$ depends on all other feature realizations $x_{t-p}^{(g)}(j)$ for $g = 1, \ldots, F$, $j \in \mathcal{V}$, and $p = 1, \ldots, P$, within a K-hop neighborhood in the product graph $\mathcal{G}_\diamond$. As such the PG-VAR model forecasts future values of $\mathbf{x}_t$ by accounting on the intra- and inter-connectivities between the different node features. Besides storage and computational benefits, model (5) has also a number of parameters $P(K+1)$ that is independent on F and N; hence, requiring little training data to estimate them. For $\mathcal{G}_\diamond$ being the Cartesian graph, the implementation complexity of (5) is $\mathcal{O}\big(PK(F|\mathcal{E}| + N|\mathcal{E}_\mathcal{F}|)\big)$ which is comparable with the cost $\mathcal{O}(PKF|\mathcal{E}|)$ of F independent G-VAR models (3), since N and $|\mathcal{E}|$ (as well as F and $|\mathcal{E}_\mathcal{F}|$) are often of the same order. But the PG-VAR model captures now additional intra-relations; hence, has the potential to improve the prediction accuracy.

Generalized Product Graph VAR. The performance of the PG-VAR heavily depends on the considered type of product graph. However, it is not clear which is the most suited product graph for a task at hand. Here, we by-pass this issue by jointly estimating the model parameters and the type of product graph. For an underlying graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{S})$ and a feature graph $\mathcal{G}_\mathcal{F} = (\mathcal{V}_\mathcal{F}, \mathcal{E}_\mathcal{F}, \mathbf{S}_\mathcal{F})$, any product graph can be expressed as:

$$\mathbf{S}_\diamond = \sum_{i=0}^{1}\sum_{j=0}^{1} s_{ij}\left(\mathbf{S}^i \otimes \mathbf{S}_\mathcal{F}^j\right), \tag{6}$$

where "$\otimes$" denotes the Kronecker product and $s_{ij} \in \{0, 1\}$. A graph filter $\mathbf{H}(\mathbf{S}_\diamond)$ of order K over the product graph $\mathcal{G}_\diamond$ has thus the form:

$$\mathbf{H}(\mathbf{S}_\diamond) = \sum_{k=0}^{K} h_k \mathbf{S}_\diamond^k = \sum_{k=0}^{K} h_k \left(\sum_{i,j=0}^{1} s_{ij}\left(\mathbf{S}^i \otimes \mathbf{S}_\mathcal{F}^j\right)\right)^k, \tag{7}$$

where we see that the coefficients s_{ij} that define the type of product graph play a role in the filtering behavior. The filter in (7) can be seen as a special case of the more general product graph filter:

$$\mathbf{H}(\mathbf{S}_\diamond) = \sum_{k=0}^{K}\sum_{l=0}^{L} h_{kl}\left(\mathbf{S}^k \otimes \mathbf{S}_\mathcal{F}^l\right), \tag{8}$$

which is now defined by the general parameters h_{kl}. The novel aspect of the general filter in (8) is in the way the shift operators $\mathbf{S}$ and $\mathbf{S}_\mathcal{F}$ are combined within the filter. We can therefore use (8) to extend (5) towards a general PG-VAR model for the F-dimensional graph process $\mathbf{x}_t$ as:

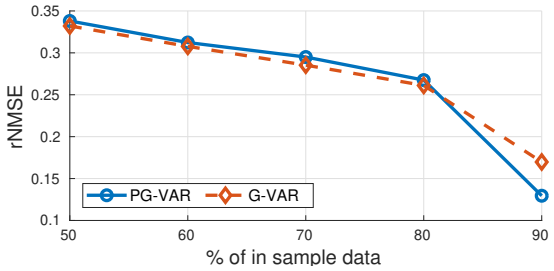
$$\mathbf{x}_t = -\sum_{p=1}^{P}\sum_{k=0}^{K}\sum_{l=0}^{L} h_{lkp}\left(\mathbf{S}^k \otimes \mathbf{S}_\mathcal{F}^l\right)\mathbf{x}_{t-p} + \boldsymbol{\varepsilon}_t. \tag{9}$$

At a slight increase in the number of parameters to estimate, i.e., PKL, compared with model (5), the GP^2-VAR model parameterizes now also the type of product graph. As such it exploits adhoc the intra- and inter-connectivities in the historical data of the F-dimensional process $\mathbf{x}_t$.

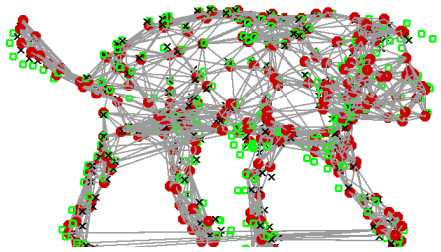
3.2. Parameter estimation

One of the main benefits of models (5) and (9) is that their respective parameters can be readily estimated following the procedure for the G-VAR developed in [6]. Let $\mathbf{H}_p(\mathbf{S}_\diamond)$ for $p = 1, \ldots, P$ be a collection of P graph filters that can represent either the forms (5) or (9). Given the historical data $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{t-1}\}$ of the F-dimensional graph process, the optimal predictor for $\mathbf{x}_t$ is given by the conditional expectation:

$$\tilde{\mathbf{x}}_t = \mathbb{E}\left[\mathbf{x}_t \middle| \{\mathbf{x}_1, \ldots, \mathbf{x}_{t-1}\}\right] = -\sum_{p=1}^{P} \mathbf{H}_p(\mathbf{S}_\diamond)\mathbf{x}_{t-p}, \tag{10}$$



(a)



(b)

Fig. 2: Test rNMSE for the one-step ahead prediction of the two algorithms considering the entire multi-dimensional graph process versus the percentage of in-sample data. (c) One-step-ahead estimated position of the walking dog for a test sample. The red points correspond to the ground-truth position, while the black crosses and the green circles correspond to the PG-VAR and G-VAR predictions, respectively.

since the innovation term $\boldsymbol{\varepsilon}_t$ has a zero mean. The mean square error $\text{MSE}\big(\mathbf{H}_p(\mathbf{S}_\diamond)\big) = \mathbb{E}\left[\|\mathbf{x}_t - \tilde{\mathbf{x}}_t\|_2^2\right]$ for the optimal predictor (10) is:

$$\text{MSE}\big(\mathbf{H}_p(\mathbf{S}_\diamond)\big) = \text{tr}\Bigg(\mathbf{R}_0 + \sum_{p=1}^{P} \mathbf{H}_p(\mathbf{S}_\diamond)_p \mathbf{R}_p + \sum_{p=1}^{P} \mathbf{R}_p \mathbf{H}_p^\top(\mathbf{S}_\diamond) + \sum_{p_1=1}^{P}\sum_{p_2=1}^{P} \mathbf{H}_{p_1}(\mathbf{S}_\diamond)\mathbf{R}_{p_2-p_1}\mathbf{H}_{p_2}^\top(\mathbf{S}_\diamond)\Bigg), \tag{11}$$

where $\mathbf{R}_p$ is the autocorrelation matrix of process $\mathbf{x}_t$ at temporal lag p, i.e., $\mathbf{R}_p = \mathbb{E}\big[\mathbf{x}_t \mathbf{x}_{t-p}^\top\big]$. We then find the parameters for model (10) as those that minimize the MSE. For the PG-VAR model, this implies solving the optimization problem:

$$\begin{aligned} \underset{\{h_{kp}\}}{\text{minimize}} \quad & \text{MSE}\big(\mathbf{H}_p(\mathbf{S}_\diamond)\big) \\ \text{subject to} \quad & \mathbf{H}_p(\mathbf{S}_\diamond) = \sum_{k=0}^{K} h_{kp} \mathbf{S}_\diamond, \end{aligned} \tag{12}$$

where the MSE expression is given in (11). Likewise, for the general PG-VAR model, the framework is similar, with the only difference in the filter expression $\mathbf{H}_p(\mathbf{S}_\diamond)$ and in the optimization variables h_{klp}.

4. NUMERICAL RESULTS

We consider the task of predicting the position of a walking dog mesh [21]. The mesh has $N = 251$ spatial points over $T = 59$ time instants. For each spatial point, we have the three (x, y, z) coordinates that change over time. We treated each point as the nodes of a ten nearest neighbor (10NN) graph built from the coordinates in the first time instant. The node coordinates represent the $F = 3$ features and our goal is to predict the one-step value. As in [6], we pre-processed the data to render them with zero mean and unitary maximum value. We compared the PG-VAR model in (5) with the G-VAR model in (3), where for this dataset the latter has yielded a superior performance compared to the standard VAR in (1) and other graph-based techniques [6]. We did not analyze the generaliz PG-VAR in (9) since the number of time instants is relatively low for the increased number of parameters; we will analyze this model on bigger data sets and present these observations in the extended version of this work.

Experimental setup. We considered the Cartesian product graph to model the relations between the underlying 10NN graph $\mathcal{G}$ and a fully connected feature graph $\mathcal{G}_\mathcal{F}$ of $F = 3$ nodes. The resulting product graph $\mathcal{G}_\diamond$ has 753 nodes connecting different (x, y, z) coordinates. We followed [6] and split the data across the temporal dimension into in-sample and out-of-sample data. We analyzed different percentages of this split ranging from 50% to 90% in-sample data. The in-sample data were used to estimate the model parameters and the out-of-sample data to test the performance. The in-sample data have been further divided into 70% training and 30% validation sets. We found the filter coefficients by fitting the model with different parameters P and K in the training set and assessing their performance in the validation set [22]. Then, we refitted the model with the best performing tuple (P, K) into the whole in-sample set. We evaluated the performance of the different algorithms with the root normalized MSE (rNMSE) defined as:

$$\text{rNMSE} = \sqrt{\frac{\sum_{t=1}^{\tau} \|\tilde{\mathbf{x}}_t - \mathbf{x}_t\|_2^2}{\sum_{t=1}^{\tau} \|\mathbf{x}_t\|_2^2}}, \tag{13}$$

where $\tilde{\mathbf{x}}_t$ is the predicted value at time t and $\mathbf{x}_t$ is the true one. For the G-VAR model in (3), we predicted each coordinate separately and report the average prediction error over all coordinates.

Results. Fig. 2a shows the rNMSE of the two methods for different percentages of in-sample data when considering the prediction of an entire multi-dimensional graph signal in the test set. Despite the considered scenario has a low number of training data and does not show high correlation between the different features, our method yields at least a comparable accuracy w.r.t. (3). However, when more training samples are available, the PG-VAR yields an improved performance. In addition, Fig. 2b depicts the true mesh position and the two estimates at a random (test) time instant, where we can observe how the proposed model matches well the real dog position.

5. CONCLUSION

We proposed a product graph-based model to forecast future values of multi-dimensional graph processes, i.e., processes which at each node have multiple time-varying features. The proposed approach builds first a feature graph between the different node features and exploits then the underlying structure between nodes to link the different features on the different nodes with product graphs. Subsequently, we incorporated into the VAR models the product graph structure to forecast the multi-dimensional process with a number of parameters that is independent of the dimensions of the graphs and therefore has a low implementation complexity. Further, we also devised a general forecasting with product graphs, which learns directly from the data also the adequate type of product graph for the task at hand. As future works, we will corroborate our methods also on larger data sets, execute numerical tests on the general PG-VAR model and work on its possible extension(s).

6. REFERENCES

[1] I. Ibáñez, J.A. Silander, J.M. Allen, S.A. Treanor, A. Wilson, "Identifying hotspots for plant invasions and forecasting focal points of further spread," *J. Appl. Ecol.*, vol. 46, pp. 1219-1228, Dec. 2009.

[2] P. A. Klein and M. Niemira, "Forecasting financial and economic cycles", *Finance editions. New York: Wiley*, 1994.

[3] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control.* John Wiley & Sons, 2015.

[4] H. Lütkepohl, *New introduction to multiple time series analysis.* Springer Science & Business Media, 2005.

[5] J. Mei and J. M. Moura, "Signal processing on graphs: Causal modeling of unstructured data," *IEEE Transactions on Signal Processing*, vol. 65, no. 8, pp. 2077–2092, 2017.

[6] E. Isufi, Andreas Loukas, N. Perraudin, G. Leus, "Forecasting Time Series with VARMA Recursions on Graphs", to appear in *IEEE Transactions on Signal Processing*, 2019.

[7] R. Hammack, W. Imrich, and S. Klavzar, 'Handbook of product graphs," *Second Edition, CRC Press, Boca Raton, FL*,2011.

[8] M. S. Kotzagiannidis, P. L. Dragotti, "Sampling and Reconstruction of Sparse Signals on Circulant Graphs - An Introduction to Graph-FRI, " *Appl. Comput. Harmon. Anal. (2017),* in press, http://dx.doi.org/10.1016/j.acha.2017.10.003, available on arXiv: arXiv:1606.08085.

[9] Guillermo Ortiz-Jiménez, Mario Coutino, Sundeep Prabhakar Chepuri, Geert Leus, "Sparse Sampling for Inverse Problems With Tensors, " in *IEEE Transactions on Signal Processing ,* Volume: 67, Jun. 15, 2019.

[10] Rohan Varma and Jelena Kovacevic, "Sampling Theory for Graph Signals on Product Graphs," *arXiv:1809.10049*, 2018.

[11] M. S. Kotzagiannidis and P. L. Dragotti, "Splines and wavelets on circulant graphs," in *Appl. Comput. Harmon. Anal.*, vol. 47, pp. 481-515, Sep. 2019.

[12] F. Grassi, A. Loukas, N. Perraudin, and B. Ricaud, "A Time-Vertex Signal Processing Framework: Scalable Processing and Meaningful Representations for Time-Series on Graphs," in *IEEE Transactions on Signal Processing* vol. 66, no. 3, pp. 817 – 829, 2018.

[13] D Romero, VN Ioannidis, GB Giannakis "Kernel-based reconstruction of space-time functions on dynamic graphs," in *IEEE Journal of Selected Topics in Signal Processing* 11 (6), 856-869.

[14] G. Connor, "The three types of factor models: A comparison of their explanatory power," *Financial Analysts Journal*, vol. 51, no. 3, pp. 42– 46, 1995.

[15] C. Lam, Q. Yao, and N. Bathia, "Factor modeling for high dimensional time series," in *in Recent Advances in Functional Data Analysis and Related Topics*. Springer, 2011, pp. 203–207.

[16] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains," *IEEE Signal Processing Magazine,*, vol. 30, no. 3, pp. 83–98, 2013.

[17] Fernando Gama, Antonio G. Marques, Alejandro Ribeiro and Geert Leus, "MIMO Graph Filters for Convolutional Neural Networks," *arXiv:1803.02247*, 2018.

[18] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs", *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[19] A. Sandryhaila J.M.F. Moura, "Big data analysis with signal processing on graphs", IEEE Signal Process. Mag. vol. 31 no. 5 pp. 80-90 2013.

[20] G. Mateos, S. Segarra, A. G. Marques and A. Ribeiro, "Connecting the Dots: Identifying Network Structure via Graph Signal Processing," in *IEEE Signal Processing Magazine,* vol. 36, no. 3, pp. 16-43, May 2019. doi: 10.1109/MSP.2018.2890143

[21] J. Gall, C. Stoll, E. De Aguiar, C. Theobalt, B. Rosenhahn, and H.- P. Seidel, "Motion capture using joint skeleton tracking and surface estimation," in *Computer Vision and Pattern Recognition*, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009, pp. 1746–1753.

[22] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*. Springer series in statistics New York, 2001, vol. 1.