Online Vector Autoregressive Models Over Expanding Graphs

Das, Bishwadeep; Isufi, Elvin

DOI
10.1109/ICASSP49357.2023.10096508

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10096508

ONLINE VECTOR AUTOREGRESSIVE MODELS OVER EXPANDING GRAPHS

Bishwadeep Das and Elvin Isufi

ABSTRACT

Current spatiotemporal learning methods for complex data exploit the graph structure as an inductive bias to restrict the function space and improve data and computation efficiency. However, these methods work principally on graphs with a fixed size, whereas in several applications there are expanding graphs where new nodes join the network; e.g., new sensors joining a sensor network or new users joining a recommender system. This paper focuses on the non-trivial extension of spatiotemporal methods to this setting, where now it is key to jointly capture both the topological and signal dynamics. Specifically, it considers a graph vector autoregressive (GVAR) model for multivariate time series. The GVAR is a multivariate linear model that leverages a bank of graph filters allowing scalability and data efficiency. To account for the dynamic nature of the graphs, the filters's parameters are learned on-the-fly via adaptive gradient descent with provable sub-linear regret. Numerical results on both synthetic and real data corroborate the proposed method.

1. INTRODUCTION

Spatiotemporal signal modeling is useful for processing real world time-varying signals like temperature or pressure over sensor networks but conventional methods have a high number of parameters and fail to capture the network structure [1]. Alternatively, graph-based spatiotemporal models jointly capture the graph and the temporal dependencies in the data and utilize fewer parameters [2]. This is achieved by using the graph as an inductive bias for the spatial network structure, thereby reducing the function space and tackling the curse of dimensionality.

Graph-based spatiotemporal models can be divided into three categories: *ARMA models*: These methods model the signal evolution as a combination of filtered past temporal signals [2]. They consider the graph topology at each time instant for a variety of applications [3–10]; *State space models*. These methods use a state transition and an obervation step to model spatiotemporal signals [11, 12]. The graph is used to model the state update equation or to build kernels as a generative model [13–16]. *Graph Construction methods*: These methods construct a new graph to capture the spatiotemporal dynamics. The works in [17–19] use product graphs to capture this relationship while some works like [20] build domain-specific graphs to represent the temporal dynamics.

All of the above approaches consier spatiotemporal models for graphs with a fixed number of nodes. However, in practice we encounter situations where the topologies expand; e.g., new time series become available or a new sensor is added to a sensor networks [21–23]. Extending these methods to a growing graph setting is non-trivial, incurring an ever-growing increase in complexity. For example, in the kernel-based methods [8, 9, 13–15], we need to recalculate the pair-wise node similarity kernels for each new node, or for the whole graph, which requires an eigendecomposition of the Laplacian. Another challenge is that the incoming nodes and their data typically arise in an online manner, making it more challenging to deploy the above batch-based approaches.

In this paper, we target the above challenge and provide a graph-based model for online learning over continuously expanding graphs. Our specific contribution is two-fold:

1. We propose expanding graph vector autoregressive (GEVAR) model for time series over expanding graphs. Such models capture multiresolution information at all nodes including the incoming ones; is linear, localized, enjoys a distributed implementation, and does not need a prior like kernels, while being adaptive to the expanding topology.
2. We perform online forecasting on the expanding graph using the GEVAR filter bank. As an alternative to batch processing, we update the filter bank online using tools from online machine learning, which has already been used for graph signal processing tasks in time invariant setting [9, 24, 25]. Specifically, we use adaptive online gradient descent, with a provable sub-linear regret upper bound w.r.t the batch solution [26].

2. GRAPH VAR MODELS OVER EXPANDING GRAPHS

In this section, we first highlight the GVAR model on a graph of fixed size and then describe its extension to an expanding graph setting.

GVAR model. Consider a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ of N nodes with shift operator $\mathbf{S} \in \mathbb{R}^{N\times N}$ and a time-varying signal $\mathbf{x}_t \in \mathbb{R}^N$. A Graph Vector Autoregressive (GVAR) model of order P [2] models the signal at time t in terms of its previous P realizations as

$$\mathbf{x}_t = -\sum_{p=1}^{P} \mathbf{H}_p(\mathbf{S})\mathbf{x}_{t-p} + \boldsymbol{\epsilon}_t \tag{1}$$

where $\mathbf{x}_{t-p}$ is the signal before p time instants and $\mathbf{H}_p(\mathbf{S}) = \sum_{k=0}^{K} h_{pk}\mathbf{S}^k$ is a graph convolutional filter of order K which filters $\mathbf{x}_{t-p}$ over the static topology represented by $\mathbf{S}$ [27]. The GVAR model combines the output of P time independent graph filters $\mathbf{H}_p(\mathbf{S})$, each acting on its corresponding past time signal $\mathbf{x}_{t-p}$ to model the signal at time t. The vector $\boldsymbol{\epsilon}_t$ is the model randomness. In each time instant, GVAR leverages multi-hop information up to K hops away. Due to the prior imposed by the graph structure, GVAR has $(K+1)P$ parameters and a complexity of order $\mathcal{O}(PK|\mathcal{E}|)$ [27].

GVAR on expanding graphs (GEVAR). The GVAR model presented above holds for a graph of fixed size. To extend it to an expanding graph, we consider a starting graph $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ of N_0 nodes, comprising a node set $\mathcal{V}_0 = \{v_1, \ldots, v_{N_0}\}$ and an edge set

The authors are with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, The Netherlands. e-mails: {b.das, e.isufi-1}@tudelft.nl

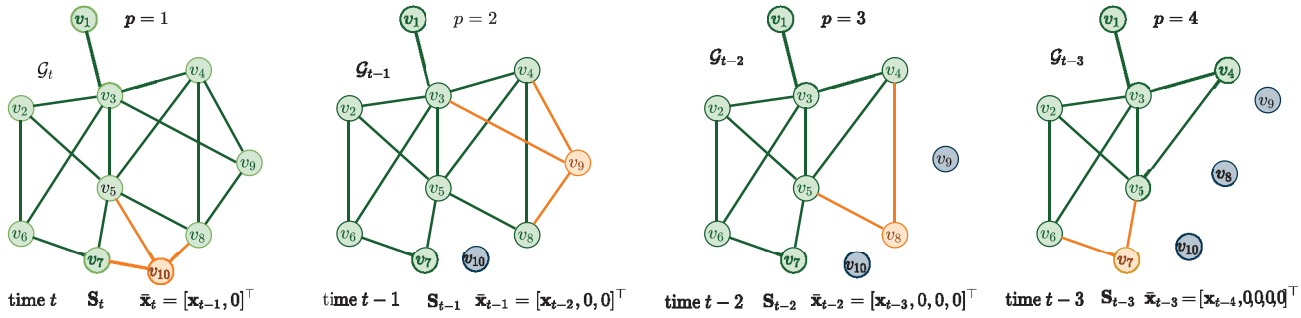

Fig. 1. A GEVAR process with $P = 4$. (Left) At time t, v_{10} (orange) joins $\mathcal{G}_{t-1}$ (green) to form $\mathcal{G}_t$ with zero-padded shift operator and signal $\mathbf{S}_t$ and $\bar{\mathbf{x}}_t$ for $p = 1$; (center left) At time $t-1$, v_9 is the incoming node and v_{10} (grey) is considered as a ghost node, i.e., we use it virtually for zero-padding and $p = 2$; (center right) v_8 joins with v_9, v_{10} as ghost nodes and $p = 3$; (right) v_7 joins with v_8,v_9, v_{10} and $p = 4$. The GVAR model predicts in this case $\mathbf{x}_t = -(\mathbf{H}_1(\mathbf{S}_t)\bar{\mathbf{x}}_t + \mathbf{H}_2(\mathbf{S}_{t-1})\bar{\mathbf{x}}_{t-1} + \mathbf{H}_3(\mathbf{S}_{t-2})\bar{\mathbf{x}}_{t-2} + \mathbf{H}_4(\mathbf{S}_{t-3})\bar{\mathbf{x}}_{t-3}) + \boldsymbol{\epsilon}_t$

$\mathcal{E}_0$. Let $\mathbf{A}_0 \in \mathbb{R}^{N_0 \times N_0}$ be its adjacency matrix with $[\mathbf{A}_0]_{ij} > 0$ if $\{v_i, v_j\} \in \mathcal{E}_0$ and $\{v_{N_0+1}, \ldots, v_{N_o+T}\}$ a sequence of T incoming nodes over time $t = 1, \ldots, T$. Node v_{N_0+t} arrives at time t and connects to the already existing graph $\mathcal{G}_{t-1}$ forming graph $\mathcal{G}_t = \{\mathcal{V}_t, \mathcal{E}_t\}$ with $N_t = N_0 + t$ nodes and the adjacency matrix $\mathbf{A}_t \in \mathbb{R}^{N_t \times N_t}$. The attachment of v_t is characterized by the vector $\mathbf{a}_t = [[\mathbf{a}_t]_1, \ldots, [\mathbf{a}_t]_{N_{t-1}}]^\top$, where $[\mathbf{a}_t]_i > 0$ indicates an undirected edge between v_t and $v_i \in \mathcal{V}_{t-1}$. The adjacency matrix $\mathbf{A}_t \in \mathbb{R}^{N_t \times N_t}$ reads relative to $\mathbf{A}_{t-1}$ as

$$\mathbf{A}_t = \begin{bmatrix} \mathbf{A}_{t-1} & \mathbf{a}_t \\ \mathbf{a}_t^\top & 0 \end{bmatrix}. \tag{2}$$

Let $\mathbf{x}_0 \in \mathbb{R}^{N_0}$ be the signal over $\mathcal{G}_0$ and $\mathbf{x}_t \in \mathbb{R}^{N_t}$ that over $\mathcal{G}_t$ at time t. Extending the GVAR model to an expanding graph requires handling an adjacency matrix and graph signal of increasing dimensions. To account for this challenge, we define the time varying zero-padded shift operator $\mathbf{S}_{t-p+1}$ and signal $\bar{\mathbf{x}}_{t-p+1}$ as

$$\mathbf{S}_{t-p+1} = \begin{bmatrix} \mathbf{A}_{t-p+1} & \mathbf{0}_{(N_{t-p+1}),(p-1)} \\ \mathbf{0}_{(p-1),(N_{t-p+1})} & \mathbf{0}_{(p-1),(p-1)} \end{bmatrix}, \; \bar{\mathbf{x}}_{t-p+1} = \begin{bmatrix} \mathbf{x}_{t-p} \\ \mathbf{0}_p \end{bmatrix}. \tag{3}$$

Given then $\mathbf{S}_{t-p+1}$, the graph filter associated to the pth lag reads as

$$\mathbf{H}_p(\mathbf{S}_{t-p+1}) = \sum_{k=0}^{K} h_{pk} \mathbf{S}_{t-p+1}^k \tag{4}$$

parameterized by the coefficients $\mathbf{h}_p = [h_{p1}, \ldots, h_{p(K+1)}]^\top \in \mathbb{R}^{K+1}$. The scalar h_{pk} is the weight given to the kth shift of the pth lag shift operator. Substituting (3) in to (4), we see that there is a coupling between the topology $\mathbf{A}_{t-p+1}$ and the signal $\mathbf{x}_{t-p}$. The subscript indices for the two differ by one. This is because the incoming node first attaches to the existing graph, updating its topology index and then filter acts on the existing graph signal over the updated topology with a zero signal at the incoming node. Then, the graph extended VAR (GEVAR) model at time t reads as

$$\mathbf{x}_t = -\sum_{p=1}^{P} \mathbf{H}_p(\mathbf{S}_{t-p+1})\bar{\mathbf{x}}_{t-p+1} + \boldsymbol{\epsilon}_t. \tag{5}$$

That is, the graph expanded VAR (GEVAR) model in (5) combines signals up to P time lags away, each coupled with multi-hop interactions over the topology of its supporting shift operator. Figure 1 showcases the GVAR model on an expanding graph with $P = 4$. We rewrite the term $\mathbf{H}_p(\mathbf{S}_{t-p+1})\bar{\mathbf{x}}_{t-p+1}$ as

$$\mathbf{H}_p(\mathbf{S}_{t-p+1})\bar{\mathbf{x}}_{t-p+1} = \mathbf{S}_{\bar{\mathbf{x}},t-p+1}\mathbf{h}_p \tag{6}$$

where $\mathbf{S}_{\bar{\mathbf{x}},t-p+1} = [\bar{\mathbf{x}}_{t-p+1}, \mathbf{S}_{t-p+1}\bar{\mathbf{x}}_{t-p+1}, \ldots, \mathbf{S}_{t-p+1}^K\bar{\mathbf{x}}_{t-p+1}]$, has as kth column the zero-padded signal $\bar{\mathbf{x}}_{t-p+1}$ shifted $(k-1)$ times over $\mathbf{S}_{t-p+1}$. Using (6) for all p, we rewrite (5) as

$$\mathbf{x}_t = -\boldsymbol{\Sigma}_t\mathbf{h} + \boldsymbol{\epsilon}_t \text{ with } \boldsymbol{\Sigma}_t = [\mathbf{S}_{\bar{\mathbf{x}},t}, \ldots, \mathbf{S}_{\bar{\mathbf{x}},t-P+1}] \in \mathbb{R}^{N_t \times (K+1)P}. \tag{7}$$

Remark 1. *We consider the most challenging case where a new node arrives at each time t, i.e., the dimension of the existing graph signal increases along with t. However, a new node can also arrive every h instants. Between t and t + h, the signal can evolve over the fixed graph expanded up to that time instant..*

3. ONLINE SPATIOTEMPORAL FORECASTING OVER EXPANDING GRAPHS

A spatiotemporal forecasting task estimates $\mathbf{x}_{t+1}$ given previous observations $\{\mathbf{x}_{t+1-p}\}$ over the expanding graph sequence $\{\mathcal{G}_{t+1-p}\}$ with $p = 1, \ldots, P$. Consider the training sequence $\mathcal{T}$ comprising T incoming nodes $\{v_{N_0+1}, \ldots, v_{N_0+T}\}$ over graphs $\{\mathcal{G}_1, \ldots, \mathcal{G}_T\}$ with graph signals $\{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$. We obtain the parameters of a GEVAR model $\mathbf{h}$ over $\mathcal{T}$ by minimizing the loss

$$\underset{\mathbf{h} \in \mathbb{R}^{(K+1)P}}{\operatorname{argmin}} \sum_{t=P+1}^{T} \frac{1}{2}||\boldsymbol{\Sigma}_t\mathbf{h} - \mathbf{x}_t||_2^2 + \mu||\mathbf{h}||_2^2 \tag{8}$$

where $(||\boldsymbol{\Sigma}_t\mathbf{h} - \mathbf{x}_t||_2^2$ measures the fitting error, $||\mathbf{h}||_2^2$ imposes an l_2 norm prior on $\mathbf{h}$, and $\mu > 0$. However, obtaining such a batch solution has a few drawbacks: i) it assumes the data is available all-at-once; ii) incorporating each incoming node can be computationally demanding as we need to solve (8) for each t; iii) the performance on a test set will degrade if its data distribution differs from that in the training set. As the incoming nodes typically arrive sequentially, next we discuss how we estimate parameters $\mathbf{h}_t$ in (7) online.

Online learning. Alternatively, we consider updating the GEVAR parameters online, i.e., based off the loss incurred on the latest incoming node. This adapts to the data sequence and the updates are also computationally tractable. Following the online machine learning principles [26], the following happens at time t: i) node v_{N_0+t}

attaches to $\mathcal{G}_{t-1}$ via $\mathbf{a}_t$ forming $\mathcal{G}_t$; $ii)$ we construct $\mathbf{\Sigma}_t$ [cf. (3), (7)] and denote the current parameters by $\mathbf{h}_{t-1}$, i.e., the parameters updated at the previous instant $t-1$ considering only past information. We then predict the signal as $\hat{\mathbf{x}}_t = \mathbf{\Sigma}_t \mathbf{h}_{t-1}$; $iii)$ the true signal $\mathbf{x}_t$ is revealed and we incur the loss which we use to update the parameters. We consider the instantaneous loss

$$l_t(\mathbf{h}, \mathbf{x}_t) = \frac{1}{2}||\mathbf{\Sigma}_t \mathbf{h} - \mathbf{x}_t||_2^2 + \mu||\mathbf{h}||_2^2 \tag{9}$$

Then, we update the parameters via a projected Adaptive Online Gradient Descent (AdOGD), where

$$\mathbf{h}_t = \prod_{\mathcal{H}} \big(\mathbf{h}_{t-1} - \eta_t \nabla l_t(\mathbf{h}_{t-1}, \mathbf{x}_t)\big) \tag{10}$$

where $\nabla l_t(\mathbf{h}_{t-1}, \mathbf{x}_t)$ denotes the gradient evaluated at $\mathbf{h}_{t-1}$, η_t the learning rate at time t, and $\prod_{\mathcal{H}}(\cdot)$ the Euclidean projection onto the set $\mathcal{H}$, respectively. The gradient is

$$\nabla l_t(\mathbf{h}_{t-1}, \mathbf{x}_t) = (\mathbf{\Sigma}_t^\top \mathbf{\Sigma}_t + \mu \mathbf{I})\mathbf{h}_{t-1} - \mathbf{\Sigma}_t^\top \mathbf{x}_t. \tag{11}$$

The adaptive nature of the online learning stems from the step-size

$$\eta_t = \frac{C}{\sqrt{\sum_{\tau=1}^{t} ||\nabla l_\tau(\mathbf{h}_{\tau-1})||_2^2}} \tag{12}$$

where $C > 0$ is a scalar. The step-size reduces with time ($\eta_{t+1} \leq \eta_t$) for all t, i.e., the importance given to a node further in the future is lower. If we have a scenario where the error grows element-wise over time as a result of the growing dimension, the learning rate in (12) can counter its effect, preventing the filter from blowing up numerically. The loss function in (9) is both strongly convex and differentiable in $\mathbf{h}$ for all t.

At time t, the complexity is dictated by the term $\mathbf{\Sigma}_t^\top \mathbf{\Sigma}_t \mathbf{h}_{t-1}$ comprising: the product $\mathbf{z}_t = \mathbf{\Sigma}_t \mathbf{h}_{t-1}$ of complexity $\mathcal{O}(N_t(K+1)P)$ and the product $\mathbf{\Sigma}_t \mathbf{z}_t$ of complexity $\mathcal{O}(N_t(K+1)P)$. This gives a total complexity of $\mathcal{O}(N_t(K+1)P)$ at time t.

Regret analysis. The static regret of an online learner w.r.t. any fixed filter bank $\mathbf{h}^\star \in \mathcal{H}$ over a T-length sequence is

$$R_T = \sum_{t=1}^{T} l_t(\mathbf{h}_{t-1}, \mathbf{x}_t) - \sum_{t=1}^{T} l_t(\mathbf{h}^\star, \mathbf{x}_t) \tag{13}$$

For a given sequence $\{\mathbf{h}_t\}$ predicted by the online learning algorithm and a fixed $\mathbf{h}^\star$, the static regret informs how much better (or worse) the learner is w.r.t the cumulative loss. For a fixed sequence $\{\mathbf{h}_t\}$, this is a function of $\mathbf{h}^\star$, typically maximum when $\mathbf{h}^\star = \underset{\mathbf{h}^\star \in \mathcal{H}}{\text{argmin}} \sum_{t=P+1}^{T} l_t(\mathbf{h}^\star, \mathbf{x}_t)$, i.e., $\mathbf{h}^\star$ is the batch solution. Since the sequence $\{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$ depends on the initialization and the online learning algorithm, often an upper bound on $R_T(\mathbf{h}^\star)$ is useful, providing a worst case performance analysis [26]. To quantify the regret of the algorithm, we need the following standard assumptions for online learning [24]:

Assumption 1. *For any t, the maximum eigenvalue of $\mathbf{\Sigma}_t^\top \mathbf{\Sigma}_t$ obeys*

$$\lambda_{max}(\mathbf{\Sigma}_t^\top \mathbf{\Sigma}_t) < 1 - \mu, \;\; \textit{where } 0 < \mu < 1.$$

and μ scales the regularization term in the loss function in (9)*.*

Assumption 2. *The filter bank parameters $\mathbf{h}_t$ lies in the bounded convex set $\mathcal{H} = \{\mathbf{h}_t : ||\mathbf{h}_t||_2 \leq H \text{ for all } t\}$*

Assumption 1 prevents the gradient from exploding, as can be seen by recursively writing equation (11) in terms of the previous filter updates. Since $\mathbf{\Sigma}_t^\top \mathbf{\Sigma}_t$ is positive semi-definite, $0 < (1-\mu)$ and $\mu > 0$ for convexity of the loss gives $0 < \mu < 1$. Assumptions 1 and 2 bound the gradient, i.e., $||\nabla(l_t(\mathbf{h}_{t-1}, \mathbf{x}_t))|| \leq L$, where L is a constant, making the loss Lipschitz. Assumption 2 further guarantees the loss function to be strongly convex and Lipschitz by considering a bounded domain. As a result, for parameters $\mathbf{h}_i$ and $\mathbf{h}_j$, $||\mathbf{h}_i - \mathbf{h}_j||_2 \leq 2H$. With this in place, the following holds true.

Proposition 1. *Given a training set $\mathcal{T}$ of T nodes, a sequence of online filters $\{\mathbf{h}_t\}$ updated over a sequence of L Lipschitz functions, the scalar C associated with the learning rate $[cf.(12)]$, set $\mathcal{H}$ with diameter $2H$ and Assumptions 1 and 2, the static regret for the AdOGD $[cf.(10)]$ for forecasting w.r.t any filterbank $\mathbf{h}^\star \in \mathcal{H}$ is upper bounded as*

$$R_T(\mathbf{h}^\star) \leq (\frac{2H^2}{C} + C)L\sqrt{T} \tag{14}$$

Proof. The proof follows a very similar approach to that shown in Section 4.2.1 in [26]. □

From (14) $\lim_{T\to\infty} \frac{R_T(\mathbf{h}^\star)}{T} = 0$, i.e., even in the worst case scenario, the online learner approaches asympotically the average cumulative loss w.r.t. any optimal $\mathbf{h}^\star$ over $\mathcal{T}$. Note however that the regret can be negative, i.e., the online method is better than $\mathbf{h}^\star$.

4. NUMERICAL RESULTS

We evaluate the online methods via experiments on two real datasets with a focus on answering the following research questions.

1. RQ1: *How does the proposed online method compare to the corresponding batch solution?* The batch solution acts as a baseline and has access to the whole sequence

$$\textbf{BS} = \underset{\mathbf{h}^\star \in \mathcal{H}}{\text{argmin}} \sum_{t=1}^{T} l_t(\mathbf{h}^\star, \mathbf{x}_t) + \gamma||\mathbf{h}^\star||_2^2 \tag{15}$$

2. RQ2: *How does the proposed method compare to a time agnostic online filter?* This, is the proposed method itself for $p = 1$, i..e, where we do not utilize only the latest observation $\bar{\mathbf{x}}_t$ over $\mathbf{S}_t$. Answering this question we aim to show the advantages of considering data from previous time lags.
3. RQ3: *How does the spatiotemporal frequency response vary over the different time lags?* We want to interpret the online learner form a spectral perspective and see which lags are contributing more to the low or high frequency components of the predictions.

For each dataset, we use independent time chunks for pre-processing, training, validation and testing. Since we have an incoming node at each time instant, the number of samples in each set is limited by the total number of nodes. To have more data samples, we construct multiple datasets like $\mathcal{T}$ [cf. Section 3], each with the same $\mathcal{V}_0$ and $\mathbf{A}_0$, but a different $\mathbf{x}_0$ and starting time, i.e., we create several replicas of each incoming node but with different signals. We combine the samples from all of these smaller datasets into a large dataset for training, validation and testing. The attachment pattern of each incoming node is fixed for each dataset, which makes the learning more robust w.r.t. starting signals.

Unless otherwise mentioned, we consider each incoming node forms three nearest neighbors based on the geographical positions of

Table 1. NOAA

Filter Order	Online(P=1)	Online(P=2)	Online(P=3)	Batch(P=1)	Batch(P=2)	Batch(P=3)
K=1	0.5	0.28	0.17	0.97	0.97	1=
K=2	0.4	0.21	0.18	0.99	1.01	1.02
K=3	0.39	0.21	0.24	0.95	0.98	0.98

Table 2. SeaSurfaceTemp

Filter Order	Online(P=1)	Online(P=2)	Online(P=3)	Batch(P=1)	Batch(P=2)	Batch(P=3)
K=1	0.6	0.58	0.65	0.38	0.24	0.21
K=2	0.6	0.59	0.7	0.45	0.31	0.28
K=3	0.66	0.64	0.74	0.53	0.41	0.36

the incoming and existing nodes. We normalize each adjacency matrix by its maximum absolute eigenvalue so its spectrum lies within $[-1, 1]$ as in [27] and each $\mathbf{\Sigma}_t$ by its maximum eigenvalue. For online training, we consider random initializations and take $H = 100$. We select the hyperparameters C and μ by validation from the interval $[10^{-3}, 1]$. The batch hyperparameter γ in (15) is similarly obtained from $[10^{-2}, 100]$.

Data and experimental setup. We perform two types of temperature forecasting: (i) hourly forecasting using the NOAA data-set comprising hourly temperature recordings over 109 stations across in the U.S [28]; (ii) monthly forecasting using sea surface temperature measured at 100 measuring stations over the Pacific over a 1700 month period [1], following [29]. For the NOAA data, we use the first 2000 temporal samples to normalize the data. For training, validation and testing, we use the time samples in the intervals $[2001, 6000]$, $[6001, 7000]$, and $[7001, 8000]$, respectively. We start with $N_0 = 10$ nodes. For the Pacific data, we normalize the data using the first 200 time samples. For training, validation and testing, we use the time samples in the intervals $[201, 1300]$, $[1301, 1500]$, and $[1501, 1700]$, respectively. We start with $N_0 = 20$ nodes. We train over different combinations of P and K and also obtain the corresponding batch solution.

Results. Tables 1 and 2 show the root normalized mean square error (rNMSE) along with the standard deviation for the online and corresponding batch method for different combinations of P and K. The standard deviation in both cases is of order 10^{-2}. Regarding **RQ1**, the batch solution performs very poorly in the NOAA data, highlighting a limitation of the batch solution when the test set has a different distribution. However, the batch method performs much better than the online method for sea temperature, suggesting a test distribution more in line with the training, maybe due to the monthly sampled data. For both the online and batch methods, an increase in the filter order across all lags results in a lower rNMSE, suggesting a simpler nature of the observed data. Regarding **RQ2**, for the NOAA data, the online method performs much better with higher values of P. For each P, the performance improves by considering higher K, with the exception of $(P = 3, K = 3)$, where we might be suffering from overparameterization. For the sea data, the online method an increase in the number of lags improves the performance only marginally, with the rNMSE increasing again for $P = 3$. To answer **RQ3**, Figure 2 shows the polynomial frequency response of the filters for each p for $K = 3$ following [27]. The x-axis corresponds to the analytical frequency over $[-1, 1]$, where frequencies approaching -1 and 1 are the high and low frequencies, respectively. [27]. For NOAA data, the response for lags $p = 1$ and $p = 2$ are similar, both having a predominantly low-pass response. For $p = 3$, there is a slightly more high-pass nature, i.e., it extracts more high frequency data from graph signals located 3 lags ago. Due to the slowly varying nature of the hourly temperature data, we can expect such a frequency response profile. For sea temperature, we see that the response for $p = 1$ attenuates more frequencies around $\lambda = 0$. For lags $p = 2, 3$ we see a comparatively flat frequency response, with a more low frequency tolerance. This might suggest that for higher lags, the filter is not contributing a lot and might be redundant, as is seen in the rNMSE performance.

[1] https://psl.noaa.gov/

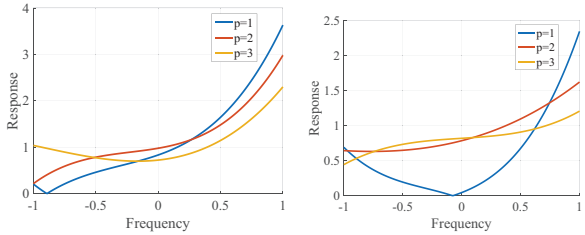

Fig. 2. Frequency response at each lag $p = \{1, 2, 3\}$ obtained at the end of the training sequence for (left) NOAA; (right) Sea Temperature Data. Each filter is of order three.

5. CONCLUSION

We propose graph vector autoregressive models for spatiotemporal learning over expanding graphs, i.e., at each time instant, we have a node attaching. We first propose the GEVAR model to accomodate for such signal variations and focus on spatiotemporal forecasting of graph signals on the expanding graphs. Due to the sequential nature of the node addition, we then consider an adaptive projected online gradient descent with provable sub-linear regret bound to update the parameters. Results on two real world data-sets shows that the online method would perform better when the test data has a different distribution to the training data; otherwise batch solutions would perform better. The frequency response learnt by the filter banks online can also throw some light on how the model learns from past time samples. Future work would focus more on understanding the dynamics in the joint frequency response. The spectrum of the growing graph also changes over time and its interaction with the evolving frequency response should be investigated.

6. REFERENCES

[1] H. Lütkepohl, "Vector autoregressive models," in *Handbook of Research Methods and Applications in Empirical Macroeconomics*. Edward Elgar Publishing, 2013, pp. 139–164.

[2] E. Isufi, A. Loukas, N. Perraudin, and G. Leus, "Forecasting time series with varma recursions on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 18, pp. 4870–4885, 2019.

[3] E. T. Güneyi, A. Canbolat, and E. Vural, "Learning parametric time-vertex graph processes from incomplete realizations," in *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2021, pp. 1–6.

[4] T. Variddhisai and D. Mandic, "Methods of adaptive signal processing on graphs using vertex-time autoregressive models," *arXiv preprint arXiv:2003.05729*, 2020.

[5] A. Natali, E. Isufi, and G. Leus, "Forecasting multidimensional processes over graphs," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 5575–5579.

[6] R. Nassif, C. Richard, J. Chen, and A. H. Sayed, "Distributed diffusion adaptation over graph signals," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4129–4133.

[7] B. Zaman, L. M. L. Ramos, D. Romero, and B. Beferull-Lozano, "Online topology identification from vector autoregressive time series," *IEEE Transactions on Signal Processing*, vol. 69, pp. 210–225, 2020.

[8] Y. Shen, G. B. Giannakis, and B. Baingana, "Nonlinear structural vector autoregressive models with application to directed brain networks," *IEEE Transactions on Signal Processing*, vol. 67, no. 20, pp. 5325–5339, 2019.

[9] V. R. Elias, V. C. Gogineni, W. A. Martins, and S. Werner, "Adaptive graph filters in reproducing kernel hilbert spaces: Design and performance analysis," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, pp. 62–74, 2020.

[10] R. Money, J. Krishnan, and B. Beferull-Lozano, "Online nonlinear topology identification from graph-connected time series," in *2021 IEEE Data Science and Learning Workshop (DSLW)*. IEEE, 2021, pp. 1–6.

[11] O. Teke and P. P. Vaidyanathan, "Joint vertex-time filtering on graphs with random node-asynchronous updates," *IEEE Access*, vol. 9, pp. 122 801–122 818, 2021.

[12] M. Coutino, E. Isufi, T. Maehara, and G. Leus, "State-space network topology identification from partial observations," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 211–225, 2020.

[13] Q. Lu, V. N. Ioannidis, G. B. Giannakis, and M. Coutino, "Learning graph processes with multiple dynamical models," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2019, pp. 1783–1787.

[14] Q. Lu, V. N. Ioannidis, and G. B. Giannakis, "Graph-adaptive semi-supervised tracking of dynamic processes over switching network modes," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2586–2597, 2020.

[15] Q. Lu and G. B. Giannakis, "Probabilistic reconstruction of spatio-temporal processes over multi-relational graphs," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 7, 2021.

[16] V. N. Ioannidis, D. Romero, and G. B. Giannakis, "Inference of spatio-temporal functions over graphs via multikernel kriged kalman filtering," *IEEE Transactions on Signal Processing*, vol. 66, no. 12, pp. 3228–3239, 2018.

[17] D. Romero, V. N. Ioannidis, and G. B. Giannakis, "Kernel-based reconstruction of space-time functions on dynamic graphs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 856–869, 2017.

[18] 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," *IEEE Transactions on Signal Processing*, vol. 66, no. 3, pp. 817–829, 2017.

[19] M. Sabbaqi and E. Isufi, "Graph-time convolutional neural networks: Architecture and theoretical analysis," *arXiv preprint arXiv:2206.15174*, 2022.

[20] Y. Hu and F. Xiao, "An efficient forecasting method for time series based on visibility graph and multi-subgraph similarity," *Chaos, Solitons & Fractals*, vol. 160, p. 112243, 2022.

[21] P. Erdos, "On the evolution of random graphs," *Bulletin of the Institute of International Statistics*, vol. 38, pp. 343–347, 1961. [Online]. Available: https://ci.nii.ac.jp/naid/10025454140/

[22] A.-L. Barabási and R. Albert, "Emergence of Scaling in Random Networks," *Science*, vol. 286, no. 5439, Oct. 1999, publisher: American Association for the Advancement of Science.

[23] A.-L. Barabási *et al.*, *Network science*. Cambridge university press, 2016.

[24] Y. Shen, G. Leus, and G. B. Giannakis, "Online Graph-Adaptive Learning With Scalability and Privacy," *IEEE Transactions on Signal Processing*, vol. 67, no. 9, pp. 2471–2483, May 2019.

[25] B. Das and E. Isufi, "Online filtering over expanding graphs," in *IEEE Asilomar Conference on Signals, Systems and Computations, Pacific Grove, USA*, 2022.

[26] F. Orabona, "A modern introduction to online learning," *arXiv preprint arXiv:1912.13213*, 2019.

[27] A. Sandryhaila and J. M. F. Moura, "Discrete Signal Processing on Graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

[28] A. Arguez, I. Durre, S. Applequist, R. S. Vose, M. F. Squires, X. Yin, R. R. Heim Jr, and T. W. Owen, "Noaa's 1981–2010 us climate normals: an overview," *Bulletin of the American Meteorological Society*, vol. 93, no. 11, pp. 1687–1697, 2012.

[29] J. H. Giraldo, A. Mahmood, B. Garcia-Garcia, D. Thanou, and T. Bouwmans, "Reconstruction of time-varying graph signals via sobolev smoothness," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 8, pp. 201–214, 2022.