Online Graph Learning In Dynamic Environments

Xiang Zhang
School of Information Science and Engineering
Southeast University
Nanjing, China
xiangzhang369@seu.edu.cn

Qiao Wang
School of Information Science and Engineering
Southeast University
Nanjing, China
qiaowang@seu.edu.cn

***Abstract*—Inferring the graph topology that characterizes structured data is pivotal to many graph-based models when predefined graphs are not available. This paper focuses on learning graphs in the case of sequential data in dynamic environments. For sequential data, an online version of classic batch graph learning method is developed. To better track dynamic graphs, we assume the graphs evolve in some certain patterns such that dynamic priors might be embedded in the online graph learning framework. When we have no knowledge of the hidden patterns, a data-driven method is leveraged to predict the evolution of graphs. Furthermore, dynamic regret analysis of the proposed method is performed, illustrating theoretically that proper dynamic priors do reduce regret. Experimental results support that our method is superior to the state-of-the-art methods.**

***Index Terms*—Graph learning, online learning, dynamic environments, regret analysis**

I. Introduction

The goal of graph learning task is to infer the hidden topology inside high dimensional data [1]–[4], from which a lot of graph based tasks such as spectral clustering [5] could be performed with a structured framework. Classic graph learning contain statistical models [6] and causality based models [7]. With the rise of graph signal processing (GSP) [8], there occur vibrant methods of learning graphs from perspective of signal processing. Many GSP based models are based on smoothness assumption [3], which is also taken into account in this paper.

Most of the aforementioned models are of batch manner, i.e., all data are collected in advance and used totally at once to learn an optimal graph. However, this scheme does not meet many practical applications. Firstly, it is not reasonable to learn a single graph for all data since graphs may not be static due to dynamic environments. Furthermore, the data are often available in a sequential way. Thus, we need to update an initial inferred graph based on partial observations, and then progressively modify it as long as new data arrive. One example is to infer connections between users through shopping behavior. The data are generated only after each user purchases, and shopping websites hope to update the graphs of connections of users after each purchase happens.

To this end, we need to learn time-varying graphs in an online fashion. While time-varying graph learning has been studied in [9], [10], these works focus on batch method. Some other excellent explorations are made on online learning graphs, e.g., causality based models [11]–[13] and GSP based models [14], [15]. However, to our understanding, most of them ignore the hidden evolutionary patterns of time-varying graphs, which might be useful for improving learning performance. A similar work to ours is [16], but differs in that our model provides a more general approach to exploit prior evolutionary patterns.

This paper endeavors to develop a novel online graph learning framework that utilizes dynamic priors to better track time-varying graphs. We first cast batch model based on the smoothness assumption into an online version via online mirror descent (OMD) framework [17] to handle sequential data. Faced with dynamic environments, we assume that graphs evolve in some prior dynamic patterns. We integrate the dynamic priors into the online graph learning framework as a prediction step to output the graphs of the next time slot. When no explicit patterns are available, a data-driven method is leveraged to predict the evolution of graphs. We also analyze the dynamic regret, an important metric to evaluate online algorithms, of our method to theoretically illustrate the power of dynamic priors. Numerical experiments on synthetic and real-world data show the superiority of our framework.

II. Problem Formulation

A. Batch Graph Learning

Batch graph learning methods first collect T observed signals $\mathbf{x}_1, \ldots, \mathbf{x}_T \in \mathbb{R}^d$ generated from undirected graph $\mathcal{G}$ with nonnegative edges. One infers the topology of $\mathcal{G}$, e.g., adjacency matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$, using all collected data globally at once. For smoothness based model, it is equivalent to solving the following problem [2]

$$\min_{\mathbf{W}\in\mathcal{W}} \|\mathbf{W}\circ\mathbf{Z}\|_1 - \alpha\mathbf{1}^\top\log(\mathbf{W}\mathbf{1}) + \beta\|\mathbf{W}\|_{\mathrm{F}}^2, \quad (1)$$

where $\circ$ is Hadamard product and $\mathbf{1} = [1, \ldots, 1]^\top \in \mathbb{R}^d$ is a column vector of ones. Moreover, α and β are predefined constant parameters. Usually, the set $\mathcal{W}$ is defined as below,

$$\mathcal{W} = \left\{\mathbf{W} : \mathbf{W} \in \mathbb{R}_+^{d\times d}, \mathbf{W} = \mathbf{W}^\top, \mathrm{diag}(\mathbf{W}) = \mathbf{0}\right\}, \quad (2)$$

where $\mathbb{R}_+$ is the set of nonnegative real numbers and $\mathbf{0} \in \mathbb{R}^d$ is a vector with all entries equal to zero. Pairwise distance matrix $\mathbf{Z} \in \mathbb{R}^{d\times d}$ in (1) is calculated using data matrix $\mathbf{X} \in \mathbb{R}^{d\times T} = [\mathbf{x}_1, \ldots, \mathbf{x}_T] = [\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_d]^\top$,

$$\mathbf{Z}_{[ij]} = \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\|_2^2, \quad (3)$$

where $\mathbf{Z}_{[ij]}$ is the (i, j) entry of $\mathbf{Z}$. The first term of (1) is used to measure the smoothness of the observed signals. Besides, the second and third term control the degrees of each node

and sparsity of edges [2]. Observing (2), we can find that the number of free variables of $\mathbf{W}$ is $p = \frac{d(d-1)}{2}$. We thus define a vector $\mathbf{w} \in \mathbb{R}^p$ whose entries are the upper right variables of $\mathbf{W}$. Problem (1) can then be rewritten as [2]

$$\min_{\mathbf{w}\in\mathcal{W}_v} f(\mathbf{w}) = \min_{\mathbf{w}\in\mathcal{W}_v} 2\mathbf{z}^\top\mathbf{w} - \alpha\mathbf{1}^\top\log(\mathbf{S}\mathbf{w}) + 2\beta\|\mathbf{w}\|_2^2, \quad (4)$$

where $\mathbf{S}$ is a linear operator satisfying $\mathbf{S}\mathbf{w} = \mathbf{W}\mathbf{1}$ and $\mathbf{z}$ is the vector form of the upper triangle variables of $\mathbf{Z}$. Additionally, the set $\mathcal{W}_v$ is defined as $\mathcal{W}_v \triangleq \{\mathbf{w} : \mathbf{w} \in \mathbb{R}^p, \mathbf{w}_{[i]} \geq 0, \text{ for } i = 1, ..., p\}$.

B. Online Graph Learning Using Dynamic Priors

Different from the batch method waiting for all T data to be ready, data arrive sequentially in online setup and we are required to update a new graph after each data $\mathbf{x}_t, t = 1, ..., T$ is received. Furthermore, graphs are assumed to be time-varying and evolve with a dynamic model Φ, i.e.,

$$\mathbf{w}_{t+1} = \Phi(\mathbf{w}_t), \quad (5)$$

where $\mathbf{w}_t$ represents the graph at moment t. The model Φ describes the evolutionary patterns of dynamic graphs and may provide additional information to help track time-varying graphs. To conclude, the problem we focus on can be formally stated as follows. At each time t, given smooth signal $\mathbf{x}_t$ and historical data $\mathbf{x}_1, ..., \mathbf{x}_{t-1}$, we are required to update a new graph $\mathbf{w}_{t+1}$ immediately by leveraging information from Φ.

III. The Proposed algorithm

We leverage online mirror descent (OMD) algorithm [17] to cast the batch graph learning problem (4) into an online version. The available data at t are $\mathbf{x}_1, ...\mathbf{x}_t$ and we utilize information of these data in an recursively way. To be specific, we first calculate $\mathbf{Z}_t$ using $\mathbf{x}_t$ through (3) and reshape $\mathbf{Z}_t$ into $\mathbf{z}_t$. We then update

$$\bar{\mathbf{z}}_t = \gamma\bar{\mathbf{z}}_{t-1} + (1-\gamma)\mathbf{z}_t, \quad (6)$$

where $\gamma \in [0, 1)$ is the forgetting factor, and $\bar{\mathbf{z}}_0$ is supposed to be $\mathbf{0}$. Note that $\bar{\mathbf{z}}_t$ carries information from $\mathbf{x}_1, ..., \mathbf{x}_t$ and is treated as the input data vector at t. With $\bar{\mathbf{z}}_t$, we can define

$$f_t(\mathbf{w}; \mathbf{x}_1, ..., \mathbf{x}_t) \triangleq 2\bar{\mathbf{z}}_t^\top\mathbf{w} - \alpha\mathbf{1}^\top\log(\mathbf{S}\mathbf{w}) + \beta\|\mathbf{w}\|_2^2. \quad (7)$$

For notational simplicity, we omit $\mathbf{x}_1, ...\mathbf{x}_t$ of f_t in the remaining of the paper. Under OMD framework, we immediately update a new graph $\breve{\mathbf{w}}_t$ through

$$\breve{\mathbf{w}}_t = \operatorname*{argmin}_{\mathbf{w}\in\mathcal{W}_v} \langle\nabla f_t(\mathbf{w}_t), \mathbf{w}\rangle + \frac{1}{2\eta_t}\|\mathbf{w} - \mathbf{w}_t\|_2^2, \quad (8)$$

where $\nabla f_t(\mathbf{w}_t)$ is the subgradient of f_t at point $\mathbf{w}_t$ and η_t is the stepsize at time t. In our problem, $\nabla f_t(\mathbf{w}_t)$ can be calculated as:

$$\nabla f_t(\mathbf{w}_t) = 2\bar{\mathbf{z}}_t + 4\beta\mathbf{w}_t - \alpha\mathbf{S}^\top(\mathbf{S}\mathbf{w}_t)^{\cdot(-1)}, \quad (9)$$

and $.(-1)$ is an element-wise operator. By the optimal condition of (8), we have that

$$\breve{\mathbf{w}}_t = \prod_{\mathcal{W}_v}(\mathbf{w}_t - \eta_t\nabla f_t(\mathbf{w}_t)), \quad (10)$$

where $\prod_{\mathcal{W}_v}(\cdot)$ means mapping variables into the set $\mathcal{W}_v$.

The core problem now becomes how to integrate evolutionary patterns (5) into the above online framework. The way we leverage Φ is to perform one more update on the basis of (10). Specifically, after obtaining $\breve{\mathbf{w}}_t$, we use dynamic model Φ to update the graph of the next time period $\mathbf{w}_{t+1}$, i.e.,

$$\mathbf{w}_{t+1} = \Phi(\breve{\mathbf{w}}_t). \quad (11)$$

We can actually regard (11) as a prediction step to output the graph of $t+1$ using prior Φ. The extra prediction step brings more information beside gradient calculated from data and hence boosts learning performance. However, we should mention that an explicit prior Φ is not always available.

In scenarios where no dynamic prior is available, inspired by [18], we employ a data-driven method to describe Φ. The basic assumption is that graphs will evolve to the one satisfying the first order optimal condition of f_{t+1}. Suppose $\mathbf{w}_{t+1|t}$ is a prediction of dynamic graph at $t+1$ given information up to t, and we assume $\mathbf{w}_{t+1|t}$ satisfies [18]:

$$\nabla_{\mathbf{w}} f_{t+1}(\mathbf{w}_{t+1|t}) + \mathcal{N}_{\mathcal{W}_v}(\mathbf{w}_{t+1|t}) \ni \mathbf{0}, \quad (12)$$

where $\nabla_{\mathbf{w}} f_{t+1}(\mathbf{w}_{t+1|t})$ is the partial derivative w.r.t. $\mathbf{w}$ and $\mathcal{N}_{\mathcal{W}_v}(\cdot)$ is the normal cone operator (the subdifferential of the indicator function) defined in [18]. However, it is impossible to immediately solve (12) since we have no information of f_{t+1} at t. An alternative way is to approximate (12) via [19]

$$\begin{aligned}&\nabla_{\mathbf{w}} f_{t+1}(\mathbf{w}_{t+1|t}) + \mathcal{N}_{\mathcal{W}_v}(\mathbf{w}_{t+1|t})\\ \approx&\nabla_{\mathbf{w}} f_t(\breve{\mathbf{w}}_t) + \nabla_{\mathbf{ww}} f_t(\breve{\mathbf{w}}_t)(\mathbf{w}_{t+1|t} - \breve{\mathbf{w}}_t)\\ &+ h\nabla_{t\mathbf{w}} f_t(\breve{\mathbf{w}}_t) + \mathcal{N}_{\mathcal{W}_v}(\mathbf{w}_{t+1|t}) \ni \mathbf{0}, \end{aligned} \quad (13)$$

where $\nabla_{\mathbf{ww}} f_t$ is the Hessian matrix of f_t w.r.t. $\mathbf{w}$, $\nabla_{t\mathbf{w}} f_t$ is the partial derivative of the gradient of f_t w.r.t. t and h is the sample interval between two slots. In practice, $\nabla_{t\mathbf{w}} f_t(\breve{\mathbf{w}}_t)$ is estimated by $\left(\nabla_{\mathbf{w}} f_t(\breve{\mathbf{w}}_t) - \nabla_{\mathbf{w}} f_{t-1}(\breve{\mathbf{w}}_t)\right)/h$ [18]. Observe that (13) is equivalent to solving the following problem

$$\begin{aligned}&\mathbf{w}_{t+1|t}\\ =&\arg\min_{\mathbf{w}\in\mathcal{W}_v}\Big\{\frac{1}{2}\mathbf{w}^\top\nabla_{\mathbf{ww}} f_t(\breve{\mathbf{w}}_t)\mathbf{w} + \big(\nabla_{\mathbf{w}} f_t(\breve{\mathbf{w}}_t)\\ &+ h\nabla_{t\mathbf{w}} f_t(\breve{\mathbf{w}}_t) - \nabla_{\mathbf{ww}} f_t(\breve{\mathbf{w}}_t)\breve{\mathbf{w}}_t\big)^\top\mathbf{w}\Big\}. \end{aligned} \quad (14)$$

This equation can be regarded as a prediction of graphs in next time slot for given $\breve{\mathbf{w}}_t$. Therefore, it is able to describe Φ when no explicit pattern is available.

Instead of calculating the exact solution of (14), which may bring high computational cost, we try to find an approximate solution $\hat{\mathbf{w}}_{t+1|t}$ as in [19]. Specifically, we set a dummy variable initialized as $\widetilde{\mathbf{w}}^0 = \breve{\mathbf{w}}_t$, and the following update steps are performed iteratively,

$$\begin{aligned}\widetilde{\mathbf{w}}^{k+1} = \prod_{\mathcal{W}_v}\big(&\widetilde{\mathbf{w}}^k - c(\nabla_{\mathbf{ww}} f_t(\breve{\mathbf{w}}_t)(\widetilde{\mathbf{w}}^k - \breve{\mathbf{w}}_t)\\ &+ h\nabla_{t\mathbf{w}} f_t(\breve{\mathbf{w}}_t) + \nabla_{\mathbf{w}} f_t(\breve{\mathbf{w}}_t))\big), \end{aligned} \quad (15)$$

for $k = 0, ..., K-1$, where K is the pre-determined number of iterations and c is a stepsize. After K updates, we have

$$\hat{\mathbf{w}}_{t+1|t} = \widetilde{\mathbf{w}}^K. \quad (16)$$

The estimated $\hat{\mathbf{w}}_{t+1|t}$ is taken as the prediction of $\mathbf{w}_{t+1}$.

The prediction step (11) of our algorithm brings extra computational burden inevitably. For explicit dynamic priors, the extra computation is not heavy but brings better learning performance as discussed in the experimental section. On the other hand, more computational burden may be incurred for the data-driven Φ, especially for large K. However, it still provides a way to construct Φ without explicit dynamic priors.

Algorithm 1 Online graph learning algorithm with prior dynamic models (OGLP)

Input:

α, β, γ, η_t, $\mathbf{z}_t$ for $t = 1, ...T$

Output:

The learned graph sequence $\mathbf{w}_t$

1: Initialize $\mathbf{w}_1$, $\bar{\mathbf{z}}_0 = \mathbf{0}$
2: **for** $t = 1, .., T$ **do**
3: Receive data $\mathbf{x}_t$
4: Update $\bar{\mathbf{z}}_t = \gamma \bar{\mathbf{z}}_{t-1} + (1-\gamma)\mathbf{z}_t$
5: Calculate the gradient $\nabla f_t(\mathbf{w}_t)$ with $\bar{\mathbf{z}}_t$
6: Update $\breve{\mathbf{w}}_t = \underset{\mathcal{W}_v}{\Pi} (\mathbf{w}_t - \eta_t \nabla f_t(\mathbf{w}_t))$
7: Update
$$\mathbf{w}_{t+1} = \Phi(\breve{\mathbf{w}}_t)$$
8: **end for**

IV. Dynamic Regret Analysis

Dynamic regret is a commonly used metric to evaluate the performance of online algorithms in dynamic environments [20]. In our problem, it is defined as

$$Reg_d(T) \triangleq \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}_t^*), \tag{17}$$

where $\mathbf{w}_t^* = \underset{\mathbf{w} \in \mathcal{W}_v}{\operatorname{argmin}} f_t(\mathbf{w})$ is the instantaneous optimal solution. Equation (17) indicates that dynamic regret quantifies the cumulative loss incurred by an online algorithm relative to the loss corresponding to the optimal instantaneous solutions. An online algorithm is admissible only if it yields a sublinear regret since online algorithms with sublinear regret perform asymptotically as well as the batch algorithms on average [12]. In this section, the dynamic regret of our online graph learning algorithm is performed.

A. Basic Assumptions

We first introduce some technical assumptions that are essential to dynamic regret analysis.

a1. All data received are bounded, i.e., there exists a constant $B_z > 0$ such that $\|\mathbf{z}_t\|_2 \leq B_z$ for all t.

a2. There is no isolated node in the graphs we update, i.e., there exist a constant $deg_{\min} > 0$ such that $\mathbf{S}\mathbf{w}_t \geq deg_{\min}\mathbf{1}$ (entry-wise inequality) for all t.

a3. The updated $\mathbf{w}_t$ is bounded, i.e., there exists a constant w_{max} such that $\mathbf{w}_t \leq w_{max}\mathbf{1}$ (entry-wise inequality) for all t.

a4. Dynamic model Φ is a contractive mapping, i.e., there exists $r \in (0, 1]$ such that

$$\|\Phi(\mathbf{w}_1) - \Phi(\mathbf{w}_2)\|_2 \leq r\|\mathbf{w}_1 - \mathbf{w}_2\|_2, \tag{18}$$

for all $\mathbf{w}_1, \mathbf{w}_2 \in \mathcal{W}_v$.

Without loss of generality, *a1* holds true naturally in real-world applications. As stated in [14], $\nabla f_t(\mathbf{w})$ is a Lipschitz-continuous function with constant $4\beta + 2\alpha(d-1)/(\min(\mathbf{S}\mathbf{w}))^2$. If the selected stepsize satisfies

$$\eta_t \leq \left(2\beta + \alpha(d-1)/(\min(\mathbf{S}\mathbf{w}_t))^2\right)^{-1},$$

the $f_t(\mathbf{w}_{t+1})$ is bounded, indicating that $\mathbf{w}_{t+1}$ is bounded and $\mathbf{S}\mathbf{w}_{t+1} > 0$. For a given feasible initial $\mathbf{w}_1$, we can hence obtain a sequence of $\mathbf{w}_t$ such that *a2* and *a3* are justified. The last assumption is made to avoid that poor prediction of dynamic models will be exacerbated by repeated update [20].

B. Dynamic Regret of OGLP

We first build the bound of the gradient $\nabla f_t(\mathbf{w}_t)$.

Proposition 1. *Under Assumption a1-a3, there exists a constant $L > 0$ such that $\|\nabla f_t(\mathbf{w}_t)\|_2 \leq L$ for all $\mathbf{w}_t \in \mathcal{W}_v$.*

Proof of Proposition 1. Clearly,

$$\begin{aligned}\bar{\mathbf{z}}_t &= \gamma \bar{\mathbf{z}}_{t-1} + (1-\gamma)\mathbf{z}_t \\ &= (1-\gamma)[\gamma^{t-1}\mathbf{z}_1 + \gamma^{t-2}\mathbf{z}_2 + ... + \mathbf{z}_t] \\ &= (1-\gamma)\sum_{\tau=1}^{t}\gamma^{t-\tau}\mathbf{z}_\tau.\end{aligned} \tag{19}$$

We can thereby represent $\nabla f_t(\mathbf{w}_t)$ as

$$\nabla f_t(\mathbf{w}_t) = 2(1-\gamma)\sum_{\tau=1}^{t}\gamma^{t-\tau}\mathbf{z}_\tau + 4\beta\mathbf{w}_t - \alpha\mathbf{S}^\top(\mathbf{S}\mathbf{w}_t)^{\cdot(-1)} \quad . \tag{20}$$

Then for $t = 1, ..., T$ and $\gamma > 0$,

$$\begin{aligned}&\|\nabla f_t(\mathbf{w}_t)\|_2 \\ \leq& 2(1-\gamma)\left\|\sum_{\tau=1}^{t}\gamma^{t-\tau}\mathbf{z}_\tau\right\|_2 + 4\beta\|\mathbf{w}_t\|_2 + \alpha\left\|\mathbf{S}(\mathbf{S}\mathbf{w}_t)^{\cdot(-1)}\right\|_2 \\ \leq& 2(1-\gamma)\left(\sum_{\tau=1}^{t}\gamma^{t-\tau}B_z\right) + 4\beta\|\mathbf{w}_t\|_2 + \alpha\left\|\mathbf{S}(\mathbf{S}\mathbf{w}_t)^{\cdot(-1)}\right\|_2 \\ \leq& 2B_z(1-\gamma^t) + 4\beta\|\mathbf{w}_t\|_2 + \alpha\|\mathbf{S}\|_2\left\|(\mathbf{S}\mathbf{w}_t)^{\cdot(-1)}\right\|_2 \\ \leq& 2B_z(1-\gamma^t) + 2\sqrt{2}\beta\sqrt{d(d-1)}w_{\max} \\ &+ \alpha\sqrt{2(d-1)}\frac{\sqrt{d}}{deg_{\min}} \\ \leq& 2B_z + 2\sqrt{2}\beta\sqrt{d(d-1)}w_{\max} + \alpha\sqrt{2(d-1)}\frac{\sqrt{d}}{deg_{\min}} \\ \triangleq& L,\end{aligned} \tag{21}$$

The second inequality holds because of assumption *a1*. The fourth inequality holds because for all $\mathbf{w}_t \in \mathcal{W}_v$, $\|\mathbf{w}_t\|_2 \leq \sqrt{\frac{d(d-1)}{2}}w_{\max}$ under assumption *a3* and $\|\mathbf{S}\|_2 = \sqrt{2(d-1)}$ (cf. [21]). In addition, $\|(\mathbf{S}\mathbf{w}_t)^{\cdot(-1)}\|_2 \leq \frac{1}{deg_{\min}}\|\mathbf{1}\|_2 = \frac{\sqrt{d}}{deg_{\min}}$ due to assumption *a2*.

We next provide the dynamic regret of OGLP algorithm.

Theorem 1. *Under assumption a1- a4, if the selected stepsize η_t is a constant satisfying $\eta_t = \eta \leq \left(2\beta + \alpha(d-1)/deg_{\min}^2\right)^{-1}$, the dynamic regret Reg_d of OGLP algorithm satisfies*

$$\begin{aligned}&Reg_d(T) = \sum_{t=1}^{T} f_t(\mathbf{w}_t) - \sum_{t=1}^{T} f_t(\mathbf{w}_t^*) \\ \leq& \frac{d(d-1)w_{\max}^2}{4\eta} + \frac{\sqrt{2d(d-1)}w_{\max}}{2\eta}C_V^d + \frac{\eta T}{2}L^2,\end{aligned} \tag{22}$$

where $C_V^d \triangleq \sum_{t=2}^{T}\|\mathbf{w}_t^* - \Phi(\mathbf{w}_{t-1}^*)\|_2$.

Proof of Theorem 1. The sketch of the proof is derived from [22] and can also be found in [23]. However, we made some

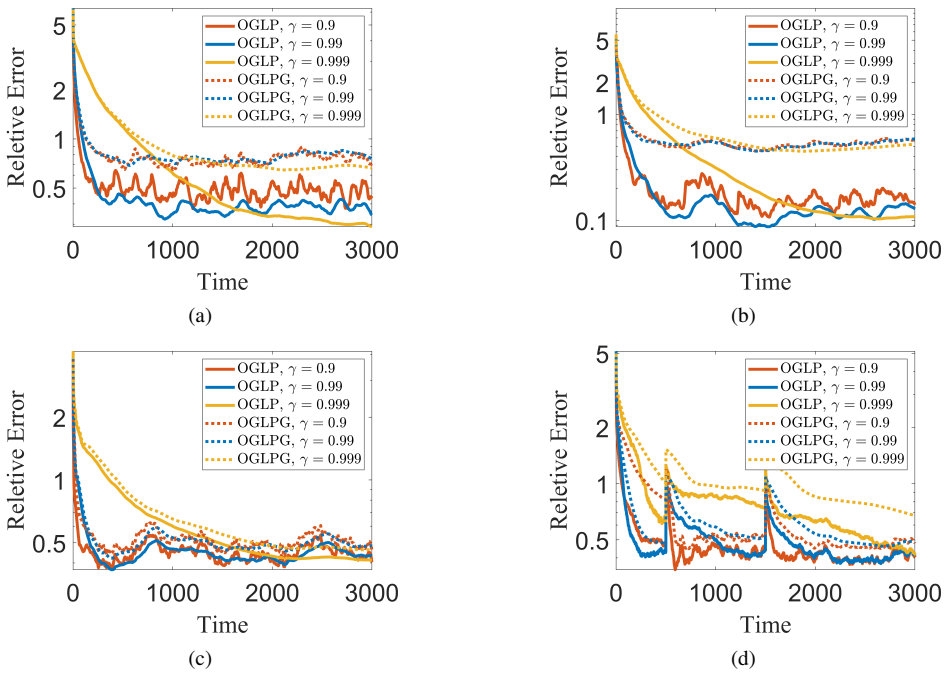

Fig. 1: Relative error of different dynamic models (a) AR model (b) Transition model (c) Social network (d) Switching model

minor modifications to the proof for our problem. The first one is the bound of $\|\nabla f_t(\mathbf{w}_t)\|_2$ used in the proof procedure is the L in Proposition 1. The other modification is that the upper bound of $\|\mathbf{w}_t\|_2$ for $\mathbf{w}_t \in \mathcal{W}_v$ is $\sqrt{\frac{d(d-1)}{2}} w_{\max}$.

From the definition of C_V^d, if Φ is in line with the real evolutionary patterns of dynamic graphs, we can conclude that $C_V^d \leq C_V \triangleq \sum_{t=2}^{T} \|\mathbf{w}_t^* - \mathbf{w}_{t-1}^*\|_2$, where C_V measures the variations of dynamic environments. Therefore, the prediction step using dynamic model Φ theoretically reduces the impact of dynamic environments, bringing a tighter regret bound compared with those without priors. Furthermore, if the selected η satisfies $\eta = \mathcal{O}(1/\sqrt{T})$, our algorithm can achieve $\mathcal{O}(\sqrt{T}(1+C_V^d))$ regret. Finally, note that the constant stepsize is required to be smaller than $\left(2\beta + \alpha(d-1)/deg_{\min}^2\right)^{-1}$. However, $deg_{\min}$ is a constant we cannot obtain in advance. An alternative is to use an adaptive stepsize introduced in [14], which is also adopted in our experiment.

V. Numerical Experiments

A. Synthetic data

We test our method on four dynamic models listed below.

1). Autoregressive (AR) model [20]. A first-order AR model is used

$$\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t, \tag{23}$$

where $\mathbf{A} \in \mathbb{R}^{p \times p}$ is an autoregressive transition matrix.

2). Transition model. In this model, graphs will evolve to a target topology known in advance, i.e,

$$\mathbf{w}_{t+1} = a\mathbf{w}_t + (1-a)\mathbf{w}_{\text{target}}, \tag{24}$$

where $0 < a < 1$ is a predefined controlling parameter and $\mathbf{w}_{\text{target}}$ is the target graph.

3). Social network model. This model describes the evolution of social networks. More details are referred in [25], and we will not introduce it in depth due to space limitation.

4). Switching model. In this model, the graph will switch to a new topology at some specific moments and remain static at other moments.

We first generate a graph following the way of [3] and take it as the initial graph at $t = 1$. Graphs of the remaining time slots are obtained by $\mathbf{w}_{t+1} = \Phi(\mathbf{w}_t)$ with Φ introduced above. All parameters of these dynamic models are carefully selected to prevent the situation where $\mathbf{w}_t \notin \mathcal{W}_v$. For switching model, we switch the topology of the graph at $t = 500$ and $t = 1500$. Smooth graph signal $\mathbf{x}_t$ is generated from graph at t by the same way introduced in [3]. We set $T = 3000$ and the obtained signals are taken as inputs sequentially. The adopted evaluation metric is relative error representing the accuracy of edge weight, i.e., $\|\mathbf{W}_t - \mathbf{W}_t^*\|_{\text{F}} / \|\mathbf{W}_t^*\|_{\text{F}}$, where $\mathbf{W}_t$ is the estimated adjacency matrix and $\mathbf{W}_t^*$ is the groundtruth of t. In our experiments, we fix $\alpha = 2$ and search the best β as [2] does. Since we have no knowledge of $deg_{\min}$ in advance, we hence adopt the adaptive stepsize $\eta_t = \left(2\beta + \alpha(d-1)/\min(\mathbf{S}\mathbf{w}_t)^2\right)^{-1}$ as [14] does. Furthermore, we take [14] as the baseline and name it as OGLPG since it is the only method we can find that learns graphs under smoothness prior in an online fashion. The parameters of OGLPG are the same as ours.

Figure 1 shows the performance of the above models. For the same γ, our method with dynamic priors outperforms OGLPG since lower relative errors are obtained as rounds increase. Furthermore, our method exhibits stronger tracking ability. For switching model, fewer iterations are required to restore previously low relative error. Since switching model has no explicit form of Φ, we adopt the data-driven method

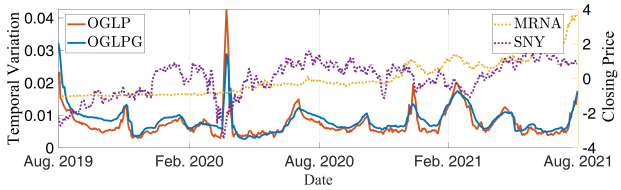

Fig. 2: Temporal variation of stock market graph

introduced in section III to construct Φ. Another interesting point is that large γ results in a "smoother" learning curve by sacrificing tracking performance while small γ is more susceptible to new data. More oscillations occur in the curves of $\gamma = 0.9$. Therefore, it is a trade-off between tracking ability and stability to choose a suitable γ.

B. Real data

Our method is also applied to the stock closing price data[1] to learn time-varying graphs of relationships among different companies. We collect stock prices data of 10 pharmaceutical companies, including Amgen (AMGN), Astrazeneca (AZN), GlaxoSmithKline (GSK), Johnson & Johnson (JNJ), Moderna (MRNA), Novavax (NVAX), Pfizer (PFE), Perrigo (PRGO), Sanofi (SNY) and Zoetis (ZTS). We focus on data from August 1st 2019 to July 30th 2021, during which COVID-19 pandemic outbroke and caused market instabilities of pharmaceutical companies. All data are standardized first and we aim to detect changes of the learned graphs in this unstable market. We set $T = 504$ (the number of working days from August 1st 2019 to July 30th 2021) and $d = 10$. Moreover, α, β and γ are set to be 2, 1.2, 0.99 respectively for both OGLP and OGLPG. The data-driven method is used to predict graph evolution. We employ a metric named temporal variation $\|\mathbf{w}_t - \mathbf{w}_{t-1}\|_2 / \|\mathbf{w}_{t-1}\|_2$ to measure the structural changes between two consecutive graphs. In Fig. 2, historical closing stock prices of two companies, i.e., MRNA and SNY, are first depicted to show the instability of the market. Compared with MRNA, the stock prices of SNY witnessed marked volatility in March, implying changes of the corresponding graph of relationship. As shown in Fig. 2, a significant temporal variation is detected in March 2020, which is consistent with changes of stock prices in March. However, OGLP tends to be more sensitive to changes, meaning better tracking ability in dynamic environments. When the market stabilizes, OGLP is able to return to the steady state (small temporal variations) more quickly.

VI. Conclusion

In this paper, a framework of learning graphs in an online fashion with dynamic priors is proposed. Where no priors are available, we use a data-driven method to predict the evolution of dynamic graphs. Dynamic regret analysis theoretically illustrates the power of dynamic priors. Experimental results show the superiority of our method.

[1]https://finance.yahoo.com/

References

[1] G. Mateos, S. Segarra, A. G. Marques, and A. Ribeiro, "Connecting the dots: Identifying network structure via graph signal processing," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp.16–43, 2019.

[2] V. Kalofolias, "How to learn a graph from smooth signals," in *Artif. Intel. and Stat. (AISTATS)*, 2016, pp. 920–929.

[3] X. Dong, D. Thanou, P. Frossard, and P. Vandergheynst, "Learning laplacian matrix in smooth graph signal representations," *IEEE Trans. Signal Process.*, vol. 64, no. 23, pp. 6160–6173, 2016.

[4] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 44–63, 2019.

[5] J. Liu and J. Han, "Spectral clustering," in *Data Clustering*, pp. 177–200. Chapman and Hall/CRC,2018.

[6] M. Yuan and Y. Lin, "Model selection and estimation in the gaussian graphical model," *Biometrika*, vol. 94,no. 1, pp. 19–35, 2007.

[7] Y. Shen, B. Baingana, and G. B. Giannakis, "Nonlinear structural vector autoregressive models for inferring effective brain network connectivity," *arXiv preprint arXiv:1610.06551*, 2016.

[8] A. Ortega, P. Frossard, and J. Kovačević, J. Moura and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proc. IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[9] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard, "Learning time varying graphs," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2017, pp. 2826–2830.

[10] K. Yamada, Y. Tanaka, and A. Ortega, "Time-varying graph learning with constraints on graph temporal variation," *arXiv preprint arXiv:2001.03346 [eess.SP]*, 2020.

[11] B. Zaman, L. Ramos, and B. Beferull-Lozano, "Dynamic regret analysis for online tracking of time-varying structural equation model topologies," in *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE, 2020,pp. 939–944.

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

[13] R. Money, J. Krishnan, and B. Beferull-Lozano, "Online nonlinear topology identification from graph-connected time series," *arXiv preprint-arXiv:2104.00030*, 2021.

[14] S. Saboksayr, G. Mateos, and M. Cetin, "Online graph learning under smoothness priors," *arXiv preprint arXiv:2103.03762*, 2021.

[15] R. Shafipour, A. Hashemi, G. Mateos,and H. Vikalo,"Online topology inference from streaming stationary graph signals," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 140–144.

[16] A. Natali, M. Coutino, E. Isufi, and G. Leus,"Online time-varying topology identification via prediction-correction algorithms," in *2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 5400–5404.

[17] E. Hazan, "Introduction to online convex optimization," *arXiv preprint arXiv:1909.05207*, 2019.

[18] A. Simonetto, E. Dall'Anese, S. Paternain, G. Leus, and G. B. Giannakis, "Time-varying convex optimization: Time-structured algorithms and applications," *Proc. IEEE*, vol.108, no. 11, pp. 2032–2048, 2020.

[19] A. Simonetto and E. Dall'Anese,"Prediction-correction algorithms for time-varying constrained optimization," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 5481–5494,2017.

[20] E. C. Hall and R. M. Willett, "Online convex optimization in dynamic environments," *IEEE J. Sel. Top. Signal Process.*, vol. 9, no. 4, pp.647–662, 2015.

[21] S. S. Saboksayr, G. Mateos, and M. Cetin, "Online discriminative graph learning from multiclass smooth signals, " *Signal Process.*, vol.186, pp. 108101, 2021.

[22] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th international conference on machinelearning (ICML)*, 2003, pp. 928–936.

[23] L. Zhang, S. Lu, and Z. Zhou, "Adaptive online learning in dynamic environments," in *Proceedings of the 32nd International Conference on Neural In-formation Processing Systems*, 2018, pp. 1330–1340.

[24] A. Mokhtari, S. Shahrampour, A. Jadbabaie, and A. Ribeiro, "Online optimization in dynamic environments: Improved regret rates for strongly convex problems," in *2016 IEEE 55th Conference onDecision and Control (CDC)*, IEEE, 2016, pp. 7195–7201.

[25] T. Snijders, "The statistical evaluation of social network dynamics," *Sociol. Methodol.*, vol. 31,no. 1, pp. 361–395, 2001.