Online Proximal ADMM for Graph Learning from Streaming Smooth Signals

Hector Chahuara[†] and Gonzalo Mateos[‡]

[†]Department of Electrical Engineering, Pontificia Universidad Católica del Perú, Lima, Perú

[‡]Department of Electrical and Computer Engineering, University of Rochester, Rochester, NY, USA

Abstract—Graph signal processing deals with algorithms and signal representations that leverage graph structures for multivariate data analysis. Often said graph topology is not readily available and may be time-varying, hence (dynamic) graph structure learning from nodal (e.g., sensor) observations becomes a critical first step. In this paper, we develop a novel algorithm for *online* graph learning using observation streams, assumed to be smooth on the latent graph. Unlike batch algorithms for topology identification from smooth signals, our modus operandi is to process graph signals sequentially and thus keep memory and computational costs in check. To solve the resulting smoothness-regularized, time-varying inverse problem, we develop online and lightweight iterations built upon the proximal variant of the alternating direction method of multipliers (ADMM), well known for its fast convergence in batch settings. The proximal term in the topology updates seamlessly implements a temporal-variation regularization, and we argue the online procedure exhibits sublinear static regret under some simplifying assumptions. Reproducible experiments with synthetic and real graphs demonstrate the effectiveness of our method in adapting to streaming signals and tracking slowly-varying network connectivity. The proposed approach also exhibits better tracking performance (in terms of suboptimality), when compared to state-of-the-art online graph learning baselines.

Index Terms—Topology identification, dynamic network tracking, proximal ADMM, smooth signal, regret analysis

I. INTRODUCTION

Graphs serve as powerful abstractions of the underlying relational structure among data entities (e.g., sensors, brain regions), thus becoming a viable modeling tool for a host of signal processing and machine learning applications. Accordingly, there has been growing interest in describing multivariate complex data, such as social networks, neural activity time series, and urban traffic flows, as graph signals; see e.g. [1]. However, this valuable (latent) graph structure may not be explicitly available, and so it needs to be estimated from nodal data before any meaningful downstream analysis can be conducted [2], [3]. Networked systems (as the examples above) are in constant flux, with increasing size and complexity, so there is a need for efficient topology inference algorithms that can process data streams generated by *dynamic* networks [4], [5].

Related work. The inverse problem of searching for a graph (encoded by e.g., an adjacency or Laplacia matrix) that in some sense best describes a given multivariate dataset is known as network topology inference or graph structure learning. Optimality criteria and constraints often arise from statistical priors, physical principles, interpretability, or downstream task goals, which result in models that tie the observations to the latent graph. These models can be probabilistic [6]–[8], assume graph signals are stationary or generated by network diffusion [9]–[13], or else they are smooth w.r.t. to the graph [14]–[18]. In this work, we rely on the latter class.

Specifically, graph learning from smooth signals is formulated as a non-smooth strongly convex optimization problem, and several *batch* solvers have been developed to date. For instance, a scalable primal-dual algorithm was proposed in [15], while alternating direction method of multipliers (ADMM) iterations were put forth in [19]. An accelerated dual-domain proximal gradient algorithm (DPG) was studied in [16], with the first convergence rate results for this problem. The best rates to date are reported in [17], for a proximal ADMM (PADMM) algorithm developed for time-varying graph learning. All these approaches operate in batch mode, they are non-recursive, and hence their computational complexity and memory storage grow linearly with time when used to estimate dynamic network topologies.

Recognizing these challenges, *online* graph learning methods have been recently developed to process data sequentially-in-time. The online proximal gradient (PG) [20] and DPG [21] approaches can track slowly-varying dynamic graph topologies from streams of smooth signal observations. We would be remiss not to mention other online graph learning methods that rely on assumptions other than signal smoothness. For instance, [22] introduces an online algorithm that uses observations from a Laplacian-based continuous-time graph process, while [23] leverages graph signal stationarity. An encompassing (model-independent) framework that exploits a prediction-correction strategy was proposed in [24].

Proposed approach and contributions. In this paper, our main contribution is to develop a novel online algorithm to track the topology of (potentially dynamic) undirected graphs using streaming smooth signals. Our method builds on PADMM [25], whose provably fast local convergence properties have been well documented in a batch graph learning setting [17]. Our technical contributions are significant for several reasons. The proposed online version of PADMM (henceforth dubbed OPADMM): (i) incorporates proximal terms in the topology updates which effectively serve as temporal-variation regularizers; (ii) yields entrywise separable and lightweight updates, with time-independent computational cost and memory footprint; and (iii) exhibits sublinear static regret under simplifying assumptions. Numerical tests with synthetic and real-world graphs demonstrate that PADMM outperforms state-of-the-art online graph learning baselines for this problem, and can seamlessly track slowly-varying dynamic network topologies. In support of reproducible research, we share the code used to generate all figures in this paper.

II. PRELIMINARIES AND PROBLEM FORMULATION

A. Graph-theoretic notions and problem statement

Consider an unknown undirected graph denoted as $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$, with vertices $\mathcal{V} = \{1, \ldots, n\}$, edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and the symmetric adjacency matrix $\mathbf{W} \in \mathbb{R}_+^{n \times n}$ that collects the edge weights with $\mathbf{W}_{i,j} = 0$ for $(i, j) \notin \mathcal{E}$. Also, $\mathbf{W}_{i,i} = 0$, $\forall i \in \mathcal{V}$, since we exclude self-loops. The graph Laplacian is $\mathbf{L} = \text{diag}(\mathbf{d}) - \mathbf{W}$, where $\mathbf{d} = \mathbf{W}\mathbf{1}_n$ represents the vector of nodal degrees

Along with the graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$, the graph signal observations are denoted as $\mathbf{x} \in \mathbb{R}^n$, where x_i is the value measured at node $i \in \mathcal{V}$. The Dirichlet energy or total variation (TV) quantifies the smoothness of graph signals supported on $\mathcal{G}$ [26], and is given by

$$\mathrm{TV}\left(\mathbf{x}\right) := \mathbf{x}^\top \mathbf{L} \mathbf{x} = \frac{1}{2} \sum_{i \neq j} W_{ij} \left(\mathbf{x}_i - \mathbf{x}_j\right)^2, \qquad (1)$$

where $\mathrm{TV}\left(\mathbf{x}\right) \in [0, \lambda_{\max}]$, where $\lambda_{\max}$ is the largest eigenvalue of the positive semidefinite (PSD) matrix $\mathbf{L}$. We say a graph signal is smooth (or low-pass bandlimited) if it has a small total variation.

We have all the ingredients to state the topology inference problem. Our goal is to learn an undirected graph $\mathcal{G}\left(\mathcal{V}, \mathcal{E}, \mathbf{W}\right)$ from a stream of graph signal measurements $\mathbf{X} := \{\mathbf{x}^{(k)}\}$ that are smooth on $\mathcal{G}$. We also consider tracking dynamic networks with slowly time-varying weight matrix $\mathbf{W}^{(k)}$, $k = 1, 2, \ldots$ ($\mathcal{V}$ remains fixed throughout).

B. Batch topology identification from smooth signals

Consider a batch of p graph signals $\mathbf{x}^{(k)}$, $k = 1, \ldots, p$ collected as columns of the data matrix $\mathbf{X} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(p)}] \in \mathbb{R}^{n \times p}$. Then form the vertex pairwise dissimilarity matrix $\mathbf{Z} \in \mathbb{R}_+^{n \times n}$, with entries $Z_{ij} = \|\mathbf{X}_{i,:} - \mathbf{X}_{j,:}\|_2^2$, $(i, j) \in \mathcal{V}$. With these definitions, the aggregate TV measure over $\mathbf{X}$ can be expressed as

$$\sum_{k=1}^p \mathrm{TV}\left(\mathbf{x}^{(k)}\right) = \mathrm{tr}\left(\mathbf{X}^\top \mathbf{L} \mathbf{X}\right) = \frac{1}{2} \|\mathbf{Z} \odot \mathbf{W}\|_{1,1}, \qquad (2)$$

where $\odot$ stands for the Hadamard product and $\|\cdot\|_{1,1}$ denotes the entrywise matrix ℓ_1 norm. Identity (2) shows that minimizing the aggregate TV measure w.r.t. $\mathbf{W}$ boils down to preferentially dropping edges (i, j) with higher pairwise nodal dissimilarities $Z_{i,j}$, thus sparsifying $\mathcal{E}$. This link between signal smoothness and edge sparsity was first recognized in [15], which advocates graph structure learning by solving the following convex inverse problem

$$\min_{\mathbf{W} \in \mathbb{R}^{n \times n}} \quad \|\mathbf{Z} \odot \mathbf{W}\|_{1,1} - \alpha \mathbf{1}_n^\top \log\left(\mathbf{W} \mathbf{1}_n\right) + \frac{\beta}{2} \|\mathbf{W}\|_F^2$$
$$\text{subject to } \mathrm{diag}\left(\mathbf{W}\right) = \mathbf{0}_n, W_{ij} = W_{ji} \geq 0, i \neq j, \qquad (3)$$

where $\alpha, \beta > 0$ are tunable regularization hyperparameters. The logarithmic barrier imposed over the nodal degrees $\mathbf{d} = \mathbf{W} \mathbf{1}_n$ excludes isolated (null degree) vertices in the learned graph. The Frobenius-norm regularization on $\mathbf{W}$ controls edge sparsity via β.

Problem (3) can be further simplified. For instance, the free-decision variables can be arranged into a compact vector $\mathbf{w} := \mathrm{vec}\left(\mathrm{triu}\left(\mathbf{W}\right)\right) \in \mathbb{R}_+^r$, $r = \frac{n(n-1)}{2}$, i.e. the upper-triangular elements of $\mathbf{W}$ (since it is symmetric and hollow). To impose $W_{ij} \geq 0$, we augment the cost with an indicator function $\iota_{\geq \mathbf{0}}(.)$, which takes the value of zero when its argument is a vector with non-negative entries, otherwise it is equal to $+\infty$. Given these definitions, one can rewrite (3) as an equivalent unconstrained, non-differentiable problem

$$\underset{\mathbf{w} \in \mathbb{R}^r}{\mathrm{argmin}} \quad 2\mathbf{z}^\top \mathbf{w} + \beta \|\mathbf{w}\|_2^2 + \iota_{\geq \mathbf{0}}\left(\mathbf{w}\right) - \alpha \mathbf{1}_n^\top \log\left(\mathbf{S} \mathbf{w}\right), \qquad (4)$$

where $\mathbf{S} \in \{0, 1\}^{n \times r}$ maps vectorized edge weights to nodal degrees.

III. PROXIMAL ADMM FOR GRAPH LEARNING

Problem (4) has a structure amenable to ADMM-type solvers, as first recognized in [19]. Here we briefly review the batch PADMM iterations [17] that inspired our online algorithm in Section IV.

A. Applying ADMM to the graph learning problem

Going back to (4), we introduce an auxiliary variable $\mathbf{v} \in \mathbb{R}^n$ (a proxy for $\mathbf{d}$) and the variable-splitting constraint $\mathbf{v} = \mathbf{S} \mathbf{w}$, to yield

$$\underset{\mathbf{w} \in \mathbb{R}^r, \mathbf{v} \in \mathbb{R}^n}{\mathrm{argmin}} \quad f\left(\mathbf{w}\right) + g\left(\mathbf{v}\right) \text{ subject to } \mathbf{S} \mathbf{w} - \mathbf{v} = \mathbf{0}_n, \qquad (5)$$

where the functions f and g are defined as

$$f\left(\mathbf{w}\right) = 2\mathbf{z}^\top \mathbf{w} + \beta \|\mathbf{w}\|_2^2 + \iota_{\geq \mathbf{0}}\left(\mathbf{w}\right)$$
$$g\left(\mathbf{v}\right) = -\alpha \mathbf{1}_n^\top \log\left(\mathbf{v}\right).$$

For $\rho > 0$, the augmented Lagrangian of the problem (5) is given by

$$\mathcal{L}_\rho\left(\mathbf{w}, \mathbf{v}, \boldsymbol{\lambda}\right) = f\left(\mathbf{w}\right) + g\left(\mathbf{v}\right) + \boldsymbol{\lambda}^\top\left(\mathbf{S} \mathbf{w} - \mathbf{v}\right) + \frac{\rho}{2} \|\mathbf{S} \mathbf{w} - \mathbf{v}\|_2^2.$$

PADMM generalizes ADMM by incorporating quadratic proximity terms in the primal subproblems [25]. Given PSD matrices $\mathbf{G}$ and $\mathbf{H}$, the k-th PADMM iteration consists of three updates

$$\mathbf{w}^{(k+1)} = \underset{\mathbf{w} \in \mathbb{R}^r}{\mathrm{argmin}} \quad \mathcal{L}_\rho(\mathbf{w}, \mathbf{v}^{(k)}, \boldsymbol{\lambda}^{(k)}) + \|\mathbf{w} - \mathbf{w}^{(k)}\|_{\mathbf{G}/2}^2, \qquad (6)$$

$$\mathbf{v}^{(k+1)} = \underset{\mathbf{v} \in \mathbb{R}^n}{\mathrm{argmin}} \quad \mathcal{L}_\rho(\mathbf{w}^{(k+1)}, \mathbf{v}, \boldsymbol{\lambda}^{(k)}) + \|\mathbf{v} - \mathbf{v}^{(k)}\|_{\mathbf{H}/2}^2, \qquad (7)$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \rho(\mathbf{S} \mathbf{w}^{(k+1)} - \mathbf{v}^{(k+1)}). \qquad (8)$$

Note that when $\mathbf{G} = \mathbf{0}_{r \times r}$ and $\mathbf{H} = \mathbf{0}_{n \times n}$, the proximity terms vanish and the updates (6)-(8) reduce to those of the standard ADMM. Typical choices for these matrices are $\mathbf{G} = \tau_1^{-1} \mathbf{I}_r - \rho \mathbf{S}^\top \mathbf{S}$ and $\mathbf{H} = (\tau_2^{-1} - \rho)\mathbf{I}_n$. Parameter values $0 < \tau_1 < \frac{1}{\rho \|\mathbf{S}\|_2^2}$ and $0 < \tau_2 < \frac{1}{\rho}$, with $\|\mathbf{S}\|_2^2 = 2(n-1)$ [16] , guarantee the PSD requirement and convergence of the PADMM algorithm [17].

B. Efficient proximal updates

The primal updates (6) and (7) can be expressed in terms of proximal operators, as

$$\mathbf{w}^{(k+1)} = \underset{\tau_1 \cdot f}{\mathrm{prox}}\left(\overline{\mathbf{w}}\right), \qquad \mathbf{v}^{(k+1)} = \underset{\tau_2 \cdot g}{\mathrm{prox}}\left(\overline{\mathbf{v}}\right),$$

where the auxiliary variables $\overline{\mathbf{w}}$ and $\overline{\mathbf{v}}$ are given by

$$\overline{\mathbf{w}} = \mathbf{w}^{(k)} - \tau_1 \cdot \left[\rho \cdot \mathbf{S}^\top\left(\mathbf{S} \mathbf{w}^{(k)} - \mathbf{v}^{(k)} + \frac{\boldsymbol{\lambda}^{(k)}}{\rho}\right)\right],$$

$$\overline{\mathbf{v}} = \mathbf{v}^{(k)} - \tau_2 \cdot \rho\left(\mathbf{S} \mathbf{w}^{(k+1)} - \mathbf{v}^{(k)} - \frac{\boldsymbol{\lambda}^{(k)}}{\rho}\right).$$

To arrive at the custom PADMM algorithm for graph learning in [17], we recognize both proximal operators admit simple expressions

$$\underset{\tau \cdot f}{\mathrm{prox}}\left(\mathbf{w}\right) = \max\left(\frac{\mathbf{w} - 2\tau\mathbf{z}}{2\tau\beta + 1}, \mathbf{0}_r\right),$$

$$\underset{\tau \cdot g}{\mathrm{prox}}\left(\mathbf{v}\right) = \frac{\mathbf{v} + \sqrt{\mathbf{v}^2 + 4\tau\alpha\mathbf{1}_n}}{2}.$$

which act entrywise on their vector arguments. All in all, the PADMM updates at the k-th iteration become

$$\mathbf{w}^{(k+1)} = \max\left(\frac{\overline{\mathbf{w}} - 2\tau_1\mathbf{z}}{2\tau_1\beta + 1}, \mathbf{0}_r\right), \qquad (9)$$

$$\mathbf{v}^{(k+1)} = \frac{1}{2}\left(\overline{\mathbf{v}} + \sqrt{\overline{\mathbf{v}}^2 + 4\tau_2\alpha\mathbf{1}_n}\right), \qquad (10)$$

$$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \rho\left(\mathbf{S} \mathbf{w}^{(k+1)} - \mathbf{v}^{(k+1)}\right). \qquad (11)$$

PADMM enjoys a global convergence rate of $\mathcal{O}(\frac{1}{k})$, and it has recently been shown to exhibit local convergence properties with a significantly faster rate of $\mathcal{O}(\mu^k)$, $0 < \mu < 1$ [27]; see also [17].

IV. ONLINE GRAPH LEARNING FROM STREAMING SIGNALS

A. Online PADMM algorithm

PADMM's favorable convergence properties in the batch topology inference setting discussed so far, motivates well its adoption for online learning. Online estimation of $\mathbf{W}$ (or even tracking $\mathbf{W}^{(k)}$ in

Algorithm 1: OPADMM for graph learning

Input: Edge weight-to-node degree map $\mathbf{S}$; new dissimilarity information $\overline{\mathbf{z}}^{(k)}$ at time k ; regularization hyperparameters α and β; PADMM hyperparameters ρ, τ_1 and τ_2; forgetting factor γ; initial values $\mathbf{w}^{(0)}$, $\mathbf{v}^{(0)}$ and $\boldsymbol{\lambda}^{(0)}$

Output: Tracking solution available at the k-th iteration $\mathbf{w}^{(k)}$

for $k = 1, 2, \ldots$ **do**

    Update $\gamma^{(k)}$

    $\mathbf{z}_{1:k} \leftarrow \left(1 - \gamma^{(k)}\right) \mathbf{z}_{1:k-1} + \gamma^{(k)} \overline{\mathbf{z}}^{(k)}$

    $\overline{\mathbf{w}} \leftarrow \mathbf{w}^{(k-1)} - \tau_1 \rho \mathbf{S}^\top \left(\mathbf{S}\mathbf{w}^{(k-1)} - \mathbf{v}^{(k-1)} + \frac{\boldsymbol{\lambda}^{(k-1)}}{\rho} \right)$

    $\mathbf{w}^{(k)} \leftarrow \frac{1}{2\tau_1\beta + 1} \max\left(\overline{\mathbf{w}} - 2\tau_1 \mathbf{z}_{1:k}, \mathbf{0}_r \right)$

    $\overline{\mathbf{v}} \leftarrow (1 + \rho\tau_2) \mathbf{v}^{(k-1)} - \rho\tau_2 \mathbf{S}\mathbf{w}^{(k)} + \tau_2 \boldsymbol{\lambda}^{(k-1)}$

    $\mathbf{v}^{(k)} \leftarrow \frac{1}{2} \left(\overline{\mathbf{v}} + \sqrt{\overline{\mathbf{v}}^2 + 4\tau_2\alpha\mathbf{1}_n} \right)$

    $\boldsymbol{\lambda}^{(k)} \leftarrow \boldsymbol{\lambda}^{(k-1)} + \rho \left(\mathbf{S}\mathbf{w}^{(k)} - \mathbf{v}^{(k)} \right)$

end

---

dynamic settings) using streaming signals $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k)}, \mathbf{x}^{(k+1)}, \ldots\}$ can be cast as the following time-varying optimization problem

$$\hat{\mathbf{w}}^{(k)} \in \underset{\mathbf{w} \in \mathbb{R}^n}{\mathrm{argmin}} \;\; \overbrace{2\mathbf{z}_{1:k}^\top \mathbf{w} + \beta \|\mathbf{w}\|_2^2 + \iota_{\geq \mathbf{0}}(\mathbf{w})}^{f^{(k)}(\mathbf{w})} \underbrace{-\alpha \mathbf{1}_n^\top \log(\mathbf{S}\mathbf{w})}_{g(\mathbf{S}\mathbf{w})}. \tag{12}$$

Notice that the non-smooth, strongly convex component $f^{(k)}$ is time-varying because of the vectorized dissimilarity matrix $\mathbf{z}_{1:k}$, which is formed using all signals $\{\mathbf{x}^{(k)}\}$ acquired by time k; see also (13).

While (12) can be solved by sequentially running a batch algorithm every time a new graph signal arrives, such procedure would incur in high delay and computational burden – an unsatisfactory solution for delay-sensitive tasks. Furthermore, searching for a high-precision solution in dynamic network settings may not be worth the effort, since the new estimate $\hat{\mathbf{w}}^{(k+1)}$ may substantially deviate from the prior estimate $\hat{\mathbf{w}}^{(k)}$. Accordingly, we propose an online (recursive) algorithm to track the solution of the time-varying problem (12), which builds on the PADMM foundations in Section III.

Our online method consists on two steps per time instant $k = 1$, $2, \ldots$. First, we recursively update the upper-triangular entries of the Euclidean distance matrix $\mathbf{z}_{1:k}$, once a new datum $\mathbf{x}^{(k)}$ is acquired at time k. Specifically, given $\mathbf{x}^{(k)}$ we form the vectorized dissimilarity matrix $\overline{\mathbf{z}}^{(k)}$ and update the running average

$$\mathbf{z}_{1:k} = \left(1 - \gamma^{(k)}\right) \mathbf{z}_{1:k-1} + \gamma^{(k)} \overline{\mathbf{z}}^{(k)}, \tag{13}$$

where $\gamma^{(k)} \in (0, 1)$ is a forgetting factor. In stationary settings (the graph is static), we choose $\gamma^{(k)} = \frac{1}{k}$, which renders (13) an infinite-memory running sample average. In dynamic environments we select $\gamma^{(k)} = 2 \times 10^{-3}$, so that (13) becomes an exponentially-weighted moving average with the ability to track topology changes. In the second step, our online algorithm updates the primal and dual vectors $\mathbf{w}^{(k)}$, $\mathbf{v}^{(k)}$ and $\boldsymbol{\lambda}^{(k)}$ by running a single iteration of the batch PADMM updates (9), (10), (11), respectively; but employing $\mathbf{z}_{1:k}$ instead of $\mathbf{z}$ in (9) to reflect the online nature of our method. The resulting topology estimate at time k is stored in $\mathbf{w}^{(k)}$, and the overall online (O)PADMM procedure is tabulated under Algorithm 1.

B. Discussion, computational complexity and regret analyses

It is worth noting that matrix-vector multiplications with $\mathbf{S}$ effectively boil down to tabular searches and additions of the vector entries. All other operations in Algorithm 1 are pointwise, so the computational cost per iteration is $\mathcal{O}(r)$ (recall $r = \frac{n(n-1)}{2}$). This is on par with online PG [20] and DPG [21], and less expensive than the prediction-correction algorithm in [24]. Unlike batch algorithms for time-varying graph learning [17], [28], OPADMM's memory storage and computational cost do not depend on the length of the data acquisition horizon. These batch algorithms augment the objective function (3) with a temporal-variation regularization, to enforce similarity between consecutive adjacency matrix estimates. Interestingly, this feature is seamlessly embedded in OPADMM by virtue of the proximity term in (6) – a significant improvement over the state-of-the-art online DPG algorithm.

To obtain performance guarantees in terms of static regret, we will henceforth assume that nodal degrees are in a compact set that is uniformly bounded away from zero (meaning $d_{\min}\mathbf{1}_n \preceq \mathbf{d} \preceq d_{\max}\mathbf{1}_n$, for some $0 < d_{\min} \leq d_{\max}$). This is a loose requirement that ensures the subgradient of the objective has bounded norm. In practice if degrees become arbitrarily small, it is prudent to apply a threshold and remove the loosely connected nodes from $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$. With this extra requirement one can show that for $\tau_2 = \rho$ so that $\mathbf{H} = \mathbf{0}_{n \times n}$, [29, Theorem 3] ensures that OPADMM attains a sublinear static regret, namely

$$\sum_{k=1}^{p} f^{(k)}(\mathbf{w}^{(k)}) + g(\mathbf{v}^{(k)}) - \min_{\mathbf{S}\mathbf{w}=\mathbf{v}} \sum_{k=1}^{p} f^{(k)}(\mathbf{w}) + g(\mathbf{v}) \leq \mathcal{O}(\sqrt{p}).$$

The extension to the case where $\mathbf{H} \neq \mathbf{0}_{n \times n}$ is left as future work.

V. EXPERIMENTAL EVALUATION

Here we first test OPADMM using various graphs ensembles that were synthetically generated using different models. We evaluate the convergence of our proposed method by montoring the suboptimality (tracking error) $\|\mathbf{w}^{(k)} - \hat{\mathbf{w}}\|_2$, where $\mathbf{w}^{(k)}$ is the solution produced by Algorithm 1 at the k-th iteration, and $\hat{\mathbf{w}}$ corresponds to the solution of the batch method using the values of hyperparameters α and β that are the best in a grid search evaluation using the method in [15]. Finally, we perform a grid search for the hyper-parameters ρ, τ_1 and τ_2 of OPADMM and compare the best case with the state-of-the-art-methods online PG and DPG. We also use real world datasets to validate our proposed method in a similar fashion to the tests with synthetic graphs. The code to generate all figures in this work is available in a public GitHub repository at https://github.com/hchahuara/opadmm.

A. Synthetic graphs

Stationary graphs. A set of $p = 1000$ smooth graph signals, each one with $n = 100$ entries, were synthetically generated and corrupted with Gaussian noise ($\mu = 0$, $\sigma^2 = 0.01$). Graphs realizations were drawn from three random graph models: random geometric with Gaussian kernel, Erdös-Rényi (ER), and preferential attachment (PA). Sparse connections were achieved with threshold 0.8 and scale 0.2 for the Gaussian model, edge probability 0.1 for the ER model, and 2 connected nodes initially and then adding new nodes one at a time for the PA model. Numerical results for the aforementioned test cases are depicted in Figure 1 (a)-(c). For stationary graphs, OPADMM outperforms online PG and DPG in terms of convergence speed.

Time-varying graphs. We generated another set of $p = 2000$ smooth graph signals, each with $n = 100$ elements, and similarly corrupted them with Gaussian noise ($\mu = 0$, $\sigma^2 = 0.01$). These graph signals were designed to be smooth on time-varying (piecewise-stationary) graphs, with 10% of the edges resampled after 1000 samples. Dynamic graphs were generated using the models with the same parameter values used for stationary graphs. The results are shown in Figure 1 (d)-(f), from where it is apparent that OPADMM
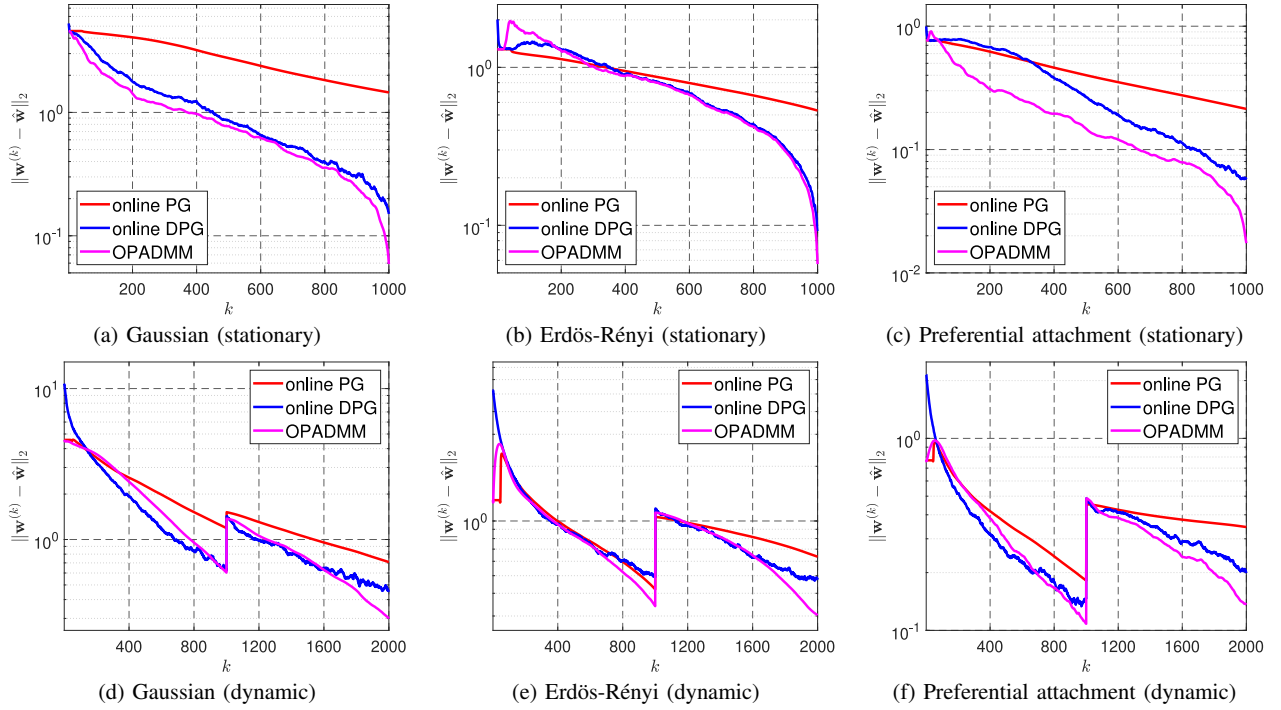
Fig. 1: Convergence behavior illustrated via the evolution of suboptimality $\|\mathbf{w}^{(k)} - \hat{\mathbf{w}}\|_2$ for synthetic random graph models with $n = 100$ nodes: (a)-(c) stationary graphs, and (d)-(f) time-varying graphs. For the dynamic networks, the topology changes at $k = 1000$. In stationary settings, our proposed method converges faster than online PG and DPG algorithms, while in dynamic settings, it demonstrates superior adaptability to changes in network topology.
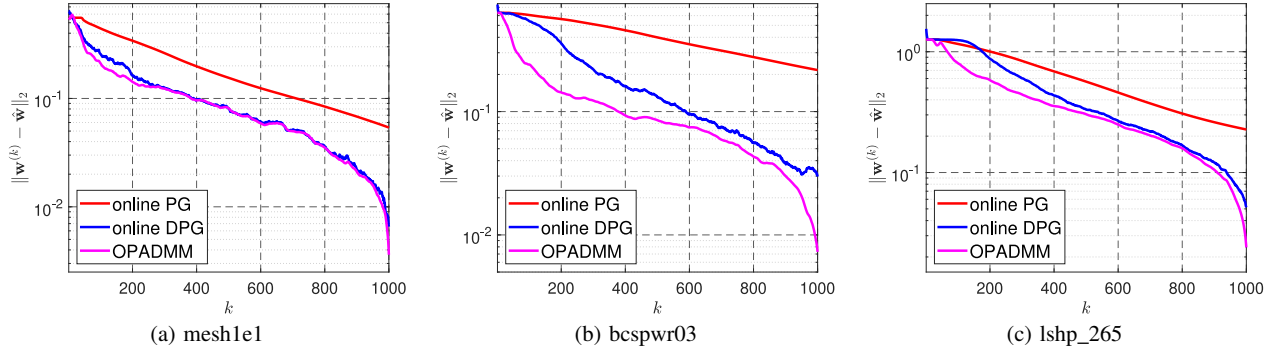

Fig. 2: Convergence behavior illustrated via the evolution of suboptimality $\|\mathbf{w}^{(k)} - \hat{\mathbf{w}}\|_2$ for real-world graphs from the datasets [30]. Our proposed method converges faster than online PG and DPG algorithms in all cases.

outperforms PG in convergence. While online DPG initially tracks the solution marginally faster, ultimately, OPADMM adapts better to the network changes and ends up converging faster than DPG.

B. Real world graphs

We also test the performance of our proposed algorithm using three datasets from the SuiteSparse Matrix Collection [30]. In particular, we use the structural engineering data mesh1e1 ($n = 48$ nodes), power network data bcspwr03 ($n = 118$ nodes), and thermal network data lshp_265 ($n = 265$ nodes). All experiments with these real graphs have been carried out by generating $p = 1000$ synthetic smooth signals corrupted with Gaussian noise ($\mu = 0$, $\sigma^2 = 0.01$). We report these results in Figure 2. It can be observed that OPADMM clearly yields a faster convergence behavior in comparison with online PG and DPG in all the performed tests. Our findings are thus fairly robust.

VI. CONCLUSIONS

In this paper, we have developed an efficient online optimization method for graph learning. Leveraging the local linear convergence properties of the PADMM and the structure of the network topology inference problem, we derive an alternating minimization algorithm (termed OPADMM) to minimize a time-varying cost in an online fashion. The proximity term in the topology updates implements a temporal-variation regularization, and we show OPADMM exhibits sublinear static regret under simplifying assumptions. Computational experiments demonstrate the effectiveness of OPADMM in stationary or dynamic settings, and that it outperforms state-of-the-art online algorithms for synthetically generated instances. Furthermore, the proposed approach exhibits robust performance across a broad range of graphs, as evidenced via tests using real world datasets.

REFERENCES

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

[2] 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.

[3] 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.

[4] G. B. Giannakis, Y. Shen, and G. V. Karanikolas, "Topology identification and learning over graphs: Accounting for nonlinearities and dynamics," vol. 106, no. 5, pp. 787–807, 2018.

[5] E. Dall'Anese, A. Simonetto, S. Becker, and L. Madden, "Optimization and learning with information streams: Time-varying algorithms and applications," *IEEE Signal Process. Mag.*, vol. 37, no. 3, pp. 71–83, 2020.

[6] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[7] E. Pavez, H. E. Egilmez, and A. Ortega, "Learning graphs with monotone topology properties and multiple connected components," *IEEE Trans. Signal Process.*, vol. 66, no. 9, 2018.

[8] J. V. de Miranda Cardoso, J. Ying, and D. Palomar, "Graphical models in heavy-tailed markets," in *Proc. Adv. Neural. Inf. Process. Syst.*, 2021, pp. 1–13.

[9] S. Segarra, A. G. Marques, G. Mateos, and A. Ribeiro, "Network topology inference from spectral templates," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 467–483, 2017.

[10] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, and M. G. Rabbat, "Characterization and inference of graph diffusion processes from observations of stationary signals," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 3, pp. 481–496, 2018.

[11] D. Thanou, X. Dong, D. Kressner, and P. Frossard, "Learning heat diffusion graphs," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 484–499, 2017.

[12] M. Wasserman, S. Sihag, G. Mateos, and A. Ribeiro, "Learning graph structure from convolutional mixtures," *Trans. Mach. Learn. Res.*, pp. 1–29, 2023.

[13] M. Navarro, S. Rey, A. Buciulea, A. G. Marques, and S. Segarra, "Joint network topology inference in the presence of hidden nodes," *IEEE Trans. Signal Process.*, vol. 72, pp. 2710–2725, 2024.

[14] 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.

[15] V. Kalofolias, "How to learn a graph from smooth signals," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2016, pp. 920–929.

[16] S. S. Saboksayr and G. Mateos, "Accelerated graph learning from smooth signals," *IEEE Signal Process. Lett.*, vol. 28, pp. 2192–2196, 2021.

[17] X. Wang, C. Yao, and A. M.-C. So, "A linearly convergent optimization framework for learning graphs from smooth signals," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 9, pp. 490–504, 2023.

[18] M. Wasserman and G. Mateos, "Graph structure learning with interpretable Bayesian neural networks," *Trans. Mach. Learn. Res.*, pp. 1–27, 2024.

[19] X. Wang, C. Yao, H. Lei, and A. M.-C. So, "An efficient alternating direction method for graph learning from smooth signals," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.*, 2021, pp. 5380–5384.

[20] S. S. Saboksayr, G. Mateos, and M. Cetin, "Online graph learning under smoothness priors," in *Proc. of European Signal Process. Conf.*, 2021, pp. 1820–1824.

[21] S. S. Saboksayr and G. Mateos, "Dual-based online learning of dynamic network topologies," in *Proc. Int. Conf. Acoustics, Speech, Signal Process.*, 2023, pp. 1–5.

[22] S. Vlaski, H. P. Maretić, R. Nassif, P. Frossard, and A. H. Sayed, "Online graph learning from sequential data," in *Proc. IEEE Data Science Workshop*, 2018, pp. 190–194.

[23] R. Shafipour and G. Mateos, "Online topology inference from streaming stationary graph signals with partial connectivity information," *Algorithms*, vol. 13, no. 9, 2020.

[24] A. Natali, E. Isufi, M. Coutino, and G. Leus, "Learning time-varying graphs from online data," *IEEE Open J. Signal Process.*, vol. 3, pp. 212–228, 2022.

[25] A. Beck, *First-Order Methods in Optimization*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2017.

[26] D. Zhou and B. Schölkopf, "A regularization framework for learning from graph data," in *ICML Workshop on Statistical Relational Learning and its Connections to Other Fields*, 2004.

[27] D. Han, D. Sun, and L. Zhang, "Linear rate convergence of the alternating direction method of multipliers for convex composite programming," *Math. Oper. Res.*, vol. 43, no. 2, pp. 622–637, 2017.

[28] J. V. d. M. Cardoso and D. P. Palomar, "Learning undirected graphs in financial markets," in *Proc. Asilomar Conf. on Signals, Systems, Computers*, 2020, pp. 741–745.

[29] H. Wang and A. Banerjee, "Online alternating direction method," in *Proc. Int. Conf. Mach. Learn.*, Madison, WI, USA, 2012, p. 1699–1706.

[30] T. A. Davis and Y. Hu, "The University of Florida sparse matrix collection," *ACM Trans. Math. Softw.*, vol. 38, no. 1, Dec. 2011.