Scalable Connected Graph Learning with Extended Frequency Ranges

Arman Lotfaslikhani, Ali Yousefi, Massoud Babaie-Zadeh, Senior Member, IEEE

***Abstract*—In this paper, the problem of inferring a graph from signals defined on a graph, commonly known as *graph learning*, is addressed with several considerations. In some cases, a connected graph is preferred, and current models perform best within a limited graph frequency range. First, connectivity constraints are imposed on a well-known graph learning algorithm by [1], which outperforms their results, even under the smoothness assumption. Additionally, evidence is provided that our algorithm retains good performance after deviating from the smoothness assumption and when handling inputs across a wider graph frequency range. Furthermore, efficient optimization algorithms for these methods are introduced, which, to the best of our knowledge, maintain the same per-iteration complexity as the fastest graph learning algorithms while delivering good performance.**

***Index Terms*—Graph Signal Processing, Graph Learning, Optimization Algorithm, Smooth Signal, ADMM.**

I. Introduction

GRAPH learning refers to inferring structures from signals that are assumed to lie on a graph, known as graph signals, when the graph is not previously available [2]. The inferred graph representation is intended to describe the relations between the different nodes, which enables applications including brain signal analysis, and image coding and compression (a list of applications can be found in [2]).

The authors of [3] have proposed a method to infer a graph from multiple observed graph signals, by assuming the signals are smooth over the desired graph. Subsequent work by [1], to enhance performance, has proposed reformulating the optimization problem of [3] and incorporating a logarithmic barrier term to encourage node connectivity in the output graph. However, this approach does not guarantee a connected graph. It should also be noted that the smoothness assumption may be restricting, and analyzing the performance of graph learning algorithms over signals with extended frequency range is necessary.

From another perspective, the graph learning method of [4] uses the frequency-domain sparsity of signals in the inferred graph to formulate an optimization problem. Also, a related approach utilizing the smoothness and graph frequency sparsity assumptions is developed in [5]. However, the computational complexity of these algorithms exceeds previously discussed techniques.

As was pointed out in [6], even algorithm complexities of $O(n^2)$ can be prohibitive for large-scale graph learning, and they considered optimizing the objective function of [1] after estimating an adequate edge pattern constraint.

In addition, to the best of our knowledge, many of the already proposed evaluation metrics (e.g. [1], [4], [5]) do not fully reflect the quality of the inferred graph. As an example, the inferred graph is often thresholded into a binary graph, where the performance is sometimes reported by tuning over different thresholds [1]. This problem is thoroughly discussed, and our proposed metrics streamline the evaluation and comparison of different algorithms.

In this paper, we first build upon the work of [1] by imposing connectivity constraints on its graph learning algorithm. This adjustment aligns our method more closely with real-world applications when connectivity is required. Moreover, by deriving efficient optimization algorithms that maintain the same per-iteration complexity as existing methods such as [1] and [3], our approach is ensured to remain computationally feasible. It is demonstrated that our algorithm complexity can be reduced in the same fashion as [6], further demonstrating the efficiency of our method. Alongside that, new insights into the class of graphs on which [1] and our algorithm work best are provided.

In summary, our contributions are as follows:

- A new graph learning cost function is proposed that outperforms widely-known graph learning algorithms.
- By introducing suitable optimization algorithms, it is shown that by using certain assumptions, the algorithm complexity per iteration remains the same as methods of [1] and [3].
- Via experimental analysis, the new graph learning model is demonstrated to retain performance over an extended frequency range, in contrast to the methods of [1] and [3].
- Analysis of several common graph learning metrics that are commonly used and discuss their shortcomings.
- Theoretical and experimental analysis of the class of graphs that state-of-the-art smoothness-based graph learning algorithms performs best on them.

The paper is organized as follows: In Section II, notation and relevant previous contributions and methodologies are reviewed briefly. Section III discusses the concept of connectivity, exploring its significance in our research and proposed cost functions. In Sections IV and V, our exact and approximate optimization algorithms are introduced, each with detailed complexity and convergence properties. In sections VI and VII, experimental results are included, showcasing the enhancements achieved in our results.

II. A Brief Review on Graph Learning

In this section, a very brief review on graph learning is presented, mainly to ensure the consistency of notations with previous works and to provide context for our proposed approach.

A. Notation

Consider a weighted and undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the node set $\mathcal{V}$ of cardinality N and edge set $\mathcal{E}$. A graph signal is defined as a function $\mathbf{x} : \mathcal{V} \to \mathbb{R}$ that assigns a scalar value to each node [7, Definition 2.6].

Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran (e-mail: armanlotfalikhani@gmail.com, 8aliyousefi@gmail.com, mbzadeh@yahoo.com).

Consider the combinatorial graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{W}$ is the weighted adjacency matrix of the graph and $\mathbf{D} = \text{diag}(\mathbf{W1})$ is the degree matrix, where $\mathbf{1} = [1, ..., 1]^T \in \mathbb{R}^N$. Denote the set of valid graph adjacency matrices considered in this paper as $\mathcal{W}_N$, identified with the following constraints:

$$\mathcal{W}_N = \left\{ \mathbf{W} \in \mathbb{R}_+^{N \times N} \;\middle|\; W_{ij} = W_{ji},\, W_{ii} = 0,\, i, j \in \mathbb{N} \right\}. \quad (1)$$

The smoothness of a graph signal $\mathbf{x} \in \mathbb{R}^N$ is a concept utilized in several graph learning methods, including [1], [3], [5]. It is based on the assumption that connected nodes have similar signal values, and can be measured using the Laplacian quadratic form

$$\mathbf{x}^T \mathbf{L} \mathbf{x} = \frac{1}{2} \sum_{i,j} W_{ij} (x_i - x_j)^2.$$

Lower values of this expression indicate the signal varies slowly at nodes connected with high weights.

Given a vector $\mathbf{a}$, $(\mathbf{a})^2, \log(\mathbf{a})$ and $\sqrt{\mathbf{a}}$ denote element-wise applications of the respective functions. For notation simplicity, the i-th column of the identity matrix is denoted as $\mathbf{e}_i$ throughout the paper, with the dimension implicit in the context.

B. A Brief Review on Previous Methods

The proposed model for learning graph structures by [3] is formulated as follows:

$$\begin{aligned} &\min \text{Tr}(\mathbf{X}^T \mathbf{L} \mathbf{X}) + \alpha \left\| \mathbf{L} \right\|_F^2 \\ &\text{s.t. } \text{Tr}(\mathbf{L}) = N,\ \mathbf{L1} = \mathbf{0}, L_{ij} = L_{ji} \leq 0\ \forall i \neq j, \end{aligned} \quad (2)$$

where $\alpha \geq 0$ controls the edge concentration of the solution. This can be reformulated in the notation used by [1]:

$$\min_{\mathbf{W} \in \mathcal{W}_N} \left\| \mathbf{W} \odot \mathbf{Z} \right\|_{1,1} + \alpha (\left\| \mathbf{W1} \right\|_2^2 + \left\| \mathbf{W} \right\|_F^2) \quad \text{s.t. } \left\| \mathbf{W} \right\|_{1,1} = N. \quad (3)$$

Here, $\mathbf{Z} \in \mathbb{R}^{N \times N}$ represents the pairwise distance matrix defined as $Z_{ij} = \left\| x_i - x_j \right\|^2$.

To ensure that each node has at least one edge connecting it to another node, the authors of [1] proposed the following model:

$$\min_{\mathbf{W} \in \mathcal{W}_N} \left\| \mathbf{W} \odot \mathbf{Z} \right\|_{1,1} - \alpha \mathbf{1}^T \log(\mathbf{W1}) + \frac{\beta}{2} \left\| \mathbf{W} \right\|_F^2. \quad (4)$$

This model can also be expressed in vector form as:

$$\min_{\mathbf{w} \in \mathbb{R}^m} 2\mathbf{z}^T \mathbf{w} - \alpha \mathbf{1}^T \log(\mathbf{Sw}) + \beta \left\| \mathbf{w} \right\|_2^2, \text{ s.t. } \mathbf{w} \succeq \mathbf{0}. \quad (5)$$

In the formulation of [1], $\mathbf{z} \in \mathbb{R}^s$, $s = \frac{N(N-1)}{2}$ is the vectorized upper-triangular part of $\mathbf{Z}$. Also, they define $\mathbf{S} \in \mathbb{R}^{N \times s}$ as a binary sparse matrix with the specific structure characterized by $\mathbf{Sw} = \mathbf{W1}$. The vector $\mathbf{w}$ captures only the non-zero elements above the main diagonal of $\mathbf{W}$, to avoid dealing with the symmetricity constraint, while also setting the diagonal elements to zero. Elements of $\mathbf{w}$ are denoted w_{ij} to correspond to W_{ij} (with $i < j$), with this indexing applied to all related variables to match the indexing of $\mathbf{W}$.

By incorporating a penalty for non-negativity in (5), as suggested by [8], a reformulation suitable for Alternating Direction Method of Multipliers (ADMM) has been derived:

$$\begin{aligned} &\min_{\mathbf{w} \in \mathbb{R}^m, \mathbf{v} \in \mathbb{R}^N} f(\mathbf{w}) + g(\mathbf{v}) \\ &\text{s.t. } \mathbf{Sw} - \mathbf{v} = \mathbf{0}, \end{aligned} \quad (6)$$

where $f(\mathbf{w}) = 2\mathbf{z}^T \mathbf{w} + \beta \left\| \mathbf{w} \right\|_2^2 + \mathbb{I}_{\{\mathbf{w} \geq 0\}}$ with

$$\mathbb{I}_{\{\mathbf{w} \geq 0\}} = \begin{cases} 0 & \mathbf{w} \geq 0, \\ +\infty & \text{otherwise}, \end{cases}$$

and $g(\mathbf{v}) = -\alpha \mathbf{1}^T \log(\mathbf{v})$.

In another relevant case, learning a connected graph has been explored [9]. It is worth noting that the work of [9] used additional simplifying assumptions on the adjacency matrix, did not provide dedicated optimization algorithms for their cost functions, and lacked comprehensive experiments.

III. Our Method

The second smallest eigenvalue of the Laplacian matrix of a graph, often referred to as the algebraic connectivity or Fiedler value [7], facilitates understanding the connectivity and structural properties of graphs.

The Fiedler value $\lambda_2(\mathbf{L})$ quantifies measures how well-connected a graph is, with a larger $\lambda_2(\mathbf{L})$ suggesting a more densely connected graph and typically a smaller diameter [10]. Also, it is well-known from spectral graph theory [11] that the number of connected components in a graph is equal to the multiplicity of its zero eigenvalue [7, Ch. 3]. It directly follows that a graph is connected if and only if its Fiedler value is larger than zero.

A. Objective and Constraints

The cost function of (4) does not ensure the continuity of the output graph. Therefore, the following cost functions are proposed, which are named in order as CGL1 (Connected Graph Learning), CGL2, and CGL3:

$$\min_{\mathbf{W} \in \mathcal{W}_N} \left\| \mathbf{Z} \odot \mathbf{W} \right\|_{1,1} + \frac{\beta}{2} \left\| \mathbf{W} \right\|_F^2 - \alpha \log(\mathbf{W1}) - \nu \lambda_2(\mathbf{L}), \quad (7)$$

$$\min_{\mathbf{W} \in \mathcal{W}_N} \left\| \mathbf{Z} \odot \mathbf{W} \right\|_{1,1} + \frac{\beta}{2} \left\| \mathbf{W} \right\|_F^2 - \alpha \log(\mathbf{W1}) - \mu \log(\lambda_2(\mathbf{L})), \quad (8)$$

$$\min_{\mathbf{W} \in \mathcal{W}_N} \left\| \mathbf{Z} \odot \mathbf{W} \right\|_{1,1} + \frac{\beta}{2} \left\| \mathbf{W} \right\|_F^2 - \alpha \log(\mathbf{W1}) \quad \text{s.t. } \lambda_2(\mathbf{L}) \geq \epsilon, \quad (9)$$

where ν, μ and ϵ are positive values regulating the linear and logarithmic connectivity terms, and the minimum connectivity constraint, respectively.

Based on experimental results, cost function (7) is demonstrated to outperform (4), even though it does not theoretically guarantee a connected output graph. Meanwhile, the last two cost functions theoretically guarantee that the output graph will always be connected due to the non-zero Fiedler value.

Proposition 1: The optimization problems of (7), (8) and (9) are convex optimization problems.

Proof: The feasible set $\mathcal{W}_N$ is convex as its constraints are affine. The Fiedler value $\lambda_2(\mathbf{L})$ can be expressed as

$$\lambda_2(\mathbf{L}) = \inf_{\substack{\mathbf{x}^T \mathbf{1} = 0 \\ \mathbf{x}^T \mathbf{x} = 1}} \mathbf{x}^T \mathbf{L} \mathbf{x} = \inf_{\substack{\mathbf{x}^T \mathbf{1} = 0 \\ \mathbf{x}^T \mathbf{x} = 1}} \sum_{i,j=1}^{N} w_{ij} (x_i - x_j)^2,$$

which is the infimum of affine functions in $\mathbf{W}$ and thus concave [12, Sec. 3.2]. The terms $\left\| \mathbf{Z} \odot \mathbf{W} \right\|_{1,1}$, $\frac{\beta}{2} \left\| \mathbf{W} \right\|_F^2$, $-\alpha \mathbf{1}^T \log(\mathbf{W1})$ and $-\mu \log(\lambda_2(\mathbf{L}))$ are proved convex by standard convexity and composition rules [12, Sec. 3.2]. ∎

B. Reducing Hyper Parameters

The computational load of the algorithm can be decreased by reducing the number of effective hyper-parameters. Consider the most general form of the cost function in our study, where f_m, g_k and h are positive homogeneous in $\mathbf{W}$, with f_m and h convex and g_k concave. Also assume that $f_m(\mathbf{W})$ are increasing over each element of $\mathbf{W}$. The optimization problem is given by

$$\begin{aligned}\min_{\mathbf{W}\in\mathcal{W}_N} \operatorname{Tr}(\mathbf{Z}^T\mathbf{W}) + \frac{\beta}{2}\|\mathbf{W}\|_F^2 - \sum_{m=1}^{M}\alpha_m \log(f_m(\mathbf{W})) \\ + \nu h(\mathbf{W}) \qquad \text{s.t. } g_k(\mathbf{W}) \geq \gamma_k.\end{aligned} \tag{10}$$

In this case, the functions $f_m(\mathbf{W})$, for $m = 1, \ldots, N$, are defined such that $\alpha_m = \alpha$, yielding

$$\sum_{m=1}^{N}\alpha_m \log(f_m(\mathbf{W})) = \alpha \mathbf{1}^T \log(\mathbf{W}\mathbf{1}). \tag{11}$$

For the problem in (8), an additional term $f_{N+1}(\mathbf{W}) = \lambda_2(\mathbf{L})$, with coefficient $\alpha_{N+1} = \alpha_c$ contributes $\alpha_c \log(\lambda_2(\mathbf{L}))$.

Proposition 2: For the optimal solution of (10) denoted as $F(\mathbf{Z}, \beta, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \nu)$, the following can be established:

$$\begin{aligned}F(\mathbf{Z}, \beta, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \nu) &= c_1 F(c_1\mathbf{Z}, c_1^2\beta, \boldsymbol{\alpha}, \frac{\boldsymbol{\gamma}}{c_1}, c_1\nu),\ c_1 > 0 \\ &= c_2 F(\mathbf{Z}, c_2\beta, \frac{\boldsymbol{\alpha}}{c_2}, \frac{\boldsymbol{\gamma}}{c_2}, \nu),\ c_2 > 0.\end{aligned}$$

Combining both for $c_1 = \frac{1}{\sqrt{\alpha_l\beta}}, c_2 = \alpha_l, \forall \alpha_l \in \boldsymbol{\alpha}$ results in

$$F(\mathbf{Z}, \beta, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \nu) = \sqrt{\frac{\alpha_l}{\beta}} F(\frac{\mathbf{Z}}{\sqrt{\alpha_l\beta}}, 1, \frac{\boldsymbol{\alpha}}{\alpha_l}, \sqrt{\frac{\beta}{\alpha_l}}\boldsymbol{\gamma}, \frac{\nu}{\sqrt{\alpha_l\beta}}). \tag{12}$$

Proof: See appendix **??**. ∎

Although our proposed cost functions have an additional hyperparameter, similar to the work by [1], Proposition 2 can be utilized to eliminate one of them in each case, resulting in the following for cost functions (7), (8), and (9), respectively:

$$F(\mathbf{Z}, \beta, [\alpha\mathbf{1}; 0], 0, \nu) = \sqrt{\frac{\alpha}{\beta}} F(\frac{\mathbf{Z}}{\sqrt{\alpha\beta}}, 1, [\mathbf{1}; 0], 0, \frac{\nu}{\sqrt{\alpha\beta}}), \tag{13}$$

$$F(\mathbf{Z}, \beta, [\alpha\mathbf{1}; \mu], 0, 0) = \sqrt{\frac{\alpha}{\beta}} F(\frac{\mathbf{Z}}{\sqrt{\alpha\beta}}, 1, [\mathbf{1}; \frac{\mu}{\alpha}], 0, 0) \tag{14}$$

$$F(\mathbf{Z}, \beta, [\alpha\mathbf{1}; 0], \epsilon, 0) = \sqrt{\frac{\alpha}{\beta}} F(\frac{\mathbf{Z}}{\sqrt{\alpha\beta}}, 1, [\mathbf{1}; 0], \epsilon\sqrt{\frac{\beta}{\alpha}}, 0). \tag{15}$$

In addition, the following theorem sets the relation between the parameters and the scale of the optimal solution of (8).

Theorem 1: Let $\theta = \frac{1}{\sqrt{\beta\alpha}}$. The optimal solution of (8) satisfies

$$\beta(\mathbf{w}^*)^T\mathbf{w}^* + \mathbf{z}^T\mathbf{w}^* = \frac{N\alpha + \mu}{2}, \tag{16}$$

$$w_{ij}^* \leq \sqrt{\frac{\alpha}{\beta}} \frac{-\theta z_{ij} + \sqrt{(\theta z_{ij})^2 + 4(1 + \frac{\mu}{2\alpha})}}{2} \leq \sqrt{\frac{\alpha + \mu/2}{\beta}}. \tag{17}$$

Proof: See appendix A-B. ∎

Corollary 1: No more than $(\alpha N + \mu)/(2\alpha + \mu) \leq N/2$ edges reach the upper bound of (17).

Proof: Rearranging the quadratic upper bound (17) gives $\beta w_{ij}^2 + z_{ij}w_{ij} \leq \alpha + \mu/2$. Combining it with (16) proves the result. ∎

IV. Exact Proximal Algorithm

A. Algorithm Derivation

One way to formulate the cost function of (7) or (8) is to introduce another equality constraint as

$$\min f(\mathbf{w}) + g(\mathbf{v}) + h(\mathbf{L}) \quad \text{s.t. } \mathbf{v} = \mathbf{S}\mathbf{w},\ \mathcal{L}(\mathbf{w}) = \mathbf{L}, \tag{18}$$

where $\mathcal{L}(\mathbf{w})$ creates the Laplacian matrix associated with graph weight vector $\mathbf{w}$. The augmented Lagrangian for (18) is

$$\begin{aligned}\mathcal{L}_t(\mathbf{w}, \mathbf{v}, \mathbf{L}; \mathbf{y}, \mathbf{H}) = f(\mathbf{w}) + g(\mathbf{v}) + h(\mathbf{L}) - \langle \mathbf{y}, \mathbf{S}\mathbf{w} - \mathbf{v}\rangle \\ - \langle \mathbf{H}, \mathcal{L}(\mathbf{w}) - \mathbf{L}\rangle + \frac{t}{2}(\|\mathbf{S}\mathbf{w} - \mathbf{v}\|_2^2 + \|\mathcal{L}(\mathbf{w}) - \mathbf{L}\|_F^2),\end{aligned} \tag{19}$$

where the explicit dependence of h on eigenvalues of $\mathbf{L}$ has been dropped for simplicity. Next, the method of [8], [13] is adapted to our cost function, using proximal gradient steps for primal variables. For notational simplicity, consider the diagonals of $\mathbf{H}$ and $\mathbf{L}$ as $\mathbf{h}_d$, $\mathbf{l}_d$ and the upper triangular elements as $-\mathbf{h}_\mathbf{w}, -\mathbf{l}_\mathbf{w}$. The steps are

$$\mathbf{w}^{k+1} = \operatorname{prox}_{\tau_1 f}(\tilde{\mathbf{w}}^{k+1}) = \max(\frac{\tilde{\mathbf{w}}^{k+1} - 2\tau_1\mathbf{z}}{2\tau_1\beta + 1}, \mathbf{0})\ [8], \tag{20}$$

$$\begin{aligned}\tilde{\mathbf{w}}^{k+1} = \mathbf{w}^k - \tau_1 t\big[\mathbf{S}^T[2\mathbf{S}\mathbf{w}^k - \mathbf{v}^k - \mathbf{l}_d^k - \frac{\mathbf{y}^k + \mathbf{h}_d^k}{t}] \\ + 2(\mathbf{w}^k - \mathbf{l}_\mathbf{w}^k - \frac{\mathbf{h}_\mathbf{w}^k}{t})\big],\end{aligned} \tag{21}$$

$$\mathbf{v}^{k+1} = \operatorname{prox}_{\tau_2 g}(\tilde{\mathbf{v}}^{k+1}) = \frac{\tilde{\mathbf{v}}^{k+1} + \sqrt{(\tilde{\mathbf{v}}^{k+1})^2 + 4\alpha\tau_2\mathbf{1}}}{2}, \tag{22}$$

$$\begin{aligned}\tilde{\mathbf{v}}^{k+1} &= \mathbf{v}^k(1 - \tau_2 t) + \tau_2 t\mathbf{S}\mathbf{w}^{k+1} - \tau_2\mathbf{y}^k \quad [8], \\ \tilde{\mathbf{L}}^{k+1} &= \mathbf{L}^k(1 - \tau_L t) - \tau_L\mathbf{H} + \tau_L t\mathcal{L}(\mathbf{w}^{k+1}), \\ \mathbf{L}^{k+1} &= \operatorname{prox}_{\tau_L h}[\tilde{\mathbf{L}}^{k+1}].\end{aligned} \tag{23}$$

The dual variables are updated via

$$\mathbf{y}^{k+1} = \mathbf{y}^k - t\left[\mathbf{S}\mathbf{w}^{k+1} - \mathbf{v}^{k+1}\right], \tag{24}$$

$$\mathbf{H}^{k+1} = \mathbf{H}^k - t\mathcal{L}\left(\mathbf{w}^{k+1}\right) + t\mathbf{L}^{k+1}, \tag{25}$$

which in vector form are

$$\begin{aligned}\mathbf{h}_d^{k+1} &= \mathbf{h}_d^k - t\left[\mathbf{S}\mathbf{w}^{k+1} - \mathbf{l}_d^{k+1}\right], \\ \mathbf{h}_\mathbf{w}^{k+1} &= \mathbf{h}_\mathbf{w}^k - t\left[\mathbf{w}^{k+1} - \mathbf{l}_\mathbf{w}^{k+1}\right].\end{aligned} \tag{26}$$

B. Laplacian Proximity Operator

The Fiedler value function is required to be convex in any $\mathbf{L}^k$ sequence, and not just the optimal solution. As the Fiedler value can be written as $\lambda_2(\mathbf{L}) = \min_{\lambda\neq 0}\boldsymbol{\lambda}(\mathbf{L})$ for all valid Laplacians, we propose to achieve convexity by forcing $\mathbf{L}\mathbf{1} = \mathbf{0}$ throughout the sequence. First, define $\tilde{h}(\mathbf{L}) = h(\mathbf{L}) + \mathbb{I}(\mathbf{L}\mathbf{1} = \mathbf{0})$ where $\mathbb{I}$ is the indicator function $\mathbb{I}(\mathbf{L}\mathbf{1} = \mathbf{0}) = \begin{cases}\infty & \mathbf{L}\mathbf{1} \neq \mathbf{0}, \\ 0 & \mathbf{L}\mathbf{1} = \mathbf{0}.\end{cases}$

Proposition 3: Suppose the Fiedler vector of the optimal solution in the dense case is unique and equal to $\mathbf{u}$. Then, the following holds for the optimal dual variable $\mathbf{H}^*$:

$$\mathbf{H}^* = \begin{cases}\frac{\mu\mathbf{u}\mathbf{u}^T}{\lambda_2} & \text{if } h(\mathbf{L}) = -\mu\log(\lambda_2(\mathbf{L})), \\ \nu\mathbf{u}\mathbf{u}^T & \text{if } h(\mathbf{L}) = -\nu\lambda_2(\mathbf{L}).\end{cases}$$

As a consequence, $\mathbf{H}^*\mathbf{1} = \mathbf{0}$.

Proof: See appendix A-C. ∎

In the view of Proposition 3, the following theorem can be reached:

Theorem 2: Suppose that for a given iteration number k_0, $\mathbf{L}^{k_0}\mathbf{1}=\mathbf{0}$, $\mathbf{H}^{k_0}\mathbf{1}=\mathbf{0}$ hold. Then $\mathbf{L}^{k}\mathbf{1}=\mathbf{0}, \mathbf{H}^{k}\mathbf{1}=\mathbf{0}$ for $k>k_0$ as well.

Proof: If the assumptions hold, $\tilde{\mathbf{L}}^{k_0+1}\mathbf{1}=\mathbf{0}$. Next, as the function $\tilde{h}(\mathbf{L})$ is always proper lower semi-continuous convex, [14, Corollary 24.65] applies and the proximity operator with respect to $\tilde{h}$ can be calculated as a function of the eigenvalues, and $\mathbf{L}^{k_0+1}\mathbf{1}=\mathbf{0}$. Finally, according to (25), $\mathbf{H}^{k_0+1}\mathbf{1}=\mathbf{0}$. In the final step, $\mathbb{I}(\mathbf{L}\mathbf{1}=\mathbf{0})$ can be effectively removed for $k>k_0$ which reverts $\tilde{h}(\mathbf{L})$ to $h(\mathbf{L})$ and concludes the proof. ∎

C. Calculating the Proximity Operator

Since $h(\mathbf{L})$ is a function of the eigenvalues of $\mathbf{L}$, according to Theorem 2 and [14, Corollary 24.65], the proximity operator in (23) can be calculated in terms of the eigenvalues alone.

Proposition 4: Assume the elements of $\mathbf{x}\in\mathbb{R}^n$ are sorted in ascending order, and the function $f(x):\mathbb{R}\to\mathbb{R}$ is non-decreasing and concave. The proximity operator for $h(\mathbf{x})=-\mu\min(f(\mathbf{x}))$, denoted as $\mathbf{y}\triangleq \text{prox}_{\tau h}(\mathbf{x})$, is unique and satisfies the following relations:

$$a\triangleq\min(\mathbf{y}),\ y_i=\begin{cases}a & i\le k\\ x_i, & i>k\end{cases},\ a<x_{k+1}$$

$$ka-\mu\tau f'(a)=\sum_{i=1}^{k}x_i,\ 1\le k\le n.$$

Proof: See appendix A-D. ∎

By substituting $f(x)=x$ and $f(x)=\log(x)$ in Proposition 4, the following corollary can be deduced:

Corollary 2: Using the assumptions of Proposition 4, the water-filling thresholds a_1, a_2 for $h_1(\mathbf{x})=-\nu\min(\mathbf{x})$, $h_2(\mathbf{x})=-\mu\min(\log(\mathbf{x}))=-\mu\log(\min(\mathbf{x}))$ satisfy

$$a_1=\frac{\sum_{i=1}^{k}x_i+\nu\tau}{k},\quad a_2=\frac{\sum_{i=1}^{k}x_i+\sqrt{(\sum_{i=1}^{k}x_i)^2+4k\mu\tau}}{2k}.$$

D. Convergence

Proposition 5: Consider a matrix $\mathbf{S}$ with norm $\|\mathbf{S}\|_2$. Then, the norm of the matrix $\tilde{\mathbf{S}}\triangleq[\mathbf{S}^T,\mathbf{S}^T,-\mathbf{I},-\mathbf{I}]^T$ satisfies $\|\tilde{\mathbf{S}}\|_2^2=2\|\mathbf{S}\|_2^2+2$.

Proof: $\tilde{\mathbf{S}}^T\tilde{\mathbf{S}}=2\mathbf{S}^T\mathbf{S}+2\mathbf{I}$, and the proposition directly follows. ∎

Adapting [8, Theorem 1] to our case, leads to:

Theorem 3: Suppose that the step sizes in Algorithm 1 satisfy

$$\tau_1<\frac{1}{t(2\|\mathbf{S}\|_2^2+2)},\ \tau_2<\frac{1}{t},\ \tau_L<\frac{1}{t}.$$

Then, the sequence $\{\mathbf{w}^k;\mathbf{v}^k;\mathbf{L}^k,\mathbf{H}^k,\mathbf{y}^k\}$ generated by Algorithm 1 converges to an optimal point for all $t>0$.

Proof: Equality conditions in (18) can be written in vector form $\tilde{\mathbf{S}}\mathbf{w}=[\mathbf{v},\,\mathbf{l}_d,\,\mathbf{l}_w,\,\mathbf{l}_w]$, where $\tilde{\mathbf{S}}$ is defined in Proposition 5. By using Proposition 5, the proof is arrived at directly. ∎

E. Practical Considerations and Final Algorithm

Using Theorem 3, by selecting a random vector $\mathbf{u}_0$ orthogonal to the vector $\mathbf{1}_N$, $\mathbf{H}$ can be initialized as $\mathbf{u}_0\mathbf{u}_0^T$. Also, after starting the weights randomly, it is easiest to set $\mathbf{L}^0=\mathcal{L}(\mathbf{w}^0)$ to meet the conditions of Theorem 2 in the first iteration.

Algorithm 1 Optimization of (7) and (8)

Input: Step sizes $\tau_1,\tau_2,\tau_L>0$, $t>0$, parameters $\alpha,\beta,\mu>0$, $\mathbf{z}\in\mathbb{R}^{N(N-1)/2}$, tolerances $\epsilon_p,\epsilon_d,\epsilon_{Ld},\epsilon_{Lw},\epsilon_{Hd},\epsilon_{Hw}>0$.
Output: Graph adjacency $\mathbf{w}\in\mathbb{R}^{N(N-1)/2}$.
Initialization : Pick a random vector $\mathbf{u}^0:(\mathbf{u}^0)^T\mathbf{1}=0$, set $\mathbf{H}^0$ as (3) using $\mathbf{u}^0$, pick random $\mathbf{w}^0$, $\mathbf{L}^0=\mathcal{L}(\mathbf{w}^0)$.
while $r_p>\epsilon_p$, $r_d>\epsilon_d$, $r_{Ld}>\epsilon_{Ld}$, $r_{Lw}>\epsilon_{Lw}$,, $r_{Hd}>\epsilon_{Hd}$, $r_{Hw}>\epsilon_{Hw}$ **do**
  $\mathbf{w}$-step (20), (21)
  $\mathbf{v}$-step (22)
  $\mathbf{L}$-step (23)
  Dual updates (24) and (25)
  Calculate residuals $r_p,r_d,r_{Ld},r_{Lw},r_{Hd},r_{Hw}$ using (27)
end while
return $\mathbf{w}$

For the convergence criteria, comparing constraint residuals with appropriate thresholds is used, which are calculated by

$$\begin{aligned}&r_p=t\|\mathbf{S}^T(\mathbf{v}^{k+1}-\mathbf{v}^k)\|_2,\ r_d=\|\mathbf{S}\mathbf{w}^k-\mathbf{v}^k\|_2,\\ &r_{Ld}=t\|\mathbf{S}^T(\mathbf{l}_d^{k+1}-\mathbf{l}_d^k)\|_2,\ r_{Lw}=t\|\mathbf{l}_{\mathbf{w}}^{k+1}-\mathbf{l}_{\mathbf{w}}^k\|_2,\\ &r_{Hd}=\|\mathbf{S}\mathbf{w}^k-\mathbf{l}_d^k\|_2,\ r_{Hw}=\|\mathbf{w}^k-\mathbf{l}_{\mathbf{w}}^k\|_2.\end{aligned}\tag{27}$$

F. Computational Complexity

The main complexity of Algorithm 1 arises from the spectral proximal step. For using the closed from without step size limits apart from Theorem 3, each iteration takes $O(n^3)$. Meanwhile, given that algorithms like the Lanczos [15] exist that calculate k eigenvalues in $O(kn^2)$ time, by using small enough τ_L, needing all eigenvalues can be avoided in the view of Proposition 4 and Corollary 2. Note that Corollary 2 also assumes only one eigenvalue may become negative for the small step expression. In practice, it can be used as an approximation of the proximal step.

V. Approximate Proximal Algorithm

A. Algorithm Derivation

Another way to formulate the cost function of (7) or (8) is to change the definition of $f(\mathbf{w})$ in [8] and add the connectivity function. To write the steps properly, differentiating $\lambda_2(\mathbf{L})$ is necessary.

Proposition 6: If $\lambda_2(\mathbf{L})$ has an algebraic multiplicity of one at any $\mathbf{w}$, its derivative with respect to the upper-diagonal weight vector $\mathbf{w}$ satisfies $\frac{\partial\lambda_2(\mathbf{L})}{\partial w_{ij}}=(u_i-u_j)^2$, where $\mathbf{u}$ is the Fiedler vector at $\mathbf{w}$.

Proof: It is known from [16] that eigenvalues are differentiable except when two of the coincide, which does not happen in practice. Also, from [16, Eq. 6.10], $\partial(\lambda_2)=\mathbf{u}^T\partial(\mathbf{L})\mathbf{u}$ where $\mathbf{u}$ satisfies $\mathbf{L}\mathbf{u}=\lambda_2\mathbf{u}$. Next, $\frac{\partial(\mathbf{L})}{\partial w_{ij}}=(\mathbf{e}_i-\mathbf{e}_j)(\mathbf{e}_i-\mathbf{e}_j)^T\Rightarrow\frac{\partial\lambda_2(\mathbf{L})}{\partial w_{ij}}=(\mathbf{u}^T(\mathbf{e}_i-\mathbf{e}_j))^2=(u_i-u_j)^2$. ∎

Now that the gradient is derived, the proximal step for $\mathbf{w}$ can be analyzed, with is formulated as

$$\begin{aligned}\mathbf{w}^{k+1}=\arg\min_{\mathbf{w}\succeq\mathbf{0}}&\frac{1}{2\tau_1}\|\mathbf{w}-\tilde{\mathbf{w}}^{k+1}\|_2^2+\beta\mathbf{w}^T\mathbf{w}+2\mathbf{w}^T\mathbf{z}\\ &-\mu\log(\lambda_2(\mathbf{L})),\end{aligned}\tag{28}$$

$$\tilde{\mathbf{w}}^{k+1}=\mathbf{w}^k-\tau_1 t\mathbf{S}^T[\mathbf{S}\mathbf{w}^k-\mathbf{v}^k-\frac{\mathbf{y}}{t}].\tag{29}$$

Algorithm 2 Optimization of (7) and (8)

Input: Step sizes $\tau_1, \tau_2, \tau_L, t$ satisfying 3, parameters α, β, μ, $\mathbf{z} \in \mathbb{R}^{N(N-1)/2}$, tolerances $\epsilon_p, \epsilon_d > 0$.
Output: Graph adjacency $\mathbf{w} \in \mathbb{R}^{N(N-1)/2}$.

```
while r_p > ε_p, r_d > ε_d do
    w-step (28), (29), (31)
    v-step (32)
    Dual updates (33)
    Calculate primal residual r_p = t‖S^T(v^{k+1} − v^k)‖_2
    Calculate dual residual r_d = ‖Sw^k − v^k‖_2
end while
return w
```

The Karush-Kuhn-Tucker (KKT) conditions [12] for (28) are

$$\begin{aligned}&(1+2\tau_1\beta)w_{ij} = \tilde{w}_{ij} - 2\tau_1 z_{ij} + \mu\frac{\tau_1}{\lambda_2}(u_i-u_j)^2 + \eta_{ij},\\ &\eta_{ij} \geq 0, \quad \eta_{ij}w_{ij} = 0.\end{aligned} \tag{30}$$

For proximal steps in (7), $\frac{1}{\lambda_2}$ needs to be removed from (30). Via projected gradient descent, (28) can be solved iteratively, using $\mathbf{w}^k$ as a warm start:

$$\begin{aligned}w_{ij}^{m+1} = \max\big(&w_{ij}^m(1-\zeta)\\ &+\zeta\big[\frac{\tilde{w}_{ij}^{k+1} - 2\tau_1 z_{ij} + \mu\tau_1\frac{(u_i^m-u_j^m)^2}{\lambda_2(\mathbf{L})^{m+1}}}{2\beta\tau_1+1}\big], 0\big)\\ &\text{If } \mathbf{w}^{m+1} \text{ satisfies } \lambda_2(\mathbf{L})^{m+1} > 0.\end{aligned} \tag{31}$$

The step size ζ can be adjusted so the constraint $\lambda_2(\mathbf{L}) > 0$ is satisfied. The rest of the algorithm is executed similar to [8], and updates to $\mathbf{v}$ and $\mathbf{y}$ will be in the following way:

$$\mathbf{v}^{k+1} = \text{prox}_{\tau_2 g}(\tilde{\mathbf{v}}^{k+1}) = \frac{\tilde{\mathbf{v}}^{k+1} + \sqrt{(\tilde{\mathbf{v}}^{k+1})^2 + 4\alpha\tau_2\mathbf{1}}}{2} \tag{32}$$

$$\tilde{\mathbf{v}}^{k+1} = \mathbf{v}^k(1-\tau_2 t) + \tau_2 t\mathbf{S}\mathbf{w}^{k+1} - \tau_2\mathbf{y}^k$$

$$\mathbf{y}^{k+1} = \mathbf{y}^k - t\left[\mathbf{S}\mathbf{w}^{k+1} - \mathbf{v}^{k+1}\right]. \tag{33}$$

B. Convergence

Assuming the proximal subproblem for $\mathbf{w}$ converges to its optimal solution, the result of [8, Theorem 1] directly applies.

Theorem 4: Suppose that the step sizes in Algorithm 2 satisfy $\tau_1 < \frac{1}{t(\|\mathbf{S}\|_2^2)}$, $\tau_2 < \frac{1}{t}$, $\tau_L < \frac{1}{t}$ and that subproblem (28) converges to its optimal solution. Then, the sequence $\{\mathbf{w}^k; \mathbf{v}^k; \mathbf{L}^k, \mathbf{H}^k, \mathbf{y}^k\}$ generated by Algorithm 2 converges to an optimal point for all $t > 0$.

C. Computational Complexity and Comparison with Algorithm 1

The main computational burden of Algorithm 2 is on (28), both in terms of determining the Fiedler value and vector, and the convergence of the subproblem steps. In practice, as the weight itself is being optimized in the overall problem, using a fixed small number of steps with a small stepsize is sufficient and works well in practice. Meanwhile, given that algorithms like the Lanczos [15] exist that calculate k eigenvalues in $O(knd)$ where d is the average number of nonzero values in a row, the Fiedler value and vector estimation take $O(\text{edges})$ time. It follows that (28) takes $O(\text{edges})$ computations, and the matrix multiplication in (33) also takes $O(\text{edges})$ computations. As was pointed out in [6], approximate algorithms can be used in $O(n\log n)$ time to select a suitable constraint set of desired edges, in which the total complexity would be $O(\text{edges}+n\log n)$. Of course, one can forgo this step and use the algorithm for the whole graph, which gives $O(n^2)$ complexity.

Algorithm 1, unlike Algorithm 2, cannot take advantage of a given sparse support set for $\mathbf{w}$, as adding structural constraints to the Laplacian proximal step removes the ability to calculate the proximal step in terms of the eigenvalues alone. This is because the cost function includes non-spectral terms as well and [14, Corollary 24.65] does not apply anymore. There is a trade-off in terms of convergence proof conditions between the two options, as the proof for Algorithm 1 is stronger and does not need assuming any subproblem routine to converge, apart from the Fiedler value and vector calculator.

VI. Experiment A

In our experiment $M = 100$ sample signals generated from a graph consisting of $N = 30$ nodes. Our results are compared to those of [1] and [3] based on specific metrics defined in the following sections. Additionally, this experiment is repeated across various types of graphs and smooth signals.

A. Signals and Graphs

Similar to [1], three distinct types of smooth signals are utilized, which are generated by filtering a base graph signal $\mathbf{x}_0$, characterized as a Gaussian independent and identically distributed (i.i.d.) signal. The filtering process is defined by

$$\mathbf{x}_f = h_{filt}(\mathbf{L})\mathbf{x} \triangleq \sum_{i=1}^{N} u_i h_{filt}(\lambda_i)u_i^T\mathbf{x} = \sum_{i=1}^{N} u_i h_{filt}(\lambda_i)\hat{\mathbf{x}}_i.$$

In this equation, $\mathbf{u}$ and $\boldsymbol{\lambda}$ represent the eigenvectors and eigenvalues of the graph Laplacian $\mathbf{L}$, respectively. The vector $\hat{\mathbf{x}} \in \mathbb{R}^N$ denotes the graph Fourier representation of $\mathbf{x}$, encapsulating its graph frequencies, where each component $\hat{\mathbf{x}}_i \in \mathbb{R}$. Low frequencies correspond to small eigenvalues, and low-pass or smooth filters are characterized by decaying functions[17].

Tikhonov: Filtering graph signals using Tikhonov regularization is equivalent to applying the filter defined by $h_{Tikhonov}(\lambda) = \frac{1}{1+\alpha\lambda}$.

In this experiment, $\alpha = 10$ is used. This approach effectively smooths the signals by attenuating high-frequency components, thereby enhancing the overall signal quality. Generative: The proposed generative model suggests that smooth signals can be generated from a colored Gaussian distribution, represented as $\mathbf{x} \sim \mathcal{N}(\bar{\mathbf{x}}, h(\mathbf{L})^2)$ where $h_{Gaussian}(\lambda) \triangleq \sqrt{\lambda^{-1}}$ if $\lambda > 0$, $h(0) = 0$, equivalent to $\bar{\mathbf{x}} = 0$.

Heat diffusion [18]: Given an initial signal $\mathbf{x}_0$, the result of heat diffusion on a graph after time t is expressed as $\mathbf{x} = e^{-\mathbf{L}t}\mathbf{x}_0$, where the corresponding filter is defined by $h_{heat}(\lambda) = e^{-t\lambda}$, for which larger values of t yield smoother signals.

For this experiment, $t = 10$ is chosen, which provided optimal performance based on previous experiments [1]. To identify the best parameters for each model, a grid search is performed. The experiment is repeated 10 times for each scenario, and the average results of the parameter values are reported that perform best across various metrics.

Connected Sensor [19], Erdős-Rényi [20] and Barabasi-Albert [21] graphs are used for testing is this part.

TABLE I: Performance on smooth signals

	Tikhonov					Heat					Generative				
	CGL1	CGL2	CGL3	base1[1]	base2[2]	CGL1	CGL2	CGL3	base1	base2	CGL1	CGL2	CGL3	base1	base2
Sensor															
W-SNR	6.62	6.75	7.08	4.95	3.94	5.67	7.61	7.92	3.97	3.70	4.47	4.72	4.43	4.00	3.46
F1-score	90.8	91.3	90.7	85.8	88.9	91.7	91.8	92.7	86.4	88.6	86.0	84.2	84.4	80.8	83.1
D-SNR	17.0	16.3	17.4	12.8	1.56	16.3	14.05	15.4	10.85	5.96	14.2	15.3	16.3	11.5	8.10
L-SNR	14.1	13.9	14.5	11.2	9.44	11.4	12.7	13.8	9.78	5.77	11.62	12.80	13.2	9.35	7.11
Erdős-Rényi															
W-SNR	3.77	3.93	3.65	1.48	1.42	3.62	3.31	3.87	1.04	1.29	2.09	2.19	2.05	1.30	1.25
F1-score	91.7	93.9	89.3	84.5	82.8	85.5	90.5	90.8	80.5	75.0	83.8	84.7	81.9	79.3	72.4
D-SNR	16.3	18.5	14.3	11.7	6.22	13.9	15.7	15.5	11.2	6.27	13.9	14.0	11.8	11.8	7.08
L-SNR	11.4	11.9	10.7	8.24	5.42	9.40	10.8	10.7	7.19	8.41	9.26	9.17	9.07	8.00	5.25
Barabási-Albert															
W-SNR	2.86	2.87	2.87	1.77	1.47	3.63	3.35	4.05	1.58	1.61	2.18	2.26	1.95	1.65	1.35
F1-score	87.2	88.9	88.6	78.3	81.4	86.1	90.5	88.8	79.1	80.6	80.2	79.2	74.8	74.2	76.6
D-SNR	15.8	13.3	15.7	11.8	5.04	13.2	11.4	13.4	10.5	6.70	11.4	10.91	12.5	10.1	5.20
L-SNR	11.2	10.1	11.5	8.90	4.63	10.8	8.90	10.7	7.80	5.73	9.01	8.19	9.30	8.17	4.29

[1] The results from base1 are derived from the work by [1].
[2] The results from base2 are based on [3].

B. Metrics

The matrix $\mathbf{W}$ is obtained as an output from our graph learning method. To enhance performance, thresholding and normalizing techniques are applied according to

$$\mathbf{W}_{\text{th}} = \begin{cases} \frac{W_{ij}}{\max(\mathbf{W})} & \text{if } W_{ij} > \frac{N}{N-1}\bar{W}_{ij} \\ 0 & \text{if } W_{ij} \leq \frac{N}{N-1}\bar{W}_{ij} \end{cases}.$$

Here, $\bar{\mathbf{W}}$ represents the mean of the matrix $\mathbf{W}$. Based on practical experiments, thresholding significantly impacts performance. It is also worth noting that several prior work [1] have used the best threshold for each graph to evaluate performance, which is not applicable to practical situations, in which the best thresholds are not known beforehand. By using a common thresholding metric for all cases, more accurate comparison between methods is aimed. To compare our results, four metrics are investigated that effectively measure the accuracy of our performance. First, the traditional F-measure (F1-score) is utilized, which is

$$F1\text{-score} = \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} = \frac{2\text{tp}}{2\text{tp} + \text{fp} + \text{fn}}.$$

Calculating the F1-score requires transforming the original and inferred graph to a binary scale, which does not indicate whether the learned graph weights are accurate. Therefore, in addition to the F1-score, we also employ three other real-valued metrics to ensure a comprehensive evaluation of performance. The first proposed metric is the W-SNR, which stands for the standard signal-to-noise ratio calculated between the normalized matrices $\mathbf{W}$. It is defined as

$$\text{W-SNR} = 10\log\big(\frac{\|\mathbf{w}_{\text{calcuated}}\|_2^2}{\|\mathbf{w}_{\text{calcuated}} - \mathbf{w}_{\text{original}}\|_2^2}\big).$$

The W-SNR metric indicates how well-defined the $\mathbf{W}$ matrix is, highlighting that a high F-score does not necessarily guarantee the accurate learning of $\mathbf{W}$ values. Similarly, L-SNR and D-SNR metrics for the Laplacian and degree matrices can be defined respectively, to assess the quality of the Laplacian matrix. More importantly, our experiments show that metrics based on the Laplacian martix are dominated by the diagonal component, and do not reflect the quality of the inferred graph edges well. As in some cases (e.g. [5]) metrics based on the Laplacian matrix was also used, these results are included to further show the metrics based on the weighted adjacency matrix are more informative of algorithm performances.

C. Results

The results indicate that our cost functions outperform those of previous works, even though signals are generated in a smooth form similar to earlier studies. While the F-scores show slight improvements, our most significant advantage lies in the W-scores, which demonstrate a marked enhancement. This indicates that the absolute values of the learned matrix $\mathbf{W}$ are more accurately captured in our approach. It is worth noting that the algorithm produces higher W-SNR on Sensor graphs, which is predictable in the light of Corollary 1, as Erdős-Rényi graphs are only binary.

VII. Experiments B and C

In this experiment, the same number of edges and signals are maintained as in the previous study while applying a heat diffusion filter to the signals. Due to the nature of this filter, as the parameter t increases, the signals become smoother. Conversely, for smaller values of t, a deviation from smoothness is observed. As in the previous section, each experiment is conducted 10 times, and the average results are calculated. In Experiment C, the signals are constructed as follows: Consider the first K eigenvectors of the graph's Laplacian matrix. The input signal is then treated as a combination of these eigenvectors, with coefficients drawn from a standard normal distribution. It is evident in this experiment that decreasing the value of K leads us closer to smoothness, while increasing it results in a loss of smoothness.

A. Results

As illustrated in Fig. 1, our cost function demonstrates its superiority despite some deviations from smoothness. This advantage arises from the fact that the cost function (4) is designed with a smoothness assumption in mind. Nonetheless, the additional terms introduced in equations (7), (8), and (9) empower the model to effectively learn and adapt to the more irregular segments of the graph.

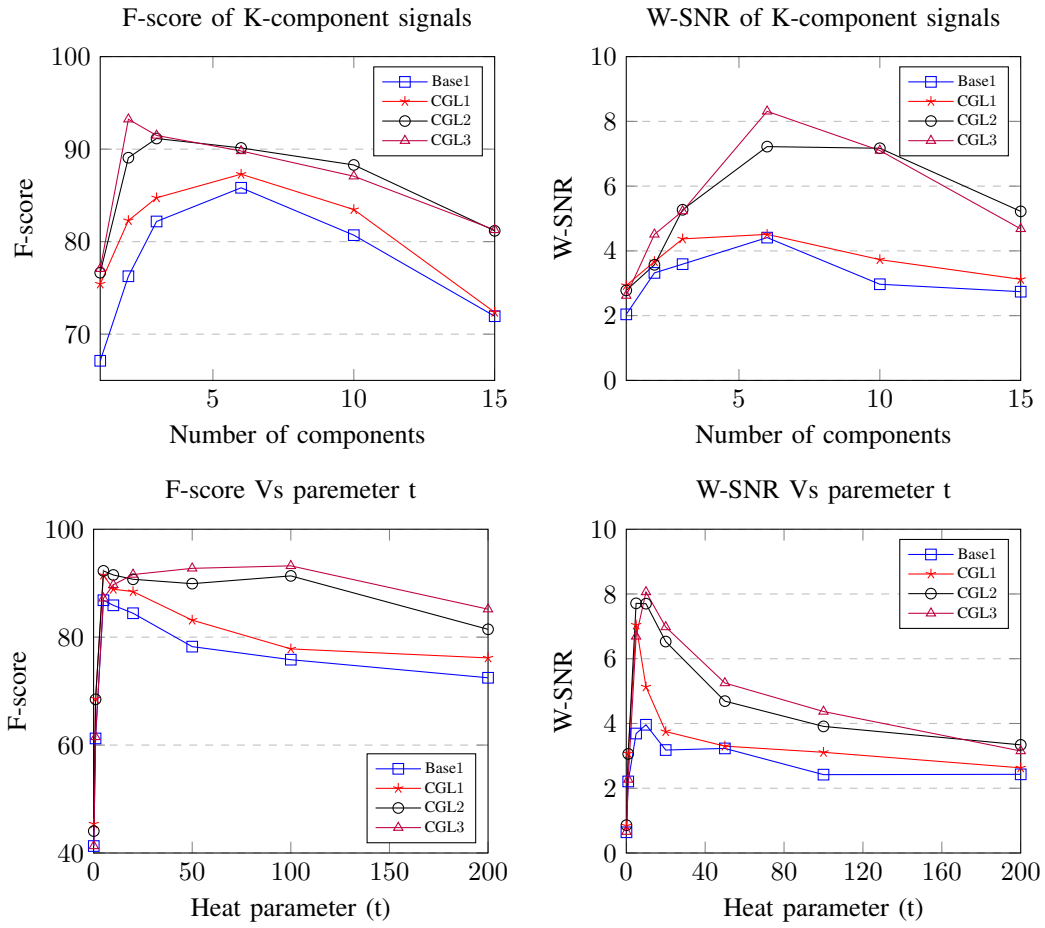

Fig. 1: The upper right subfigure depicts the W-SNR as a function of the number of graph components involved in signal creation. The upper left subfigure presents the corresponding F-score for the same number of components. The lower left subfigure shows W-SNR in relation to the parameter t in the heat diffusion filter, while the lower right panel illustrates the F-score associated with varying t values. These figures collectively highlight the impact of deviating from smoothness on graph learning performance.

VIII. Conclusion

A new set of cost functions for the problem of smoothness-based graph learning is introduced. Next, hyperparameter reduction results and ADMM-based optimization algorithms are provided. These algorithms come with proofs of convergence, and by utilizing reasonable assumptions, the per-iteration complexity of our method is demonstrated to remain the same as previous algorithms. Furthermore, the frequency behavior of our method is compared with others [1], [3], which remains adequate for a wider range. Another topic of interest included is the class of graphs this algorithm class works on best, of which the theoretical analysis was missing from the literature.

Appendix A
Theoretical Proofs

A. Proof of Proposition 2

Suppose $(\mathbf{w}, \mathbf{z})$ are vectorized form of upper triangular of matrix $(\mathbf{W}, \mathbf{Z})$ so symmetric and diagonal conditions are eliminated. The set $\mathcal{V} = \{\mathbf{w} \succeq \mathbf{0}\}$ is a cone, and $c\mathbf{w} \in \mathcal{V} \Leftrightarrow \mathbf{w} \in \mathcal{V}$, $c > 0$. Consider the following reformulated problem, with overloaded notation for the functions:

$$\begin{aligned} \underset{\mathbf{w}\in\mathcal{V}}{\arg\min}\ & 2\mathbf{w}^T\mathbf{z} + \beta\|\mathbf{w}\|_2^2 - \sum_{m=1}^{M} \alpha_m \log(f_m(\mathbf{w})) + \nu h(\mathbf{w}) \\ & \text{s.t. } g_k(\mathbf{w}) \geq \gamma_k. \end{aligned}$$

Substituting $\mathbf{w} = c\mathbf{w}'$ leads to the equivalent problem

$$\begin{aligned} \underset{\mathbf{w}'\in\mathcal{V}}{\arg\min}\ & 2c\mathbf{w}'^T\mathbf{z} + \beta c^2\|\mathbf{w}'\|_2^2 - \sum_{m=1}^{M} \alpha_m \log(f_m(\mathbf{w}')) \\ & + c\nu h(\mathbf{w}') \quad \text{s.t. } g_k(\mathbf{w}') \geq \frac{\gamma_k}{c}. \end{aligned} \tag{34}$$

Since (34) is equivalent to the original problem with scaled hyperparameters, the following result is obtained:

$$F(\mathbf{Z}, \beta, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \nu) = cF(c\mathbf{Z}, c^2\beta, \boldsymbol{\alpha}, \frac{\boldsymbol{\gamma}}{c}, c\nu) = cF(\mathbf{Z}, c\beta, \frac{\boldsymbol{\alpha}}{c}, \frac{\boldsymbol{\gamma}}{c}, \nu). \tag{35}$$

Since a constant factor c can be removed from the cost function terms, the second scaling relation is

$$F(\mathbf{Z}, \beta, \boldsymbol{\alpha}, \boldsymbol{\gamma}, \nu) = cF(\mathbf{Z}, c\beta, \frac{\boldsymbol{\alpha}}{c}, \frac{\boldsymbol{\gamma}}{c}, \nu). \tag{36}$$

B. Proof of Theorem 1

The KKT conditions for (8), introducing a slack variable $\boldsymbol{\eta}$ with the same dimension as $\mathbf{w}$, and denoting are given by

$$\begin{aligned} &2\beta w_{ij} + 2z_{ij} - \alpha(\frac{1}{d_i} + \frac{1}{d_j}) - \frac{\mu}{\lambda_2}(u_i - u_j)^2 - \eta_{ij} = 0 \\ &\eta_{ij} \geq 0,\ \eta_{ij} w_{ij} = 0, \quad \mathcal{L}(\mathbf{w})\mathbf{u} = \lambda_2 \mathbf{u}. \end{aligned} \tag{37}$$

Multiplying (37) by w_{ij} and summing over i, j, results in

$$2\beta \mathbf{w}^T\mathbf{w} + 2\mathbf{z}^T\mathbf{w} - \alpha \sum_{i<j} w_{ij}(\frac{1}{d_i} + \frac{1}{d_j}) - \frac{\mu}{\lambda_2}\sum_{i<j} w_{ij}(u_i - u_j)^2 = 0.$$

Simple calculations show that

$$\sum_{i<j} w_{ij}(\frac{1}{d_i} + \frac{1}{d_j}) = \frac{1}{2}\sum_{i,j} w_{ij}(\frac{1}{d_i} + \frac{1}{d_j}) = N$$

$$\lambda_2 = \sum_{i<j} w_{ij}(u_i - u_j)^2,$$

from which (16) follows.

For (17), an idea similar to [1] is used. For $w_{ij} > 0$, consider

$$2q_{ij} \coloneqq \frac{1}{w_{ij}}\big[\alpha(2 - \frac{w_{ij}}{d_i} - \frac{w_{ij}}{d_j}) + \mu(1 - \frac{w_{ij}(u_i - u_j)^2}{\lambda_2})\big].$$

Next, (37) simplifies to $\beta w_{ij} + z_{ij} - \frac{\alpha + \mu/2}{w_{ij}} + q_{ij} = 0$, with solution

$$w_{ij} = \frac{-(q_{ij} + z_{ij}) + \sqrt{(q_{ij} + z_{ij})^2 + 4\beta(\alpha + \mu/2)}}{2\beta}. \tag{38}$$

Using $\lambda_2 = \sum_{i<j} w_{ij}(u_i - u_j)^2 \geq w_{ij}(u_i - u_j)^2$ and $d_j, d_i \geq w_{ij}$, $q_{ij} \geq 0$ quickly follows. Since the function $\frac{-x + \sqrt{x^2+4}}{2}$ is non-increasing for $x \geq 0$, w_{ij} can be upper bounded by using $q_{ij} = 0$ in (38). The simpler bound in (17) follows by setting $z_{ij} = 0$ as well.

C. Proof of Proposition 3

For the problem in (18), the augmented Lagrangian has the optimal solution $(\mathbf{w}^*, \mathbf{v}^*, \mathbf{L}^*, \mathbf{y}^*, \mathbf{H}^*)$, with $\mathbf{L}^*$ having Fiedler vector $\mathbf{u}$ ($\mathbf{u}^T\mathbf{1} = 0$) and $\mathcal{L}(\mathbf{w}^*) = \mathbf{L}^*$, the optimality condition is $\mathbf{H}^* \in \partial h(\mathbf{L}^*)$.

For $h(\mathbf{L}) = -\mu \log(\lambda_2(\mathbf{L}))$, assuming $\lambda_2(\mathbf{L}^*)$ is simple, the gradient is

$$\nabla h(\mathbf{L}^*) = -\frac{\mu}{\lambda_2(\mathbf{L}^*)}\mathbf{u}\mathbf{u}^T, \tag{39}$$

so $\mathbf{H}^* = \frac{\mu}{\lambda_2}\mathbf{u}\mathbf{u}^T$.

For $h(\mathbf{L}) = -\nu\lambda_2(\mathbf{L})$, the subdifferential is

$$\partial h(\mathbf{L}^*) = -\nu\{\mathbf{u}\mathbf{u}^T\}, \tag{40}$$

so $\mathbf{H}^* = \nu\mathbf{u}\mathbf{u}^T$.

In both cases, $\mathbf{H}^*\mathbf{1} = c\mathbf{u}(\mathbf{u}^T\mathbf{1}) = \mathbf{0}$, where $c = \frac{\mu}{\lambda_2}$ or $c = \nu$, since $\mathbf{u}^T\mathbf{1} = 0$. Thus, the proposition holds.

D. Proof of Proposition 4

The uniqueness follows from the convexity of the function $h(\mathbf{x})$, based on convexity rules for function compositions [12], and uniqueness of proximity operators for convex functions [14, Proposition 12.15]. Now, sub-gradient rules [22, Proposition 2.3.12] can be applied to the proximal operator cost function, resulting in

$$\mathbf{0} \in \mathbf{y} - \mathbf{x} - \mu\tau f'(\min(\mathbf{y}))\,\mathrm{co}\{\sum_i -\mathbf{e}_i, \forall i : \min(\mathbf{y}) = y_i\}. \tag{41}$$

Since elements in $\mathbf{x}$ are sorted in ascending order, the set over which the convex hull is taken are the first k elements, with k to be determined. By summing over $\{i : \min(\mathbf{y}) = y_i\}$, the following relation is reached, considering $a \triangleq \min(\mathbf{y})$:

$$\sum_{i=1}^{k} y_i - x_i = \mu\tau f'(a) \Rightarrow ka - \sum_{i=1}^{k} x_i = \mu\tau f'(a).$$

REFERENCES

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

[2] 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.

[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] S. Sardellitti, S. Barbarossa, and P. Di Lorenzo, "Graph topology inference based on transform learning," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2016, pp. 356–360.

[5] P. Humbert, B. Le Bars, L. Oudre, A. Kalogeratos, and N. Vayatis, "Learning Laplacian matrix from graph signals with sparse spectral representation," *J. Mach. Learn. Res.*, vol. 22, no. 195, pp. 1–47, 2021.

[6] V. Kalofolias and N. Perraudin, "Large scale graph learning from smooth signals," in *Proc. Int. Conf. Learn. Representations*, 2019.

[7] A. Ortega, *Introduction to Graph Signal Processing*. Cambridge, U.K.: Cambridge University Press, 2022.

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

[9] M. Sundin, A. Venkitaraman, M. Jansson, and S. Chatterjee, "A connectedness constraint for learning sparse graphs," in *Proc. Eur. Signal Process. Conf.*, 2017, pp. 151–155.

[10] C. Godsil and G. Royle, *Algebraic Graph Theory*. New York, NY, USA: Springer, 2001.

[11] F. R. K. Chung, *Spectral Graph Theory*, ser. CBMS Regional Conference Series in Mathematics. Providence, RI, USA: Amer. Math. Soc., 1997, vol. 92.

[12] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, UK: Cambridge University Press, 2004.

[13] 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.

[14] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. New York, NY, USA: Springer, 2017.

[15] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Natl. Bur. Stand. B*, vol. 45, pp. 255–282, 1950.

[16] T. Kato, "Perturbation theory in a finite-dimensional space," in *Perturbation Theory for Linear Operators*. Berlin, Heidelberg, Germany: Springer, 1995, pp. 62–126.

[17] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.

[18] F. Zhang and E. R. Hancock, "Graph spectral image smoothing using the heat kernel," *Pattern Recognit.*, vol. 41, no. 11, pp. 3328–3342, 2008.

[19] N. P. Mahalik, Ed., *Sensor Networks and Configuration: Fundamentals, Standards, Platforms, and Applications*. Berlin, Heidelberg, Germany: Springer, 2007.

[20] P. Erdos and A. Renyi, "On the evolution of random graphs," *Publ. Math. Inst. Hungary. Acad. Sci.*, vol. 5, pp. 17–61, 1960.

[21] A.-L. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–12, 1999.

[22] F. H. Clarke, *Optimization and Nonsmooth Analysis*, ser. Canadian Mathematical Society Series of Monographs and Advanced Texts. New York, NY, USA: Wiley, 1983.