# An Efficient Optimization Framework for Learning General Signed Graphs from Smooth Signals

Shi-Yuk Fong
*Department of Computer Science and Engineering*
*The Chinese University of Hong Kong*
Hong Kong SAR, China
syfong1@cse.cuhk.edu.hk

Anthony Man-Cho So
*Department of Systems Engineering and Engineering Management*
*The Chinese University of Hong Kong*
Hong Kong SAR, China
manchoso@se.cuhk.edu.hk

***Abstract*—Graph learning has been widely used in many fields to study the relationships between different entities in a dataset. We present an optimization framework based on the proximal alternating direction method of multipliers (pADMM) for learning general signed graphs from smooth signals. We show that our proposed pADMM enjoys global convergence and a local linear convergence rate. Then, we demonstrate the effectiveness of the proposed framework through numerical experiments on signed graphs. Our proposed framework provides a promising approach for learning general signed graphs from smooth signals and can be a valuable tool for data analysis and decision-making in various fields.**

***Index Terms*—Signed graph, graph learning, optimization, graph signal processing, proximal ADMM, time-varying graphs**

## I. Introduction

Graph is a fundamental mathematical construct whose nodes and edges can represent objects and their relationships, respectively. We consider the scenario where each node has a signal, and the similarity (or dissimilarity) of the signals on two adjacent nodes is represented by a weight on the edge joining the two nodes. In many real-world scenarios, the signals on the nodes, referred to as the graph signals, can be observed, but the set of edges and the corresponding weights, also known as the topology of the graph, may be unknown. Therefore, graph learning, which aims to infer graph topology from graph signals, is an important problem in graph signal processing.

Most recent developments in graph learning focus only on unsigned graph settings, where the edge weights are assumed to be non-negative, such as [1]–[5]. There are some approaches that extend the discussion to signed graph settings, such as [6], which used the signed Laplacian matrix [7] to evaluate the smoothness of signals over signed graphs. This is under the assumption that for a positive (resp. negative) edge weight with a large magnitude, the signals on the two nodes connected by the edge have the same (resp. different) signs and similar magnitudes. However, such an assumption is rather restrictive. Under this assumption, an edge weight is negative if and only if the signals between the nodes connected by the edge are of different signs [6]. In particular, negative weights will not exist in the graph if the signal values are all positive.

In light of this, a more general assumption for signed graphs is introduced in [8], where the signed graph is split into two different unsigned graphs based on the sign of the edge weight. Graph signals are assumed to be smooth (i.e. low signal total variation) on the graph with positive edge weights and non-smooth (i.e., high graph signal total variation) on the graph with negative edge weights. Several optimization frameworks are built upon these assumptions, such as [8], [9]. The work [10] further extended the discussion to dynamic graphs (or time-varying graphs), where the graph structures evolve over time. Due to the process of splitting the signed graph into two unsigned graphs, complementarity constraints are introduced, resulting in a non-convex optimization framework. Although it is shown in [11] that the alternating direction method of multipliers (ADMM) possesses a global convergence guarantee when applied to solve the non-convex formualtion in [10], there is no guarantee that an optimal solution can be found.

In this paper, we build upon the earlier work [2] and propose a novel convex optimization formulation for general signed graph settings, where we combine the two unsigned weights into one variable to eliminate the complementarity constraint. It is then solved using the globally convergent pADMM algorithm. Numerical results show that our model outperforms the state-of-the-art framework [10] in terms of both model accuracy and algorithm runtime. In addition, we demonstrate that our algorithm exhibits a local linear convergence rate.

## II. Background

### A. Signed Graph Learning

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with $\mathcal{V} = [m]$ being the set of nodes and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ being the set of edges. The topology of the graph is characterized by an edge weight matrix $\boldsymbol{W} \in \mathbb{R}^{m \times m}$, where $W_{ij} \neq 0$ if and only if $(i, j) \in \mathcal{E}$. We only consider undirected graphs with no self-loop, which implies that $\boldsymbol{W}$ is symmetric and has only zero entries over the main diagonal. The difference between signed and unsigned graphs is that the former can take negative edge weights, whereas the latter cannot. A graph signal is usually represented by $\boldsymbol{x} \in \mathbb{R}^m$, whose $i$-th coordinate $x_i$ is the signal value on node $i$. The task of signed graph learning can be formulated as follows: Given a collection of $\bar{n}$ graph signals $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\bar{n}} \in \mathbb{R}^m$ from the same underlying graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ associated with unknown $\boldsymbol{W}$ and $\mathcal{E}$, learn the topology of the graph from graph signals $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\bar{n}}$.

To learn the graph topology, the relationships between edge weights and graph signals are required. We adopt the assumption in [8], where we split $\mathcal{G}$ into two unsigned graphs $\mathcal{G}^+ = (\mathcal{V}, \mathcal{E}^+)$ and $\mathcal{G}^- = (\mathcal{V}, \mathcal{E}^-)$, associated with two weight matrices $\boldsymbol{W}^+, \boldsymbol{W}^- \in \mathbb{R}_+^{m\times m}$, respectively that satisfy $\boldsymbol{W} = \boldsymbol{W}^+ - \boldsymbol{W}^-$ and

$$\mathcal{E}^s = \{(i,j) \in \mathcal{E} \mid W_{ij}^s > 0\} \text{ for } s \in \{+,-\}.$$

It is assumed that the graph signal varies smoothly (resp. nonsmoothly) accross $\mathcal{G}^+$ (resp. $\mathcal{G}^-$). The Lagragian matrices $\boldsymbol{L}^s \coloneqq \mathrm{Diag}(\boldsymbol{W}^s\mathbf{1}) - \boldsymbol{W}^s$ with $s \in \{+,-\}$ are used to define the signal smoothness. The overall smoothness of the graph signals $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\bar{n}}$ can be measured by total variation (also known as Dirichlet energy), which is evaluated as

$$\begin{aligned}\sum_{k=1}^{\bar{n}} \boldsymbol{x}_k^\top \boldsymbol{L}^s \boldsymbol{x}_k &= \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} W_{ij}^s \cdot ||\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j||_2^2 \\ &= \frac{1}{2}||\boldsymbol{W}^s \odot \boldsymbol{D}||_{1,1}. \end{aligned} \tag{1}$$

Here, $\tilde{\boldsymbol{x}}_i \coloneqq [(\boldsymbol{x}_{\mathbf{1}})_i, \ldots, (\boldsymbol{x}_{\bar{\boldsymbol{n}}})_i]^\top$ is the data vector associated with the $i$-th node and $D_{ij} \coloneqq ||\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j||_2^2$. The signals are smooth (resp. nonsmooth) if (1) is small (resp. high). Based on the above, we can formally state the assumption of smooth signal below:

**Assumption 1.** *Given the graphs $\mathcal{G}^+, \mathcal{G}^-$ and the signals $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{\bar{n}}$ on the graphs, the signals are assumed to have low total variation on $\mathcal{G}^+$ ($\frac{1}{2}||\boldsymbol{W}^+ \odot \boldsymbol{D}||_{1,1}$) and high total variation on $\mathcal{G}^-$ ($\frac{1}{2}||\boldsymbol{W}^- \odot \boldsymbol{D}||_{1,1}$).*

Note that splitting the signed weight matrix $\boldsymbol{W}$ into two unsigned weight matrices $\boldsymbol{W}^+$ and $\boldsymbol{W}^-$ will introduce complementarity constraints in the optimization formulation (see, e.g., [10]). Therefore, in actual implementation, we do not split it into two matrices, but this does not affect our developments.

### *B. Model for Learning Signed Graphs from Smooth Signals*

We introduce a novel model for learning the underlying structure of signed graph from smooth signals. The model considers general graphs, where the graph topology may vary over time. Such graphs are called dynamic graphs, or time-varying graphs. This leads to another assumption when we are dealing with dynamic graph learning problems:

**Assumption 2.** *Given dynamic graphs $\mathcal{G}_T = \{\mathcal{G}^t\}_{t=1}^T$, where $\mathcal{G}^t = (\mathcal{V}^t, \mathcal{E}^t)$ with $|\mathcal{V}^t| = m$, $\mathcal{G}^t$ and $\mathcal{G}^{t-1}$ are similar to each other, in the sense that $||\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^{(t)}||$ is assumed to be small.*

Suppose that the graph signals $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^m$ are sequentially collected from $T$ non-overlapping time slots. Within the $t$-th time slot, where $t = 1, \ldots, T$, the weight matrix $\boldsymbol{W}^{(t)} \in \mathbb{R}^{m\times m}$ of the underlying graph remains static. The $n$ graph signals are partitioned into $T$ disjoint groups and the $t$-th group $\{\boldsymbol{x}_1^{(t)}, \ldots, \boldsymbol{x}_{n_t}^{(t)}\}$ consists of signals collected in the $t$-th time slot. It follows that $\sum_{t=1}^T n_t = n$. For $t = 1, \ldots, T$, let $\boldsymbol{D}^{(t)} \in \mathbb{R}^{m\times m}$ be given by $\boldsymbol{D}_{ij}^{(t)} \coloneqq ||\tilde{\boldsymbol{x}}_i^{(t)} - \tilde{\boldsymbol{x}}_j^{(t)}||_2^2$ with $\tilde{\boldsymbol{x}}_i^{(t)} \coloneqq [(\boldsymbol{x}_1^{(t)})_i, \ldots, (\boldsymbol{x}_{n_t}^{(t)})_i]^\top$ and $\boldsymbol{D} \coloneqq [\boldsymbol{D}^{(1)}, \ldots, \boldsymbol{D}^{(T)}]$. The goal then is to infer the (possibly time-varying) graphs represented by $\Xi_T \coloneqq [\boldsymbol{W}^{(1)}, \ldots, \boldsymbol{W}^{(T)}]$. Motivated by the developments in [2], we consider the following formulation of the problem:

$$\begin{aligned} \min_{\Xi_T} \ & \mathcal{F}_T(\Xi_T) + \mathcal{H}_T(\Xi_T) + \mathcal{R}_T(\Xi_T) && (2)\\ \text{s.t.} \ & \boldsymbol{W}^{(t)} = (\boldsymbol{W}^{(t)})^\top, \ \mathrm{diag}(\boldsymbol{W}^{(t)}) = \mathbf{0}, \\ & \mathbf{1}^\top \boldsymbol{W}^{(t)} \mathbf{1} = 2\delta, \text{ for } t = 1, \ldots, T. && (\text{C1}) \end{aligned}$$

Here,

$$\mathcal{F}_T(\Xi_T) \coloneqq -\frac{1}{2}\sum_{t=1}^{T} \mathbf{1}^\top(\boldsymbol{W}^{(t)} \odot \boldsymbol{D}^{(t)})\mathbf{1}, \tag{3}$$

$$\begin{aligned}\mathcal{H}_T(\Xi_T) \coloneqq -\frac{1}{2}\sum_{t=1}^{T} \Big[&\alpha||\boldsymbol{W}^{(t)}||_{1,1} + \beta_1||\boldsymbol{W}^{(t)}||_F^2 \\ &+ \beta_2||\,\mathrm{Diag}(\boldsymbol{W}^{(t)}\mathbf{1})||_F^2\Big], \end{aligned} \tag{4}$$

$$\mathcal{R}_T(\Xi_T) \coloneqq -\frac{\gamma}{2}\sum_{t=1}^{T-1} ||\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^{(t)}||_{1,1}, \tag{5}$$

and $\alpha, \beta_1, \beta_2, \gamma > 0, \delta \in \mathbb{R}$ are given parameters. For the ease of demonstration in this paper, we let $\beta = \beta_1 = \beta_2/2 > 0$ to simplify the above formulation. The first term $\mathcal{F}_T$ evaluates the overall signal smoothness on the learned graphs $\Xi_T$. It aims to minimize the signal total variation on $\mathcal{G}^+$ and maximize the signal total variation on $\mathcal{G}^-$. The second term $\mathcal{H}_T$ consists of the $L_{1,1}$ norm and the squared Frobenius norm of the weight matrices to balance weight magnitude and sparsity of the learned weight matrices associated with the underlying graphs. The squared Frobenius norm of $\mathrm{Diag}((\boldsymbol{W}^{(t)})\mathbf{1})$ aims to ensure that the edge weights are evenly distributed among all nodes in the original signed graph. The third term $\mathcal{R}_T$ is a temporal regularization term that is based on Assumption 2. The $L_{1,1}$ norm promotes sparsity in the temporal variation of the graph. The constraint (C1) balances the density of the positive and negative weight matrices. Observe that problem (2) is convex because it consists of only convex functions in the objective and affine equalities in the constraints.

## III. Proposed Optimization Framework for Signed Graph Learning

### *A. Problem Reformulation*

Let $\boldsymbol{w}^{(t)}$ and $\boldsymbol{d}^{(t)}$ be the vectors formed by stacking the entries above the main diagonal of $\boldsymbol{W}^{(t)}$ and $\boldsymbol{D}^{(t)}$, respectively. Set $\boldsymbol{w} \coloneqq [\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(2)}, \ldots, \boldsymbol{w}^{(T)}] \in \mathbb{R}^{Tp}$ and $\boldsymbol{d} \coloneqq [\boldsymbol{d}^{(1)}, \boldsymbol{d}^{(2)}, \ldots, \boldsymbol{d}^{(T)}] \in \mathbb{R}^{Tp}$ with $p \coloneqq m(m-1)/2$. Let $\boldsymbol{B} \in \{0,1\}^{Tm\times Tp}$, $\boldsymbol{B}' \in \{0, \pm 1\}^{Tm\times Tp}$, $\boldsymbol{B}'' \in \{0,1\}^{T\times Tp}$ satisfy

$$\boldsymbol{B}\boldsymbol{w} = [\boldsymbol{W}^{(1)}\mathbf{1}; \ldots; \boldsymbol{W}^{(T)}\mathbf{1}], \tag{6}$$

$$\boldsymbol{B}'\boldsymbol{w} = \boldsymbol{w} - \boldsymbol{w}', \tag{7}$$

$$\boldsymbol{B}''\boldsymbol{w} = [\mathbf{1}^\top\boldsymbol{w}^{(1)}; \ldots; \mathbf{1}^\top\boldsymbol{w}^{(T)}], \tag{8}$$

with $\boldsymbol{w}' :- [\boldsymbol{w}^{(1)}, \boldsymbol{w}^{(1)}, \ldots, \boldsymbol{w}^{(T-1)}] \in \mathbb{R}^{Tp}$. Then, it can be verified that problem (2) can be reformulated as

$$\min_{\boldsymbol{w}\in\mathbb{R}^{Tp}} \boldsymbol{d}^\top \boldsymbol{w} + \alpha||\boldsymbol{w}||_1 + \beta(||\boldsymbol{w}||_2^2 + ||\boldsymbol{B}\boldsymbol{w}||_2^2) + \gamma||\boldsymbol{B}'\boldsymbol{w}||_1 \quad \text{s.t. } \boldsymbol{B}''\boldsymbol{w} = \delta\mathbf{1}. \tag{9}$$

By defining $\boldsymbol{v} :- [\boldsymbol{v}_1; \boldsymbol{v}_2]$ with $\boldsymbol{v}_1 \in \mathbb{R}^{Tm}, \boldsymbol{v}_2 \in \mathbb{R}^{Tp}$ and $\boldsymbol{B}\boldsymbol{w} = \boldsymbol{v}_1$, $\boldsymbol{B}'\boldsymbol{w} = \boldsymbol{v}_2$, $\boldsymbol{C} :- [\boldsymbol{B}; \boldsymbol{B}']$, $\boldsymbol{C}' :- [\boldsymbol{C}; \boldsymbol{B}'']$, problem (2) can be written as

$$\min_{\boldsymbol{w}\in\mathbb{R}^{Tp}, \boldsymbol{v}\in\mathbb{R}^{T(m+p)}} f_T(\boldsymbol{w}) + g_T(\boldsymbol{v}) \quad \text{s.t.} \quad \boldsymbol{C}'\boldsymbol{w} - [\boldsymbol{v}; \delta\mathbf{1}] = \mathbf{0}. \tag{10}$$

Here,

$$f_T(\boldsymbol{w}) :- \boldsymbol{d}^\top \boldsymbol{w} + \alpha||\boldsymbol{w}||_1 + \beta||\boldsymbol{w}||_2^2, \tag{11}$$

$$g_T(\boldsymbol{v}) :- \beta||\boldsymbol{v}_1||_2^2 + \gamma||\boldsymbol{v}_2||_1. \tag{12}$$

### *B. Algorithmic Development*

Let $\boldsymbol{\lambda} :- [\boldsymbol{\lambda}_1; \boldsymbol{\lambda}_2] \in \mathbb{R}^{T(m+p+1)}$ be the dual variable associated with the constraint $\boldsymbol{C}'\boldsymbol{w} - [\boldsymbol{v}; \delta\mathbf{1}] = \mathbf{0}$ in problem (10), where $\boldsymbol{\lambda}_1 \in \mathbb{R}^{T(m+p)}$, $\boldsymbol{\lambda}_2 \in \mathbb{R}^T$. By denoting the augmented Lagrangian function with penalty parameter $\rho > 0$ by

$$\mathcal{L}_\rho(\boldsymbol{w}, \boldsymbol{v}; \boldsymbol{\lambda}) :- f_T(\boldsymbol{w}) + g_T(\boldsymbol{v}) - \boldsymbol{\lambda}^\top(\boldsymbol{C}'\boldsymbol{w} - [\boldsymbol{v}; \delta\mathbf{1}]) + \frac{\rho}{2}||\boldsymbol{C}'\boldsymbol{w} - [\boldsymbol{v}; \delta\mathbf{1}]||_2^2,$$

our proposed algorithm, termed pADMM-SGL, updates the variables in the $k$-th iteration by

$$\boldsymbol{w}^{k+1} = \arg\min_{\boldsymbol{w}\in\mathbb{R}^{Tp}} \mathcal{L}_\rho(\boldsymbol{w}, \boldsymbol{v}^k; \boldsymbol{\lambda}^k) + \frac{1}{2}||\boldsymbol{w} - \boldsymbol{w}^k||_{\boldsymbol{G}}^2, \tag{13a}$$

$$\boldsymbol{v}^{k+1} = \arg\min_{\boldsymbol{v}\in\mathbb{R}^{T(m+p)}} \mathcal{L}_\rho(\boldsymbol{w}^{k+1}, \boldsymbol{v}; \boldsymbol{\lambda}^k) + \frac{1}{2}||\boldsymbol{v} - \boldsymbol{v}^k||_{\boldsymbol{H}}^2, \tag{13b}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k - \rho(\boldsymbol{C}\boldsymbol{w}^{k+1} - \boldsymbol{v}^{k+1}), \tag{13c}$$

where $\boldsymbol{G} \in \mathbb{R}^{Tp\times Tp}$ and $\boldsymbol{H} \in \mathbb{R}^{T(m+p)\times T(m+p)}$ can be chosen as any positive semidefinite matrices. As in [2], with carefully selected $\boldsymbol{G}$ and $\boldsymbol{H}$, we may find analytical solution to the subproblems (13a), (13b) and solve them at each iteration efficiently.

To motivate the algorithmic development below, we need to define the proximal operator. Given a proper extended real-valued function $f$, we use

$$\text{prox}_f(\boldsymbol{a}) :- \arg\min_{\boldsymbol{b}} \left\{ f(\boldsymbol{b}) + \frac{1}{2}||\boldsymbol{b} - \boldsymbol{a}||_2^2 \right\}$$

to denote its proximal mapping evaluated at $\boldsymbol{a} \in \text{dom}(f)$.

Let $\boldsymbol{G} = \boldsymbol{I}/\tau_1 - \rho\boldsymbol{C}'^\top\boldsymbol{C}'$ with $0 < \tau_1 < 1/\rho||\boldsymbol{C}'||_2^2$. Then, we can rewrite (13a) as

$$\boldsymbol{w}^{k+1} = \text{prox}_{\tau_1 f_T}\left[\boldsymbol{w}^k - \tau_1\rho\boldsymbol{C}'^\top\left(\boldsymbol{C}'\boldsymbol{w}^k - \begin{bmatrix}\boldsymbol{v}^k \\ \delta\mathbf{1}\end{bmatrix} - \frac{\lambda'^k}{\rho}\right)\right]. \tag{14}$$

Similarly, by letting $\boldsymbol{H} = (1/\tau_2 - \rho)\boldsymbol{I}$ with $0 < \tau_2 < 1/\rho$, (13b) can be rewritten as

$$\boldsymbol{v}^{k+1} = \text{prox}_{\tau_2 g_T}\left[\boldsymbol{v}^k + \tau_2\rho\left(\boldsymbol{C}\boldsymbol{w}^k - \boldsymbol{v} - \frac{\lambda'^k}{\rho}\right)\right]. \tag{15}$$

Then, we state the following two propositions, which are used to solve the proximal form subproblems (14), (15).

**Proposition 1.** *If $f_T(\boldsymbol{w}) = \boldsymbol{d}^\top\boldsymbol{w} + \alpha||\boldsymbol{w}||_1 + \beta||\boldsymbol{w}||_2^2$ for $\boldsymbol{w} \in \mathbb{R}^{Tp}$, then given any $\tau > 0$, we have*

$$\text{prox}_{\tau f_T}(\boldsymbol{w}) = \frac{1}{2\tau\beta + 1}\mathcal{S}_{\tau\alpha}(\boldsymbol{w} - \tau\boldsymbol{d}),$$

*where*

$$(\mathcal{S}_\kappa(\boldsymbol{a}))_i = \text{sgn}(a_i)\max\{(|a_i| - \kappa), 0\}$$

*is the soft-thresholding operator [12].*

**Proposition 2.** *If $g_T(\boldsymbol{v}) :- \beta||\boldsymbol{v}_1||_2^2 + \gamma||\boldsymbol{v}_2||_1$ for $\boldsymbol{v} = [\boldsymbol{v}_1; \boldsymbol{v}_2]$ with $\boldsymbol{v}_1 \in \mathbb{R}^{Tm}$ and $\boldsymbol{v}_2 \in \mathbb{R}^{Tp}$, then given any $\tau > 0$, we have*

$$\text{prox}_{\tau g_T}(\boldsymbol{v}) = \begin{bmatrix}\frac{1}{2\tau\beta + 1}\boldsymbol{v}_1 \\ \mathcal{S}_{\tau\gamma}(\boldsymbol{v}_2)\end{bmatrix}.$$

Due to the space limit, the proofs of Propositions 1 and 2 are deferred to the full version of this paper. By Propositions 1 and 2, we obtain

$$\boldsymbol{w}^{k+1} = \frac{1}{2\tau\beta + 1}\mathcal{S}_{\tau\alpha}(\tilde{\boldsymbol{w}}^{k+1} - \tau\boldsymbol{d}), \tag{16a}$$

$$\boldsymbol{v}^{k+1} = \begin{bmatrix}\frac{1}{2\tau\beta + 1}\tilde{\boldsymbol{v}}_1^{k+1} \\ \mathcal{S}_{\tau\gamma}(\tilde{\boldsymbol{v}}_2^{k+1})\end{bmatrix}, \tag{16b}$$

where

$$\tilde{\boldsymbol{w}}^{k+1} :- \frac{\boldsymbol{w}^k - \tau_1\rho\boldsymbol{C}'^\top\left(\boldsymbol{C}'\boldsymbol{w}^k - \begin{bmatrix}\boldsymbol{v}^k \\ \delta\mathbf{1}\end{bmatrix} - \frac{\boldsymbol{\lambda}^k}{\rho}\right) - \tau_1\boldsymbol{d}}{2\tau_1\beta + 1},$$

$$\tilde{\boldsymbol{v}}^{k+1} :- (1 - \tau_2\rho)\boldsymbol{v}^k + \tau_1\rho\boldsymbol{C}\boldsymbol{w}^{k+1} - \tau_2\lambda^k.$$

The algorithm for solving (10) is summarized in Algorithm 1. By adapting the analysis in [2], we can establish the global convergence and local linear convergence of Algorithm 1.

## IV. Numerical Results

### *A. Data Generation and Evaluation Metrics*

In this subsection, we introduce the methods to construct dynamic signed graphs and to generate synthetic signal data that satisfy Assumptions 1 and 2. Due to space limit, the numerical experiments regarding static graphs are deferred to the full version of this paper. We adopt the signal generation model in [10] with slight modifications. Given a signed graph $\mathcal{G}$ with $m$ nodes and its corresponding Laplacians $\boldsymbol{L}^+$ and $\boldsymbol{L}^-$, let $\boldsymbol{L}^+ = \boldsymbol{V}^+\boldsymbol{\Lambda}^+(\boldsymbol{V}^+)^\top$ and $\boldsymbol{L}^- = \boldsymbol{V}^-\boldsymbol{\Lambda}^-(\boldsymbol{V}^-)^\top$ be their eigen-decompositions. A graph signal that follows Assumption 1 can be generated by $\boldsymbol{x} = \frac{1}{2}(\boldsymbol{V}^+h_1(\boldsymbol{\Lambda}^+)(\boldsymbol{V}^+)^\top + \boldsymbol{V}^-h_2(\boldsymbol{\Lambda}^-)(\boldsymbol{V}^-)^\top)\boldsymbol{x}_0 + \boldsymbol{\epsilon}$, where $h_1(\Lambda) = \sqrt{\boldsymbol{\Lambda}^\dagger}/||\sqrt{\boldsymbol{\Lambda}^\dagger}||_F$, $h_2(\Lambda) = \sqrt{\boldsymbol{\Lambda}}/||\sqrt{\boldsymbol{\Lambda}}||_F$ are low-pass and high-pass filters,

**Algorithm 1** pADMM-SGL for problem (10)

**Input:** model parameters $\alpha, \beta > 0$, $T \geq 1$, $\gamma \geq 0$, and $\delta \in \mathbb{R}$; penalty parameter $\rho > 0$; step sizes $\tau_1, \tau_2 > 0$; tolerances $\epsilon_p, \epsilon_d > 0$;
**Initialize:** $k = 0$, randomly pick $w^0 \in \mathbb{R}^{Tp}, v^0 \in \mathbb{R}^{T(m+p)}$, and $\lambda^0 \in \mathbb{R}^{T(m+p+1)}$, and pick sufficiently large $r_p, r_d$;
**while** $r_p \geq \epsilon_p$ or $r_d \geq \epsilon_d$ **do**
    update $w$ according to (16a);
    update $v$ according to (16b);
    update $\lambda$ according to (13c);
    set primal residual $r_p \leftarrow ||\boldsymbol{C}'\boldsymbol{w}^{k+1} - [\boldsymbol{v}^{k+1}; \delta\mathbf{1}]||_2$;
    set dual residual $r_d \leftarrow \rho||\boldsymbol{C}^\top(\boldsymbol{v}^{k+1} - v^k)||_2$;
**end while**

respectively; $\boldsymbol{\Lambda}^\dagger$ is the Moore-Penrose pseudoinverse, $\boldsymbol{x}_0 \sim \mathcal{N}(0, \boldsymbol{I})$; and $\epsilon$ is an additive white Gaussian noise. For dynamic signed graph learning problem with $T \geq 2$ timeslots, we first construct the base model $\mathcal{G}^1$ by either the Erdős-Rényi (ER) model with connectivity probability set at $p_{ER} = 0.2$, or the preferential attachment (PA) model with $0.1m$ edges attached per iteration so that the edge density is around 0.1 for each sign. The corresponding edge weights are set as $W_{ij}^{(1)} \sim Uniform(0, 1)$. Half of the edges are then selected randomly and set to be negative edges. After that, we define the dynamic graph $\mathcal{G}_T = \{\mathcal{G}^t\}_{t=1}^T$, where each $\mathcal{G}^t$ is constructed by perturbing $r$ fractions of edges and resampling corresponding edge weights of $\mathcal{G}^{t-1}$ with the same distribution $Uniform(0, 1)$ so that Assumption 2 is satisifed. The numbers of positive and negative edges remain unchanged among all $\mathcal{G}^t$. Finally, $n$ graph signals $\boldsymbol{x}_1^{(t)}, \ldots, \boldsymbol{x}_n^{(t)}$ are generated from each $\mathcal{G}^t$.

We compare our proposed approach with dynSGL in [10][1] and a MATLAB toolbox CVX [13], [14][2]. Our evaluation consists of two metrics, algorithm efficiency and model accuracy. To measure algorithm efficiency, we compare the algorithm execution time to find the optimal solution for pADMM-SGL, CVX, and dynSGL. Note that the execution time for CVX at $m = 80$ and $m = 100$ is not reported, as the algorithm runs exceedingly slowly. To measure model accuracy, we employ the multiclass F1 score as defined in [10] to compare our model and dynSGL. This score is obtained by averaging the F1+ and F1- scores, which are computed by comparing the positive and negative edges of the learned graph to those of the ground truth, respectively. Note that we do not report the F1 score for CVX, as the objective function for CVX is the same as that of pADMM-SGL, and the results are expected to be the same. The hyperparameters in pADMM-SGL and dynSGL are tuned in order to make the edge density and similarity between consecutive graphs close to those of the ground truth.

We set the magnitude of variation of $\epsilon$ to be $10\%$ of the signal power, and set $r = 0.05, T = 10, n = 100$. All tests are conducted 50 times, and average values are recorded. Our code is implemented in MATLAB[3].

[1]https://github.com/SPLab-aviyente/dynSGL
[2]http://cvxr.com/cvx/

### B. Performance of the algorithm

Table I(a) shows that pADMM-SGL is generally faster than CVX and dynSGL. Table I(b) shows that pADMM-SGL is significantly more accurate than dynSGL, and such result is supported statistically by a $t$-test with significance level $p = 0.01$. Also note that our algorithm performs generally better on ER graphs than on PA graphs. This is due to $||\operatorname{Diag}((\boldsymbol{W}^{(t)})\mathbf{1})||_F^2$ in (4). Moreover, Fig. 1 indicates that pADMM-SGL exhibits a local linear convergence rate.

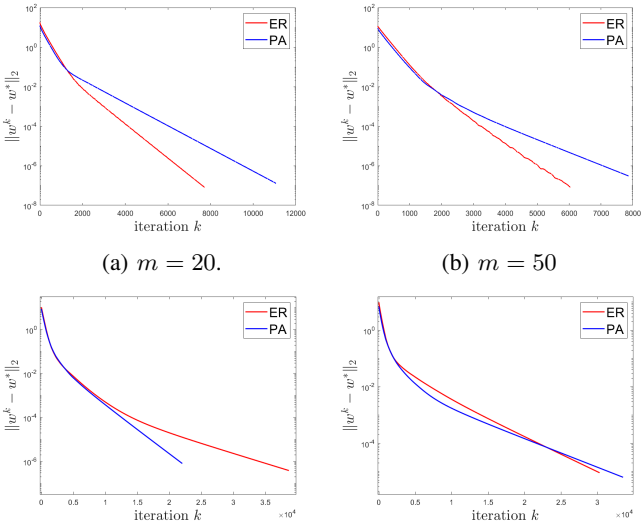
TABLE I: Performance comparison on dynamic signed graphs.

| | $m$ | CVX | dynSGL | pADMM-SGL |
|---|---|---|---|---|
| ER | 20 | 6.1889 | 2.3334 | **0.2518** |
| | 50 | 178.9360 | 4.6157 | **2.5264** |
| | 80 | – | 9.0473 | **6.3870** |
| | 100 | – | 13.0137 | **9.9193** |
| PA | 20 | 6.0456 | 2.3107 | **0.2723** |
| | 50 | 185.0518 | 4.7493 | **3.3423** |
| | 80 | – | 9.4161 | **9.3056** |
| | 100 | – | **14.0884** | 15.4484 |

I(a) Runtime comparison on synthetic graphs.

| | $m$ | dynSGL | pADMM-SGL |
|---|---|---|---|
| ER | 20 | 0.3623 | **0.5354** |
| | 50 | 0.3566 | **0.6124** |
| | 80 | 0.3400 | **0.6064** |
| | 100 | 0.3231 | **0.5881** |
| PA | 20 | 0.2957 | **0.5026** |
| | 50 | 0.2783 | **0.5411** |
| | 80 | 0.2684 | **0.5202** |
| | 100 | 0.2574 | **0.4924** |

I(b) F-score comparison on synthetic graphs.




(a) $m = 20$. (b) $m = 50$

(c) $m = 80$. (d) $m = 100$

Fig. 1: Convergence performance of dynamic signed graphs.

[3]Our code can be found at https://github.com/dmddjack/padmm_sgl.

## REFERENCES


[1] V. Kalofolias, "How to learn a graph from smooth signals," in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Gretton and C. C. Robert, Eds., vol. 51. Cadiz, Spain: PMLR, 09–11 May 2016, pp. 920–929. [Online]. Available: https://proceedings.mlr.press/v51/kalofolias16.html

[2] X. Wang, C. Yao, and A. M.-C. So, "A linearly convergent optimization framework for learning graphs from smooth signals," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 9, pp. 490–504, 2023.

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

[4] G. Fatima, A. Arora, P. Babu, and P. Stoica, "Learning sparse graphs via majorization-minimization for smooth node signals," *IEEE Signal Process. Lett.*, vol. 29, pp. 1022–1026, 2022. [Online]. Available: https://doi.org/10.1109/LSP.2022.3165468

[5] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, "Learning graphs from data: A signal representation perspective," *IEEE Signal Processing Magazine*, vol. 36, pp. 44–63, 05 2019.

[6] G. Matz and T. Dittrich, "Learning signed graphs from data," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 5570–5574.

[7] J. Kunegis, S. Schmidt, A. Lommatzsch, J. Lerner, E. W. D. Luca, and S. Albayrak, "Spectral analysis of signed graphs for clustering, prediction and visualization," in *Proceedings of the 2010 SIAM International Conference on Data Mining (SDM)*, 2010, pp. 559–570. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611972801.49

[8] A. Karaaslanli, S. Saha, S. Aviyente, and T. Maiti, "scSGL: Kernelized signed graph learning for single-cell gene regulatory network inference," *Bioinformatics*, vol. 38, no. 11, pp. 3011–3019, 04 2022. [Online]. Available: https://doi.org/10.1093/bioinformatics/btac288

[9] A. Karaaslanli, S. Saha, T. Maiti, and S. Aviyente, "Kernelized multiview signed graph learning for single-cell rna sequencing data," *BMC Bioinformatics*, vol. 24, no. 1, p. 127, Apr 2023. [Online]. Available: https://doi.org/10.1186/s12859-023-05250-y

[10] A. Karaaslanli and S. Aviyente, "Dynamic signed graph learning," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[11] Y. Wang, W. Yin, and J. Zeng, "Global convergence of ADMM in nonconvex nonsmooth optimization," *J. Sci. Comput.*, vol. 78, no. 1, p. 29–63, jan 2019. [Online]. Available: https://doi.org/10.1007/s10915-018-0757-z

[12] N. Parikh and S. Boyd, "Proximal algorithms," *Found. Trends Optim.*, vol. 1, no. 3, p. 127–239, jan 2014. [Online]. Available: https://doi.org/10.1561/2400000003

[13] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[14] ——, "Graph implementations for nonsmooth convex programs," in *Recent Advances in Learning and Control*, ser. Lecture Notes in Control and Information Sciences, V. Blondel, S. Boyd, and H. Kimura, Eds. Springer-Verlag Limited, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.