Recovery of Time Series of Graph Signals Over Dynamic Topology

Eisuke Yamagata* and Shunsuke Ono†
* Tokyo Institute of Technology, Tokyo, Japan
E-mail: yamagata.e.ab@m.titech.ac.jp
† Tokyo Institute of Technology, Tokyo, Japan
E-mail: ono@c.titech.ac.jp

***Abstract*—Conventional studies on time-varying graph signal recovery involve leveraging priors of both temporal and vertex domains for effective estimations. However, these methods all assume a static graph, in spite of the time-varying signals. We believe that such assumption, a static graph signal model, is insufficient to represent some cases where the underlying graph is explicitly dynamic. In this paper, we propose a novel recovering framework for dynamic graph signal models that leverage both temporal and vertex-domain priors. To achieve this, we introduce regularization terms in a convex optimization problem that capture behaviors of graph signals in the two domains, respectively, and integrate the dynamics of the dynamic graph topology into the formulation. We compare the proposed framework to the conventional framework through experiments on synthetic datasets to show the advantageous results of our method in numerous settings.**

I. Introduction

The concept of *graph signal*, defined by signal values observed on the vertex set V of a graph G, has been intensely researched as an approach to represent data of irregular structures. Conventional signal processing is based on spatially or temporally regular structures, e.g. images and sounds, and thus, the relations between signal values are also regular, which provides no further information for us to leverage. On the other hand, graph signal representations and graph signal processing [1]–[3] explicitly represent relations between signal values with vertices and weighed edges, which we can exploit as priors in the vertex domain. Data of irregular structures such as traffic and sensor network data, mesh data, and biomedical data all benefit from such representation.

Following the history of conventional signal processing, recovering the true graph signals from the often corrupted observations is a necessity in processing the data for further use. Many algorithms have been proposed to address the problem, such as methods based on vertex [4]–[8] and spectral domains [9]–[11]. However, these methods usually ignore the temporal domain. In real-life scenarios, many of the above-mentioned data can easily be sampled continuously to form time-coherent data, and therefore, priors based on the temporal domain should be effectively utilized to improve estimation results.

As a step forward from these algorithms, several approaches have been proposed to tackle time-varying graph signals. The earlier studies involve filtering and Fourier transform of product graphs [12] and visualization based on graph wavelet theory [13]. The more recent studies include spectral based approaches like Joint time-vertex Fourier transform (JFT) [14]–[16] and vertex-domain based approaches that leverages smoothness on the vertex and temporal domains [17]. JFT jointly applies Fourier transform to both the temporal direction and vertex direction to fully leverage priors of the two domains. In [17], the formulation is based on leveraging the smoothness of the temporal difference signal on the underlying graph, which effectively utilizes both domains for the estimation.

Although these methods succeed in improving the estimation accuracy by leveraging temporal and vertex-domain priors, they are all proposed under the assumption that the underlying graph remains unchanged despite the time-varying signals. We believe that this assumption is not adequate in modeling some real-life scenarios.

In this paper, we propose a method to tackle a novel graph signal recovery problem based on a dynamic graph signal model, a model where the underlying graph is also time-varying like the graph signals. As opposed to the static graph signal model, where the graph remains unchanged, we believe that the dynamic model is highly practical in applications such as remote sensing using moving sensors. Sensor-loaded drones, cars, smartphones, or any other mobile devices that can form a dynamic network are very feasible and realistic sensing methods.

The main contributions of this paper are as follows.

- We propose a novel problem setting (recovering a time-varying graph signals observed on dynamic graphs).
- We propose an optimization based recovery method that leverage priors of both vertex and temporal domains, and integrate dynamic graph topology into the formulation.

In the following section II, we first propose our formulation based on the dynamic graph signal model and also explain the optimization algorithm in detail. In section III, we illustrate the experimental results both quantitively and visually and discuss the obtained results.

II. Proposed Method

A. Problem Formulation

Consider the following graph signal observation model:

$$\mathbf{X} = \Phi(\overline{\mathbf{X}}) + \mathbf{n}_\sigma, \tag{1}$$

where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_p] \in \mathbb{R}^{n\times p}$ is the observation of a n nodes $\times$ p time-slots time-varying graph signal, and $\overline{\mathbf{X}} \in \mathbb{R}^{n\times p}$ is the true signal. $\Phi(\cdot)$ and $\mathbf{n}_\sigma$ are a masking operator and an additive white Gaussian noise of variance σ^2, respectively. Each graph signal $\mathbf{x}_k$ is observed on a corresponding graph Laplacian $\mathbf{L}_k$ at time $t = k$.

We propose the following optimization problem to estimate the true graph signal in the above model:

$$\begin{aligned}\min_{\mathbf{Y}\in\mathbb{R}^{n\times p}} &\sum_{k=1}^{p} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k + \lambda\, \mathrm{R}\left(\mathrm{D}\left(\mathbf{Y}\right)\right) \\ &\text{s.t.} \|\Phi(\mathbf{Y}) - \mathbf{X}\|_2 \le \varepsilon,\end{aligned} \tag{2}$$

where $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_p] \in \mathbb{R}^{n\times p}$ is the estimated graph signal, $\mathbf{L}_k \in \mathbb{R}^{n\times n}$ is the graph Laplacian at time k, and λ is the positive regularization parameter. $\|\cdot\|_F$ is the Frobenius-norm of a matrix.

The first regularization term enforces the smoothness of the graph signal $\mathbf{y}_k$ over $\mathbf{L}_k$ by penalizing the graph Laplacian quadratic form, an often-used measure of graph signal smoothness [18]. The second term penalizes the temporal difference signal:

$$\mathrm{D}\left(\mathbf{Y}\right) = [\mathbf{y}_2 - \mathbf{y}_1, \mathbf{y}_3 - \mathbf{y}_2, ..., \mathbf{y}_p - \mathbf{y}_{p-1}], \tag{3}$$

where $\mathrm{D}\left(\cdot\right)$ is a temporal difference operator, to leverage signal smoothness in the temporal domain. We consider $\mathrm{R}\left(\cdot\right)$ to be either $\|\cdot\|_1$ or $\|\cdot\|_F^2$ depending on the nature of the observed signal.

The key idea of the formulation is the integration of the time-varying graph Laplacian. Previous works on time-varying graph signal recovery all assume a static graph Laplacian, which we believe is inappropriate for applications where the underlying graph Laplacian is rapidly changing.

B. Algorithm

1) ℓ_1 Regularization: When $\mathrm{R}\left(\cdot\right)$ is $\|\cdot\|_1$, we use a primal-dual splitting method (PDS) [19] to solve (2). PDS can solve optimization problems in the form of:

$$\min_{\mathbf{u}} f_1(\mathbf{u}) + f_2(\mathbf{u}) + f_3(\mathbf{A}\mathbf{u}), \tag{4}$$

where f_1 is a differentiable convex function with the β-Lipschitzian gradient ∇f_1 for some $\beta > 0$, proximal operators for f_2 and f_3 are efficiently computable (proximable), and $\mathbf{A}$ is a matrix. The problem is solved by the algorithm:

$$\begin{aligned}\mathbf{u}^{(n+1)} &= \mathrm{prox}_{\gamma_1 f_2}\left[\mathbf{u}^{(n)} - \gamma_1(\nabla f_1(\mathbf{u}^{(n)}) + \mathbf{A}^\top \mathbf{v}^{(n)})\right], \\ \mathbf{v}^{(n+1)} &= \mathrm{prox}_{\gamma_2 f_3^*}\left[\mathbf{v}^{(n)} + \gamma_2 \mathbf{A}(2\mathbf{u}^{(n+1)} - \mathbf{u}^{(n)}\right],\end{aligned}$$

where f_3^* is the Fenchel-Rockafellar conjugate function[1] of f_3, $\gamma_1, \gamma_2 > 0$ satisfy $\frac{1}{\gamma_1} - \gamma_2\lambda_1(\mathbf{A}^\top\mathbf{A}) \ge \frac{\beta}{2}$ ($\lambda_1(\cdot)$ is the maximum eigenvalue of $\cdot$). The sequence $(\mathbf{u}^{(k)})_{k\in\mathbb{N}}$ converges to a solution of (4) under some conditions on f_2, f_3, and $\mathbf{A}$.

By vectorizing $\mathbf{X}$ and using indicator function[2] $\iota_{B_{\mathbf{x},\varepsilon}}$ where:

$$\mathbf{\Phi}\mathbf{z} \in B_{\mathbf{x},\varepsilon} := \{\mathbf{z} \in \mathbb{R}^{np} \mid \|\mathbf{z} - \mathbf{x}\|_2 \le \varepsilon\},$$

we can reformulate (2) as:

$$\begin{aligned}\min_{\mathrm{vec}\,(\mathbf{Y})\in\mathbb{R}^{n\times p}} &\sum_{k=1}^{p} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k + \lambda\|\mathbf{D}\,\mathrm{vec}\,(\mathbf{Y})\|_1 \\ &+\iota_{B_{\mathrm{vec}\,(\mathbf{X}),\varepsilon}}(\mathbf{\Phi}\,\mathrm{vec}\,(\mathbf{Y})),\end{aligned} \tag{5}$$

Matrices $\mathbf{D} \in \mathbb{R}^{n(p-1)\times np}$ and $\mathbf{\Phi} \in \mathbb{R}^{np\times np}$ are the temporal linear difference and masking operators for vectorized variables, respectively. By defining $\mathbf{u} := \mathrm{vec}\,(\mathbf{Y}) \in \mathbb{R}^{np}$, and $\mathbf{v} := [\mathbf{v}_1^\top\ \mathbf{v}_2^\top]^\top$ ($\mathbf{v}_1 \in \mathbb{R}^{p-1}, \mathbf{v}_2 \in \mathbb{R}^{np}$), and $f_1, f_2, f_3, \mathbf{A}$ as:

$$\begin{aligned} f_1(\mathbf{u}) &:= \sum_{k=1}^{p} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k, \\ f_2(\mathbf{u}) &:= 0, \\ f_3(\mathbf{v}) &:= \lambda\|\mathbf{v}_1\|_1 + \iota_{B_{\mathrm{vec}\,(\mathbf{X}),\varepsilon}}(\mathbf{v}_2), \\ \mathbf{A} &:= \begin{bmatrix}\mathbf{D} \\ \mathbf{\Phi}\end{bmatrix},\end{aligned} \tag{6}$$

(5) is reduced to (4), which can be solved by PDS.

The ∇f_1 is given by:

$$\nabla f_1(\mathrm{vec}\,(\mathbf{U})) = 2[\mathbf{L}_1\mathbf{u}_1^\top\ \mathbf{L}_2\mathbf{u}_2^\top\ ...\ \mathbf{L}_p\mathbf{u}_p^\top]^\top,$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_p]$.

Proximity operator for f_3 can be represented by proximity operators for each term. The proximity operator of $\|\cdot\|_1$ is equivalent to the soft-thresholding operation:

$$[\mathrm{prox}_{\gamma\|\cdot\|_1}(\mathbf{z})]_i := [\mathrm{ST}\,(\mathbf{z}, \gamma)]_i = \mathrm{sgn}\,(z_i)\max\{0, \|\mathbf{z}_i\| - \gamma\},$$

and the proximity operator for $\iota_{B_{\mathbf{x},\varepsilon}}$ is given by:

$$\mathrm{prox}_{\gamma\iota_{B_{\mathbf{x},\varepsilon}}}(\mathbf{z}) = P_{B_{\mathbf{x},\varepsilon}}(\mathbf{z}) = \begin{cases}\mathbf{z}, & \text{if } \mathbf{z} \in B_{\mathbf{x},\varepsilon}, \\ \mathbf{x} + \frac{\varepsilon(\mathbf{z}-\mathbf{x})}{\|\mathbf{z}-\mathbf{x}\|_2}, & \text{otherwise},\end{cases}$$

where, $P_{B_{\mathbf{x},\varepsilon}}(\mathbf{z})$ is the projection to the ℓ_2-norm ball $B_{\mathbf{x},\varepsilon}$.

2) Frobenius Regularization: Similarly, when $\mathrm{R}\left(\cdot\right)$ is $\|\cdot\|_F^2$, we also use PDS to solve (2). In this case, f_1, f_3 and $\mathbf{A}$ are redefined as follows:

$$\begin{aligned} f_1(\mathbf{u}) &:= \sum_{k=1}^{p} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k + \lambda\|\mathbf{D}\,\mathrm{vec}\,(\mathbf{Y})\|_2^2, \\ f_3(\mathbf{v}) &:= \iota_{B_{\mathrm{vec}\,(\mathbf{X}),\varepsilon}}(\mathbf{v}), \\ \mathbf{A} &:= \mathbf{\Phi},\end{aligned} \tag{7}$$

where $\mathbf{v} \in \mathbb{R}^{np}$.

The ∇f_1 is now given by:

$$\nabla f_1(\mathrm{vec}\,(\mathbf{U})) = 2[\mathbf{L}_1\mathbf{u}_1^\top\ \mathbf{L}_2\mathbf{u}_2^\top\ ...\ \mathbf{L}_p\mathbf{u}_p^\top]^\top + 2\lambda\mathbf{D}^\top\mathbf{D}\,\mathrm{vec}\,(\mathbf{U}).$$

Refer to algorithm (2) for details.

[1] The proximity operator of f^* can be stated as $\mathrm{prox}_{\gamma f^*}(\mathbf{x}) = \mathbf{x} - \gamma\,\mathrm{prox}_{\gamma^{-1}f}(\gamma^{-1}\mathbf{x})$.

[2] The indicator function of a given nonempty closed convex set C is defined by $\iota_C(\mathbf{x}) := 0$, if $\mathbf{x} \in C; \infty$, otherwise.

Algorithm 1 Algorithm for solving $\|\cdot\|_1$ version of (2)

Input: Input signal $\mathbf{X}$, graph Laplacian $\mathbf{L}_k (k = 1, 2, ..., p)$
Output: Output signal $\mathbf{Y}^{(n)}$
Initialization: $\mathbf{Y}^{(0)} = \mathbf{X} \in \mathbb{R}^{n \times p}$
while *A stopping criterion is not satisfied* **do**
  $\mathrm{vec}\,(\mathbf{Y})^{(n+1)} = \mathrm{vec}\,(\mathbf{Y})^{(n)} - \gamma_1(\nabla f_1(\mathrm{vec}\,(\mathbf{Y})^{(n)}) + \mathbf{D}^\top \mathbf{v}_1 + \mathbf{\Phi}^\top \mathbf{v}_2)$
  $\mathbf{v_1}^{(n)} \leftarrow \mathbf{v_1}^{(n)} + \gamma_2 \mathbf{D}(2\,\mathrm{vec}\,(\mathbf{Y})^{(n+1)} - \mathrm{vec}\,(\mathbf{Y})^{(n)})$
  $\mathbf{v_2}^{(n)} \leftarrow \mathbf{v_2}^{(n)} + \gamma_2 \mathbf{\Phi}(2\,\mathrm{vec}\,(\mathbf{Y})^{(n+1)} - \mathrm{vec}\,(\mathbf{Y})^{(n)})$
  $\mathbf{v}_1^{(n+1)} = \mathbf{v}_1^{(n)} - \gamma_2\,\mathrm{ST}\,(\frac{1}{\gamma_2}\mathbf{v}_1^{(n)}, \frac{\lambda}{\gamma_2})$
  $\mathbf{v}_2^{(n+1)} = \mathbf{v}_2^{(n)} + \gamma_2 P_{B_{\mathrm{vec}\,(\mathbf{X}),\varepsilon}}(\frac{1}{\gamma_2}\mathbf{v}_2^{(n)})$
  $n \leftarrow n + 1$
end while
return $\mathbf{Y}^{(n)}$

Algorithm 2 Algorithm for solving $\|\cdot\|_F^2$ version of (2)

Input: Input signal $\mathbf{X}$, graph Laplacian $\mathbf{L}_k (k = 1, 2, ..., p)$
Output: Output signal $\mathbf{Y}^{(n)}$
Initialization: $\mathbf{Y}^{(0)} = \mathbf{X} \in \mathbb{R}^{n \times p}$
while *A stopping criterion is not satisfied* **do**
  $\mathrm{vec}\,(\mathbf{Y})^{(n+1)} = \mathrm{vec}\,(\mathbf{Y})^{(n)} - \gamma_1(\nabla f_1(\mathrm{vec}\,(\mathbf{Y})^{(n)}) + \mathbf{\Phi}^\top \mathbf{v})$
  $\mathbf{v}^{(n)} = \mathbf{v}^{(n)} + \gamma_2 \mathbf{\Phi}(2\,\mathrm{vec}\,(\mathbf{Y})^{(n+1)} - \mathrm{vec}\,(\mathbf{Y})^{(n)})$
  $\mathbf{v}^{(n+1)} = \mathbf{v}^{(n)} + \gamma_2 P_{B_{\mathrm{vec}\,(\mathbf{X}),\varepsilon}}(\frac{1}{\gamma_2}\mathbf{v}^{(n)})$
  $n \leftarrow n + 1$
end while
return $\mathbf{Y}^{(n)}$

III. Experiments

A. Experimental Settings

In the experiments, we evaluate our method on a synthetic dataset. Regarding the implementation for both Algorithm 1 and 2, the stopping criterion is set to $\|\mathbf{y}^{(k)} - \mathbf{y}^{(k-1)}\| \leq 1.0 \times 10^{-5}$. The regularization parameter λ is tuned prior to the following experiments. The performance of the methods is measured in $\mathrm{PSNR\,[dB]} = 20 \times \log_{10} \frac{\mathrm{MAX}_{\overline{\mathbf{X}}}}{\|\overline{\mathbf{X}} - \mathbf{Y}\|_F}, (\mathrm{MAX}_{\overline{\mathbf{X}}} = 1)$.

We compare the proposed methods (Methods (H) and (J) in table I) to other methods (Methods (A) $\sim$ (J)). For the temporal domain, we use either $\|\,\mathrm{D}\,(\mathbf{Y})\|_1$ or $\|\,\mathrm{D}\,(\mathbf{Y})\|_F^2$, and for the vertex domain, we use $\mathrm{Tr}\,(\mathbf{Y}^\top \mathbf{L}\mathbf{Y})$ or $\sum_{k=1}^{p-1} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k$. The trace of a matrix is denoted by $\mathrm{Tr}\,(\cdot)$. For $\mathrm{Tr}\,(\mathbf{Y}^\top \mathbf{L}\mathbf{Y})$, where it uses only one graph Laplacian, we input $\mathbf{L}_1$, the first graph constructed at $t = 1$.

Method (E) is a method proposed in [17], which enforces the smoothness of the temporal difference signal $\mathrm{D}\,(\mathbf{Y})$ over the graph by penalizing $\mathrm{Tr}\,((\mathrm{D}\,(\mathbf{Y}))^\top \mathbf{L}(\mathrm{D}\,(\mathbf{Y})))$. It leverages priors of both vertex and temporal domains but is based on a static graph model. Thus, to evaluate the effectiveness of the dynamic graph model, we also reimplement Method (E) by altering the regularization term to $\sum_{k=1}^{p-1}(\mathbf{y}_{k+1} - \mathbf{y}_k))^\top \mathbf{L}_k(\mathbf{y}_{k+1} - \mathbf{y}_k)$ (Method (F)). In Method (F), the smoothness term is considered separately for each time-slot, thus can leverage the dynamics of the time-varying graph Laplacian.

TABLE I
The methods

Method	Regularization	
	Vertex domain	Temporal domain
(A)	-	$\|\,\mathrm{D}\,(\mathbf{Y})\|_1$
(B)	-	$\|\,\mathrm{D}\,(\mathbf{Y})\|_F^2$
(C) [18]	$\mathrm{Tr}\,(\mathbf{Y}^\top \mathbf{L}\mathbf{Y})$	-
(D)	$\sum_{k=1}^{p-1} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k$	-
(E) [17]	$\mathrm{Tr}\,((\mathrm{D}\,(\mathbf{Y}))^\top \mathbf{L}(\mathrm{D}\,(\mathbf{Y})))$	
(F)	$\sum_{k=1}^{p-1}(\mathbf{y}_{k+1} - \mathbf{y}_k)^\top \mathbf{L}_k(\mathbf{y}_{k+1} - \mathbf{y}_k)$	
(G)	$\mathrm{Tr}\,(\mathbf{Y}^\top \mathbf{L}\mathbf{Y})$	$\|\,\mathrm{D}\,(\mathbf{Y})\|_1$
(H) Ours	$\sum_{k=1}^{p-1} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k$	$\|\,\mathrm{D}\,(\mathbf{Y})\|_1$
(I)	$\mathrm{Tr}\,(\mathbf{Y}^\top \mathbf{L}\mathbf{Y})$	$\|\,\mathrm{D}\,(\mathbf{Y})\|_F^2$
(J) Ours	$\sum_{k=1}^{p-1} \mathbf{y}_k^\top \mathbf{L}_k \mathbf{y}_k$	$\|\,\mathrm{D}\,(\mathbf{Y})\|_F^2$

Note that although the original implementation of [17] enforces signal fidelity by an additive regularization, Method (E) and Method (F) are enforced by a fidelity constraint $\|\Phi(\mathbf{Y}) - \mathbf{X}\|_2 \leq \varepsilon$, like the rest of the methods (A) $\sim$ (J). This is to avoid parameter tuning in cases where the noise intensity is known a priori, and in our experiments, ε is set to $\varepsilon = 0.9\sigma\sqrt{n}$. All of the methods are implemented by either PDS or FISTA [20] depending on whether the formulation includes a non-differentiable term or not.

B. Synthetic Dataset

To test the methods on a dynamic graph signal model, we designed a dataset that simulates observations from moving sensors. In a 2D-plane, 64 vertices are generated randomly from a uniform distribution. Each vertex represents a sensor, which observes a signal value give by an underlying smooth distribution, in this case, a combination of several multivariate normal distributions (Fig. 1 (a)). The signal values are calculated by inputting the coordinates of the points to the distribution. The graph is constructed by k-nearest neighbors algorithm, where $k = 4$ in our implementation, to capture the geographical adjacency of the vertices. The vertices move across the 2D-plane at random degrees and a pre-set velocity v, and the signal values and the graph Laplacian are calculated at each time-slot to generate a 64×100 time-varying graph signal $\overline{\mathbf{X}}$ and the corresponding graph Laplacians. After $\overline{\mathbf{X}}$ is scaled to $[0, 1]$, it is masked by $\Phi(\cdot)$ at a masking probability of P, and corrupted with an additive white Gaussian noise of variance σ^2 : $\mathbf{n}_\sigma$. We also constructed a piece-wise flat alternative (Fig. 1 (b)).

C. Results and Discussion

1) Dynamic vs Static Graph Laplacian: From table II, the formulations integrated with a time-varying graph Laplacian (Methods (D), (F), (H), (J)) performed better than their counterparts that only consider a static graph (Methods (C), (E), (G), (I)). Especially looking at Methods (C) and (D), a pure comparison between static and dynamic graph based regularization, the dynamic regularization is almost always

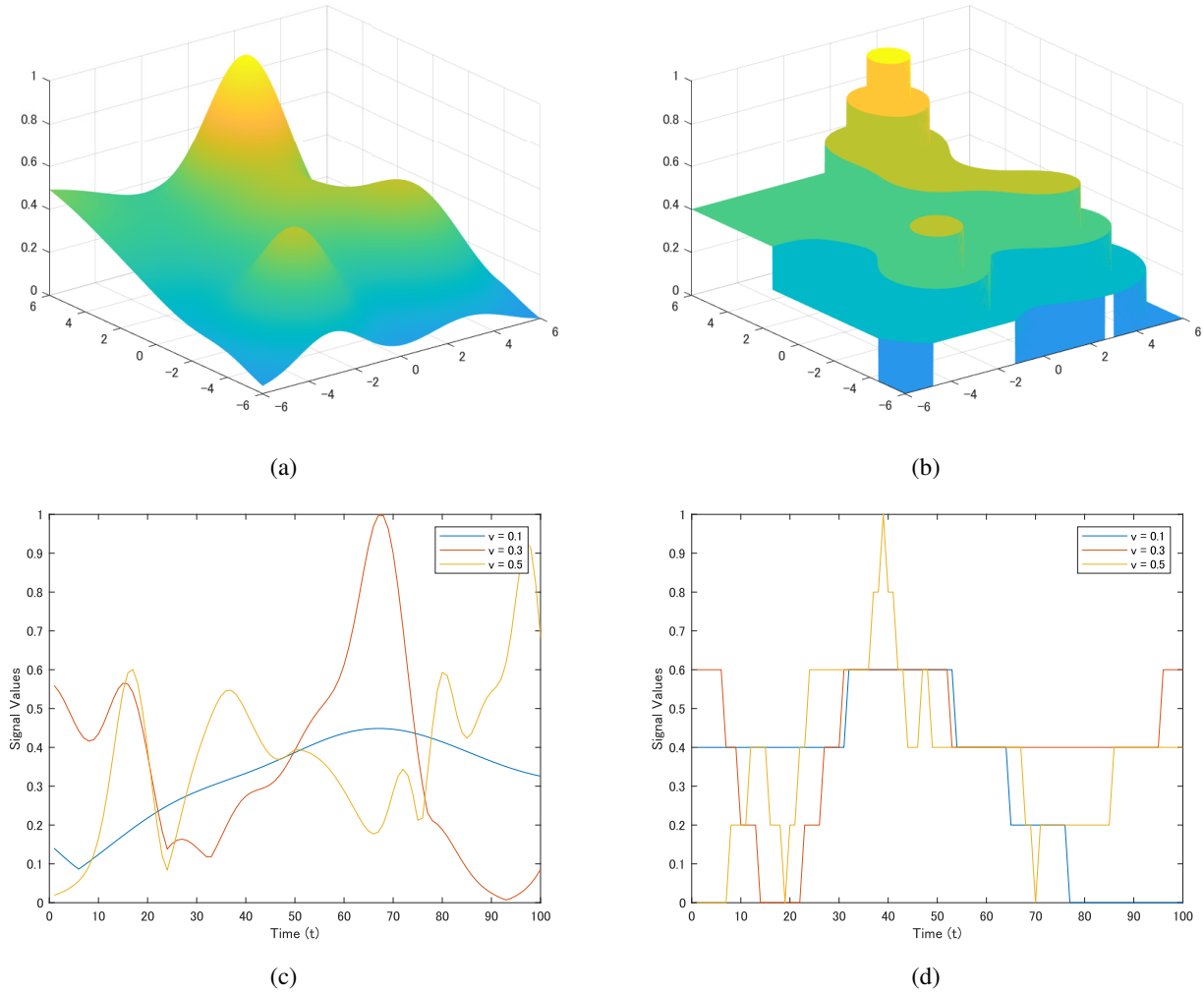

Fig. 1. (a): The smooth distribution constructed by a combination of several multivariate normal distributions. (b): A piece-wise flat version of (a). (c): A time-varying signal values of a random vertex, observed by moving across (a) at velocity v. (d): A time-varying signal values of a random vertex, observed by moving across (c) at velocity v.

better than the static counterpart by $2 \sim 3$dB. In fact, comparing Methods (G) and (I) to Methods (A) and (B), the static based vertex domain even seems to be hindering the overall recovery performance. Using $\mathbf{L}_1$, an inaccurate representation of the graph at time t, for every time-slot should clearly be discouraged. Furthermore, performance gaps between the static and dynamic graph based methods get larger as the velocity gets larger. This is most likely because the faster the points are moving, the more different graph gets as the time progresses, which enlarges the difference between $\mathbf{L}_1$ and $\mathbf{L}_t$ at time t. Thus, a larger velocity leads to a more inaccurate graph Laplacian being used at each time-slot, resulting in worse performance for the static methods.

2) Vertex vs Temporal Domain Priors: Comparing Methods (A) and (B) to Methods (C) and (D), a comparison between vertex and temporal domain regularizations, leveraging the temporal prior is much more effective than leveraging the vertex-domain prior. This can also be observed from the relatively small performance gains between Methods (A), (B) and Methods (H), (J). However, Methods (A) and (B) rely on the assumption that the time-varying graph signals are smooth in the temporal domain. Thus, as the velocity gets higher and the signals get less smooth (refer Fig. 1: (c) and (d)), Methods (A) and (B) drops its performance, whereas Methods (C) and (D) remain unchanged. Likewise, the performance differences between Methods (H), (J) and (A), (B) get larger as the velocity gets higher. This indicates how the proposed Methods (H) and (J) are robust to various types of graph signals.

3) Smooth and Piece-wise Flat Signals: For the methods that leverage the temporal prior, we expected methods that use $\|\mathrm{D}(\mathbf{Y})\|_F^2$ to perform better on the smooth dataset, and the methods that use $\|\mathrm{D}(\mathbf{Y})\|_1$ to perform better on piece-wise flat dataset. However, for some cases with higher noise intensities and higher velocities, $\|\mathrm{D}(\mathbf{Y})\|_F^2$ performed better even for the piece-wise flat dataset. Looking at how methods performed as expected in lower noise intensity settings, we

TABLE II
THE AVERAGE RECOVERY PERFORMANCE MEASURED IN PSNR [dB]

Method	Noise σ	Smooth			Piece-wise Flat		
		$P = 0$					
		$v = 0.1$	$v = 0.3$	$v = 0.5$	$v = 0.1$	$v = 0.3$	$v = 0.5$
(A)	0.1	28.81	25.23	23.63	27.92	24.72	23.31
(B)		30.38	26.61	24.66	26.95	24.65	23.54
(C) [18]		19.74	19.48	19.46	19.52	19.37	19.38
(D)		22.35	22.40	22.35	21.76	21.75	21.79
(E) [17]		26.84	23.78	22.61	24.94	22.94	22.04
(F)		26.63	23.92	22.74	24.93	22.92	22.09
(G)		29.23	25.36	23.69	27.78	24.71	23.32
(H) Ours		30.30	26.13	24.79	**28.00**	**25.19**	**24.10**
(I)		29.94	26.52	24.64	26.90	24.63	23.53
(J) Ours		**30.62**	**27.12**	**25.38**	27.07	24.92	24.02
Observ.		19.98	20.00	20.00	19.99	20.00	19.99
(A)	0.2	24.37	21.73	20.16	23.11	20.81	19.49
(B)		25.23	22.69	20.89	23.43	21.43	20.07
(C) [18]		16.10	15.84	15.70	15.80	15.51	15.45
(D)		18.64	18.73	18.69	18.19	18.16	18.20
(E) [17]		22.39	19.86	18.57	21.24	19.10	18.04
(F)		22.22	19.95	18.80	21.22	19.22	18.31
(G)		24.46	21.81	20.24	23.15	20.87	19.55
(H) Ours		25.28	23.22	21.92	23.70	21.90	20.92
(I)		25.13	22.66	20.90	23.42	21.43	20.08
(J) Ours		**26.00**	**23.85**	**22.32**	**23.93**	**22.23**	**21.19**
Ovserv.		13.96	14.01	13.98	13.97	13.96	13.96

TABLE III
THE AVERAGE RECOVERY PERFORMANCE MEASURED IN PSNR [dB] WITH THE MASKED OBSERVATIONS

Method	Noise σ	Smooth			Piece-wise Flat		
		$P = 0$	$P = 0.25$	$P = 0.5$	$P = 0$	$P = 0.25$	$P = 0.5$
		$v = 0.3$					
(A)	0.1	25.23	22.48	19.20	24.72	21.91	18.72
(B)		26.61	23.45	19.74	24.65	22.55	19.27
(C) [18]		19.48	16.57	14.87	19.37	16.12	14.22
(D)		22.40	20.06	17.50	21.75	19.52	16.92
(E) [17]		23.78	21.38	18.39	22.94	20.65	17.81
(F)		23.92	21.36	17.66	22.92	20.69	17.24
(G)		25.36	21.57	18.77	24.71	21.02	18.17
(H) Ours		26.13	23.36	20.05	**25.19**	22.71	19.54
(I)		26.52	23.40	19.75	24.63	22.52	19.26
(J) Ours		**27.12**	**24.07**	**20.43**	24.92	**23.03**	**19.88**
Observ.		20.00	13.10	10.50	20.00	12.87	10.33
(A)	0.2	21.73	18.76	15.34	20.81	18.43	15.00
(B)		22.69	19.34	15.43	21.43	19.03	15.14
(C) [18]		15.84	14.69	14.16	15.51	14.29	13.48
(D)		18.73	16.93	14.39	18.16	16.55	14.16
(E) [17]		19.86	17.64	14.41	19.10	17.28	14.26
(F)		19.95	17.18	13.37	19.22	17.04	13.26
(G)		21.81	18.39	15.41	20.87	17.94	15.05
(H) Ours		23.22	19.94	15.75	21.90	19.55	15.55
(I)		22.66	19.37	15.51	21.43	19.04	15.22
(J) Ours		**28.85**	**20.34**	**15.82**	**22.23**	**19.90**	**15.63**
Observ.		14.01	11.45	9.79	13.96	11.30	9.70

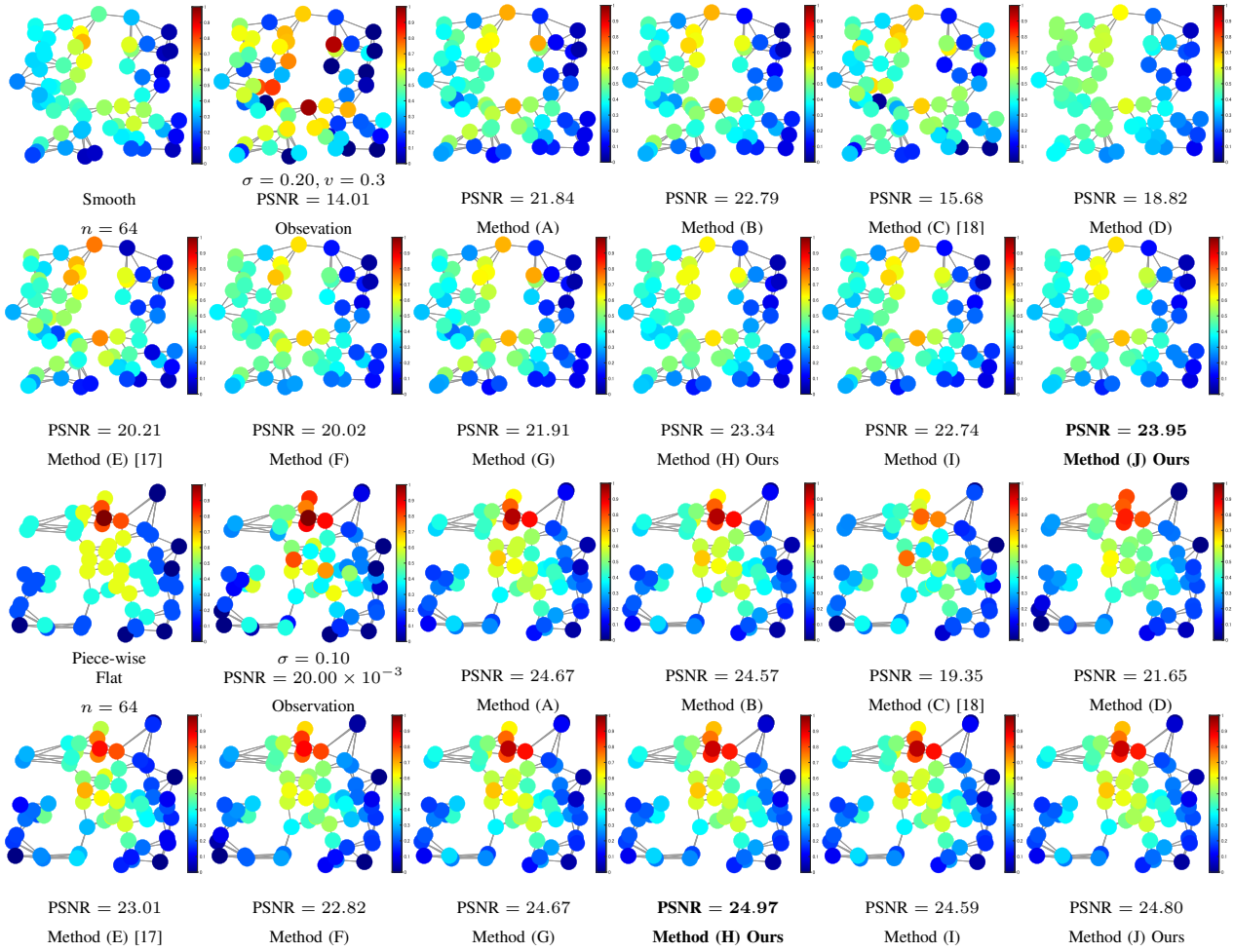

Fig. 2. Graph signal recovery results. Note that the colar range for the observations are clipped to [0,1] for a better illustrative comparison. The actual observations are noisier than the figures show. The top two and bottom two rows are the results on the smooth and piece-wise flat datasets, respectively.

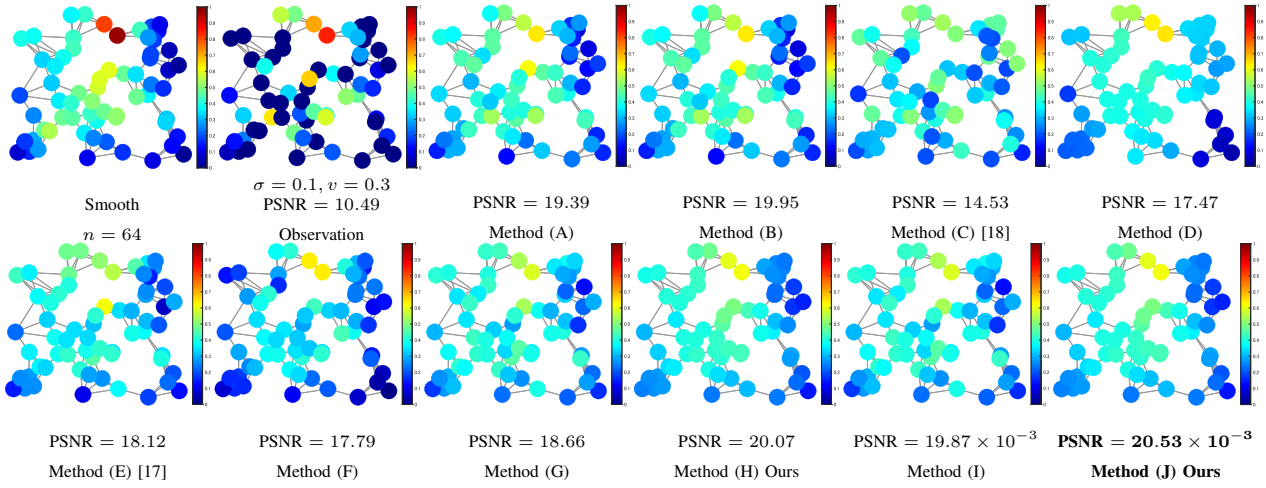

Fig. 3. Graph signal recovery results for $P = 0.5$. Note that the color range for the observations are clipped to [0,1] for a better illustrative comparison. The actual observations are noisier than the figures show.

speculate that high noise intensity compromises the sparsity of the temporal difference signal. For the same reason, higher velocity diminishing the sparsity of the temporal difference signal may have lead to the unexpected results.

Furthermore, looking at the visual results for the piece-wise flat dataset (Fig. 2 bottom two rows), the outputs seem to lack the piece-wise flatness they are supposed to have. This is most likely because the formulations are based on a second-order regularization in the vertex domain. We believe that a first-order regularization, such as Graph-Total Variation [4], can better estimate piece-wise flat graph signals. Although we did not implement a such formulation, the dynamic graph model should be applicable to Graph-Total Variation as well.

4) Masked vs Non-Masked: Although the performance does drop compared to the non-masked setting, it shows considerably good recovery results taking into account the difficult setting. In terms of comparisons between methods and parameters, the masked setting shows similar results to that of the non-masked setting.

D. Summary

To summarize,

- The dynamic model based methods performed better than the static counterpart, and our proposed method performed better than the other methods in all situations.
- Temporal domain contributes more to the performance, but was weak to the increasing velocity, reinforcing our proposal as a method more robust to various signal types.
- The results for the piece-wise flat data would probably improve by using a first-order regularization.
- We obtained similar results for the masked and non-masked settings.

IV. Conclusions

In this paper, we proposed a novel time-varying graph signal recovery problem based on a dynamic graph model, a model that assumes the underlying graph to be time-variant like the signals. We proposed a method to solve the problem that effectively leverages the priors of both the vertex and temporal domains, while also taking advantage of the dynamics of the graph. Through experiments on synthetic data, we compared our method to other methods based on regularization on the vertex and temporal domains, and analyzed how the priors of the two domains contribute to the recovery performance. We believe that the proposed problem setting is realistic, especially in situations like remote sensing using mobile sensors. Dynamic sensor networks are very feasible and practical sensing methods, and we place our research as the first step in tackling the problem and encourage further research on the problem.

Acknowledgment

This work was supported in part by JST CREST under Grant JPMJCR1662 and JPMJCR1666, and in part by JSPS KAKENHI under Grant 20H02145, 19H04135 and 18H05413.

References

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

[2] 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 Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[3] S. K. Narang and A. Ortega, "Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4673–4685, 2013.

[4] C. Couprie, L. Grady, L. Najman, J.-C. Pesquet, and H. Talbot, "Dual constrained TV-based regularization on graphs," *SIAM J. Imag. Sci*, vol. 6, no. 3, pp. 1246–1273, 2013.

[5] S. Ono, I. Yamada, and I. Kumazawa, "Total generalized variation for graph signals," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5456–5460.

[6] P. Berger, G. Hannak, and G. Matz, "Graph signal recovery via primal-dual algorithms for total variation minimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 842–855, 2017.

[7] C. Dinesh, G. Cheung, and I. V. Bajić, "3d point cloud color denoising using convex graph-signal smoothness priors," in *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, 2019, pp. 1–6.

[8] S. Chen and Y. C. Eldar, "Graph signal denoising via unrolling networks," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5290–5294.

[9] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 872–876.

[10] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka, "Graph signal denoising via trilateral filter on graph spectral domain," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 137–148, 2016.

[11] M. Nagahama, K. Yamada, Y. Tanaka, S. H. Chan, and Y. C. Eldar, "Graph signal denoising using nested-structured deep algorithm unrolling," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5280–5284.

[12] A. Sandryhaila and J. M. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Processing Magazine*, vol. 31, no. 5, pp. 80–90, 2014.

[13] P. Valdivia, F. Dias, F. Petronetto, C. T. Silva, and L. G. Nonato, "Wavelet-based visualization of time-varying data on graphs," in *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, 2015, pp. 1–8.

[14] N. Perraudin, A. Loukas, F. Grassi, and P. Vandergheynst, "Towards stationary time-vertex signal processing," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 3914–3918.

[15] A. Loukas and D. Foucard, "Frequency analysis of time-varying graph signals," in *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2016, pp. 346–350.

[16] F. Grassi, A. Loukas, N. Perraudin, and B. Ricaud, "A time-vertex signal processing framework: Scalable processing and meaningful representations for time-series on graphs," *IEEE Transactions on Signal Processing*, vol. 66, no. 3, pp. 817–829, 2018.

[17] K. Qiu, X. Mao, X. Shen, X. Wang, T. Li, and Y. Gu, "Time-varying graph signal reconstruction," *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 6, pp. 870–883, 2017.

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

[19] L. Condat, "A primal–dual splitting method for convex optimization involving lipschitzian, proximable and linear composite terms," *Journal of Optimization Theory and Applications*, vol. 158, 08 2013.

[20] A. Beck and M. Tebouile, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci*, vol. 2, no. 1, pp. 183–202, 2009.