Inpainting-Driven Graph Learning via Explainable Neural Networks

Subbareddy Batreddy, Pushkal Mishra, Yaswanth Kakarla, and Aditya Siripuram

***Abstract*—Given partial measurements of a time-varying graph signal, we propose an algorithm to simultaneously estimate both the underlying graph topology and the missing measurements. The proposed algorithm operates by training an interpretable neural network, designed from the unrolling framework. The proposed technique can be used as a graph learning and/or a graph signal reconstruction algorithm. This work builds on prior work in graph learning by tailoring the learned graph to the signal reconstruction task; and also enhances prior work in graph signal reconstruction by allowing the underlying graph to be unknown.**

***Index Terms*—Graph temporal data, graph signal processing, graph topology inference, unrolling.**

I. Introduction

GRAPHS are a natural way to represent a large class of irregularly structured signals obtained from numerous sources such as health monitoring devices [1], economic networks [2], meteorological stations [3], transportation networks, and biological networks [4]. The graph's vertices represent the signal components, and the edges encode the relation between various signal components. Graph signal processing (GSP) extends techniques and concepts from classical signal processing (e.g., the Fourier transform and frequencies) to such graph signals [5] and by exploiting the information from the underlying graph, GSP aims to improve upon the traditional techniques [3], [6]. Some applications include graph filters [7], sampling [8], [9], graph neural networks [10], [11], computer vision [12] and graph learning from data [13], [14].

In this work, we investigate the problem of *graph learning* via *graph-based data inpainting*. They key idea is that we use the learned graph from the graph learning algorithm on a signal processing task (like data inpainting), and use the feedback to improve the graph learning algorithm itself. As such the proposed technique here can be used both as a graph-learning and a graph-based data inpainting algorithm.

A. Inpainting

Graph-based data inpainting is the process of reconstructing missing data points from datasets that are derived from an underlying graph structure. Examples of such datasets include neurological data like fMRI (where the underlying graph is the functional connectivity map of the brain) [15], temperature data (the underlying graph corresponds to geographical similarity) [16] and neuroskeletal data (underlying graph is based on neural connectivity) [3]. Recent works [17], [18], [19] have proposed an inpainting algorithm for time-varying graph signals. The inpainting task is accomplished by assuming some prior on the signal, typically via graph-variation minimization: which requires knowledge of the underlying graph. However, in many real-world applications, the underlying graph is unknown; as evident in neurological and biological datasets (such as fMRI and MEG), sensor measurements, and others.

B. Graph Learning

The above discussion naturally points us to the problem of graph learning: constructing an approximate graph given a dataset assumed to be derived from an underlying graph. Techniques from GSP have offered a new perspective on the problem of graph learning by assuming certain priors on the data model [13], [20]. For example, well-known methods, such as [16], [21], [22], [23], assume that the data is smooth on the graph (or that the signal has low graph frequencies): similar to assumptions made for data inpainting. Other techniques exploit statistical properties [24], [25] or spectral characteristics [26], [27], [28]. While most of these methods are not specific to time-varying data, techniques in [29], [30] give algorithms for learning graphs from time-varying data.

We identify two key challenges in deploying these graph learning algorithms for data inpainting. First, note that graph learning techniques are typically evaluated on a GSP task (such as classification or data inpainting) using the learned graph [19], [31], [32]. However, graph learning techniques may not be tailored to the GSP task (data-inpainting) at hand; as this is an *open-loop system*. Secondly, there is a multitude of signal priors to deploy for graph learning: take for instance the global smoothness-based techniques from [16] which assume the signals are smooth on the graph, or the data inpainting techniques in [17], [19] where the temporal difference of the signal is assumed to be smooth on the graph. It may not be clear which model fits the given dataset. Our work aims to investigate a systematic solution to address these challenges.

Consider starting with a parameterized graph-learning model (which encompasses existing models as special cases) and a dataset with missing entries. We first perform data-inpainting using an initial graph structure: If the estimation results do not seem satisfactory, we update the graph parameters based on the received feedback. This iterative process forms a *closed loop system* as opposed to the open loop system discussed earlier; forming the core of our approach. The parameter updates are

Received 11 September 2024; revised 28 October 2024; accepted 5 November 2024. Date of publication 18 November 2024; date of current version 16 December 2024. The associate editor coordinating the review of this article and approving it for publication was Dr. Aiping Liu. *(Subbareddy Batreddy and Pushkal Mishra contributed equally to this work.) (Corresponding auhtor: Pushkal Mishra.)*

The authors are with the Indian Institute of Technology Hyderabad, Hyderabad 502284, India (e-mail: pushkalblr@gmail.com).

The dataset and code used in this work can be downloaded from https://github.com/PushkalM11/Graph-Learning-via-Unrolling.githere.

Digital Object Identifier 10.1109/LSP.2024.3501273

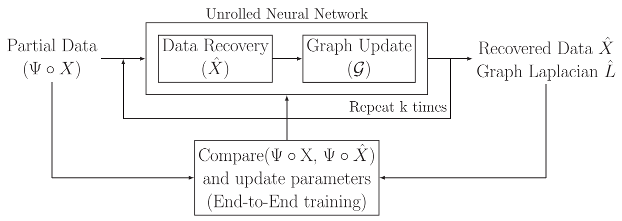

Unrolled Neural Network
Partial Data $(\Psi \circ X)$
Data Recovery $(\hat{X})$
Graph Update $(\mathcal{G})$
Recovered Data $\hat{X}$
Graph Laplacian $\hat{L}$
Repeat k times
Compare$(\Psi \circ X, \Psi \circ \hat{X})$ and update parameters (End-to-End training)

Fig. 1. Proposed model with a closed-loop feedback system.

accomplished via the unrolling framework [33]: where each iteration in an iterative technique is interpreted as a layer of a deep neural network. End-to-end training of this neural network accomplishes the role of feedback. The block diagram as shown in Fig. 1 provides an illustration of the proposed model.

Thus the proposed approach *jointly* accomplishes both data-inpainting and graph learning tasks. Note that this technique can also be used as a standalone graph-learning technique (by artificially removing a small number of entries from the given dataset).

Evaluation of our results show that:

1) The proposed algorithm estimates the missing entries better than techniques that first learn the graph and then use the learned graph for data inpainting (on real datasets),
2) The proposed algorithm recovers a graph closer to the ground truth compared with other graph learning techniques (on synthetic datasets), thus suggesting a potential use for the algorithm in both data-inpainting and graph-learning tasks.

Thus suggesting a potential use for the algorithm in both data-inpainting and graph-learning tasks. The proposed framework also extends to various other GSP tasks beyond inpainting, such as classification.

II. Problem Setup

A data matrix $\mathbf{X} = [\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_M]$ is an $N \times M$ matrix of real numbers, with each of the M columns corresponding to signals at M successive timestamps. We work with undirected graphs $\mathcal{G} = (V, E)$ with vertices $V = \{1, 2, \ldots, N\}$ and edges $E \subseteq V \times V$. Each column $\bar{x}_i : V \mapsto \mathbb{R}$ is interpreted as a graph signal defined on the vertices V of $\mathcal{G}$. The Laplacian matrix $\mathbf{L}$ of the graph has a diagonal entry at (i,i) as the degree of vertex i, an off-diagonal entry at (i,j) as -1 if vertices i and j are connected by an edge, and 0 otherwise. The graph-variation of the data matrix $\mathbf{X}$ is defined as

$$\mathcal{V}_{\mathcal{G}}(\mathbf{X}) = Tr(\mathbf{X}^{\mathsf{T}}\mathbf{L}\,\mathbf{X}) = \frac{1}{2}\sum_{t\,=\,1}^{M}\sum_{(i,j)\in E}(x_{ti} - x_{tj})^2$$

Graph-based data-inpainting algorithms [17], [19] and graph learning algorithms [16], [21], [22] assume that $\mathcal{V}_{\mathcal{G}}(f(\mathbf{X}))$ is small, for a suitably defined f, thus linking the data matrix with the graph structure. This assumption is then used to recover missing entries and the graph.

We denote Ψ as the mask matrix where $\Psi_{ij} = 1$ if the entry at that location (i,j) in $\mathbf{X}$ is known and $\Psi_{ij} = 0$ otherwise. With this setup, the problem under investigation is: 'Given $\Psi \circ \mathbf{X} + \mathbf{E}$ (where $\mathbf{E}$ is the noise) and Ψ, design an algorithm that outputs the graph $\mathcal{G}$ and the entire data matrix $\mathbf{X}$; such that $\mathcal{V}_{\mathcal{G}}(f(\mathbf{X}))$ is small'. Some general notations: We denote by $\mathbf{A} \circ \mathbf{B}$ the Hadamard (elementwise) product of $\mathbf{A}$ and $\mathbf{B}$, by $\mathbf{1}$ a vector of all $1'$s and tr(.) as the trace of the matrix.

III. Proposed Algorithm

A. Framing the Optimization Problem

Motivated by [17] and [19], we define the function f referenced in the previous section as a higher-order temporal difference. Let the temporal difference operator be defined as:

$$\mathbf{X}\Delta = \begin{bmatrix}\bar{x}_2 - \bar{x}_1 & \bar{x}_3 - \bar{x}_2 & \ldots & \bar{x}_M - \bar{x}_{M-1}\end{bmatrix}$$

Thus, we can take $f(\mathbf{X}) = \mathbf{X}\Delta$, and get $\mathcal{V}_{\mathcal{G}}(\mathbf{X}) = \text{Tr}(\mathbf{X}^{\mathsf{T}}\mathbf{L}\mathbf{X}\Delta\Delta^{\mathsf{T}})$. This variation measures the smoothness of the temporal difference on the graph. We generalize the $\Delta\Delta^{\mathsf{T}}$ operator above by introducing powers of $\Delta\Delta^{\mathsf{T}}$ and their polynomial combinations- this has the effect of accounting for smoothness of higher order temporal differences:

$$Z(\bar{\alpha}) = \alpha_0\mathbf{I} + \sum_{i=1}^{k}\alpha_i(\Delta\Delta^{\mathsf{T}})^i. \tag{1}$$

Hence, we use $\mathcal{V}_{\mathcal{G}}(\mathbf{X}, \bar{\alpha}) = \text{Tr}(\mathbf{X}^{\mathsf{T}}\mathbf{L}\mathbf{X}Z(\bar{\alpha}))$. Note that this is a semi-norm for $\bar{\alpha} \geq 0$, and generalizes the regularizer (based on difference operators) from [19]. Now given the partial data $\Psi \circ \mathbf{X}$ and Ψ, we frame the following optimization problem to obtain the complete data $\hat{\mathbf{X}}$ (output of the neural network), the graph Laplacian $\mathbf{L}$ (learned graph) and the model parameters $\bar{\alpha}$ (graph learning parameters):

$$\min_{\hat{\mathbf{X}},\mathbf{L},\bar{\alpha}} \left\|\Psi \circ (\mathbf{X} - \hat{\mathbf{X}})\right\|_F^2 + \lambda\mathcal{V}_{\mathcal{G}}(\hat{\mathbf{X}}, \bar{\alpha}) + \beta\left\|\mathbf{L}\right\|_F^2 + \gamma\left\|Z(\bar{\alpha})\right\|_F^2$$

$$\text{s.t.} \quad \mathbf{L}_{ii} \geq 1, \mathbf{L}_{ij} = \mathbf{L}_{ji} \leq 0\ \forall\ i \neq j, \mathbf{L}\mathbf{1} = \mathbf{0} \tag{2}$$

In the objective function, the first term ensures data fidelity, the second term is the graph variation, the third term ensures that the learned graph is sparse and the last term imposes a norm-bound constraint on the solution space and ensures stability of the solution. The constraints ensure that the learned $\mathbf{L}$ is a valid Laplacian matrix.

B. Solving the Optimization Problem

We adopt an alternating minimization technique to proceed.

1) Estimating $\hat{\mathbf{X}}$*:* In the first step, we estimate unknown entries of the data matrix $\Psi \circ \mathbf{X}$, assuming $\mathbf{L}$ (initialized using covariance matrix from available data) and $\bar{\alpha}$ are known. The corresponding optimization problem is as follows:

$$\min_{\hat{\mathbf{X}}} \left\|\Psi \circ (\mathbf{X} - \hat{\mathbf{X}})\right\|_F^2 + \lambda\ \mathcal{V}_{\mathcal{G}}(\hat{\mathbf{X}}, \bar{\alpha}) \tag{3}$$

Since $\mathbf{L}$ is assumed known, this step is similar to prior works [17]. The optimization problem above is solved using the conjugate gradient descent method [34]. Refer to Appendix A for more the modified steps (EMD: Estimate Missing Data).

2) Updating $\mathbf{L}$*:* In the second step, we update $\mathbf{L}$ by fixing the other two variables $\hat{\mathbf{X}}$ and $\bar{\alpha}$. The corresponding optimization problem is as follows:

$$\arg\min_{\mathbf{L}} \quad \mathcal{V}_{\mathcal{G}}(\hat{\mathbf{X}}, \bar{\alpha}) + \beta\left\|\mathbf{L}\right\|_F^2$$

$$\text{s.t. } \mathbf{L}_{ii} \geq 1, \mathbf{L}_{ij} = \mathbf{L}_{ji} \leq 0\ \forall\ i \neq j, \mathbf{L}\cdot\mathbf{1} = \mathbf{0}$$

To solve this optimization problem, we employ the Projected Gradient Descent method [34]. The update steps are as follows:

We use the following approximate projection operator $\text{Proj}_{\mathcal{M}}(\mathbf{L})$: first extract the off-diagonal entries of $\mathbf{L}$ and multiply them by -1 (hadamard product with $\mathcal{M}$), second pass them through a ReLU unit to retain edges with positive weights,

Algorithm 1: Graph Learning.

```
1: function GL  $\Psi \circ \mathbf{X}$, $\hat{\mathbf{X}}$, $\mathbf{L}$, $\bar{\alpha}$, $k$, $\beta$, $\eta$
2:     Set: $\mathbf{L}_1 = \mathbf{L}$ and $\mathcal{M} = \mathbf{I} - \mathbf{1}.\mathbf{1}^\mathsf{T}$
3:     for $i \leftarrow 1$ to $k$ do
4:         Compute gradient $\nabla f(\mathbf{L}_i) \leftarrow \hat{\mathbf{X}}\, Z(\bar{\alpha})\, \hat{\mathbf{X}}^\mathsf{T} + \beta\, \mathbf{L}_i$
5:         Update Laplacian $\bar{\mathbf{L}}_{i+1} \leftarrow \mathbf{L}_i - \eta \nabla f(\mathbf{L}_i)$
6:         Project to feasible space $\mathbf{L}_{i+1} \leftarrow \text{Proj}_{\mathcal{M}}(\bar{\mathbf{L}}_{i+1})$
7:     end for
9:     return $\mathbf{L}_k$
10: end Function
```

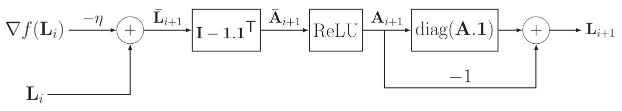


Fig. 2. Illustration of the proposed Graph learning update step using linear operations and a ReLU unit.

and finally converting resulting matrix into a Laplacian matrix by replacing the diagonal entries suitably. This approximate projection operator $\text{Proj}_{\mathcal{M}}(\mathbf{L})$ and the update procedure for $\mathbf{L}$ is better illustrated in Fig. 2. Note that this projection operator is not necessarily an orthogonal projection on the space of Laplacian matrices. However, it enables us to map the update iterations above to layers of a neural network. The projection operator proposed satisfies the following properties which are desirable for the algorithm:

1) $\text{Proj}_{\mathcal{M}}(\mathbf{L})$ is always a valid Graph Laplacian for any $\mathbf{L}$,
2) It is an idempotent i.e. $\text{Proj}^2_{\mathcal{M}}(\mathbf{L}) = \text{Proj}_{\mathcal{M}}(\mathbf{L})$, and importantly
3) $\text{Proj}_{\mathcal{M}}(\mathbf{L})$ can be realized by linear combinations and ReLUs, resulting in a neural network interpretation.

The effectiveness of the $\text{Proj}_{\mathcal{M}}(\mathbf{L})$ operator has been empirically validated through experimental results.

C. Setting up the Unrolled Neural Network

Now, we have an iterative technique to find the graph and missing entries assuming $\bar{\alpha}$ is fixed. Next, we unroll this algorithm i.e. map each iteration to a single network layer. For the resulting neural network, $\Psi \circ \mathbf{X}$ is the input, $\hat{\mathbf{X}}$ and $\mathbf{L}$ are the outputs, and $\bar{\alpha}$ are the parameters of the neural network.

Our iterations consist of those that update $\hat{\mathbf{X}}$ (Section III-B1) and update $\mathbf{L}$ (Section III-B2). The design of our iterative technique for graph learning (Section III-B2) involves linear operations mixed with ReLU (Fig. 2), which enables us to map iterations to neural network layers easily.

The feed-forward process of this neural network is thus equivalent to iteratively reconstructing a graph and time-varying graph signal inpainting with the given parameters $\bar{\alpha}$. We use the following loss function to train the neural network:

$$\text{Loss}(\bar{\alpha}) = \left\|\Psi \circ (\mathbf{X} - \hat{\mathbf{X}})\right\|_F^2 + \lambda \mathcal{V}_{\mathcal{G}}(\hat{\mathbf{X}}, \bar{\alpha}) + \beta \left\|\mathbf{L}\right\|_F^2 + \gamma \left\|Z(\bar{\alpha})\right\|_F^2 \quad (4)$$

A forward pass of this neural network minimizes the above loss for a fixed $\bar{\alpha}$ and estimates $\hat{\mathbf{X}}$ and $\mathbf{L}$, and the backward pass updates these parameters $\bar{\alpha}$ for a better fit of the learned graph to the given data. The forward pass is summarized in the pseudocode below:

Algorithm 2: Unrolled Neural Network (Forward pass).

```
1: Given: Missing data matrix $\Psi \circ \mathbf{X}$ and $\Psi$
2: Input: $\mathbf{Y} = \Psi \circ \mathbf{X}$ and $\Psi$
3: Hyperparameters: $k$, $k_1$, $k_2$, $\eta$, $\beta$, $\gamma$ and $\lambda$
4: Parameters of the neural network: $\bar{\alpha}$
5: Initialization: $\mathbf{L}_1 \leftarrow$ Covariance graph from $\Psi \circ \mathbf{X}$
6: for $i \leftarrow 1$ to $k$ do                                  ▷ Unrolled iterations
7:     $\hat{\mathbf{X}}_i \leftarrow$ EMD $(\mathbf{Y}, \mathbf{L}_i, \bar{\alpha}, \lambda, k_1)$          ▷ See Appendix A
8:     $\mathbf{L}_{i+1} \leftarrow$ GL $(\mathbf{Y}, \hat{\mathbf{X}}_i, \mathbf{L}_i, \bar{\alpha}, k_2, \beta, \eta)$     ▷ Graph update
9: end for
10: Output $\hat{\mathbf{X}}$, $\mathbf{L}$
```

D. Differences From Prior Work

As a graph learning algorithm, to our knowledge, there is no similar work for time-varying data, and applying the unrolling framework for graph learning via feedback from data inpainting is new.

As a data-inpainting algorithm, our work is motivated from and builds on [17]. The main differences stem from the assumption that $\mathbf{L}$ is unknown, and the resulting formulation of the objective function (2). In addition, we omit the polynomial terms in $\mathbf{L}$ (to promote convexity of the alternate minimization steps) in favour of a regularization term involving both $\mathbf{L}$ and the graph parameters via $Z(\bar{\alpha})$. Further, the cost function used to train the neural network is the same as the objective in (2), unlike [17] where only one term in the objective is used to train the neural network. The proposed unrolled neural network consists of additional layers corresponding to the graph learning which are interlaced with the ones seen in [17] for data inpainting.

Unrolling has also been used for graph learning in different contexts: e.g. [35] proposes a distributed learning model for multi-agent collaborative setup. The goal here is to enable the agents to detect appropriate collaborators for performance gains autonomously. The graphs here denote pairwise collaborative relations which are obtained using a graph learning network; and unrolling is employed by introducing trainable attention for each model parameter at the agent. This is in contrast to our setup where the signals involved at each graph node have a time-based interpretation.

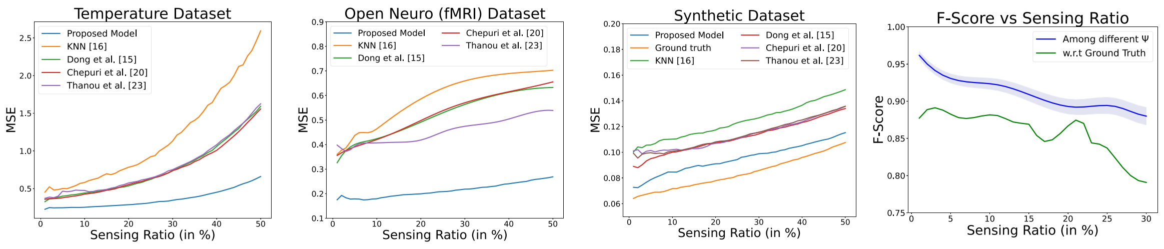
The work [36] introduces an unrolling-based technique to learn graphs with certain topological properties. The unrolling model is trained with node data and graph samples. From the graph learning perspective, this technique operates in a supervised framework. In contrast, we do not have access to graph samples, and thus operate in an unsupervised framework. Our end-to-end training is done based on how well the graph model fits the data within the inpainting process. Likewise, recent work in [37] develops a unrolling based transformer model; however it deals with image data unlike the work here which is developed for time-varying data.

IV. Results and Discussion

In this section, we evaluate the performance of the proposed graph learning algorithm on both synthetic and real data sets.

A. Evaluation Metrics:

- *Normalized error (MSE):* We evaluate the data inpainting performance by comparing the MSE error of unknown entries: $\text{MSE} = \|(\mathbf{1}\mathbf{1}^\mathsf{T} - \Psi) \circ (\hat{\mathbf{X}} - \mathbf{X})\|_F \Big/ \|\mathbf{1}\mathbf{1}^\mathsf{T} - \Psi\|_1$



(a) Temperature Dataset: MSE (b) Open Neuro Dataset: MSE (c) Synthetic Dataset: MSE (d) Synthetic Dataset: F-Score

Fig. 3. Normalized Error (MSE) as a function of the sensing ratio on (a) Temperature dataset (b) fMRI (Open Neuro) dataset (c) Synthetic dataset (d) F-score for different sensing ratios on Synthetic dataset.

- *F-Score:* We use this to measure the similarity of the learned graph with the ground truth graph (synthetic dataset only) [38]: higher F-Score relates to better graph learning.

Note that the sensing ratio is referred to as $\|\mathbf{1}\mathbf{1}^\top - \Psi\|_1/NM$, i.e. fraction of missing entries.

B. Competing Methods

We compare our joint graph learning and data inpainting algorithm with techniques that separately perform the two tasks. We compare with various graph learning techniques from literature including global smoothness-based learning [16], [22] and diffusion-based learning [26], [28]. We also compare with a nearest neighbour-based graph learning (each vertex is connected to its k nearest neighbours in $2-$norm): as used in [17]. Once the graph is learned, we artificially remove some entries and recover these missing entries using the learned graph as in [17]. This process is repeated 20 times for various mask matrices and the computed MSE is averaged.

C. Results on Real Datasets

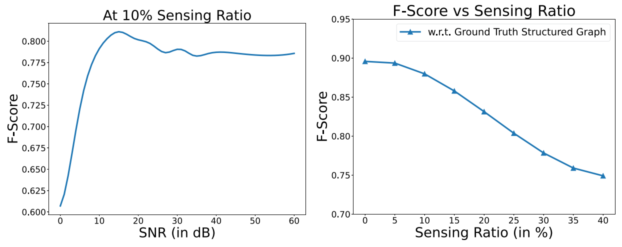
We evaluated the proposed graph learning model on two publicly available datasets. The first is the Brittany temperature dataset [22] (the temperature measurements collected across 32 weather stations, with 744 observations per weather station). The second dataset is the Open Neuro dataset [39] (this is fMRI data from 32 brain regions and 152 observations per region of interest).

Fig. 3(a) and (b) compare the performance of the proposed and competing methods on these datasets as a function of the sensing ratio. The performance gains are evident; we hypothesize that this is because the proposed algorithm can better learn the underlying graph topology, as motivated in the introduction.

D. Results on Synthetic Datasets

Synthetic datasets are generated by first constructing a graph and then generating data that is a temporally smooth on the constructed graph. The (unweighted) graph is generated according Erdős-Rényi (ER) model [40] with $N = 20$ vertices and probability of edge 0.3. On this graph, we generate time-varying data $\mathbf{X}$ as per the model in (2). We generate $M = 500$ time varying observations. The polynomial parameters are set as $\alpha_1 = 4$ and $\alpha_2 = 1.66$. We have verified that the proposed algorithm recovers these ground truth α's.

In addition to the competing methods from before, we also compare the performance of our algorithm with prior graph signal reconstruction work [17] by giving the *ground truth* graph as input. This is marked as a lower bound for the MSE of our algorithm.



(a) Data on Random graph (b) Data on structured graph

Fig. 4. F-score as a function of Sensing Ratio for the proposed algorithm on synthetic data generated on (a) a random (ER) graph and (b) a structured synthetic graph.

Fig. 3(c) compares the data inpainting performance of the proposed and competing methods. The proposed technique shows better performance in data inpainting tasks and is quite close to the lower bound as defined above. Fig. 3(d) depicts the performance of graph learning. The green plot depicting the closeness of learned graph with ground truth and blue plot showing the similarity of graphs generated with 100 different mask matrices Ψ. Thus we notice that at low sensing ratios, the impact of Ψ is minimal, establishing that the learned graph is stable w.r.t where the entries in the dataset are removed.

To demonstrate the robustness of graph learning task, we add Gaussian noise to the (synthetic) data matrix and then apply our algorithm to learn the graph. We then plot the F-score as a function of the SNR as seen in the Fig. 4(a). We evaluated the proposed graph learning algorithm on structured graphs as well (we used Zachary's Karate Club social network dataset [41], also used in prior work [42]) in Fig. 4(b). As we can see, the performance is similar to that of ER graphs.

Appendix

A. Estimate Missing Data (EMD: Section III-B1)

Since our optimization equation has different regularizers compared to the one in [17], we recompute the gradient of f. For the data inpainting block, we have the optimization function as follows:

$$f(\mathbf{X}) = \left\|\Psi \circ (\mathbf{X} - \hat{\mathbf{X}})\right\|_F^2 + \lambda\ \mathcal{V}_{\mathcal{G}}(\hat{\mathbf{X}}, \bar{\alpha}).$$

Computing the gradient of f with respect to $\mathbf{X}$ yields:

$$\nabla_{\mathbf{X}} f(\mathbf{X}) = 2\Psi \circ (\mathbf{X} - \hat{\mathbf{X}}) + 2\lambda \mathbf{L}\mathbf{X}Z(\bar{\alpha})$$

Further, we use 4a to 4d from [17], replace the gradient with the above expression and follow the exact iteration steps as mentioned in [17].

REFERENCES

[1] D. Ahmedt-Aristizabal, M. A. Armin, S. Denman, C. Fookes, and L. Petersson, "Graph-based deep learning for medical diagnosis and analysis: Past, present and future," *Sensors*, vol. 21, no. 14, 2021, Art. no. 4758.

[2] D. Michael and S. Battiston, "From graph theory to models of economic networks. A tutorial," in *Networks, Topology and Dynamics: Theory and Applications to Economics and Social Systems*. Berlin, Germany: Springer, 2009, pp. 23–63.

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

[4] C. Hu et al., "A spectral graph regression model for learning brain connectivity of Alzheimer's disease," *PLoS One*, vol. 10, no. 5, 2015, Art. no. e0128136.

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

[6] G. Leus, A. G. Marques, J. M. F. Moura, A. Ortega, and D. I. Shuman, "Graph signal processing: History, development, impact, and outlook," *IEEE Signal Process. Mag.*, vol. 40, no. 4, pp. 49–60, Jun. 2023.

[7] J. Liu, E. Isufi, and G. Leus, "Filter design for autoregressive moving average graph filters," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 1, pp. 47–60, Mar. 2019.

[8] A. G. Marques, S. Segarra, G. Leus, and G. Ribeiro, "Sampling of graph signals with successive local aggregations," *IEEE Trans. Signal Process.*, vol. 64, no. 7, pp. 1832–1843, Apr. 2016.

[9] A. Anis, A. Gadde, and A. Ortega, "Towards a sampling theorem for signals on arbitrary graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 3864–3868.

[10] F. Gama, A. G. Marques, G. Leus, and A. Ribeiro, "Convolutional neural network architectures for signals supported on graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 4, pp. 1034–1049, Feb. 2019.

[11] F. Gama, J. Bruna, and A. Ribeiro, "Stability properties of graph neural networks," *IEEE Trans. Signal Process.*, vol. 68, pp. 5680–5695, 2020.

[12] J. H. Giraldo, S. Javed, M. Sultana, S. K. Jung, and T. Bouwmans, "The emerging field of graph signal processing for moving object segmentation," in *Proc. Int. Workshop Front. Comput. Vis.*, 2021, pp. 31–45.

[13] 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, May 2019.

[14] B. Subbareddy, S. Aditya, and J. Zhang, "Robust graph learning for classification," *Signal Process.*, vol. 211, 2023, Art. no. 109120.

[15] J. Richiardi, S. Achard, H. Bunke, and D. Van De Ville, "Machine learning with brain graphs: Predictive modeling approaches for functional imaging in systems neuroscience," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 58–70, May 2013.

[16] 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, Dec. 2016.

[17] S. Chen and Y. C. Eldar, "Time-varying graph signal inpainting via unrolling networks," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2021, pp. 8092–8097.

[18] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovačević, "Signal recovery on graphs: Variation minimization," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4609–4624, Sep. 2015.

[19] K. Qiu, X. Mao, X. Shen, X. Wang, T. Li, and Y. Gu, "Time-varying graph signal reconstruction," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 870–883, Sep. 2017.

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

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

[22] S. P. Chepuri, S. Liu, G. Leus, and A. O. Hero, "Learning sparse graphs under smoothness prior," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 6508–6512.

[23] B. Subbareddy, S. Aditya, and J. Zhang, "Learning bipartite graphs from spectral templates," *Signal Process.*, vol. 227, 2024, Art. no. 109732.

[24] B. Lake and J. Tenenbaum, "Discovering structure by learning sparse graphs," in *Proc. 32nd Annu. Meeting Cogn. Sci. Soc.*, Portland, OR, USA, 2010, pp. 778–784.

[25] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from data under Laplacian and structural constraints," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 6, pp. 825–841, Sep. 2017.

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

[27] H. E. Egilmez, E. Pavez, and A. Ortega, "Graph learning from filtered signals: Graph system and diffusion Kernel identification," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 5, no. 2, pp. 360–374, Jun. 2019.

[28] H. P. Maretic, D. Thanou, and P. Frossard, "Graph learning under sparsity priors," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 6523–6527.

[29] V. Kalofolias, A. Loukas, D. Thanou, and P. Frossard, "Learning time varying graphs," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2017, pp. 2826–2830.

[30] K. Yamada, Y. Tanaka, and A. Ortega, "Time-varying graph learning with constraints on graph temporal variation," in *Proc. 2019 IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2019, pp. 5411–5415, doi: 10.1109/ICASSP.2019.8682762.

[31] P. Humbert, B. B. Le, 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.

[32] A. Kroizer, Y. C. Eldar, and T. Routtenberg, "Modeling and recovery of graph signals and difference-based signals," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2019, pp. 1–5.

[33] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, Mar. 2021.

[34] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[35] E. Zhang, S. Tang, X. Dong, S. Chen, and Y. Wang, "Unrolled graph learning for multi-agent collaboration," 2022, *arXiv:2210.17101*.

[36] X. Pu, T. Cao, X. Zhang, X. Dong, and S. Chen, "Learning to learn graph topologies," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 4249–4262.

[37] D. Thuc, P. Eftekhar, S. A. Hosseini, G. Cheung, and P. Chou, "Interpretable lightweight transformer via unrolling of learned graph smoothness priors," 2024, *arXiv:2406.04090*.

[38] Y. Sasaki et al., "The truth of the F-measure," *Teach. Tut. Mater.*, vol. 1, no. 5, pp. 1–5, 2007.

[39] P. Bellec, C. Chu, F. Chouinard-Decorte, Y. Benhajali, D. S. Margulies, and R. C. Craddock, "The neuro bureau ADHD-200 preprocessed repository," *Neuroimage*, vol. 144, pp. 275–286, 2017.

[40] P. Erdos et al., "On the evolution of random graphs," *Pub. Math. Inst. Hung. Acad. Sci*, vol. 5, no. 1, pp. 17–60, 1960.

[41] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropological Res.*, vol. 33, no. 4, pp. 452–473, 1977.

[42] W. L. Hamilton, *Graph Representation Learning*. San Rafael, CA, USA: Morgan & Claypool, 2020.