Learning Product Graphs From Spectral Templates

Aref Einizade and Sepideh Hajipour Sardouie

Abstract—Graph Learning (GL) is at the core of leveraging connections in machine learning (ML). By observing a dataset of graph signals and considering specific assumptions, Graph Signal Processing (GSP) provides practical constraints in GL. Inferring a graph with desired frequency signatures, i.e., spectral templates, from stationary graph signals has gained great attention. However, a severe computational burden is a challenging barrier, especially for inference from high-dimensional product graph signals, i.e., graph signals live on the product of smaller factor graphs. Few product GL methods have been proposed for mostly inference with smoothness assumption, while they are limited to learning only two factor graphs, handle only the Cartesian products, and have not addressed GL with desired spectral templates. To bridge the mentioned gaps, we propose a method for learning product graphs from product graph signals in which the product GL problem can be broken into separated optimizations associated with each (significantly smaller) factor graph. Besides, unlike the current approaches, our method can learn from any type of product graph (possibly with more than two factor graphs) without needing to know the type of graph products beforehand, and with significantly reduced complexity than learning directly from product graph signals. In addition to devising theoretical sufficient recovery conditions and validating them numerically, experimental results on synthetic and real-world data, i.e., multi-view image and brain signal analysis, illustrate meaningful factor graphs supported by expert-related research and also superiority over the current methods for learning from spectral templates. The implementation codes are available at https://github.com/ArefEinizade2/ProdSpecTemp

Index Terms—Graph signal processing (GSP), Spectral templates, Product graph learning (PGL), Brain connectivity, Multi-view object images.

I. INTRODUCTION

Growing demand for generating and recording irregular (structured) data, which live on meaningful underlying graph structures, has led to extending the classic processing tools to the emerging field of Graph Signal Processing (GSP) [1]. For instance, temperature measurement sites with close geographic locations usually record near air quality measures and temperatures [2], and functionally (or spatially) connected brain regions behave rather similarly in a specific brain activity [3]. Exploiting such connections among data components via GSP tools can severely improve the processing performance and also pave the way for expert interpretation in a more comprehensive scheme. However, in many real-world applications, these meaningful graphs are not available beforehand, or estimating their connections in a predefined pairwise manner (e.g., Pearson correlation-based measures) leads to erroneous inferences and sensitivity to possible noise [2], [4]. In this way and to leverage meaningful data connections, some GSP-based Graph Learning (GL) approaches have recently been proposed to address these issues by considering the global behavior of graph signals over the underlying graphs, unlike their traditional pairwise correlation-based counterparts [2].

Different GL frameworks learn the underlying graphs based on prior specific assumptions about the (global or local) behavior of the graph signals. One of the widely-used GSP-based GL categories relies on the smoothness of the graph signals (i.e., the more connected the nodes, the more similar the nodes’ values) and, due to the amenable computational aspects, has facilitated the smooth GL [2]. Another popular category, which makes it possible to have desired graph frequency characteristics, learns the underlying graphs from prior (predefined or estimated from observations) spectral templates of diffused (filtered) graph signals [5]. For a comprehensive review of GL approaches, please refer to [4], [6] and the references therein.

Despite the insightful and applicable advantages of the different GL categories, in many prevalent real-world applications, the observations at hand admit meaningful couplings across different domains, in which learning a single underlying graph leads to interpretation issues and a severe increase in the dimensionality of the (vectorized) graph signals [7]. For example, in many brain signal processing applications [8], there is specific temporal connectivity between multi-channel brain signals, as well as interpretable spatial connection signatures, in which learning a single graph associated with the whole spatiotemporal data for describing temporal and spatial couplings makes it computationally intractable and extremely hard to interpret. As another instance, this is the case with multi-view object images [9], in which similar objects construct the clusters, and specific connections exist between degree intervals (e.g., a cycle graph describing the typical close degrees or connections between the complete front and back to the camera for rather symmetric objects). A few more examples of the real-world multi-domain couplings under the category of

2373-776X © 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.
multi-layer graphs include dynamic opinion modeling \[10\] and timely temperature pattern inference \[11\]. These multi-domain coupling connections can be well modeled by the concept of product graphs \[7, 12\] and their associated factor graphs \[11\].

**Related work:** In \[13\], the problem of inferring Cartesian product Precision matrices was addressed by proposing the Bigraphical Lasso (BiGLasso) optimization. In another work, the covariance matrix is modeled as a sum of Kronecker structure terms for better estimation within the proposed permuted rank-penalized least-squares (PRLS) method \[14\]. However, interpreting the Precision matrix instead of the underlying graph itself has major drawbacks, e.g., possibly irrelevant sparsity pattern between a Precision matrix and its relevant Laplacian graph \[15\]. As another category, learning product graphs (PGL) from smooth multi-domain graph signals has recently been addressed in \[11\]. However, it is limited to only Cartesian product graphs and deals with only two underlying factor graphs.

From another point of view, graph frequency-aware GL approaches are also another desired category \[4, 6\], which investigate the frequency characteristics of applicable graph processes, e.g., Graph Moving Average (GMA) \[16\], and their impacts on the underlying typologies \[5\]. Such stationary-based modelings have shown popular real-world practical applications such as inferring the topological properties of the proteins \[5\], image reconstruction \[16\], and learning brain connectivity patterns \[6\]. The stationarity of the observed graph signals serves a vital role in formulating the problem because, in such cases, the eigenvectors of the estimated observation covariance matrix share with that of the Graph Shift Operator (GSO) of interest, e.g., the adjacency matrix \[5, 16\]. However, to the best of our knowledge, learning product graphs with desired (estimated from observations) spectral templates from stationary diffused (filtered) multi-domain graph signals has not been addressed.

In the following, the contributions and organization of the present paper are summarized as:

- Firstly, for learning an underlying graph from observed stationary graph signals, in Section III-A, we propose a more computationally appropriate approach rather than the baseline SpecTemp \[5\] method via inexact Augmented Lagrange Multipliers (IALM) \[17, 18\], referred as SpecTemp-IALM, which is crucial for acceleration \[19, 20\].

- Then, based on the developed SpecTemp-IALM, in Section III-B, we present a method, named ProdSpecTemp, to efficiently learn the product graphs from stationary multi-domain diffused tensorial product graph signals living on possibly more than two interacting factor graphs with desired graph frequency characteristics, i.e., spectral templates \[5\], unlike the rare current methods which have addressed the recovery of only two factor graphs \[11, 13\]. In fact, our approach generalizes and extends the SpecTemp \[5\] to infer product graphs, which, to the best of our knowledge, has not been addressed\(^1\). In the proposed ProdSpecTemp, the product GL problem can be broken into separated optimizations associated with each (significantly smaller) factor graph under some imposed (mild) assumptions. In Section III-C, the theoretical justifications behind these imposed assumptions for efficiently formulating the problem are discussed.

- The theoretical sufficient recovery conditions are devised in Section III-D and evaluated experimentally in Section IV-D. Besides, based on the computational analysis presented in Section III-E, our proposed approach reduces the computational cost against directly learning from high-dimensional product graph signals and, therefore, enjoy applicability in real-world scenarios, e.g., Big Data processing \[7\].

- The proposed method can infer factor graphs from any kind of graph product, e.g., Cartesian, Kronecker, or Strong, without prior knowledge about the type of graph products, in contrast to the current methods, which are restricted to a specific type (mostly Cartesian) \[11, 13\] or need to know the product type beforehand \[21\].

- Experimental results on both synthetic and real-world data in Section IV demonstrate that the proposed ProdSpecTemp approach outperforms the state-of-the-art for learning type-free product graphs from high-dimensional product spectral templates. Precisely, its applicability of revealing sleep-related brain functional connectivity \[8\] and also multi-view object connections \[9\] illustrates the strength of the proposed approach in learning meaningful real-world factor graphs. Additional theoretical and experimental discussions are also presented in the Appendix and the provided Supplemental Text Material (STM).

**Notation:** Vectors, matrices, tensors, and sets are denoted by boldface lowercase, boldface uppercase, boldface uppercase with desired graph frequency characteristics, i.e., spectral templates \[5\], and Frobenius norm, respectively, but the \(p\)-norm of a matrix is \(\|\cdot\|_p\). The notation \(\langle \cdot , \cdot \rangle\) for inner product, inner product, mathematical expectation, Kronecker product, Khatri-Rao product, Moore-Penrose pseudo-inverse of a matrix, the \(p\)-norm of a vector or vectorized form of a matrix, and Frobenius norm, respectively, but the \(p\)-norm of a matrix induced by its vector form is stated by \(\|\cdot\|_M(p)\). The set of non-negative real numbers is \(\mathbb{R}_+\). The matrices \(\text{diag}(a)\) and \(\text{diag}(A)\) are diagonal matrices with the entries \(a\) and diagonal entries of \(A\) on their main diagonal, respectively. The \((i, j)\)-th and \(i\)-th entries of a matrix \(M\) and a vector \(v\) are denoted as \(M_{ij}\) and \(v_i\), respectively. The all-zero and all-one vectors of size \(N\) are referred as \(0_N\) and \(1_N\). The notations \(\text{vec}(\cdot)\) and \(\text{vech}(\cdot)\) denote the vectorization and half-vectorization operators \[2\], respectively, while \(\text{vecm}(\cdot)\) performs non-diagonal half-vectorization \[19, 20\]. Besides, \(\text{mat}(a, [M, N])\) returns a matrix of size \(M \times N\) containing entries of the vector \(a\) (of size \(MN\)) in column-wise order. For two square matrices \(A \in \mathbb{R}^{N \times N}\) and \(B \in \mathbb{R}^{M \times M}\), Cartesian \((\otimes)\), Kronecker \((\otimes)\), and Strong \((\boxdot)\) products have the size of \(MN \times MN\) and are denoted as \((A \otimes B) + (I_N \otimes B), (A \otimes B)\), and \((A \otimes I_M) + (I_N \otimes B) + (A \otimes B)\), respectively. The mode-\(r\) unfolding (matricization) form \[22\] of a tensor \(X\) is

---

\(^1\)The implementation codes are available at https://github.com/ArefEinizade2/ProdSpecTemp
denoted as \( \mathbf{X}_f \). For a matrix \( \mathbf{X} \), \( \mathbf{X}_f \) is obtained by selecting the rows of \( \mathbf{X} \) indexed by the set \( \mathcal{I} \) with cardinality \( |\mathcal{I}| \). In this way, \( \mathbf{X}_f \) is formed by selecting rows of \( \mathbf{X} \) which do not belong to the set \( \mathcal{I} \). \( \ker(\mathbf{X}) \) and \( \text{Im}(\mathbf{X}) \) express the null and column spaces of matrix \( \mathbf{X} \). The block-diagonal matrix \( \mathbf{A} = \text{bdiag}(\{\mathbf{B}_i\})_{i=1}^M \) has the matrices \( \{\mathbf{B}_i\}_{i=1}^M \) on its main diagonal blocks.

**II. PRELIMINARIES**

A. GSP Background

An undirected graph \( \mathcal{G} \) with \( N \) vertices is characterized with the vertex set \( \mathcal{V} \), the edge set \( \mathcal{E} \), and the symmetric GSO \( \mathbf{S} \in \mathbb{R}^{N \times N} \). The sparsity pattern of \( \mathcal{G} \) is encoded by the GSO \( \mathbf{S} \), where \( S_{ij} = 0 \) if the \( i \)-th and \( j \)-th nodes are disconnected. In the present article, the GSO of interest is the adjacency matrix \( \mathbf{W} \in \mathbb{R}^{N \times N} \), in which \( \mathbf{W}_{ij} \) models the similarity measure between the \( i \)-th and \( j \)-th nodes. Precisely, the set \( \mathcal{W} \), i.e., the valid non-zero adjacencies, can be expressed as

\[
\mathcal{W} := \left\{ \mathbf{W} \in \mathbb{R}^{N \times N} : \mathbf{W} = \mathbf{W}^\top, \{\mathbf{W}_{ii} = 0\}_{i=1}^N \sum_{j=1}^N \mathbf{W}_{ij} = 1 \right\},
\]

where, the constraint \( \sum_{j=1}^N \mathbf{W}_{ij} = 1 \) is included to avoid zero adjacencies. The other well-defined constraints, e.g., \( \max_{i,j} \{\mathbf{W}_{ij}\}_{i=1}^N \sum_{j=1}^N \text{ or } \|\mathbf{W}\|_F = 1 \), can also be used to rule out trivial adjacencies. It has been shown [5, 16] that the GSO (here, adjacency \( \mathbf{W} \)) of an undirected graph \( \mathcal{G} \) is diagonalizable by its orthogonal eigenmatrix, i.e., spectral templates, \( \mathbf{V} \) as

\[
\mathbf{W} = \mathbf{V} \Lambda \mathbf{V}^\top,
\]

where \( \Lambda := \text{diag}(\lambda) \in \mathbb{R}^{N \times N} \) collects the eigenvalues \( \lambda := (\lambda_1, \lambda_2, \ldots, \lambda_N)^\top \) of \( \mathbf{W} \). A graph signal \( \mathbf{x} \in \mathbb{R}^{N \times 1} \) is a mapping \( x : \mathcal{V} \rightarrow \mathbb{R}^N \) that assigns the vertices of \( \mathcal{G} \) the entries of \( \mathbf{x} \). Besides, the adjacency \( \mathbf{W} \) represents the structure of the stationary graph signal \( \mathbf{x} \) if it is the output of a (L-degree) graph (diffusion) filter \( \mathcal{H}(\mathbf{W}) := \sum_{i=0}^L h_i \mathbf{W}^i \) with the scalar coefficients \( \{h_i\}_{i=0}^L \) and input zero-mean white signal \( \mathbf{y} \) as

\[
\mathbf{x} = \mathcal{H}(\mathbf{W})\mathbf{y} = \sum_{i=0}^L h_i \mathbf{W}^i \mathbf{y} = \mathbf{V} \left[ \sum_{i=0}^L h_i \Lambda^i \right] \mathbf{V}^\top \mathbf{y},
\]

where \( \mathbf{y} \) is the Graph Fourier Transform (GFT) of \( \mathbf{x} \).

It has been shown [5] that, in the case of the input excitation \( \mathbf{y} \) being white in (3), i.e., \( \mathcal{C}_Y := \mathbb{E}\{\mathbf{y} \mathbf{y}^\top\} = \mathbf{I}_K \), the eigenvectors of the GSO \( \mathbf{W} \), i.e., \( \mathbf{V} \) in (2), share with that of the observations covariance matrix \( \mathbf{C}_X := \mathbb{E}\{\mathbf{x} \mathbf{x}^\top\} \in \mathbb{R}^{N \times N} \). Note that, due to the stationarity, the latter is equivalent to the commutativity of \( \mathbf{W} \) and \( \mathbf{C}_X \), i.e., \( \mathbf{W} \mathbf{C}_X = \mathbf{W} \mathbf{C}_X \) [23].

B. Learning Graphs From Stationary Diffused Graph Signals

1) SpecTemp [5]: In [5], the problem of inferring network structure from \( T \) independent observed diffused graph signals \( \mathcal{X} := \{\mathbf{x}_t \in \mathbb{R}^{N \times 1}\}_{t=1}^T \) has been addressed. In this way, the output covariance matrix \( \mathbf{C}_X \) can be estimated via the sample covariance over the observed graph signals \( \mathcal{X} \) as \( \mathbf{C}_X \approx \frac{1}{T} \sum_{t=1}^T \mathbf{x}_t \mathbf{x}_t^\top \), and the Eigenvalue-decomposition (EVD) of \( \mathbf{C}_X \) obtains the orthogonal eigenmatrix (i.e., spectral templates) \( \mathbf{V} \). Then, the following convex optimization was proposed [5] to recover the underlying sparse adjacency matrix \( \mathbf{W} \) and its eigenvalues \( \lambda \) as

\[
\{\mathbf{W}^*, \lambda^*\} := \arg\min_{\mathbf{W}, \lambda} \|\mathbf{W}\|_1
\]

s.t. \( \mathbf{W} = \mathbf{V} \text{diag}(\lambda) \mathbf{V}^\top, \mathbf{W} \in \mathcal{W} \). (4)

C. Product Graphs

The graph product of \( n \) factor graphs \( \{\mathcal{G}_i\}_{i=1}^n \) (with adjacencies \( \{\mathcal{W}_i\}_{i=1}^n \)) is denoted as \( \mathcal{G}_\oplus := \mathcal{G}_1 \circ \ldots \circ \mathcal{G}_n \), where \( \circ \) can be any type of Cartesian, Kronecker or Strong graph products with the adjacency \( \mathbf{W}_\oplus = \mathbf{W}_1 \circ \ldots \circ \mathbf{W}_n \in \mathbb{R}^{[n \times n] \times [n \times n]} \). Two examples are presented in the following to get insights about the real-world applicability of the product graphs. Note that other than the given examples, the modeling of graph-time dynamics [24], in which the time graph is typically a path or cyclic graph, demonstrates the flexibility of product graphs in modeling time-varying systems.

Example 1 (Dynamic network control): The dynamic controllability of a complex network consisted of two simpler (Leader-Follower Signed Factor) systems is expressed as [25]:

\[
\frac{dx(t)}{dt} = -\left(\mathbf{L}_1 \oplus \mathbf{L}_2\right) \mathbf{x}(t) + \left(\mathbf{B}_1 \oplus \mathbf{B}_2\right) \mathbf{u}(t),
\]

where \( \mathbf{x}(t) \) is the stacked system states at time \( t \), \( \mathbf{u}(t) \) denote the external input, \( \mathbf{B}_i \) states the endowed leaders by external controls for the \( i \)-th system, and \( \mathbf{L}_i \) is a (normalized) Laplacian encoding the system nodes’ connections. With the definitions \( x := \lim_{t \rightarrow \infty} \mathbf{x}(t) \) and \( \mathbf{u} := \lim_{t \rightarrow \infty} \mathbf{u}(t) \), the equilibrium expression of (5) is stated as:

\[
x = (\mathbf{I} - \mathbf{A}_\oplus)^{-1} \mathbf{y},
\]

where \( \mathbf{y} := \mathbf{B}_\oplus \mathbf{u} \) and \( \mathbf{A}_\oplus := \mathbf{I} - \mathbf{L}_\oplus \) is the Cartesian adjacency.

Example 2 (Brain signal processing): The dynamic diffusion process described in [26] demonstrates the evolution of brain region signals at time \( t \), i.e., \( x(t) \in \mathbb{R}^{N \times 1} \) with \( N \) being the number of brain regions, in terms of a priori known spatial connectivity associated with the well-known (brain signal-specific) sub-band frequencies of Delta (2–4 Hz) and Theta (4.5–7.5 Hz) with adjacencies \( \mathbf{A}^{(\delta)} \) and \( \mathbf{A}^{(\theta)} \), respectively, and an excitation signal \( \mathbf{y}(t) \) as:

\[
\frac{dx(t)}{dt} = -\mathbf{x}(t) + \left(\mathbf{A}^{(\delta)} \oplus \mathbf{A}^{(\theta)}\right) \mathbf{x}(t) + \mathbf{y}(t),
\]

where, with some mild node degree normalizations, has a unique equilibrium, with \( x := \lim_{t \rightarrow \infty} \mathbf{x}(t) \) and \( \mathbf{y} := \lim_{t \rightarrow \infty} \mathbf{y}(t) \), as [27]:

\[
x = (\mathbf{I}_N^2 - \mathbf{A}_\oplus)^{-1} \mathbf{y}.
\]
Note that both the equilibrium expressions (6) and (8) are special cases of (3), where \( \mathbf{W} \) has product structure.

### III. The Proposed Methods

Before addressing the product graph learning problem, we first propose a more computationally appropriate approach rather than the baseline SpecTemp [5] in (4) via inexact Augmented Lagrange Multipliers (IALM) [17], [18], as:

#### A. SpecTemp-IALM

Due to the symmetry and zero-diagonality of \( \mathbf{W} \), we propose a more simplified form of (4) with fewer optimization parameters (with the details in Appendix A) as:

\[
\{ \mathbf{w}^*, \lambda^* \} := \operatorname{argmin}_{\mathbf{w}, \lambda} \| \mathbf{w} \|_1 \quad \text{s.t.} \quad \mathbf{w} = \mathbf{\Gamma} \lambda, \quad \mathbf{w} \in \mathcal{W}_r,
\]

where \( \mathbf{w} := \operatorname{vechn}(\mathbf{W}) \in \mathbb{R}^{(N)^2} \) collects the non-diagonal upper triangular entries of \( \mathbf{W} \), \( \mathbf{\Gamma} := (\mathbf{M}_d \mathbf{M}_h) \mathbf{\bar{V}} \in \mathbb{R}^{(N)^2} \) with \( \mathbf{M}_d \in \mathbb{R}^{N^2 \times \frac{N(N+1)}{2}} \) and \( \mathbf{M}_h \in \mathbb{R}^{N^2 \times \frac{N(N+1)}{2}} \) being the duplication matrix [2], [28] and a matrix that \( \operatorname{vechn}(\mathbf{Z}) = \mathbf{M}_d \operatorname{vechn}(\mathbf{Z}) \) for a symmetric zero diagonal matrix \( \mathbf{Z} \), respectively, and also \( \mathbf{\bar{V}} := \mathbf{V} \odot \mathbf{V} \in \mathbb{R}^{N^2 \times N} \). Besides, \( \mathcal{W}_r \) is defined as the reduced version of \( \mathcal{W} \), containing non-diagonal upper triangular entries of valid non-zero adjacencies defined as:

\[
\mathcal{W}_r := \left\{ \mathbf{w} = \left( w_1, \ldots, w_{(N)} \right)^T \in \mathbb{R}^{(N)^2} : \sum_{i=1}^{N-1} w_i = 1 \right\}.
\]

The proposed simplified optimization (9) has \( (N)^2 + N \) optimization variables, compared to (4) with \( N^2 + N \) ones. We propose to optimize (9) via splitted IALM [17], [18] as:

\[
\{ \mathbf{w}^*, \mathbf{s}^*, \lambda^* \} := \operatorname{argmin}_{\mathbf{w}, \mathbf{s}, \lambda} \| \mathbf{w} \|_1 \quad \text{s.t.} \quad \mathbf{w} = \mathbf{\Gamma} \lambda, \quad \mathbf{s} = \mathbf{w}, \quad \lambda \in \mathcal{W}_r,
\]

where \( \mathbf{s} \) is an auxiliary variable vector. The augmented Lagrangian of (11) with \( \gamma_1 \) and \( \gamma_2 \) being the Lagrange multipliers can be written as:

\[
\mathcal{L}_{\rho}(\mathbf{w}, \mathbf{s}, \lambda) := \| \mathbf{w} \|_1 + \frac{\rho}{2} \| \mathbf{w} - \mathbf{\Gamma} \lambda \|_2^2 + \frac{\rho}{2} \| \mathbf{s} - \mathbf{\Gamma} \lambda \|_2^2 - \langle \gamma_1, \mathbf{w} - \mathbf{\Gamma} \lambda \rangle - \langle \gamma_2, \mathbf{s} - \mathbf{\Gamma} \lambda \rangle.
\]

Based on the defined Lagrangian in (12), the iteration updates of the involved optimization variables (with the details in Appendix B) are summarized in Algorithm 1, where \( \operatorname{prox} \) denotes the proximal function [29], [30] and \( \Pi_C(\mathbf{a}) \) is the Euclidean projection of vector \( \mathbf{a} \) onto the set \( \mathcal{C} \).

#### B. ProdSpecTemp

After proposing the more computationally appropriate approach SpecTemp-IALM, in this subsection, we express the product graph learning problem as:

**Problem 1:** Learn the factor graphs \( \{ \mathcal{G}_i \}_{i=1}^N \) with \( \{ \mathcal{P}_i \}_{i=1}^N \) nodes, respectively, from a stream of stationary multi-domain graph signals \( \mathcal{X} := \{ \mathbf{x}_t \in \mathbb{R}^{N \times 1} \} \).

**Input:** Graph signals

\( \mathcal{X}(\mathbf{x}_t) := \{ \mathbf{x}_t \in \mathbb{R}^{N \times 1} \}_t \) := \( \mathbf{X} \in \mathbb{R}^{N \times T} \)

**Output:** Adjacency matrix \( \mathbf{W} \in \mathbb{R}^{N \times N} \)

1. Estimate the observation covariance \( \mathbf{C}_\mathcal{X} := \mathbb{E}\{\mathbf{xx}^\top\} \) via sample covariance matrix over \( \{ \mathbf{x}_t \}_{t=1}^T \)
2. Obtain the orthogonal eigenvectors \( \mathbf{V} \) via EVD on \( \mathbf{C}_\mathcal{X} \)
3. \( \mathbf{\Gamma} := (\mathbf{M}_d \mathbf{M}_h) \mathbf{V} \), where \( \mathbf{M}_d \) and \( \mathbf{M}_h \) are the duplication matrix [28] and a matrix that \( \operatorname{vechn}(\mathbf{Z}) = \mathbf{M}_d \operatorname{vechn}(\mathbf{Z}) \) for a symmetric zero diagonal matrix \( \mathbf{Z} \), respectively, and \( \mathbf{V} := \mathbf{V} \odot \mathbf{V} \)
4. Initialization:
   \( \rho^{(0)} = 1 \), \( \lambda^{(0)} \) := \( \text{eigenvalues of } \mathbf{C}_\mathcal{X}, k = 0 \),
   \( \text{cnt} = 10^3 \), \( s^{(0)} = \mathbf{1}_N, w^{(0)} = \mathbf{1}_N \),
   \( \gamma_1^{(0)} = w^{(0)} - \mathbf{\Gamma} \lambda^{(0)} , \gamma_2^{(0)} = w^{(0)} - s^{(0)} \)
5. While convergence do
   6. \( w^{(k+1)} := \operatorname{prox}_{\rho^{(k)}}\left( \frac{\rho^{(k)} \mathbf{\Gamma} \lambda^{(k)} + \rho^{(k)} \mathbf{w}^{(k)} + \gamma_1^{(k)} + \gamma_2^{(k)}}{2 \rho^{(k)}} \right) \)
   7. \( \lambda^{(k+1)} = \mathbf{\Gamma}^\top \left( \frac{\rho^{(k)} w^{(k+1)} + \gamma_1^{(k)}}{\rho^{(k)}} \right) \)
   8. \( s^{(k+1)} = \Pi_{\mathcal{W}} \left( \frac{\rho^{(k)} w^{(k+1)} + \gamma_2^{(k)}}{\rho^{(k)}} \right) \)
   9. \( \gamma_1^{(k+1)} = \gamma_1^{(k)} - \rho^{(k)} (w^{(k+1)} - \mathbf{\Gamma} \lambda^{(k+1)}) \)
   10. \( \gamma_2^{(k+1)} = \gamma_2^{(k)} - \rho^{(k)} (w^{(k+1)} - s^{(k+1)}) \)
   11. \( \rho^{(k+1)} = \rho^{(k)} \times \text{cnt} \)
   12. \( k \leftarrow k + 1 \)
6. End while
7. Return \( \mathbf{W} = \mathbf{w}^{(k)} \)

**Algorithm 1:** SpecTemp-IALM.

From a general point of view, the product graph filter \( \mathcal{H}(\mathbf{W}) := \sum_{l=0}^{L} \hat{h}_l \mathbf{W}^l \) in (3), where \( \mathbf{W} = \mathbf{W}_1 \odot \ldots \odot \mathbf{W}_n \), is...
In this way, the following theorem helps in recovering \( \{V_{\nu_{r+1}}\}_{r=1}^{n} \) from (23).

**Theorem 1:** The covariance matrix \( C_{Z(r)} \) in (23) is diagonal with non-negative diagonal entries.

**Proof:** From (20), it can be seen that

\[
E\{zz^{\top}\} = \mathcal{H}(\{A_{i}^{\top}\}_{i=1}^{n}) V_{\varphi} \mathcal{H}(\{A_{i}^{\top}\}_{i=1}^{n})^{\top} = \mathcal{H}(\{A_{i}^{\top}\}_{i=1}^{n}) V_{\varphi} \mathcal{H}(\{A_{i}^{\top}\}_{i=1}^{n})^{\top}.
\]

Note that \( V_{\varphi}^{\top} V_{\varphi} = I_{N} \), because the orthogonality of \( V_{\varphi} = V_{1} \otimes \ldots \otimes V_{n} \) holds under the orthogonality of \( \{V_{i}\}_{i=1}^{n} \) [32]. Due to the diagonality of \( E\{zz^{\top}\} \) in (24), the entries of \( z = (z_{1}, \ldots, z_{N})^{\top} \) are uncorrelated. To better illustrate the entries of \( \mathcal{Z} \in \mathbb{R}^{P_{n} \times P_{n-1} \times \ldots \times P_{1}} \) in (21) (with dropping the index \( t \) for simplification), to relate the relation \( \mathcal{Z} = vcc(\mathcal{Z}) \) and with the notations of \( \mathcal{Z}_{(i_{1}, \ldots, i_{n})} \) and \( z_{i} \) being the \((i_{1}, \ldots, i_{n})\)-th and \( i \)-th entries of \( \mathcal{Z} \) and \( z \), respectively, it holds that [33]

\[
\mathcal{Z}_{(i_{1}, \ldots, i_{n})} = zf(i_{1}, \ldots, i_{n}), \tag{25}
\]

where

\[
f(i_{1}, \ldots, i_{1}) := i_{n} + \sum_{k=2}^{n} (i_{n-k+1} - 1) J_{k}, \tag{26}
\]

\[
J_{k} := \prod_{m=1}^{k-1} P_{n-m+1}. \tag{27}
\]

Besides, the \((i_{n-r+1}, j)\)-th element of \( \mathcal{Z}_{(r)} \in \mathbb{R}^{P_{n-r+1} \times \tilde{P}_{r}} \), with \( \tilde{P}_{r} := \prod_{m=1}^{r} P_{n-m+1} \), is\( \mathcal{Z}_{(r)}(i_{1}, \ldots, i_{n-r+1}, i_{1}) \) [22], where

\[
j := 1 + \sum_{k=1}^{n} (i_{n-k+1} - 1) \tilde{J}_{k}, \tag{28}
\]

\[
\tilde{J}_{k} := \prod_{m=1}^{k-1} P_{n-m+1}, \quad \tilde{J}_{1} := 1. \tag{29}
\]

Therefore, from (25)–(29), in summary, we have

\[
\mathcal{Z}_{(r)(m,j)} = \mathcal{Z}_{(r)}(i_{1}, \ldots, i_{n-r+2}, m, i_{n-r}, \ldots, i_{1}), \tag{30}
\]

where \( \mathcal{Z}_{(r)(m,j)} \) is the \((m, j)\)-th element of \( \mathcal{Z}_{(r)} \). In constructing the matrix \( C_{\mathcal{Z}(r)} \) in (23), the \((p, q)\)-th entry is \( \mathbb{E}\{\mathcal{Z}_{(r)(p,q)}\} \), where \( \mathcal{Z}_{(r)(p,q)} \) denotes the \( \tilde{p} \)-th row of \( \mathcal{Z}(r) \). In this way, due to the uncorrelatedness of \( z \) from (24), all the non-diagonal entries, i.e., \( p \neq q \), are zero. Regarding the diagonal entries of \( C_{\mathcal{Z}(r)} \), it can be seen that the \( m \)-th diagonal element of \( C_{\mathcal{Z}(r)} \) is \( \sum_{j=1}^{P_{r}} \mathbb{E}\{\mathcal{Z}_{(r)(m,j)}^{2}\} \), for \( m = 1, \ldots, P_{n-r+1} \).
Algorithm 2: ProdSpecTemp.

Input: Tensor graph signals \( \{X_t \in \mathbb{R}^{P_r \times \ldots \times P_1} \}_{t=1}^{\tau} \), or \( \{x_u = vec(X_t) \in \mathbb{R}^{N \times 1} \}_{t=1}^{\tau} \), and \( \{P_t\}_{r=1}^{\tau} \), where \( N = \prod_{r=1}^{\tau} P_r \).

Output: Factor adjacencies \( \{W_r \in \mathbb{R}^{P_r \times P_r} \}_{r=1}^{\tau} \)

1: for \( r = 1 \ldots n \) do
2: Perform the mode-\( r \) unfolding (matricization) on \( \{X_t\}_{t=1}^{\tau} \) and compute \( C_{X(r)} = E[X(r)X(r)^\top] \)
3: Perform EVD on \( C_{X(r)} \) and obtain \( V_{n-r+1} \)
4: Learn factor adjacency \( W_{n-r+1} \) from (9) with obtained \( V_{n-r+1} \) by performing SpecTemp-IAML (Lines 3-14 of Algorithm 1)
5: end for, and return \( \{W_r\}_{r=1}^{\tau} \)

Thus, the covariance matrix \( C_{X(r)} \) is diagonal with non-negative diagonal entries.

Therefore, due to \( C_{X(r)} \) being positive semi definite, \( V_{n-r+1} \) can be recovered by performing EVD on \( C_{X(r)} \) in (23) for \( r = 1 \ldots n \). Then, the SpecTemp-IAML can be separately applied for each factor graph to recover it via optimizing (11). The ProdSpecTemp approach is summarized in Algorithm 2. However, to ensure that the eigenvalues \( \{V_{n-r+1}\}_{r=1}^{\tau} \) can be completely recovered in (23), the following assumptions are imposed, serving as sufficient conditions for recovering the complete factor eigensystem.

**Assumption 1:** (A1) The magnitudes of filtering functions \( h(\lambda_1^{(1)}, \lambda_2^{(1)}, \ldots, \lambda_n^{(1)}) \) with the definition (18) for all \( \{i_r = 1 \ldots n \} \) are distinct values. This assumption implies that the factor adjacencies have distinct eigenvalues when \( h(\{W_i\}_{i=1}^{\tau}) \) in (15) is a function of \( W_0 = W_1 \ldots \ldots W_n \).

**Assumption 2:** (A2) The graph filter \( h(\lambda_1^{(1)}, \ldots, \lambda_n^{(1)}) \) with the definition (18) is separable [24], i.e., \( h(\lambda_1^{(1)}, \ldots, \lambda_n^{(1)}) = \prod_{r=1}^{\tau} h_r^{(r)}(\lambda_r^{(1)}) \), where \( \{h_r^{(r)}(\cdot)\} \) are factor graph filters.

Rigorous theoretical analysis of the justifications behind Assumptions A1 and A2 are presented in the next subsection, and extensive experimental analysis (e.g., practical examples) are provided in the STM A and B.

C. Theoretical Justifications of the Assumptions A1 and A2

Firstly, we should mention that the EVD form of a matrix can be useful only in the case of distinct eigenvalues [5], [24]. This reason justifies the need of A1, because, from (19), it can be admitted that the values of \( \{h(\lambda_1^{(1)}, \ldots, \lambda_n^{(1)})\} \) directly serve as the eigenvalues of \( C_X \). Also, in the generic graph learning method, i.e., SpecTemp in (4), it is assumed that the eigenvalues of the underlying graph are distinct [5]. Regarding further discussions, we first present and prove a proposition that helps in rigorous justifications, especially for A2.

**Proposition 1:** Under the Assumption A1, and the excitation signal being zero-mean and white, i.e., \( E(yy^\top) = I_N \), \( C_{X(r)} \) in (23), for \( r = 1 \ldots n \) can be presented as:

\[
C_{X(r)} = \sum_{m=1}^{P_{n-r+1}} v_m^{(n-r+1)}v_m^{(n-r+1)^\top} \lambda_m^{(n-r+1)},
\]  

where \( v_m^{(n-r+1)} \) is the \( m \)-th eigenvector of factor adjacency \( W_{n-r+1} \) (and, eventually, \( C_{X(r)} \)), and

\[
\hat{\lambda}_m^{(n-r+1)} := \sum_{i_1=1}^{P_1} \ldots \sum_{i_{n-r+1}=1}^{P_{n-r+1}} \sum_{i_{n-r+2}=1}^{P_{n-r+2}} \ldots \sum_{i_n=1}^{P_n} h_m^{(n-r+1)} \left( \{\lambda_{i_k}^{(k)}\}_{k=1}^{n} \right),
\]

(32)

\[
\hat{\lambda}_m^{(n-r+1)} := h \left( \lambda_1^{(1)}, \ldots, \lambda_{n-r}^{(n-r)}, \lambda_{n-r+1}^{(n-r+1)}, \lambda_{n-r+2}^{(n-r+2)}, \ldots, \lambda_n^{(n)} \right).
\]

(33)

**Proof:** Building upon the Theorem of Proposition 1 and without redundant explanations, it only needs to express the diagonal entries of \( C_{X(r)} \) (i.e., \( \sum_{j=1}^{P_r} E[z_j^{(i_1, \ldots, i_r)}z_j^{(i_1, \ldots, i_r)^\top}] \)) in terms of the graph eigenvalues. From (24) and leveraging the definitions (26)–(30) for indexing the entries of \( z \), one can write

\[
E[z_j^{(i_1, \ldots, i_r)}z_j^{(i_1, \ldots, i_r)^\top}] = \sum_{j=1}^{P_r} E[z_j^{(i_1, \ldots, i_r)}z_j^{(i_1, \ldots, i_r)^\top}] = |h(\lambda_1^{(1)}, \ldots, \lambda_n^{(n)})|^2.
\]

Then, it holds that

\[
\hat{\lambda}_m^{(n-r+1)} := \sum_{j=1}^{P_r} E[z_j^{(i_1, \ldots, i_r)}z_j^{(i_1, \ldots, i_r)^\top}] = \sum_{i_1=1}^{P_1} \ldots \sum_{i_{n-r+1}=1}^{P_{n-r+1}} \sum_{i_{n-r+2}=1}^{P_{n-r+2}} \ldots \sum_{i_n=1}^{P_n} h_m^{(n-r+1)} \left( \{\lambda_{i_k}^{(k)}\}_{k=1}^{n} \right)^2,
\]

(35)

where we defined

\[
\hat{\lambda}_m^{(n-r+1)} := \sum_{j=1}^{P_r} E[z_j^{(i_1, \ldots, i_r)}z_j^{(i_1, \ldots, i_r)^\top}] = h(\lambda_1^{(1)}, \ldots, \lambda_{n-r}^{(n-r)}, \lambda_{n-r+1}^{(n-r+1)}, \lambda_{n-r+2}^{(n-r+2)}, \ldots, \lambda_n^{(n)}),
\]

which concludes the proof.

From the obtained EVD form of (31) for \( r = 1 \ldots n \) in Proposition 1, the justifications for needing the assumptions, especially A2, are well-illustrated. Precisely, considering \( n = 2 \) for simplicity, with a separable graph filter \( h(\lambda_1^{(1)}, \lambda_2^{(1)}) = h(\lambda_1^{(1)})h(\lambda_2^{(1)}) \) (i.e., Assumption A2), the eq. (31) with \( r = 1 \) turns to

\[
C_{X(1)} = \sum_{m=1}^{P_2} v_m^{(2)}v_m^{(2)^\top}h(\lambda_2^{(2)})^2 \sum_{i=1}^{P_1} |h(\lambda_1^{(1)})|^2.
\]

(37)

Next, the assumptions of distinct magnitudes \( \{h(\lambda_1^{(1)}, \lambda_2^{(1)})\}_{i=1}^{P_1} \) and also distinct factor graph eigenvalues \( \{\lambda_1^{(1)}, \lambda_2^{(1)}\}_{i=1}^{P_1} \) (i.e., Assumption A1) imply that \( |h(\lambda_2^{(2)})|^2 \sum_{i=1}^{P_1} |h(\lambda_1^{(1)})|^2 \) in (31) have distinct values for \( m = 1 \ldots P_2 \). Therefore, the eigenvectors \( \{v_m^{(2)}\}_{m=1}^{P_2} \) can be fully recovered and used in further graph learning optimizations. Similar procedure for \( C_{X(2)} \) and \( \{v_m^{(1)}\}_{i=1}^{P_1}, \), i.e., \( r = 2 \) in (31), gets the desired results. In STM A and B, we provide several intuitive and experimental examples of what could occur if these
assumptions are violated, especially in the case of Kronecker product graphs.

Remark 1: In the following, some well-used separable graph filters associated with different kinds of product graphs and their representation in the graph frequency domain (specified with left-right arrows (i.e., \( \leftrightarrow \)) are listed: 

\[
\mathcal{H}_\otimes(W_\otimes) := e^{\tau W_\otimes} \leftrightarrow h_\otimes(\{\lambda_{ir}\}_{r=1}^n) := \prod_{r=1}^n e^{\tau \lambda_{ir}}, \\
\mathcal{H}_\circ(W_\circ) := W_\circ \leftrightarrow h_\circ(\{\lambda_{ir}\}_{r=1}^n) := \prod_{r=1}^n \lambda_{ir}, \\
\mathcal{H}_\boxplus(W_\boxplus) := I_N + W_\boxplus \leftrightarrow h_\boxplus(\{\lambda_{ir}\}_{r=1}^n) := \prod_{r=1}^n (1 + \lambda_{ir}),
\]

(38)

where \( \tau := \frac{1}{\max_{n=1,..,N} d_i} \) and \( d_i \) denote the degree of the \( i \)-th node of \( W_\otimes \). Due to obtaining the graph eigenvalues of the Kronecker product graph solely from multiplied eigenvalues of the factor graphs (i.e., \( h_\otimes(.) \) in (38), learning Kronecker-structured products is more challenging than the other types, especially in the case of existing zero (or even almost zero) eigenvalues which is quite common in practical settings. Further related experimental discussions and examples can be found in STM A and B.

Remark 2: Recovering factor graphs is often the main purpose of real-world multi-domain data applications. However, if one also needs to infer the existing type of graph product, a naive (and computationally expensive) approach is as follows: After learning factor graphs using the proposed ProdSpecTemp method, recovering the product graph \( W_\otimes \) directly from the multi-domain graph signals (via optimizing (4) by SpecTemp-IALM) is needed. Then, the next step is checking which type of Cartesian, Kronecker, or Strong products leads to minimum error \( \|W_\otimes - W_\circ \circ ... \circ W_n\|_F \). Devising more computationally appropriate approaches is of interest, but it is beyond the scope of the current paper and will be addressed via the concept of graph filter identification [34] in future work.

D. Sufficient Conditions for Recovery of Product Graphs

Defining \( w := (w_1^T, ..., w_n^T)^T \in \mathbb{R}^{\sum_{i=1}^n (\binom{\mathcal{F}}{2})^1} \) to collect the non-diagonal upper triangular entries of the adjacencies \( \{W_i \in \mathbb{R}^{P_i \times P_i}\}_{i=1}^n \), the main goal here is to recover the unique and sparsest version of \( w \), refereed as \( w_0^* \), for learning all the factor graphs in a unified minimization. Using the following definitions (39)–(43), Theorem 2 (as a comprehensive generalization of the results in [5] to the case of product graphs) provides sufficient conditions for the minimizations within Algorithm 2, which can be cast as a unified minimization, to obtain a unique solution \( w \) with coincident to \( w_0^* \).

\[
\Psi := \text{bdia} \{(\alpha_i \otimes I_{\binom{\mathcal{F}}{2}})^n\}_{i=1}^n \in \mathbb{R}^{(\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2) \times (\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2)},
\]

(39)

\[
\mathbf{M}^{(i)} := \left( C_{X_{(n+1)}^i} \oplus -C_{X_{(n+i)}^i} \right) M^0_i M^0_h \in \mathbb{R}^{P_i \times (\binom{\mathcal{F}}{2})^2},
\]

(40)

\[
\Theta := \text{bdia} \left( \left\{ M^{(i)}_{j=1}^n \right\} \right) \in \mathbb{R}^{\sum_{i=1}^n P_i \times (\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2)},
\]

(41)

\[
\Phi := (\Theta^T, E^T)^T \in \mathbb{R}^{(n+\sum_{i=1}^n P_i) \times (\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2)},
\]

(42)

\[
b := (0^T, 1^T)^T \in \mathbb{R}^{(n+\sum_{i=1}^n P_i) \times 1}; \quad \tilde{n} := \sum_{i=1}^n P_i^2,
\]

(43)

where, in the \( i \)-th row of \( E \in \mathbb{R}^{n \times (\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2)}, \) the \( (1 + \sum_{j=1}^i (P_j^2)) - \text{th to} (P_i - 1 + \sum_{j=1}^i (P_j^2)) \)-th entries are one, and the others are zero. The \( \{\alpha_i\}_{i=1}^n \) are some appropriate positive scalars. Also, we define the sets \( \mathcal{I} \) and \( \mathcal{J} \) containing the indices of non-zero entries of \( \Psi w_0^* \) and \( w_0^* \), respectively.

Theorem 2: If the minimizations within the proposed Prod-SpecTemp method related to learning factor adjacencies, i.e., Line 4 of Algorithm 2, are feasible and the following conditions are satisfied, then \( w = w_0^* \).

C1: \( \text{rank}(\Phi_{\mathcal{J}}) = |\mathcal{J}| \), i.e., \( \Phi_{\mathcal{J}}^T \) is full row rank,

C2: There exists a constant \( \delta > 0 \) such that

\[
\|\Psi_{\mathcal{I}}^T (\delta^{-2} \Phi_{\mathcal{J}}^T \Phi + \Psi_{\mathcal{I}}^T \Psi_{\mathcal{I}})^{-1} \Psi_{\mathcal{I}}\|_{M(\infty)} < 1.
\]

(44)

Proof: For some appropriate positive constants \( \{\alpha_i\}_{i=1}^n \), the separated optimizations for learning factor adjacencies in Algorithm 2 can be unified in a single optimization as

\[
\min_{\{W_i\}_{i=1}^n} \sum_{i=1}^n \frac{\alpha_i}{2} \|W_i\|_1 \s.t. C_{X_{(n+1)}^i} W_i = W_i C_{X_{(n+i)}^i}, \quad W_i \in \mathcal{W}; \quad i = 1, ..., n
\]

(45)

where, for the sake of simplicity, the equality of the eigenbasis of \( C_{X_{(n+1)}^i} \) with the adjacency \( W_i \) in (23) has been equivalently replaced by their commutativity [23]. Then, using some algebraic simplifications (with details in Appendix C) and definitions (39)–(43), the optimization (45) is observed as a special case of the \( \ell_1 \)-analysis [35] problem

\[
\min_{w} \|\Psi w\|_1 \quad \text{s.t.} \quad \Phi w = b.
\]

(46)

in which the sufficient conditions for recovery of the unique and sparsest solution, i.e., the solution to the \( \ell_0 \) alternative of (46), are stated as [35]

a) \( \ker(\Psi_{\mathcal{I}}) \cap \ker(\Phi) = \{0\} \),

b) There exist \( y \in \mathbb{R}^{(\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2)} \) such that \( \Psi_{\mathcal{I}}^T y \in \text{Im}(\Phi^T), \)

\[
y_{\mathcal{I}} = \text{sign}(\Psi_{\mathcal{I}} w_0^*), \quad \|y_{\mathcal{I}}\|_\infty < 1.
\]

Note that although the constraint system in (46) is overdetermined (i.e., \( \Phi \in \mathbb{R}^{(n+\sum_{i=1}^n P_i) \times (\sum_{i=1}^n (\binom{\mathcal{F}}{2})^2)} \)), the feasibility of (45) guarantees the compatibility of the equation system [5]. In the following, we show that satisfying conditions C1 and C2 imply that of the conditions a) and b), respectively. Besides, the non-negativity constraint in \( W \) or \( W_i \) has been excluded since if minimization (46) can recover the sparsest solution, its more (reasonably) constrained alternative can also recover it [5].
Due to the diagonal structure of $\Psi$ and $\{\alpha_i > 0\}_{n=1}^n$, if a generic vector $s \neq 0$ belongs to the null space of $\Psi_{x'}$, it must satisfy $s_{x'} = 0$. Let $\Phi_{i,:}$ denote the $i$-th column of $\Phi$, then one can write $\Phi s = \sum_{i \in J} s_i \Phi_{i,:}$. However, under the linearity independence of $\{\Phi_{i,:}\}_{i \in J}$ or equivalently full row rank of $[\Phi_{i,:}]_{J}$ in the condition $C1, \Phi s \neq 0$. Therefore, this $s \neq 0$ can not be in null space of $\Phi$, and consequently the condition a) holds.

To show that satisfying the condition $C2$ implies b), let a vector $r$ such that $\Psi^\top r = \Phi^\top r$. Then, to satisfy the other constraints in b), one can consider the following constrained $\ell_2$-minimization problem [5]

$$\min_{r,y} \delta^2 \|r\|_2^2 + \|y\|_2^2 \text{ s.t. } \Psi^\top y = \Phi^\top r, \ y_{I_2} = \text{sign}(\Psi_{x'w_0}^\top),$$

where $\delta > 0$ is a tuning constant. The term $\delta^2 \|r\|_2^2$ brings numerical stability and also guarantees a closed-form solution. Due to holding the constraints in b) by satisfying constraint in (47), only holding $\|y_{I_2}\|_\infty < 1$ remains while $y^*$ is the solution to (47). To do so, firstly, the equality $\Psi^\top y = \Phi^\top r$ can be rewritten as $\Psi_{I_2}^\top \text{sign}(\Psi_{x'w_0}^\top) = -\Psi_{I_2}^\top y_{I_2} + \Phi^\top \delta^{-1} \delta r$. Then, the minimization (47) can be rewritten as

$$\min_{t} \|t\|_2^2 \text{ s.t. } \Psi_{I_2}^\top \text{sign}(\Psi_{x'w_0}^\top) = Qt,$$

where $t := (\delta y_{I_2}^\top - y_{I_2}^\top)^\top$ and $Q := (\delta^{-1} \Phi^\top, \Psi_{I_2}^\top)$. The optimal minimum-norm solution to (48) is stated as $t^* = Q^\perp \text{sign}(\Psi_{x'w_0}^\top)$, where it obtains

$$y_{I_2}^* = -\Psi_{I_2}^\top (\delta^{-2} \Phi^\top + \Phi \Psi_{I_2}^\top)^{-1} \Phi^\top \text{sign}(\Psi_{x'w_0}^\top).$$

The existence of the inverse in (49) is guaranteed by condition a). One can bound the $\ell_\infty$ norm of $y_{I_2}^*$ as $\|y_{I_2}^*\|_\infty \leq \|\Psi_{I_2}^\top (\delta^{-2} \Phi^\top + \Phi \Psi_{I_2}^\top)^{-1} \Phi^\top \text{sign}(\Psi_{x'w_0}^\top)\|_{M(\infty)} = \gamma$, because $\|\text{sign}(\Psi_{x'w_0}^\top)\|_\infty = 1$. Therefore, holding the condition $C2$ implies the condition b), and the proof is concluded.

Condition $C1$ in Theorem 2 ensures the uniqueness of the solution to (46). Precisely, violating $C1$ implies that there would be a zero-non zero vector $s$ in null space of $[\Phi_{i,:}]_{J}$ such that $[\Phi_{i,:}]_{J} s_{x'} = 0$ and $s_{x'} = 0$ satisfying $\Phi (w_0^* + s) = \Phi w_0^*$. Therefore, there would exist a non-empty interval $S = [w_0^* - \mu s, w_0^* + \mu s]$ in which $\mu > 0$ is sufficiently small such that $\|\Psi \|_1$ is linear in $S$ and $\Phi w_0^* = \Phi s$ for $s \in S$ [34] implying that the solution $w_0^*$ to (46) is not unique. Condition $C2$ introduces a strictly-complementary dual certificate for ensuring the optimal and coincident with the $\ell_0$ norm minimization counterpart to (46), as detailed in the proof of Theorem 2.

The presented unification in (46) provides analysis of different aspects of the problem within the concept of $\ell_1$-analysis [5, 35]. For example, when having possible (covariance) estimation error, the equality constraint in (46) can be relaxed with a (noise level-related) constant $\epsilon$ as

$$\min_{w} \|\Psi w\|_1 \text{ s.t. } \|\Phi w - b\|_2 \leq \epsilon,$$

Defining $\alpha_{\max i} := \max_i \{\alpha_i\}_{i=1}^n = 1$ (after scaling) and $\alpha_{\min i} := \min_i \{\alpha_i\}_{i=1}^n$ in (39), the recovery robustness under relaxing the equality constraint in (50) is investigated in the next theorem as follows:

**Theorem 3:** Let $\Phi w_0^* - b = \eta$ and $\epsilon = \|\eta\|_2$, for arbitrary noise vector $\eta$. If all the stated conditions in Theorem 2 hold, then any minimizer $w$ to (50) satisfies

$$\|\Psi (\hat{w} - w_0^*)\|_1 \leq C\epsilon; \ C := 2C_1 + 2C_2 \sqrt{C_3},$$

where $w_0^*$ is the true (and sparsest) solution to (46), and

$$C_1 := \frac{\sqrt{\|\eta\|_2}}{\sigma_{\min} (\Phi)}, \ C_2 := \frac{1 + \frac{\alpha_{\max}}{\alpha_{\min}} \|\Phi\|_{M(2)} C_1}{1 - \gamma},$$

and $\sigma_{\min} (A)$ denotes the smallest singular value of matrix $A$.

**Proof:** The proof follows from applying Claim 2 of Theorem 2 in [35]. Note that since $\Phi$ is not a full row rank matrix, $C_3$ relies on $\|\Phi\|_{M(2)}$ instead of $\|\Phi\|^{-1}$ in [35]. Besides, by assuming $\|y_{I_2}\|_\infty \leq \gamma$ (as shown in Theorem 2) and $\|y\|_2 < \sqrt{\sum_{i=1}^{n} (P_i^2)}$ (because the magnitudes of the entries in $y$ are at most 1 as shown in Theorem 2), $C_2$ and $C_3$ are independent of the dual certificate $y$ in Theorem 2.

From the findings of Theorem 3, it can be admitted that the desired GL performance is upper bounded by the noise power $\epsilon$, and implicitly graph sizes $\{P_i\}_{i=1}^n$.

**E. Computational Complexity Analysis**

It can be seen that ignoring product graph structures in minimizing (4) optimizes $N^2 + N = \prod_{i=1}^{n} P_i^2 + \prod_{i=1}^{n} P_i$ free variables. However, ProdSpecTemp considers $n$ GL problems on the factor graphs $\{W_i\}_{i=1}^n$ separately, which reduces the number of optimization variables to $\sum_{i=1}^{n} P_i^2 + \sum_{i=1}^{n} P_i$. This number can be even more reduced to $\sum_{i=1}^{n} (P_i^2 + P_i)$ by exploiting the proposed SpecTemp-IALM (Algorithm 1), instead of SpecTemp [5], in the mentioned separated optimizations. Besides, it has been shown [5] that the computational complexity of (4) is dominated by EVD of the observation covariance matrix, which requires $O(N^3) = O(\prod_{i=1}^{n} P_i^3)$ operations, whereas, due to the separability of recovering factor graphs in the proposed Algorithm 2, the proposed approach requires significantly reduced $\sum_{i=1}^{n} O(P_i^3)$ operations.

**IV. EXPERIMENTAL RESULTS AND DISCUSSION**

This section examines the performance of product graph learning in different scenarios. Note that further analyses of different scenarios are given in Appendix D-F, and, since recovering Kronecker product graphs are more challenging (as presented in Remark 1), additional related discussions are provided in STM A and C.
and $PGL$

denotes the set $\{0, 2, 0\}$ \{0, 2, 0\}.

As acceptable approximation and choose...

$H$ is typically a Precision

$G$ in such a

$\approx \times (1$ as:

product graph signals $(\times$ over the generated 100 independent instances of 15 and 10-node ER factor graphs. (c) The effect of unknown (not accurately investigated by the F1-measure of the resulted product graph averaged over 20 noisy independent realizations (SNR = 1.5) across $P_1$ and $P_2$ and different number of graph signals $T$, where $P_1^{\text{err}} = 15$ and $P_2^{\text{err}} = 10$.

A. Comparison to the Related Work

In this subsection, the proposed ProdSpecTemp method is

compared with the related methods BiGLasso [13] and PGL [11] in Fig. 1(a) for learning Erdős-Rényi (ER) factor graphs $G_1$ and $G_2$ ($P_1 = 15$, $P_2 = 10$, $N = P_1P_2 = 150$) with the edge probability $p_{\text{ER}} = 0.15$ simulated by GPSPBOX [36]. Note that the output of the BiGLasso method [13] is typically a Precision matrix $(P)$, and thus we first project onto the space of valid Laplacian matrices $(L)$, as follows:

$$\min_{L \in L} \|P - L\|_F^2, \quad (53)$$

where the set

$L := \{L \in S_{+}^{N \times N} : L1_N = 0_N, L_{ij} = L_{ji} \leq 0; \forall i \neq j\} \quad (54)$

is defined as the set of valid Laplacian matrices of the undirected and weighted graphs with $N$ nodes, and $S_{+}^{N \times N}$ denotes the set of Positive Semi Definite matrices of size $N \times N$. Then, the obtained Laplacian matrices $(L)$ from PGL and BiGLasso methods get the (undirected) adjacencies $(W)$ as $W = \text{diag}(L) - L$. The $T$ product graph signals $(T \in \{10^{2.5}\})$ are generated from Cartesian, Kronecker, and Strong product graph diffusion processes with excitation signals $(\{x_t \sim N(0, \mathbf{I}_N)\})$ and separable graph filters in (38). However, we note that exploiting the separable graph filter $h_\bigoplus(.)$ in (38) could cause severe practical issues in recovering Kronecker product graphs, which has been discussed in detail in STM A and B. In summary, existing zero (or even almost zero) eigenvalues of the involved factor graphs are directly duplicated in the spectrum of the Kronecker product graphs, which obscures the effect of the corresponding product eigenvectors in the output data challenging the exact recovery of the factor eigenvectors. There is an alternative separable graph filter to $H_\bigotimes(.)$ as:

$$H_\bigotimes^{(\alpha)}(\{W_r\}_{r=1}^n) := \frac{1}{\alpha^n} (I_N + [(\alpha W_1) \boxplus \ldots \boxplus (\alpha W_n)]), \quad (55)$$

$$h_\bigotimes^{(\alpha)}(\lambda_{i_r})_{r=1}^n := \frac{1}{\alpha^n} \prod_{r=1}^n (1 + \alpha \lambda_{i_r}). \quad (55)$$

Next, we must choose a sufficiently large $\alpha > 0$ in such a way that $H_\bigotimes^{(\alpha)}(.) \approx H_\bigotimes(.)$, where it is obvious that $\lim_{\alpha \to \infty} H_\bigotimes^{(\alpha)}(.) = H_\bigotimes(.)$. However, in the case of severely small eigenvalues (e.g., sparse (ER) graphs), choosing an appropriate $\alpha$ could be computationally expensive. Another more numerically stable for $n = 2$ binary factor graphs could be

$$H_\bigotimes^{(\alpha)}(W_1, W_2) := \frac{1}{\alpha^2} (I_N + \alpha^2 W_1 \otimes W_2).$$

To tune $\alpha$, we vary it in the span of $\{4 : 12\}$, and monitor the Relative Graph Error (RGE) $\ell_2^{err} := \frac{\|H_\bigotimes^{(\alpha)}(W_1, W_2) - H_\bigotimes(W_1, W_2)\|_F^2}{\|H_\bigotimes(W_1, W_2)\|_F^2}$. In this way, the averaged RGE (in the error-bar form to monitor the standard deviations as well) over the generated 100 independent instances of 15 and 10-node ER factor graphs are depicted in Fig. 1(b).

We consider $\ell_2^{err} \approx 0.01$ as acceptable approximation and choose...
\(\bar{\alpha} = 10\). To make the settings more challenging and also investigate the asymptotic behavior of the involved methods on noisy observations, we add Gaussian noise with Signal to Noise Ratio (SNR) = -20db to the resulted product graph signals, and the product and factor GL results are illustrated in Fig. 1(a), in which the GL performance is evaluated via the popular F1-measure metric \[ F1 := \frac{tp}{tp + fn + fp}, \] where \(tp\), \(fn\) and \(fp\) stand for true positive, false negative and false positive, respectively. We recall that the ER graphs are typically binary, and the learned graphs have continuous edge values. Therefore, it makes sense to use F1 metric \[2\], which measures the graph topology inference harmonic accuracy. As can be seen in this figure, with increasing the number of graph signals \(T\) at hand, the performances in (almost) all cases improve. Due to the suitability of the BiGLasso \[13\] and PGL \[11\] methods for recovering imperfect products \(P \neq 9\), we also estimate them in the case of having ground truth product graphs, Fig. 1(c) shows the F1-measure of the resulted product graph averaged over 20 noisy independent realizations (SNR = -20db) across \(P_1\) and \(P_2\) in the span of \(\in (5, 10, 15, 30)\) and the different number of graph signals \(T \in \{10^{0.3}\}\), where \(P_1^{true} = 15\) and \(P_2^{true} = 10\). As can be seen in this figure, the point \((P_1, P_2) = (P_1^{true}, P_2^{true}) = (15, 10)\) has the highest F1-measure in all values of \(T\). Besides, even in the points \((P_1, P_2) \neq (P_1^{true}, P_2^{true})\), the F1-measure does not drop drastically in many cases, especially in the case of having a fair number of graph signals, e.g., \(T \geq 1000\). This observation implies the robustness of the proposed method against the unknown (or not accurately estimated) hyperparameters.

### B. Hyperparameter Analysis \((\{P_i\}_{i=1}^n)\)

First, we recall that the only notable hyperparameter of the proposed ProdSpecTemp method is the number of nodes of the factor graphs, i.e., \(\{P_i\}_{i=1}^n\). By considering \(n = 2\), to analyze the effect of unknown \(P_1\) and \(P_2\) and also estimate them in the case of having ground truth product graphs, Fig. 1(c) shows the F1-measure of the resulted product graph averaged over 20 noisy independent realizations (SNR = -20db) across \(P_1\) and \(P_2\) in the span of \(\in (5, 10, 15, 30)\) and the different number of graph signals \(T \in \{10^{0.3}\}\), where \(P_1^{true} = 15\) and \(P_2^{true} = 10\). As can be seen in this figure, the point \((P_1, P_2) = (P_1^{true}, P_2^{true}) = (15, 10)\) has the highest F1-measure in all values of \(T\). Besides, even in the points \((P_1, P_2) \neq (P_1^{true}, P_2^{true})\), the F1-measure does not drop drastically in many cases, especially in the case of having a fair number of graph signals, e.g., \(T \geq 1000\). This observation implies the robustness of the proposed method against the unknown (or not accurately estimated) hyperparameters.

### C. Learning More Than Two Factor Graphs

To illustrate the flexibility of the proposed ProdSpecTemp method in recovering more than two factor graphs, we generate three factor ER graphs with \(P_1 = 12\), \(P_2 = 10\) and \(P_3 = 9\) number of nodes with the edge probability \(p_{ER} = 0.3\) for generating Cartesian and Strong product graphs \(G_1, G_2, G_3\). Then, \(T\) product graph signals, where \(T \in \{10^{0.3}\}\), are simulated via (3) with the excitation signals \(\{y_t \sim \mathcal{N}(0, I_N)\}_{t=1}^T\) and separable graph filters in (38). Afterward, the Gaussian additive noise with SNR = -10db is added to the generated graph signals. The average of the graph recovery performance corresponding to the three factor graphs over twenty independent realizations are illustrated in Fig. 2 (with added Area Under Curve (AUC) and RGE metrics). This figure shows that, although recovery of the Strong products is more challenging than the Cartesian ones, the proposed ProdSpecTemp method has successfully recovered the factor graphs with improved performance by increasing the number of observed graph signals \(T\). Besides, the average of the true and learned third factor graphs \((P_3 = 9)\) are provided in Fig. 3, which supports the success in graph recovery results and deductions in Fig. 2.

### D. Evaluating Sufficient Recovery Conditions

To evaluate the derived sufficient conditions for recovery of the sparsest graph in Theorem 2, we simulated 500 Cartesian product graphs from two factor ER graphs \((P_1 = 15, P_2 = 10, p_{ER} = 0.5)\) and generated \(T = 10000\) independent noiseless graph signals from them. The other settings were similar to Section IV-A. We selected the value of \(\delta\) in (44) so as to obtain a minimum value for \(\gamma\). Note that the condition \(C_1\) was satisfied in all realizations, as is (mostly) the case with ER graphs \([5, 23]\). The histogram of the product graph recovery state (perfect or imperfect) across the values of \(\gamma\) (rounded with a step of 0.25)

---

*Fig. 2. The graph recovery performance of the three factor graphs averaged over twenty independent realizations.*

*Fig. 3. The average of the true and learned third factor adjacencies \((P_3 = 9)\) from Cartesian (top row) and Strong (bottom row) products across varying number of signals \((T)\).*
is depicted in Fig. 4(a). As can be seen in this figure, realizations associated with $\gamma < 1$ led to perfect recovery, and a (small) portion with $\gamma > 1$ did not recover the graph topologies perfectly. Besides, since a (small) portion of realizations with $\gamma$ slightly more than one (black dashed line in this figure) led to imperfect recovery, the tightness of the derived bound in Theorem 2 is also experimentally validated. In Fig. 4(b), the averaged RGE over $T = 50$ independent realizations across different SNR (related to the inverse noise level in Theorem 3) associated with different graph sizes ($P_1 = 7$ and $P_2 \in \{7, 14, 28\}$) are plotted. It can be seen that the recovery quality is related to both the noise levels (SNRs here) and graph sizes, where the consistency of the product graph recovery experimentally validates the stated claims in Theorem 3.

**E. Learning Brain Functional Connectivity of Sleep Stages**

To investigate the applicability of the proposed ProdSpecTemp on real-world Electroencephalogram (EEG) brain signals, we select the MASS-SS3 dataset [8] consisting of Polysomnogram (PSG) data from 62 healthy subjects during sleep, i.e., 20 EEG, 3 Electromyogram (EMG), 2 Electrooculogram (EOG), and 1 Electrocardiogram (ECG) channels with a sampling frequency of 256 Hz. The sleep scoring of the thirty-second sleep epochs into five sleep stages (Wake, Rapid Eye Movement (REM), N1, N2, and N3) is performed by sleep experts using the American Academy of Sleep Medicine (AASM) standard [37]. With no specific preprocessing on the PSG signals, Differential Entropy (DE) features in 9 crossed frequency bands 0.5–4 Hz, 2–6 Hz, 4–8 Hz, 6–11 Hz, 8–14 Hz, 11–22 Hz, 14–31 Hz, 22–40 Hz, and 31–50 Hz were extracted from each of PSG channels. As well as the spatial EEG electrodes’ connectivities having 26 spatial factor graph nodes, the temporal connections of neighbor thirty-second sleep epochs have also been reported in the relevant literature to be efficient in sleep staging [38], [39], [40]. We consider four epochs before and after the target epoch; therefore, temporal factor graphs have nine nodes.

It has been shown that analysis of the spatial graphs can represent valuable information about brain functional connectivity during sleep [3], [41], where Nguyen et al. [42] showed that the functional connectivity of the brain varies across different sleep stages. In this way, we plot and analyze the averaged binarized learned spatial/temporal graphs based on a specific threshold determined in a way that leads to the most significant differences between pairs of sleep stage graphs. For more details, please refer to the provided STM D.

Fig. 5(a), i.e., the averaged learned spatial graphs related to the EEG channels, shows that during non-REM sleep stages (i.e., N1-3), the brain connectivities decrease compared to the Wake state. This observation is consistent with the findings of previous studies admitting reduced hypothalamic functional connectivity, which may be synchronized for establishing and maintaining sleep [43], [44]. Moreover, this figure shows an increase in connectivity and activity of the occipital region corresponding to the REM vs. non-REM, which is consistent with the results of [43], [45] that illustrate the occipital metabolism. Therefore, its connectivity with the other brain regions increases during REM vs. non-REM. Besides, the N1 stage has the most connection-based activity vs. N2 and N3. Also, N3 has the lowest activity. These observations are also quite consistent and supported by the neuroscientific research literature [46], [47] implying that the N1 stage is a complicated stage and the brain is still active, unlike the N3 stage, which is a typical deep stage when the brain is in its low active mode.

Fig. 5(b) shows the averaged learned temporal graphs. It can be seen that the target epoch is (mostly) connected with its before/after epochs. However, the key point of this figure is that even distant neighboring epochs (e.g., $t - 4$ and $t + 3$) do not necessarily follow a cycle-like graph structure, which means that different sleep epochs could have connections, where it is not limited to only before/after neighbor epochs.

**F. COIL-20 Dataset**

As another real-world application of the proposed method, in this subsection, we consider the COIL-20 dataset [9] consisting of grayscale images (with a size of $128 \times 128$ pixels) of 20 objects captured on a 5-degree interval of a turntable in front of
Fig. 5. (a) The averaged learned brain spatial graphs related to the connectivity between EEG channels. (b) The averaged learned temporal graphs illustrating the connections between neighbor thirty-second sleep epochs.

Fig. 6. (a) The learned 6400 × 6400 and 36 × 36 adjacencies corresponding to the pixel (left panel) and degree (right panel) graphs, respectively. (b) Two examples of cases where the objects are quite in front and back view positions. (c) The learned object graph shows the meaningful connectivities (clusters) between similar objects, e.g., cars.

a fixed camera. For this experiment, we selected eight objects in 10-degree intervals, resulting in 36 ° images per object. Besides, we downsampled the images to the size of 80 × 80 via the bicubic interpolation approach. The mentioned structure of the data at hand allows us to consider the whole dataset as one four-way tensor $X \in \mathbb{R}^{8 \times 36 \times 80 \times 80}$. Therefore, we seek to learn meaningful and interpretable object, degree, and pixel connectivities by performing the proposed ProdSpecTemp on $X$, where $T = 1$ in this experiment.

The learned 6400 × 6400 and 36 × 36 adjacencies corresponding to the pixel and degree graphs are illustrated in the left and right panels of Fig. 6(a), respectively. Note that the shown pixel graph is the Strong product of the learned 80-node row and column graphs, as mentioned to be appropriate and comprehensive to model the pixel connectivities in image processing literature [7]. As can be seen in this figure, the resulted pixel graph is approximately divided into 80 × 80 strong connection intervals (roughly), showing the actual size of the images. This implies that neighboring pixels within the images are internally more connected rather than the pixels of the other ones. On the other hand, in the resulted degree graph, notable path structure connections along the main diagonal (black circles in the figure) rely on the connections between before/after degree images. The blue circles show the connection between zero and 360-degree turns, which is quite expected. Another interesting point is the connections corresponding to the green circle area, which is associated with the cases where objects are quite (or almost) in front and back view positions. Fig. 6(b) shows two examples of this scenario. These findings are consistent and supported by previous pioneer work on this kind of data, e.g., [13]. In
Fig. 6(c), the learned object graph shows meaningful connectivities between similar objects and in similar poses, e.g., cars. This shows the opportunity to modify the proposed method to learn rank-constrained structures (Laplacians) for use in clustering applications (Multi-View Object Clustering) in future work. More analysis of this dataset can be found in STM E.

V. CONCLUSION

In this article, we proposed a GL approach inferring product graphs from spectral templates of multi-domain graph signals for possibly more than two factor graphs with significantly reduced computational complexity than the generic method. Our method is not limited to any specific type of graph product, in contrast to the rare current approaches for inference from smooth graph signals defined only on Cartesian product graphs. A rigorous theoretical analysis of the proposed method for recovering sparse graphs was devised and experimentally validated. In addition to outperforming the currently limited methods in the synthetic diffused stationary graph signals, our approach also infers (possibly more than two) meaningful and interpretable factor graphs from sleep brain signals and multi-view object images, which are supported by expert-related pioneer previous work.

APPENDIX

A. Simplifications of Optimization (9)

The constraint $W = V \Lambda V^T$, where $V := (v_1, \ldots, v_N)$, can be rewritten using the vectorized operator vec as

$$vec(W) = vec(V \Lambda V^T) = vec(\lambda_1 v_1 v_1^T + \cdots + \lambda_N v_N v_N^T)$$

$$= vec(V)^\top \cdot vec(V)$$

$$\Gamma := \left[vec(v_1 v_1^T), \ldots, vec(v_N v_N^T)\right] = \left[\lambda_1, \ldots, \lambda_N\right]^T.$$ (56)

Also, vectorized form of $W$ can be written based on its non-diagonal upper triangular form $w \in \mathbb{R}^{(\frac{N}{2}) \times 1}$ explained in (9) as

$$vec(W) = M_d vech(W) = M_d M_h vech_n(W) = M_d M_h w.$$ (57)

Therefore, using a combination of the two previous equations, the constraint $W = V \Lambda V^T$ can be turned into $w = \Gamma \lambda$, where $\Gamma := (M_d M_h)^\top \cdot \overline{V}$.

B. Derivatives of Iteration Updates of Lagrangian (12)

From the Lagrangian function (12), the update steps of the $(k + 1)$th iteration can be expressed as

$$\lambda^{(k+1)} = \frac{\rho(k)}{\rho(k)} \frac{w^{(k+1)} - \gamma_1^{(k)}}{\rho(k)}$$

$$\gamma_1^{(k+1)} = \Gamma^T \left( \frac{\rho(k)}{\rho(k)} w^{(k+1)} - \frac{\rho(k)}{\rho(k)} \gamma_1^{(k)} \right).$$ (58)

$$\|w^{(k+1)} - \gamma_1^{(k)}\|_2$$

$$= argmin_\omega \left\{ \|w\|_1 \right\}$$

$$+ \rho(k) \left( \frac{\|w\|_2^2 \left( \sigma_1^{(k)} \right)^2 \left( \sigma_2^{(k)} \right)^2 \right)$$

$$= argmin_\omega \left\{ \|w\|_1 \right\}$$

$$+ \frac{1}{2} \left( \frac{\rho(k)}{\rho(k)} \Gamma^{(k)} + \frac{\rho(k)}{\rho(k)} \sigma_1^{(k)} + \frac{\rho(k)}{\rho(k)} \sigma_2^{(k)} \right)$$

$$= \frac{1}{2} \left( \frac{\rho(k)}{\rho(k)} \Gamma^{(k)} + \frac{\rho(k)}{\rho(k)} \sigma_1^{(k)} + \frac{\rho(k)}{\rho(k)} \sigma_2^{(k)} \right).$$ (59)

$$s^{(k+1)} = argmin_{s \in W} \left\{ \|s\|_2^2 - 2(s, \frac{\rho(k)}{\rho(k)} w^{(k+1)} - \gamma_2^{(k)}) \right\}$$

$$= argmin_{s \in W} \left\{ \|s\|_2^2 - 2(s, \frac{\rho(k)}{\rho(k)} w^{(k+1)} - \gamma_2^{(k)}) \right\}$$

$$= \Pi_{W} \left( \frac{\rho(k)}{\rho(k)} w^{(k+1)} - \gamma_2^{(k)} \right).$$ (60)

C. Simplifications of Theorem 2

The commutativity constraints for $r = 1, \ldots, n$ can be stated using (23) as

$$C_{\Xi_{n+1}} W_r = W_r C_{\Xi_{n+1}}.$$ (61)

Then, the vectorized form of (61), for $r = 1, \ldots, n$, is

$$\left( C_{\Xi_{n+1}} \otimes I_{P_r} \right) = \left( C_{\Xi_{n+1}} \otimes I_{P_r} \right) = \left( M_d^{(r)} Gem_{(r)}^{\top} \right).$$ (62)

Therefore, the recovery of the adjacencies $\{W_i \in \mathbb{R}^{P_i \times P_i}\}_{i=1}^n$ in (43) can be written in more simplified manner w.r.t. the non-diagonal upper triangular entries $\{w_i = (w^{(i)}_1, w^{(i)}_2, \ldots, w^{(i)}_{\binom{n}{2}}) \in \mathbb{R}^{\binom{P_i}{2}}\}_{i=1}^n$ as

$$\min_{\{W_i\} \in \mathbb{R}^{P_i \times P_i}} \sum_{i=1}^n \alpha_i \|W_i\|_1$$

s.t. \left\{ \begin{array}{l}
\sum_{i=1}^n w^{(i)}_i = 1 \end{array} \right\}. (63)
Fig. 7. Comparison of the GL performance and executed run-times between the proposed SpecTemp-IALM, the generic SpecTemp-CVX [5], and also SpecTemp-ADMM [49], [50] in (a) varying number of graph signals ($T$) and (b) varying number of nodes ($N$).

which, using the definitions (39)–(43), can be clearly unified as in (46).

D. Comparison of SpecTemp [5] and the Proposed SpecTemp-IALM

To illustrate the graph learning performance and complexity improvement of the proposed SpecTemp-IALM solver over the baseline SpecTemp [5], referred to SpecTemp-CVX (which uses the of-the-shelf CVX solver [5], [48]) and also SpecTemp-ADMM [49], [50] (which exploits Alternating Direction Method of Multipliers (ADMM) for accelerate over the generic SpecTemp-CVX), we apply them for learning single (not product) ER graphs with the settings $N = 20$, $p_{ER} = 0.6$ and $T \in \{10^{3\pm 6}\}$. Fig. 7(a) provides the GL performance and executed run-times (in seconds) averaged over 40 independent noiseless realizations from diffusion process (3) with $L = 2$, $h_0 = 1$, and $h_1 = 0.5$, which clearly illustrates that the significant improvement over the baseline, especially for drastically reducing the computational run-time. Besides, the proposed method performs rather faster than the SpecTemp-ADMM, which is probably because, unlike the proposed SpecTemp-IALM, the number of optimization variables does not reduce in this method, leading to a slightly higher run-time. We should remind that the number of samples varies in a power of 10, and

the transition from $10^4$ to $10^5$ and $10^6$ means a tremendous computational increase, which the other settings seem flat against $10^5$ and $10^6$. Note that analyzing the executed run times with a varying number of samples can give insights into the consistency performance in possible online applications, i.e., processing the new batch of samples.

From another point of view, we fix the number of samples $T = 10^5$ and vary the number of nodes in the span of $\{15, 20, 25, 30\}$ to compare the robustness and scalability of the involved methods. Fig. 7(b) shows the GL performance and executed run-times averaged over 40 noiseless realizations. Similar findings from Fig. 7(a) about the superior robustness and computational efficiency of the proposed SpecTemp-IALM method over the compared ones can be admitted here, too, especially over the SpecTemp-CVX baseline.

E. Ignoring Product Structure Vs. Product GL

In this section, we compare the product GL and also the execution run times with ignoring the product structure. Note that due to prohibitive high computational complexity and time of applying SpecTemp [5] method for learning rather large graphs ($N \geq 40$) [51], we consider rather small factor graphs, i.e., $G_1$ and $G_2$ are 3-node path and $\{7, 10, 15, 20, 25, 30\}$-node ER graphs (with $p_{ER} = 0.3$), respectively. In this way, the averaged results over 40 independent realizations, generated as in Section IV-A, are depicted in Fig. 8. As can be seen in this figure, ignoring product structure (referred to as Direct method in this figure), which deals with many more optimization variables as discussed in Section III-E of the manuscript, has led to poor GL robustness and also a significant computational burden in all kinds of graph products.

F. Effect of the Graph Size ($N$)

We study the robustness of the involved PGL [11] and BiGLasso [13] and the proposed ProdSpecTemp methods against
varying \( P_2 \in \{7, 14, 21, 28, 35\} \) (and fixing \( P_1 = 7 \)) in terms of both the F1-measure and run-time execution for learning Strong product graphs. The averaged results over fifty independent realizations are illustrated in Fig. 9. As can be observed, the proposed ProdSpecTemp method has superior robustness against graph size in both F1-measure and run times.

REFERENCES

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

[2] 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.

[3] W. Huang et al., “A graph signal processing perspective on functional brain imaging,” Proc. IEEE, vol. 106, no. 5, pp. 868–885, May 2018.

[4] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, “Learning graphs from data: A spectral representation perspective,” IEEE Signal Process. Mag., vol. 36, no. 3, pp. 44–63, May 2019.

[5] S. S. Parsegov, A. V. Proskurnikov, R. Tempo, and N. E. Friedkin, “Novel learning Strong product graphs. The averaged results over fifty independent realizations are illustrated in Fig. 9. As can be observed, the proposed ProdSpecTemp method has superior robustness against graph size in both F1-measure and run times.

REFERENCES

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

[2] 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.

[3] W. Huang et al., “A graph signal processing perspective on functional brain imaging,” Proc. IEEE, vol. 106, no. 5, pp. 868–885, May 2018.

[4] X. Dong, D. Thanou, M. Rabbat, and P. Frossard, “Learning graphs from data: A spectral representation perspective,” IEEE Signal Process. Mag., vol. 36, no. 3, pp. 44–63, May 2019.

[5] S. S. Parsegov, A. V. Proskurnikov, R. Tempo, and N. E. Friedkin, “Novel learning Strong product graphs. The averaged results over fifty independent realizations are illustrated in Fig. 9. As can be observed, the proposed ProdSpecTemp method has superior robustness against graph size in both F1-measure and run times.

REFERENCES

[1] A. Ortega, P. Frossard, J. Kovačević, J. M. F. Moura, and P. Vandergheynst, “Graph signal processing: Overview, challenges, and applications,” Proc. IEEE, vol. 106, no. 5, pp. 808–828, May 2018.
[42] T. Nguyen, O. Babawale, T. Kim, H. J. Jo, H. Liu, and J. G. Kim, “Exploring brain functional connectivity in rest and sleep states: A fairs study,” Sci. Rep., vol. 8, no. 1, pp. 1–10, 2018.

[43] C. Kaufmann et al., “Brain activation and hypothalamic functional connectivity during human non-rapid eye movement sleep: An eeg/fmri study,” Brain, vol. 129, no. 3, pp. 655–667, 2006.

[44] A. Tarun et al., “NREM sleep stages specifically alter dynamical integration of large-scale brain networks,” Iscience, vol. 24, no. 1, 2021, Art. no. 101923.

[45] T. W. Kjaer et al., “Regional cerebral blood flow during light sleep–a H215O-PET study,” J. Sleep Res., vol. 11, no. 3, pp. 201–207, 2002.

[46] Z. Jia et al., “Multi-view spatial-temporal graph convolutional networks with domain generalization for sleep stage classification,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 29, pp. 1977–1986, 2021.

[47] L. J. Larson-Prior et al., “Modulation of the brain’s functional network architecture in the transition from wake to sleep,” Prog. Brain Res., vol. 193, pp. 277–294, 2011.

[48] M. Grant and S. Boyd, “CVX: MATLAB software for disciplined convex programming, version 2.1,” Mar. 2014. [Online]. Available: http://cvxr.com/cvx

[49] R. Shafipour, A. Hashemi, G. Mateos, and H. Vikalo, “Online topology inference from streaming stationary graph signals,” in Proc. IEEE Data Sci. Workshop, 2019, pp. 140–144.

[50] R. Shafipour and G. Mateos, “Online topology inference from streaming stationary graph signals with partial connectivity information,” Algorithms, vol. 13, no. 9, 2020, Art. no. 228.

[51] 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.