Signal Processing 186 (2021) 108101

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Online discriminative graph learning from multi-class smooth signals☆,☆☆

Seyed Saman Saboksayr[a,*], Gonzalo Mateos[a], Mujdat Cetin[a,b]

[a] Dept. of Electrical and Computer Eng., Univ. of Rochester, Rochester, USA
[b] Goergen Institute for Data Science, Univ. of Rochester, Rochester, USA

ARTICLE INFO

Article history:
Received 31 December 2020
Revised 24 March 2021
Accepted 26 March 2021
Available online 29 March 2021

Keywords:
Graph learning
Network topology inference
Graph signal processing
Online optimization
Discriminative learning
Smooth signals

ABSTRACT

Graph signal processing (GSP) is a key tool for satisfying the growing demand for information processing over networks. However, the success of GSP in downstream learning and inference tasks such as supervised classification is heavily dependent on the prior identification of the relational structures. Graphs are natural descriptors of the relationships between entities of complex environments. The underlying graph is not readily detectable in many cases and one has to infer the topology from the observed signals which admit certain regularity over the sought representation. Assuming that signals are smooth over the latent class-specific graph is the notion we build upon. Firstly, we address the problem of graph signal classification by proposing a novel framework for discriminative graph learning. To learn discriminative graphs, we invoke the assumption that signals belonging to each class are smooth with respect to the corresponding graph while maintaining non-smoothness with respect to the graphs corresponding to other classes. Discriminative features are extracted via graph Fourier transform (GFT) of the learned representations and used in downstream learning tasks. Secondly, we extend our work to tackle increasingly dynamic environments and real-time topology inference. To this end, we adopt recent advances of GSP and time-varying convex optimization. We develop a proximal gradient (PG) method which can be adapted to situations where the data are acquired on-the-fly. Beyond discrimination, this is the first work that addresses the problem of dynamic graph learning from smooth signals where the sought network alters slowly. The introduced online framework is guaranteed to track the optimal time-varying batch solution under mild technical conditions. The validation of the proposed frameworks is comprehensively investigated using both synthetic and real data. The proposed classification pipeline outperforms the-state-of-the-art methods when applied to the problem of emotion classification based on electroencephalogram (EEG) data. We also perform network-based analysis of epileptic seizures using electrocorticography (ECoG) records. Moreover, by applying our method to financial data, our approach infers the relationships between the stock-price behavior of leading US companies and the recent events including the COVID-19 pandemic.

1. Introduction

We are witnessing an uptick of recent interest in signal and information processing tasks involving network data, which go all the way from clustering similar actors in a social network to classifying brain connectomes in computational neuroscience studies [3]. Complex relational structures arising with network datasets are naturally mapped out using graph representations. These graphs are essential for understanding how different system elements interact with each other, especially in dynamic environments. Moreover, they often provide useful context (e.g., in the form of a prior or regularizer) to effectively extract actionable information from graph signals, i.e., nodal attributes such as sensor measurements, infected case counts across different localities, or discipline-related labels assigned to scientific papers [4]. Some graphs may be readily obtained based on prior surveys and experiments (e.g., road networks and molecule structures), or they could be (at least partially) directly observable as with social and citation networks. In other cases, however, there is a need to utilize *network topology inference* algorithms to estimate the graph structure from graph signal observations; see e.g., [3,5,6]. This paper tackles a classification problem involving network data, where class-

☆ This work was supported in part by the National Science Foundation under awards CCF-1750428, CCF-1934962 and ECCS-1809356.
☆☆ Part of the results in this paper were submitted to the *2021 ICASSP* and *EUSIPCO* conferences [1,2].
* Corresponding author.
E-mail addresses: ssaboksa@ur.rochester.edu (S.S. Saboksayr), gmateosb@ece.rochester.edu (G. Mateos), mujdat.cetin@rochester.edu (M. Cetin).

https://doi.org/10.1016/j.sigpro.2021.108101
0165-1684/

specific graphs are learned from labeled signals to obtain discriminative representations.

1.1. Related work

Popular graph construction schemes rely on ad hoc thresholding of user-defined edgewise similarity measures or kernels. Albeit intuitive, these informal approaches may fail to capture the actual intrinsic relationships between entities and do not enjoy any sense of optimality. Recognizing this shortcoming, a different paradigm is to cast the graph-learning problem as one of selecting the best representative from a family of candidate graphs by bringing to bear elements of statistical modeling and inference [3]. This path usually leads to inverse problems that minimize criteria stemming from different models binding the (statistical) signal properties to the graph topology. Noteworthy approaches from the Graph Signal Processing (GSP) literature usually assume the signals are stationary with respect to the sought sparse network [7,8], or that they are smooth in the sense that a desirable graph is one over which the nodal attributes exhibit limited variations; see e.g., [9,10] and Section 2.3. Graph learning via signal smoothness maximization is closely related to Gaussian graphical model selection when the precision matrix is constrained to have a combinatorial Laplacian structure; see also the recent review papers [5,11].

The graph Fourier transform (GFT) is defined in terms of the network's eigenvectors [4], so through this lens, topology inference is tightly related to the problem of learning efficient graph signal representations for denoising and compression, just to name several relevant downstream tasks. Acknowledging this perspective, [12] puts forth a graph learning approach facilitating accurate reconstruction of sampled bandlimited (i.e., smooth) graph signals. Laplacian-regularized dictionary learning was considered in [13], where the graph can be learned to effect sparse signal representations by better capturing the irregular data geometry. Naturally, some works have dealt with supervised classification of network data. Given a known graph over which the signals in all classes reside, the approach of [14] is to learn class-specific parametric graph sub-dictionaries. The discriminative graphical lasso algorithm in [15] constructs class-specific graphs from labeled signals, by minimizing the conventional Gaussian maximum likelihood objective augmented with a term that boosts the discrimination ability of the learnt representations. Related work in [16] advocates a graph Laplacian mixture model for data that reside on multiple networks, and tackles the problem of jointly clustering a set of signals while learning a graph for each of the clusters. Here we advance this line of work in several directions as discussed in Section 1.2.

1.2. Proposed approach and contributions

We tackle a multi-class network topology inference problem for graph signal classification; see Section 3.1 for a formal statement of the problem. The signals in each class are assumed to be smooth (or bandlimited) with respect to unknown class-specific graphs. Given training signals the goal is to recover the class-specific graphs subject to signal smoothness constraints (Section 3.2), so that the obtained GFT bases can be subsequently used to classify unseen (and unlabeled) graph signals effectively as discussed in Section 3.3. This problem bears some similarities with subspace clustering [17]. Indeed, smoothness with respect to class-specific graphs can be interpreted as data living in multiple low-dimensional subspaces spanned by the respective low-frequency GFT components. However, here the problem is supervised and we contend there is value in learning the graph topologies beyond the induced discriminative transforms (i.e., subspaces) to reveal important structure in the data for each class. To this end, we develop a new framework for *discriminative graph learning* in which we encourage smoothness of each class signals on their corresponding inferred network representation as well as non-smoothness over all other classes' topologies. Our convex criterion for graph learning builds on [10], which we augment with a judicious penalty term to effect class discriminability [15]. Remark 1 summarizes the distinctive features of our approach relative to [10,15]. From an algorithmic standpoint, we develop lightweight proximal gradient iterations (Section 3.2) with well documented rates of convergence, acceleration potential, and that can be readily adapted to *process streaming signals in an online fashion* – our second main contribution.

As network data sources are increasingly generating real-time streams, training of models (here discriminative graph learning) must often be performed online, typically with limited memory footprint and without waiting for a batch of signals to become available. To accommodate this pragmatic scenario, in Section 4.1 we develop online proximal gradient iterations to track the (possibly time-varying) network structure per class. Since the dynamic topology inference cost is strongly convex, we establish that the online graph estimates closely hover within a ball centered at the optimal time-varying batch solution (Section 4.2), and characterize the convergence radius even in a dynamic setting [18–20]. This algorithmic contribution is of independent interest beyond the classification theme of this paper. Dropping from the objective the regularization term that encourages discriminability, yields for the very first time an online graph learning algorithm under smoothness priors. Existing algorithms to identify dynamic network topologies from smooth signals operate in a batch fashion, and so their computational cost and memory requirements grow with time (i.e., the dataset size) [21–23].

The effectiveness and convergence of the proposed algorithms is corroborated through a comprehensive performance evaluation carried out in Section 5. We include numerical test cases with synthetic and real data in both batch and online settings, spanning several tasks across timely application domains such as a emotion classification from electroencephalogram (EEG) data, online graph learning from financial time-series, and network-based analysis of epileptic seizures from electrocorticography (ECoG) records. Concluding remarks are given in Section 6, while some technical details are deferred to the appendices. This journal paper offers a more thorough treatment of online discriminative graph learning relative to its short conference precursors [1,2]. This is achieved by means of expanded technical details, discussions and insights, as well as through more comprehensive performance evaluation studies with synthetic and real data experiments. In particular, focus in [1] is on emotion classification from EEG signals (the subject of Section 5.3), while [2] deals with online topology identification without a classification task in mind; see also Remark 2.

Notational conventions. The entries of a matrix $\mathbf{X}$ and a (column) vector $\mathbf{x}$ are denoted by X_{ij} and x_i, respectively. Sets are represented by calligraphic capital letters. The notation $^\top$ and † stand for transpose and pseudo-inverse, respectively; $\mathbf{0}$ and $\mathbf{1}$ refer to the all-zero and all-one vectors; while $\mathbf{I}_N$ denotes the $N \times N$ identity matrix. For a vector $\mathbf{x}$, $\mathrm{diag}(\mathbf{x})$ is a diagonal matrix whose ith diagonal entry is x_i. The operators $\circ$, $\mathrm{tr}(\cdot)$, and $\mathrm{vec}(\cdot)$ stand for Hadamard (element-wise) product, matrix trace, and vectorization, respectively. Lastly, $\|\mathbf{X}\|_p$ denotes the ℓ_p norm of $\mathrm{vec}(\mathbf{X})$ and $\|\mathbf{X}\|_F$ refers to the Frobenius norm. To avoid overloading the notation, on occasion $\|\mathbf{x}\|$ is used to denote the ℓ_2 norm of vector $\mathbf{x}$.

2. Graph signal processing background

We start by briefly reviewing the necessary GSP concepts and tools that be will used recurrently in the ensuing sections.

2.1. Graphs, signals and shift operators

We define the graph signal $\mathbf{x} = [x_1, \ldots, x_N]^\top \in \mathbb{R}^N$ over a weighted, undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$, where $\mathcal{V} = \{1, \ldots, N\}$ represents the node set of cardinality N and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ the set of edges. Following this notation, $x_i \in \mathbb{R}$ and $\mathbf{W} \in \mathbb{R}_+^{N \times N}$ denote the signal value at node $i \in \mathcal{V}$ and the adjacency matrix of edge weights, respectively. The symmetric and non-negative coefficients $W_{ij} = W_{ji} \in \mathbb{R}_+$ indicate the strength of the connection (or similarity) between node i and node j. In the absence of connection [i.e., $(i, j) \notin \mathcal{E}$] one has $W_{ij} = 0$. Moreover, we assume that $\mathcal{G}$ does not include any self-loops which implies $W_{ii} = 0$, $\forall i \in \mathcal{V}$. We will henceforth assume nodal degrees $\mathbf{d} := \mathbf{W}\mathbf{1}$ are uniformly lower bounded away from zero, i.e., $\mathbf{d} \geq d_{\min}\mathbf{1}$ (entry-wise inequality) for some prescribed $d_{\min} > 0$. Else if degrees become arbitrarily small, it is prudent to apply a threshold and remove the loosely connected nodes from $\mathcal{G}$. Extensions to directed graphs could be important [24], but are beyond the scope of this paper. Complex-valued signals can be accommodated as well [4].

The adjacency matrix $\mathbf{W}$ encodes the topology of $\mathcal{G}$. More generally it is possible to define the so-called *graph-shift operator* $\mathbf{S}$, which captures the sparsity pattern of $\mathcal{G}$ without any specific assumption on the value of its non-zero entries. The shift $\mathbf{S} \in \mathbb{R}_+^{N \times N}$ is a symmetric matrix whose entry S_{ij} can be non-zero only if $i = j$ or if $(i, j) \in \mathcal{E}$. Beyond the adjacency matrix $\mathbf{W}$, results in spectral graph theory often motivate choosing the combinatorial graph Laplacian $\mathbf{L} := \text{diag}(\mathbf{d}) - \mathbf{W}$ as well as their various degree-normalized counterparts. Other application-dependent alternatives have been proposed as well; see [4] and the references therein. In particular, $\mathbf{L}$ plays a central role in defining a useful and intuitive graph Fourier transform (GFT) as described next.

2.2. Graph Fourier transform and signal smoothness

In order to introduce the network's spectral basis and define the GFT, we decompose the (symmetric and positive semi-definite) combinatorial graph Laplacian as $\mathbf{L} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top$, where $\mathbf{\Lambda} := \text{diag}(\lambda_1, \ldots, \lambda_N)$ denotes the diagonal matrix of non-negative eigenvalues and $\mathbf{V} := [\mathbf{v}_1, \ldots, \mathbf{v}_N]$ the orthonormal matrix of eigenvectors. The GFT of $\mathbf{x}$ with respect to $\mathbf{L}$ is the signal

$$\tilde{\mathbf{x}} := \mathbf{V}^\top \mathbf{x}. \tag{1}$$

The inverse (i)GFT of $\tilde{\mathbf{x}} := [\tilde{x}_1, \ldots, \tilde{x}_N]^\top$ is given by $\mathbf{x} = \mathbf{V}\tilde{\mathbf{x}} = \sum_{k=1}^N \tilde{x}_k \mathbf{v}_k$, which is a proper inverse due to the orthonormality of $\mathbf{V}$. The GFT encodes a notion of signal variability over $\mathcal{G}$ (akin to frequency in Fourier analysis of temporal signals) by synthesizing $\mathbf{x}$ as a sum of orthogonal frequency components $\mathbf{v}_k$ (see the iGFT definition above). The GFT coefficient $\tilde{x}_k$ is the contribution of $\mathbf{v}_k$ to the graph signal $\mathbf{x}$.

To elaborate on the notion of frequency for graph signals, consider the total variation (or *Dirichlet energy*) of $\mathbf{x}$ with respect to the combinatorial graph Laplacian $\mathbf{L}$ defined as

$$\text{TV}(\mathbf{x}) := \mathbf{x}^\top \mathbf{L}\mathbf{x} = \frac{1}{2}\sum_{i \neq j} W_{ij}\big(x_i - x_j\big)^2 = \sum_{k=1}^N \lambda_k \tilde{x}_k^2. \tag{2}$$

The quadratic form (2) acts as a smoothness measure, because it effectively quantifies how much the graph signal $\mathbf{x}$ changes with respect to $\mathcal{G}$'s topology. If we evaluate the total variation of eigenvector $\mathbf{v}_k$ (itself a graph signal), one immediately obtains $\text{TV}(\mathbf{v}_k) = \lambda_k$. Accordingly, the Laplacian eigenvalues $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_N$ can be viewed as graph frequencies indicating how the eigenvectors (i.e., frequency components) vary with respect to $\mathcal{G}$ [4].

Smoothness is a cardinal property of many real-world network processes; see e.g. [3]. The last equality in (2) suggests that smooth (or bandlimited) signals admit a sparse representation in the graph spectral domain. Intuitively, they tend to be spanned by few Laplacian eiegenvectors associated with small eigenvalues. Exploiting this structure by means of priors or TV-based regularizers is at the heart of several graph-based statistical learning tasks including nearest-neighbor prediction (also known as graph smoothing), denoising, semi-supervised learning, and spectral clustering [3,4]. More germane to our graph-learning problem is to use smoothness as the criterion to construct graphs on which network data admit certain regularity.

2.3. Learning graphs from observations of smooth signals

Consider the following network topology identification problem. Given a set $\mathcal{X} := \{\mathbf{x}_p\}_{p=1}^P$ of possibly noisy graph signal observations, the goal is to learn an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E}, \mathbf{W})$ with $|\mathcal{V}| = N$ nodes such that the observations in $\mathcal{X}$ are smooth on $\mathcal{G}$. In this section we review the solution proposed in [10], that we build on in the rest of the paper.

Given $\mathcal{X}$ one can form the data matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_P] \in \mathbb{R}^{N \times P}$, and let $\bar{\mathbf{x}}_i^\top \in \mathbb{R}^{1 \times P}$ denote its ith row collecting those P measurements at vertex i. The key idea in [10] is to establish a link between smoothness and sparsity, namely

$$\sum_{p=1}^P \text{TV}(\mathbf{x}_p) = \text{tr}(\mathbf{X}^\top \mathbf{L}\mathbf{X}) = \frac{1}{2}\|\mathbf{W} \circ \mathbf{Z}\|_1, \tag{3}$$

where the Euclidean-distance matrix $\mathbf{Z} \in \mathbb{R}_+^{N \times N}$ has entries $Z_{ij} := \|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j\|^2$, $i, j \in \mathcal{V}$. The intuition is that when the given distances in $\mathbf{Z}$ come from a smooth manifold, the corresponding graph has a sparse edge set, with preference given to edges (i, j) associated with smaller distances Z_{ij}.

Given these considerations, a general purpose framework for learning graphs under a smoothness prior is advocated in [10], which entails solving

$$\min_{\mathbf{W}} \|\mathbf{W} \circ \mathbf{Z}\|_1 + f(\mathbf{W}) \tag{4}$$
$$\text{s. t.} \quad \text{diag}(\mathbf{W}) = \mathbf{0},\ W_{ij} = W_{ji} \geq 0,\ i \neq j.$$

The convex objective function $f(\mathbf{W})$ augments the smoothness criterion $\|\mathbf{W} \circ \mathbf{Z}\|_1$, and several choices have been proposed to e.g., recover common graph constructions based on the Gaussian kernel [25], accommodate time-varying graphs [22], or to scale other related graph learning algorithms [9]. Identity (3) offers a favorable way of formulating the inverse problem (4), because the space of adjacency matrices can be described via simpler (meaning entry-wise decoupled) constraints relative to its Laplacian counterpart. As a result, the convex optimization problem (4) can be solved efficiently with complexity $\mathcal{O}(N^2)$ per iteration, by leveraging provably-convergent primal-dual solvers amenable to parallelization [26]; see also Section 3.2 for a different optimization approach based on proximal gradient algorithms.

The framework in [10] has been shown to attain state-of-the-art performance and is not only attractive due to is computational efficiency but also due to its generality. Henceforth, we leverage and expand these ideas to tackle two important graph signal and information processing problems.

3. Discriminative graph learning

In an effort to address classification problems involving network data (Section 3.1), we bring to bear GSP insights to propose a new framework for learning discriminative graph-based representation of the signals. To this end, a proximal gradient (PG) algorithm is developed to solve the separable graph-learning problems per class (Section 3.2). After this training phase, the GFTs of the optimum graphs can be used to extract discriminative features of the test

signals. In the test phase, we exploit the extracted GFT-based features in order to classify the test signals (Section 3.3).

3.1. Optimization problem formulation

Consider a dataset $\mathcal{X}=\bigcup_{c=1}^{C}\mathcal{X}_c$ comprising labeled graph signals $\mathcal{X}_c:=\{\mathbf{x}_p^{(c)}\}_{p=1}^{P_c}$ from C different classes. The signals in each class possess a very distinctive structure, namely they are assumed to be smooth (or bandlimited) with respect to unknown class-specific graphs $\mathcal{G}_c=(\mathcal{V},\mathcal{E}_c,\mathbf{W}_c)$, $c=1,\ldots,C$. This notion is analogous to a multiple linear sub-space model in which graph signals of each class are assumed to be spanned by a few vectors (namely, the basis of the corresponding low-dimensional subspace) [17]. In fact, the closing discussion in Section 2.2 implies one can equivalently restate the class-conditional signal model assumptions as follows: there is a network $\mathcal{G}_c$ such that graph signals in $\mathcal{X}_c$ are spanned by a few Laplacian eigenvectors (associated with small eigenvalues), for $c=1,\ldots,C$. Similar to [15], given $\mathcal{X}$ the goal is to learn the class-specific adjacency matrices $\mathbf{W}_c$ under signal smoothness priors, so that the obtained GFT bases can be subsequently used to classify unseen (and unlabeled) graph signals effectively; see Section 3.3.

Our approach blends elements and ideas from the graph learning framework in [10] [cf. (4)] along with the discriminative graphical lasso estimator [15]. Indeed, the algorithm in [10] optimizes network topology recovery under smoothness assumptions, but is otherwise agnostic to the performance of a potential downstream (say, classification) task the learnt graph may be integral to. Inspired by [15], we seek graph representations that capture the underlying network topology (i.e., the class structure), but at the same time are discriminative to boost classification performance. To this end, we propose to learn a graph representation $\mathcal{G}_c$ per class by solving the following convex optimization problems [cf. (4)]

$$\min_{\mathbf{W}_c\in\mathcal{W}_m}\ \|\mathbf{W}_c\circ\mathbf{Z}_c\|_1-\alpha\mathbf{1}^\top\log(\mathbf{W}_c\mathbf{1})+\beta\|\mathbf{W}_c\|_F^2-\gamma\sum_{k\neq c}^{C}\|\mathbf{W}_c\circ\mathbf{Z}_k\|_1, \tag{5}$$

where $\mathbf{W}_c$ is the adjacency matrix of $\mathcal{G}_c$, constrained to the set $\mathcal{W}_m=\{\mathbf{W}\in\mathbb{R}_+^{N\times N}:\mathbf{W}=\mathbf{W}^\top,\operatorname{diag}(\mathbf{W})=\mathbf{0}\}$. Moreover, $\mathbf{Z}_c$ is the distance matrix constructed from class c signals $\mathbf{X}_c:=[\mathbf{x}_1^{(c)},\ldots,\mathbf{x}_{P_c}^{(c)}]\in\mathbb{R}^{N\times P_c}$, while α, β, and γ are positive regularization parameters.

Taking a closer look at the objective function, minimizing $\|\mathbf{W}_c\circ\mathbf{Z}_c\|_1$ encourages a graph $\mathbf{W}_c$ over which the signals in $\mathcal{X}_c$ are smooth. At the same time, the last term enforces non-smoothness of the signals in the other $C-1$ classes. This composite criterion will thus induce a GFT with better discrimination ability than the baseline $\gamma=0$ case. In fact, the energy of class c signals will be predominantly concentrated in lower frequencies, while the spectral content of the other classes is pushed towards high-pass regions of the spectrum; see Section 5.1 for corroborating simulations. Under Gaussian assumptions this can be interpreted as a Fisher discrimination criterion used in Linear Discriminant Analysis (LDA) [25], which entails the other way around; see [15].

The logarithmic barrier on the nodal degree sequence $\mathbf{W}_c\mathbf{1}$ precludes the trivial all-zero solution. Moreover, it ensures the estimated graph is devoid of isolated vertices. The Frobenius-norm regularization on the adjacency matrix $\mathbf{W}_c$ controls the graphs' edge sparsity pattern by penalizing larger edge weights (the sparsest graph is obtained for $\beta=0$). Overall, this combination forces degrees to be positive but does not prevent most individual edge weights from becoming zero [10].

Remark 1 (Comparison with [10,15])**.** The topology inference problem (5) is subsumed by the general framework in [10], for a particular choice of the regularization function $f(\mathbf{W}_i)$ in (4). The distinctive goal here is that of discriminative transform learning, to effectively tackle a classification problem involving network data. Different from [10], in the next section we solve (5) using efficient PG iterations that can afford Nesterov-type acceleration and are readily adapted to process streaming data in an online fashion during training (Section 4). A discriminative graph learning approach, sharing similar objectives to ours was formulated in [15], but under the lens of probabilistic graphical model selection. Therein, graph signals are viewed as random vectors adhering to a Gaussian Markov random field distribution, where the unknown class-specific precision matrices typically play the role of graph Laplacians. However, the discriminative graphical lasso estimator in [15] is not guaranteed to return a valid graph Laplacian for each class, since the search is performed over the whole positive semi-definite cone. Incorporating Laplacian constraints may challenge the block coordinate-descent algorithm in [15]. Accordingly, one misses on the GSP insights offered here in terms of signal smoothness and bandlimitedness in the graph spectral domain. Numerical tests in Section 5 show that the proposed approach outperforms the algorithms in [10] and [15] when it comes to classification, while it recovers interpretable graphs offering insights into the structure of the data classes.

3.2. Proximal-gradient algorithm

Given the optimization problem formulation in (5), in this section we develop a PG algorithm to recover the network topology of the graphs $\mathcal{G}_c$ per class $c=1,\ldots,C$; see [27] for a tutorial on proximal methods and their applications. PG methods have been popularized for ℓ_1-norm regularized linear regression problems, and their desirable features of computational simplicity as well as suitability for online operation are starting to permeate naturally to the topology identification context of this paper [28].

To make (5) amenable to this optimization method, recall first that the adjacency matrix $\mathbf{W}_c\in\mathcal{W}$ is symmetric with diagonal elements equal to zero. Therefore, the independent decision variables are effectively the upper-triangular elements $[\mathbf{W}_c]_{ij}$, $j>i$, which we collect in the vector $\mathbf{w}_c\in\mathbb{R}_+^{N(N-1)/2}$. Second, it will prove convenient to enforce the non-negativity constraints by means of a penalty function augmenting the original objective. Just like [10], we add an indicator function $\mathbb{I}\{\mathbf{w}_c\geq\mathbf{0}\}=0$ if $\mathbf{w}_c\geq\mathbf{0}$, and $\mathbb{I}\{\mathbf{w}_c\geq\mathbf{0}\}=\infty$ otherwise. Given all these considerations we reformulate the objective in (5) as the function $F(\mathbf{w}_c)$ of a vector variable, and write the equivalent composite, non-smooth optimization problem:

$$\min_{\mathbf{w}_c}F(\mathbf{w}_c):=\overbrace{-\alpha\mathbf{1}^\top\log(\mathbf{S}\mathbf{w}_c)+\beta 2\|\mathbf{w}_c\|^2}^{g(\mathbf{w}_c)}+\underbrace{\mathbb{I}\{\mathbf{w}_c\geq\mathbf{0}\}+2\mathbf{w}_c^\top\mathbf{z}_c-\gamma\sum_{k\neq c}^{C}2\mathbf{w}_c^\top\mathbf{z}_k}_{h(\mathbf{w}_c)}, \tag{6}$$

where $\mathbf{z}_c$ is a vector containing the upper-triangular entries of $\mathbf{Z}_c$, and $\mathbf{S}\in\{0,1\}^{N\times N(N-1)/2}$ is such that $\mathbf{d}_c=\mathbf{W}_c\mathbf{1}=\mathbf{S}\mathbf{w}_c$.

To arrive at the PG iterations, first note that the gradient of g in (6) has the simple form

$$\nabla g(\mathbf{w}_c)=4\beta\mathbf{w}_c-\alpha\mathbf{S}^\top\left(\frac{\mathbf{1}}{\mathbf{S}\mathbf{w}_c}\right), \tag{7}$$

where $\mathbf{1}/(\mathbf{S}\mathbf{w}_c)$ stands for element-wise division. Moreover, the gradient ∇g is a Lipschitz-continuous function with constant $\eta=\left(4\beta+\frac{2\alpha(N-1)}{d_{\min}^2}\right)$; see Appendix A for a proof. With $\lambda>0$ a fixed positive scalar, introduce the proximal operator of a closed proper

Algorithm 1: PG for graph learning, class c.

Input parameters $\alpha, \beta, \gamma, \eta$, data $\{\mathbf{z}_c\}_{c=1}^C$, initial $\mathbf{w}_{c,0}$.
Set $k=0$, $\mu \leq 2/\eta$ and $\bar{\boldsymbol{\mu}} = 2\mu\big(\mathbf{z}_c - \gamma \sum_{k\neq c}^C \mathbf{z}_k\big)$.
while *not converged* **do**
  Compute $\nabla g(\mathbf{w}_{c,k}) = 4\beta \mathbf{w}_{c,k} - \alpha \mathbf{S}^\top \Big(\frac{\mathbf{1}}{\mathbf{S}\mathbf{w}_{c,k}}\Big)$.
  Update $\mathbf{w}_{c,k+1} = \big[\mathbf{w}_{c,k} - \mu \nabla g(\mathbf{w}_{c,k}) - \bar{\boldsymbol{\mu}}\big]_+$.
  Increment $k \leftarrow k+1$.
end

convex function $f: \mathbb{R}^N \to \mathbb{R} \cup \{+\infty\}$ evaluated at point $\boldsymbol{\upsilon} \in \mathbb{R}^N$ as

$$\mathbf{prox}_{\lambda f}(\boldsymbol{\upsilon}) := \operatorname*{argmin}_{\mathbf{x}} \left\{ f(\mathbf{x}) + \frac{1}{2\lambda}\|\mathbf{x}-\boldsymbol{\upsilon}\|^2 \right\}. \tag{8}$$

With this definition, the PG updates with fixed step size $\mu < \frac{2}{\eta}$ to solve each of the class-specific graph learning problems (6) are given by (henceforth $k = 0, 1, 2, \ldots$ denote iterations)

$$\mathbf{w}_{c,k+1} := \mathbf{prox}_{\mu h}\big(\mathbf{w}_{c,k} - \mu \nabla g(\mathbf{w}_{c,k})\big). \tag{9}$$

It follows that graph estimate refinements are generated via the composition of a gradient-descent step and a proximal operator. Efficient evaluating of the latter is key to the success of PG methods. The proximal operator of h in (6) is given by

$$\mathbf{prox}_{\mu h}(\mathbf{w}_c) = \left[\mathbf{w}_c - 2\mu\left(\mathbf{z}_c - \gamma \sum_{k\neq c}^C \mathbf{z}_k\right)\right]_+, \tag{10}$$

where $[\boldsymbol{\upsilon}]_+ := \max(\mathbf{0}, \boldsymbol{\upsilon})$ denotes the projection operator onto the non-negative orthant (the max operator is applied entry-wise). The non-negative soft-thresholding operator in (10) sets to zero all edge weights in $\mathbf{w}_c$ that fall below the data-dependent thresholds in vector $\bar{\boldsymbol{\mu}} := 2\mu\big(\mathbf{z}_c - \gamma \sum_{k\neq c}^C \mathbf{z}_k\big)$. The resulting iterations are tabulated under Algorithm 1, which will also serve as the basis for the online algorithm in Section 4.

The computational complexity is dominated by the gradient evaluation in (7), incurring a cost of $\mathcal{O}(N^2)$ per iteration k due to scaling and additions of vectors of length $N(N-1)/2$. For sparse graphs $\mathcal{G}_c$ the iterates $\mathbf{w}_{c,k}$ tend to become (and remain) quite sparse at early stages of the algorithm by virtue of the soft-thresholding operations (a sparse initialization $\mathbf{w}_{c,0}$ is useful to this end). Hence, it is possible to reduce the complexity further if Algorithm 1 is implemented carefully using sparse vector operations. All in all, Algorithm 1 scales well to large graphs with thousands of nodes and it is competitive with the state of the art primal-dual solver in [10]. In terms of convergence, as $k \to \infty$ the sequence of iterates (9) provably approaches a minimizer of the composite cost F in (6); see e.g., [27] for the technical details. Moreover, the worst-case convergence rate of PG algorithms is well documented (namely $\mathcal{O}(1/\varepsilon)$ iteration complexity to return a ε-optimal solution measured in terms of F values), and can be boosted to $\mathcal{O}(1/\sqrt{\varepsilon})$ via Nesterov-type acceleration techniques that are readily applicable to Algorithm 1.

3.3. Classification via low-pass graph filtering

During the training phase of the classification task, the goal is to learn C class-specific graphs $\mathcal{G}_c$ from labeled graph signals $\mathcal{X}_c := \{\mathbf{x}_p^{(c)}\}_{p=1}^{P_c}$. This can be accomplished by running C parallel instances of Algorithm 1. Let $\hat{\mathbf{W}}_c$ denote the estimated adjacency matrix of the graph representing class c, and likewise let $\hat{\mathbf{L}}_c = \mathrm{diag}(\hat{\mathbf{W}}_c\mathbf{1}) - \hat{\mathbf{W}}_c$ be the combinatorial graph Laplacian. Finally, let $\hat{\mathbf{V}}_c$ denote the orthonormal GFT basis of Laplacian eigenvectors for class c; see Section 2.2.

In the operational or test phase, we are presented with an unseen and unlabeled graph signal $\mathbf{x}$ which we wish to classify into one of the C classes. To that end, we will process $\mathbf{x}$ with a filter-bank comprising C graph filters. The cth branch yields the graph-frequency domain output

$$\tilde{\mathbf{x}}_{F,c} = \tilde{\mathbf{H}}\mathbf{V}_c^\top \mathbf{x} = \mathrm{diag}(\tilde{\mathbf{h}})\tilde{\mathbf{x}}_c = \tilde{\mathbf{h}} \circ \tilde{\mathbf{x}}_c, \tag{11}$$

where $\tilde{\mathbf{x}}_c$ are the GFT coefficients of $\mathbf{x}$ with respect to graph $\mathcal{G}_c$, and $\tilde{\mathbf{h}} = [\tilde{h}_1, \ldots, \tilde{h}_N]^\top$ is the frequency response of an ideal low-pass filter with bandwidth $w \in \{1, 2, \ldots, N\}$, i.e., $\tilde{h}_i := \mathbb{1}\{i \leq w\}$. The indicator function $\mathbb{1}\{i \leq w\} = 1$ if $i \leq w$, and $\mathbb{1}\{i \leq w\} = 0$ otherwise. Typically, one chooses the tunable parameter w to be $N/2$ or smaller in order to implement a low-pass filter. Notice that while the frequency response $\tilde{\mathbf{h}}$ is the same for all C branches, the graph filters $\mathbf{H}_c = \mathbf{V}_c \mathrm{diag}(\tilde{\mathbf{h}})\mathbf{V}_c^\top$ differ because the learnt graphs (hence the GFT transforms) vary across classes. From the definition of $\tilde{\mathbf{h}}$, it immediately follows that $\tilde{\mathbf{x}}_{F,c}$ is nothing else than the projection of $\mathbf{x}$ onto the eigenvectors of $\hat{\mathbf{L}}_c$ corresponding to the smallest w eigenvalues.

If $\mathbf{x}$ belongs to class $c^\star$, say, then this graph signal should be smoothest with respect to $\mathcal{G}_{c^\star}$. Equivalently, for fixed (appropriately low) bandwidth w we expect the signal power to be largest when projected onto the GFT basis constructed from $\hat{\mathbf{L}}_c$. Accordingly, the adopted classification rule is simply

$$\hat{c} = \operatorname*{argmax}_c \left\{ \|\tilde{\mathbf{x}}_{F,c}\|^2 \right\}. \tag{12}$$

A classification error occurs whenever $\hat{c} \neq c^\star$.

4. Online discriminative graph learning

As network data sources are increasingly generating real-time streams, training of models (here discriminative graph learning) must often be performed on-the-fly, typically (i) not affording to store or revisit past measurements; and (ii) without waiting for a whole batch of signals to become available. To accommodate this envisioned scenario, we switch gears to online estimation of $\mathbf{W}_c$ (or even tracking $\mathbf{W}_{c,t}$ in a non-stationary setting when the class-conditional distributions exhibit variations) from streaming data $\{\mathbf{x}_1^{(c)}, \ldots, \mathbf{x}_t^{(c)}, \mathbf{x}_{t+1}^{(c)}, \ldots\}$.

A viable approach is to solve at each time instant $t = 1, 2, \ldots$, the composite, time-varying optimization problem [cf. (6)]

$$\mathbf{w}_{c,t}^\star := \operatorname*{argmin}_{\mathbf{w}_c} F_{c,t}(\mathbf{w}_c) := \overbrace{-\alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w}_c) + 2\beta\|\mathbf{w}_c\|^2}^{g(\mathbf{w}_c)} + \underbrace{\mathbb{I}\{\mathbf{w}_c \succeq \mathbf{0}\} + 2\mathbf{w}_c^\top \mathbf{z}_{c,1:t} - \gamma \sum_{k\neq c}^C 2\mathbf{w}_c^\top \mathbf{z}_{k,1:t}}_{h_{c,t}(\mathbf{w}_c)}. \tag{13}$$

In writing $\mathbf{z}_{c,1:t} \in \mathbb{R}_+^{N(N-1)/2}$ we make explicit that the Euclidean-distance matrix is computed using all class-c signals acquired up to time t. In this infinite-memory case there is a need for normalization, for instance by using instead the time average $\bar{\mathbf{z}}_{c,t} := \frac{\mathbf{z}_{c,1:t}}{t}$. Otherwise the signal-dependent terms in (13) would explode due to data accumulation in $\mathbf{z}_{c,1:t} = \mathbf{z}_{c,1:t-1} + \mathbf{z}_{c,t}$ as $t \to \infty$. This and other limited-memory averaging alternatives that are better suited for tracking variations in the graphs will be discussed in Section 4.1. Either way, as data come in, the edge-wise ℓ_1-norm weights in $\|\mathbf{W}_{c,t} \circ \mathbf{Z}_{c,1:t}\|_1 = 2\mathbf{w}_c^\top \mathbf{z}_{c,1:t}$ will fluctuate explaining the time dependence of $F_{c,t}(\mathbf{w}_c)$ through its non-differentiable component $h_{c,t}(\mathbf{w}_c)$. Notice that $g(\mathbf{w}_c)$ is time-invariant for this particular instance of the topology inference problem, but this need not be the case in general [28].

4.1. Online algorithm construction

A naive sequential estimation approach consists of solving (13) repeatedly using the batch PG algorithm in Section 3.2. However (pseudo) real-time operation in delay-sensitive applications may not tolerate running multiple inner PG iterations per time interval, so that convergence to $\mathbf{w}^{\star}_{c,t}$ is attained for each t. For time-varying graphs it may not be even prudent to obtain $\mathbf{w}^{\star}_{c,t}$ with high precision (hence incurring high delay and unnecessary computational cost), since at time $t+1$ a new datum arrives and the solution $\mathbf{w}^{\star}_{c,t+1}$ may be well off the prior estimate. These reasons motivate devising an efficient online and recursive algorithm to solve the time-varying optimization problem (13).

To this end, we build on recent advances in time-varying convex optimization, in particular the online PG methods in [18], [20]. Our algorithm construction approach entails two steps per time instant $t = 1, 2, \ldots$. First, we recursively update the upper-triangular entries $\mathbf{z}_{c,1:t} = \mathbf{z}_{c,1:t-1} + \mathbf{z}_{c,t}$ of the Euclidean-distance matrix via an exponential moving average (EMA), namely

$$\bar{\mathbf{z}}_{c,t} = (1-\theta)\bar{\mathbf{z}}_{c,t-1} + \theta \mathbf{z}_{c,t}. \tag{14}$$

The constant $\theta \in (0,1)$ is a discount factor, which downweighs past data to facilitate tracking dynamic graphs in non-stationary environments. The closer θ becomes to 1, the faster EMA discounts past observations. Another viable alternative is to use a fix-length sliding window [20], whereby the Euclidean distance matrix $\mathbf{z}_{c,t-L+1:t}$ is computed using the most recent $L \geq 1$ graph signals from class c. For the infinite-memory case (suitable for time-invariant class-specific data distributions), one can rely on recursive updates of the sample mean $\bar{\mathbf{z}}_{c,t} = \frac{\mathbf{z}_{c,1:t}}{t} = \frac{t-1}{t}\bar{\mathbf{z}}_{c,t-1} + \frac{\mathbf{z}_{c,t}}{t}$. In any case, recursive updates of the vectorized Euclidean-distance matrix can be carried out in $\mathcal{O}(N^2)$ complexity (and memory) per instant t. We initialize $\bar{\mathbf{z}}_{c,0}$ from a few graph signals acquired before the online algorithm starts.

In the second step, we run a single iteration

$$\mathbf{w}_{c,t+1} = \mathbf{prox}_{\mu_t h_{c,t}}(\mathbf{w}_{c,t} - \mu_t \nabla g(\mathbf{w}_{c,t})), \tag{15}$$

of the batch graph learning algorithm developed in Section 3 to update $\mathbf{w}_{t+1}$, with $\mu_t < \frac{2}{\eta} = \left(2\beta + \frac{\alpha(N-1)}{d^2_{\min}}\right)^{-1}$. An adaptive step-size rule μ_t whereby $d_{\min}$ is replaced with $\min(\mathbf{S}\mathbf{w}_{c,t-1})$ is an alternative that works well in practice. In a nutshell, (15) amounts to letting iterations $k = 1, 2, \ldots$ in Algorithm 1 match the time instants t of signal acquisition. Especially when dealing with slowly-varying graphs or in delay-tolerant applications, running a few PG iterations during $(t-1, t]$ would likely improve recovery performance; see also the discussion following Theorem 1. The gradient calculation is unchanged from the batch case [see (7)], and the non-negative soft-thresholding operator is now defined in terms of a time-varying threshold $\tilde{\boldsymbol{\mu}}_t := 2\mu_t\left(\bar{\mathbf{z}}_{c,t} - \gamma \sum_{k\neq c}^{C} \bar{\mathbf{z}}_{k,t}\right)$. This is a direct manifestation of the fact that data enters the cost in (13) as time-varying ℓ_1-norm weights. Accordingly, the computational complexity remains $\mathcal{O}(N^2)$ per instant t – just as in the batch setting of Section 3. The resulting online iterations are tabulated under Algorithm 2 .

Unlike recent approaches that estimate dynamic graph topologies from the observation of smooth signals [21–23], Algorithm 2's memory footprint and computational cost per data sample $\mathbf{x}^{(c)}_t$ do not grow with t. Specifically, the algorithms in [21], [22], [23] operate in batch fashion and the number of optimization variables $(\mathbf{W}_1, \ldots, \mathbf{W}_T$ or their Laplacian counterparts) increase by a factor of T when time-varying graphs are learnt over a horizon $t = 1, \ldots, T$. Moreover, these algorithms can only start once all data have been acquired and stored, rendering them unsuitable for online operation using streaming signals. Finally, the formulations in [21], [22], [23] rely on explicit regularization terms such as $\sum_{t=2}^{T} \|\mathbf{W}_t - \mathbf{W}_{t-1}\|_F^2$ in order to constrain the temporal variation of the estimated graphs; see also [29] for a recent approach whereby signals are assumed to be stationary over the dynamic networks. The proposed online PG iterations offer an implicit proximal regularization, so there is no need to enforce similarity between consecutive graphs explicitly in the criterion. Details can be found in [30], but the argument is that the iteration (15) can be viewed as a proximal regularization of the linearized function g at $\mathbf{w}_{c,t}$, namely

$$\mathbf{w}_{c,t+1} = \operatorname*{argmin}_{\mathbf{w}} \Big\{ g(\mathbf{w}) + (\mathbf{w} - \mathbf{w}_{c,t})^{\top}\nabla g(\mathbf{w}_{c,t}) + \frac{1}{2\mu_t}\|\mathbf{w} - \mathbf{w}_{c,t}\|^2 + h_{c,t}(\mathbf{w}) \Big\}.$$

Notice how the third term in the objective function imposes said implicit temporal regularization between consecutive graphs. Before moving on to issues of convergence, a remark is in order.

Algorithm 2: Online discriminative graph learning, class c.

Input parameters $\alpha, \beta, \gamma, \theta$, stream $\mathbf{z}_{c,1}, \mathbf{z}_{c,2}, \ldots$, initial $\mathbf{w}_{c,1}, \bar{\mathbf{z}}_{c,0}$.
for $t = 1, 2, \ldots,$ **do**
  Update $\bar{\mathbf{z}}_{c,t} = (1-\theta)\bar{\mathbf{z}}_{c,t-1} + \theta \mathbf{z}_{c,t}$.
  Update $\mu_t = \left(4\beta + \frac{2\alpha(N-1)}{\min(\mathbf{S}\mathbf{w}_{c,t})^2}\right)^{-1}$.
  Update $\tilde{\boldsymbol{\mu}}_t = 2\mu_t\left(\bar{\mathbf{z}}_{c,t} - \gamma \sum_{k\neq c}^{C} \bar{\mathbf{z}}_{k,t}\right)$.
  Compute $\nabla g(\mathbf{w}_{c,t}) = 4\beta\mathbf{w}_{c,t} - \alpha\mathbf{S}^{\top}\left(\frac{\mathbf{1}}{\mathbf{S}\mathbf{w}_{c,t}}\right)$.
  Update $\mathbf{w}_{c,t+1} = [\mathbf{w}_{c,t} - \mu_t\nabla g(\mathbf{w}_{c,t}) - \tilde{\boldsymbol{\mu}}_t]_+$.
end

Remark 2 (Online graph learning from smooth signals)**.** By setting $\gamma = 0$ in (13) and Algorithm 2, (to the best of our knowledge) we obtain the very first online algorithm to learn graphs from the observation of streaming smooth signals. This is a novel contribution of independent interest beyond the (discriminative) classification task-oriented focus of this paper. As we were preparing the final version of this manuscript, we became aware of recent related work on online graph learning via prediction-correction algorithms [31]. While the framework proposed therein is undoubtedly general, concrete iterations are developed for the problem of time-varying GMRF model selection. Similar to the discriminative graphical lasso algorithm in [15], lacking Laplacian constraints on the sought precision matrices, the connections with smoothness priors are tenuous. Moreover, Algorithm 2 is a first-order method while prediction-correction iterations also require (second-order) Hessian information. This is likely to result in increased computational load. Finally, [31] does not offer convergence guarantees. A related online topology inference framework was put forth in [28], but observations therein are modeled as stationary graph signals generated by local diffusion dynamics on the sought network. Altogether, [28], [31] along with the ideas presented here underscore the potential of recent advances in time-varying convex optimization towards tackling dynamic network topology inference problems in various relevant settings.

4.2. Convergence analysis

Here we establish that Algorithm 2 can closely track the sequence of batch minimizers $\mathbf{w}^{\star}_{c,t}$ [recall (13)] for large enough t; see also the corroborating simulations in Sections 5.2 and 5.5. Our results build on the performance guarantees derived in [20] for online subspace clustering algorithms. To simplify notation, we henceforth drop the subindex c from all relevant quantities. The

results hold as stated for the estimated graphs in all classes $c = 1, \ldots, C$.

Instrumental to stating bounds for the tracking error $\|\mathbf{w}_t - \mathbf{w}_t^\star\|$ is to note that $g(\mathbf{w})$ in (13) is 4β-strongly convex (see Appendix B) and its gradient is η-Lipschitz continuous. It thus follows that $\mathbf{w}_t^\star$ is the unique minimizer of (13). Next, let us define $\nu_t := \|\mathbf{w}_{t+1}^\star - \mathbf{w}_t^\star\|$ to quantify the temporal variability of the optimal solution of (13). Since $g(\mathbf{w})$ is strongly convex, we have the following (non-asymptotic) performance guarantee for Algorithm 2.

Theorem 1. *For all $t \geq 2$, the sequence of iterates $\mathbf{w}_t$ generated by Algorithm 2 satisfies:*

$$\|\mathbf{w}_t - \mathbf{w}_t^\star\| \leq \tilde{L}_{t-1}\left(\|\mathbf{w}_1 - \mathbf{w}_1^\star\| + \sum_{\tau=1}^{t-1} \frac{\nu_\tau}{\tilde{L}_\tau}\right), \tag{16}$$

where $L_t := \max\{|1-4\mu_t\beta|, |1-\mu_t\eta_t|\}$, $\tilde{L}_t := \prod_{\tau=1}^{t} L_\tau$. Moreover, for the sequence of objective values we can write $F_t(\mathbf{w}_t) - F_t(\mathbf{w}_t^\star) \leq \frac{\eta_t}{2}\|\mathbf{w}_t - \mathbf{w}_t^\star\|$; see [32].

Theorem 1 is adapted from [20], and the proof can be obtained via straightforward modifications to the arguments therein. If one could afford taking i_t PG iterations (instead of a single one as in Algorithm 2) per time step t, the performance gains can be readily evaluated by substituting $\tilde{L}_t = \prod_{\tau=1}^{t} L_\tau^{i_\tau}$ in Theorem 1 to use the bound (16).

To gain further insights on the tracking bound, let us define $\hat{L}_t := \max_{\tau=1,\ldots,t} L_\tau$, $\hat{\nu}_t := \max_{\tau=1,\ldots,t} \nu_\tau$. With the aid of these upper bounds, the sum of the geometric series in the right-hand side of (16) can be simplified to

$$\|\mathbf{w}_t - \mathbf{w}_t^\star\| \leq \left(\hat{L}_{t-1}\right)^t \|\mathbf{w}_1 - \mathbf{w}_1^\star\| + \frac{\hat{\nu}_t}{1-\hat{L}_{t-1}}. \tag{17}$$

Accordingly, $\hat{L}_t = (\eta_t - 4\beta)/\eta_t < 1$ since in Algorithm 2 we set $\mu_t = \eta_t^{-1}$. Therefore, $(\hat{L}_{t-1})^t \to 0$ and Algorithm 2 hovers on the vicinity of the optimal solution with a misadjustment $\hat{\nu}_t/(1-\hat{L}_{t-1})$. It follows that the tracking error increases with $\hat{\nu}_t$ (rapidly-varying class-conditional graphs are more challenging to track) and also if the problem is badly conditioned (i.e., $\beta \to 0$ or $\eta_t \to \infty$ in which case $\hat{L}_t \to 1$).

5. Numerical experiments

Here we test both the discriminative graph learning and the online graph learning approaches on different synthetic and real-world signals. A comprehensive performance evaluation is carried out for the proposed frameworks. In the case of discriminative graph learning we: (i) assess the classification accuracy; (ii) evaluate the similarity of the learned graph and the ground truth via the F-measure of the detected edges (defined as the harmonic mean between edge precision and recall) [33]; (iii) illustrate the distribution of the resulting GFT coefficients; (iv) compare with state-of-the-art methods; and (v) check if our findings are aligned with the literature. Also, we investigate the performance of the online discriminative graph learning approach by evaluating how it follows its offline counterpart. For online graph learning we: (i) investigate how the online algorithm tracks the optimum objective value; and (ii) study the similarity of the learned graph and the ground truth, even in dynamic environments.

For the synthetic data experiments, we generate i.i.d. smooth signals with respect to the underlying graphs we wish to recover. The signals are drawn from a Gaussian distribution

$$\mathbf{X} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{L}^\dagger + \sigma_e^2 \mathbf{I}_N\right), \tag{18}$$

where σ_e represents the noise level; see e.g., [9]. Throughout, we perform a grid search to determine the best regularization parameters α, β, γ and report the results. The optimality criterion depends on the task at hand. Specifically, in classification tasks (Sections 5.1, 5.3, and 5.4) we choose the parameters that lead to the best classification accuracy. When the goal is to track a time-varying graph topology (Section 5.2), we select the parameters that result in the most accurate graph recovery in terms of F-measure. Also, after learning the graph, we remove the weak connections by thresholding edge weights below 10^{-3}.

5.1. Learning discriminative graphs from synthetic data

In this section, we illustrate the effectiveness of our graph learning framework in controlled synthetic settings. We carry out comparisons with other state-of-the-art methods such as the approach developed in [10] (Kalofolias) and the method proposed in [15] (DISC). Moreover, to determine if the learned graphs and the adopted smoothness assumption are beneficial in boosting classification accuracy, we also compared against a network-agnostic baseline whereby the raw signals are used as features in a Support Vector Machine (SVM) [34] classifier. The evaluation is conducted in two different scenarios as follows.

5.1.1. Scenario one

Consider two families of random graphs: (i) Erdős–Rényi (ER) [35], and (ii) Barabási–Albert (BA) [36]. In this experiment, for $N = 60$ we generate an ER graph with edge-formation probability $p = 0.1$, and a BA graph by adding a new node to the graph each time, connecting to 3 existing nodes in the graph. This will result in sparse graphs with edge density of approximately 0.1. Then, 100 independent random signals are generated using (18); half of which are smooth over the ER graph and the other half over the BA graph. The signals have different levels of noise $\sigma_e \in [0.05, 3]$. We use 80% of the data for training and the remaining 20% is used for testing. This procedure is repeated over 50 trials, with 50 different ER and BA graph realization.

A summary of our results can be found in Fig. 1. Fig. 1(a–c) depicts the classification accuracy and F-measure (for each graph family) averaged over 50 trials. Mean values are accompanied by 95% confidence intervals. It is apparent from Fig. 1(a) that our discriminative graph learning approach outperforms state-of-the-art methods in terms of classification accuracy. As expected, by increasing the noise level we can see the drop in the performance of all methods (still, our algorithm uniformly outperforms the competing alternatives). The algorithm in [10] is designed to accurately recover the class-conditional graphs, but it is not optimized for a downstream classification task. Inspection of Fig. 1(b,c) shows that the proposed method – despite introducing biases to effect discriminability – can still accurately recover the ground-truth graph for both model classes (the gap with [10] is indistinguishable). On the other hand, the DISC method [15] focuses more on discrimination and its graph recovery performance markedly degrades (especially for in the low-noise regime); see Fig. 1(b,c). As expected, DISC is competitive in terms of classification accuracy. Overall, these results suggest that our method is more robust in noisy scenarios and it can accurately recover the class-conditional graphs while maintaining high discriminative power.

As mentioned in Section 3.3, we use the cumulative relative energy of the first one-third GFT coefficients in order to classify the test signals. To assess the discrimination ability of these spectral features, we compute the GFT of the test signals with respect to each of the learned graphs (ER and BA) and then evaluate the respective energy distributions across GFT coefficients. For test signals in the ER class, Fig. 1(d) shows the mean magnitude of the GFT coefficients obtained with respect to each of the learned graphs. The signals are noticeably smoother with respect to the learnt ER class graph; see the higher magnitude of the GFT coefficients corresponding to the lower frequencies. On the other hand,

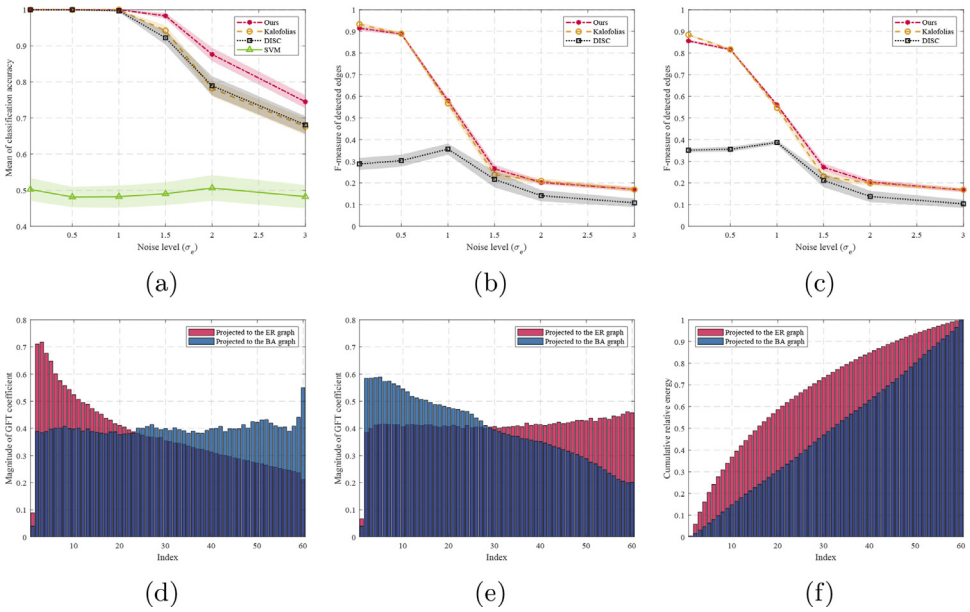

Fig. 1. The mean of (a) signal classification accuracy, (b) F-measure of detected edges for ER graph, and (c) F-measure of detected edges for BA graph as a function of the noise level. Our method outperforms other methods in both tasks. Magnitude of GFT coefficients averaged over all trials for (d) ER class, and (e) BA class. Apparently, the test signals of each class are smooth over the corresponding learned graph while they are non-smooth over the other class. (f) The cumulative relative energy of projected test signals on the ER graph. The cumulative relative energy is used for the classification and must be higher for the corresponding class in low frequencies (first couple of indices).

for the learnt BA class graph we observe strong energy coefficients in the higher-end of the spectrum. To reinforce these observations, we plot the cumulative relative energy distributions in Fig. 1(f). A similar trend is observed for the test signals in the BA class; see Fig. 1(e) where now projections onto the first few eigenvectors of the BA class graph carry most of the signal energy.

5.1.2. Scenario two

Different from the previous scenario, here we generate 100 random graphs (50 from each class, all with $N = 100$ nodes) by replacing a proportion of edges with new ones drawn from the same family of random graphs. This way we try to replicate a more realistic setting whereby graphs have some common patterns, but otherwise they can exhibit small fluctuation in their topology across samples. The 100 smooth signals over these graphs are synthetically generated using (18), where $\sigma_e = 1$. The ER graphs have edge-formation probability $p = 0.1$, and BA graphs are generated by adding a new node to the graph each time, connecting to 5 existing nodes in the graph. We repeat this procedure 50 times while varying the proportions of edges replaced.

Since we generate 100 different random graphs, the evaluation of the F-measure is not feasible or even meaningful here. Therefore, in Fig. 2 we depict the mean classification accuracy averaged over 50 trials (along with the corresponding 95% confidence intervals) as a function of the proportion of edges redrawn from sample to sample. Once more, our algorithm uniformly outperforms all the aforementioned state-of-the-art methods. For all graph-based approaches, it is noticeable how the classification performance degrades as the perturbations in the graph topology increase.

Departing from batch solutions, next we analyze the online discriminative graph learning approach in a similar setting. To this end, we create 100 random graphs (50 for each class, among which 40 are used for training and 10 for testing) with $N = 100$ nodes by retaining 25% of edges, and each time redrawing the others from the same family of random graphs. The ER and BA parameters are identical to those used in the previous test case. We assume that the training signals are acquired in a streaming fashion (so the class-conditional graphs are learnt on the fly), but we have access

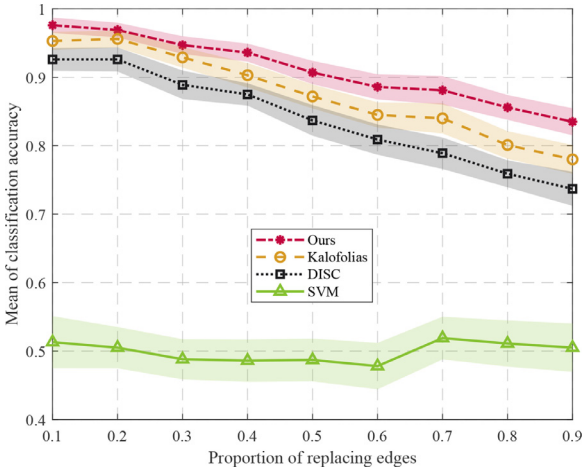

Fig. 2. The average signal classification accuracy as a function of the proportion of edge replacing. The proposed method outperforms competing alternatives in all cases.

to the whole test data to evaluate the classification performance at each time slot. For each class we generate 50 training samples per graph, so the training horizon is $2000 = 50 \times 40$ in Fig. 3. We train online using Algorithm 2, and the obtained time-varying classification accuracy over the test set (averaged over 50 independent trials) is depicted in Fig. 3. Notice how after an initial learning period of around 1200 time slots, the online classification algorithm attains the performance of the optimal batch classifier (which learns the class-conditional graphs in one-shot using Algorithm 1, jointly using all the training signals acquired so far).

5.2. Online graph learning from synthetic data

To assess the performance of the proposed online graph learning algorithm (without discriminative term; $\gamma = 0$ see Remark 2), we test Algorithm 2 on simulated streaming data. There is no classification problem we are solving here, the goal is to recover the topology of a time-varying network. To this end, we generate a

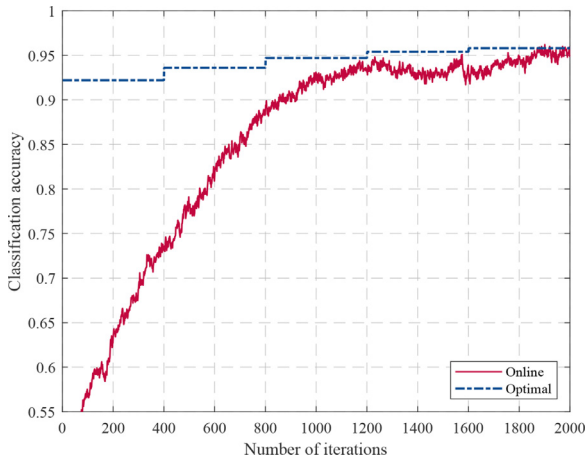

Fig. 3. The average signal classification accuracy as a function of iterations. After acquiring enough time samples the online algorithm converges to its offline counterpart.

piecewise-constant sequence of two random ER graphs (edge formation probability $p = 0.1$) with $N = 30$ nodes. The initial graph switches in the middle of the experiment (specifically after $t = 10,000$ time slots). The final graph is obtained from the initial one by redrawing 40% of its edges. Then, we synthetically generate i.i.d. smooth signals with respect to the time-varying underlying graph. The signals are drawn according to (18) by setting $\sigma_e = 0.05$. Results are averaged over 10 independent Monte Carlo trials. As a baseline we compute the optimal time-varying solution F_t [cf. (13)], by running the batch PG algorithm (Algorithm 1) until convergence every 500 time samples. For this experiment we set $\theta = 0.003$ in Algorithm 2.

Fig. 4 (a) shows that after around 500 time samples (iterations) the objective value of the online Algorithm 2 comes close to the optimal value F_t of its time-varying batch counterpart. The observed oscillation depends on the step size μ_t in Algorithm 2. As expected, increasing the step size leads to faster convergence with larger oscillations. Conversely, reducing the step size leads to smoother and slower convergence. The other noteworthy observation is that the objective value markedly increases when the graph changes (after $10,000$ time samples), but the online algorithm can effectively track the dynamic graph after a sufficient number of samples have been acquired. A similar trend can be observed for the F-measure of the detected edges; see Fig. 4(b). For visual comparison we illustrate the snapshots of the estimated adjacency matrices at time samples $10,000$ and $20,000$, and compare them with the ground-truth topologies before and after the switch point; see Fig. 4(c). The similarities are noticeable.

5.3. EEG emotion recognition

In this section, we apply the discriminative graph learning algorithm on a real-world problem, namely emotion recognition using EEG signals. To this end, we resort to a widely used and publicly available EEG data-set called DEAP [37].

The DEAP data-set contains EEG and peripheral physiological signals of 32 participants. The data were recorded while subjects were watching one-minute long music videos. Each participant is subjected to 40 trials (music videos) and rates each video in terms of the levels of valence, arousal, like/dislike, dominance, and familiarity [37]. In this numerical test, we focus on valence and arousal classification. Ratings are decimal numbers between 1 and 9 and in order to make this a classification task we divide the ratings into two classes: *low* when the ratings are smaller than 5; and *high* when ratings are larger than or equal to 5. We exploit the pre-processed version of the data set which contains $N = 32$ EEG channels with 128 Hz sampling rate and the 3 s pre-trial baseline is discarded.

Table 1
The average of EEG emotion classification accuracy over all subjects.

Study	Valence	Arousal
Proposed method	**92.73**	**93.44**
Kalofolias [10]	86.56	88.91
Chao and Liu [38]	77.02	76.13
Koelstra et al. [37]	57.60	62.00
Chung and Yoon [39]	66.60	66.40
Rozgić et al. [40]	76.90	69.10
Chen et al. [41]	76.17	73.59
Tripathi et al. [42]	81.40	73.36

We perform the classification task in leave-one-trial-out scheme where for each subject we use 39 trials as the training set and test on the one remaining trial. By cycling over the left-out trial, We repeat this 40 times for each subject and report the mean classification accuracy. Consequently, this is a subject dependent procedure which means that we perform training and classification for each subject separately. Classification follows the same procedure described in Section 3.3 and we project the test signals on the first 1/4th of the GFT bases (i.e., the Laplacian eigenvectors associated to the smallest 8 eigenvalues). The discriminative graph is learned for the training set via Algorithm 1 and then normalized to have unit Frobenius norm. The average classification accuracy over all the trials and all the participants is reported in Table 1. Results for other state-of-the-art methods are also reported for comparison. Results of the DISC algorithm in [15] are omitted since the provided code did not converge for some of the DEAP dataset trials.

As Table 1 shows, the proposed discriminative graph learning approach outperforms state-of-the-art methods in the classification of emotions. Moreover, we can see 6% and 5% improvement relative to [10] in the classification of valence and arousal, respectively. The added discriminative term in the cost function is key towards enhancing classification performance. Additionally, there exist some other emotion classification studies using the DEAP dataset. However, their classification settings are different from what we considered in our experimental analysis, e.g. [43,44] divide the valence-arousal space into four sub-spaces and perform a 4-class emotion recognition.

Now that we established the superior performance of our graph-based classifier, it is of interest to investigate whether there is any useful information in the learned topologies. To this end, we first discard the trials that have ratings in the interval (4.5,5.5), where the participants themselves are not confident enough in rating the trials. Secondly, we study the connections that are significantly different between classes. Finally, we decompose the EEG signals using low-, band-, and high-pass graph filters to visualize spatial activations and investigate whether our findings are aligned with the literature.

Accordingly, we learn two graphs corresponding to low and high emotions per person, using the parameters that led to the best classification performance. In Fig. 5, we show the mean of the connections over all subjects and the significantly different connections between low and high emotions. Interestingly, it appears that the edge weights are related to the intensity of the emotions, i.e., valence and arousal. As shown in Fig. 5, the graphs corresponding to high valence/arousal (Fig. 5(b,e)) tend to exhibit stronger connections than the learned graphs for low valence/arousal (Fig. 5(a,d)). We also identify significantly different connections across the class-conditional networks. To this end, we apply the non-parametric Wilcoxon rank-sum test [45]. Fig. 5(c,f) show connections that are significantly different between low and

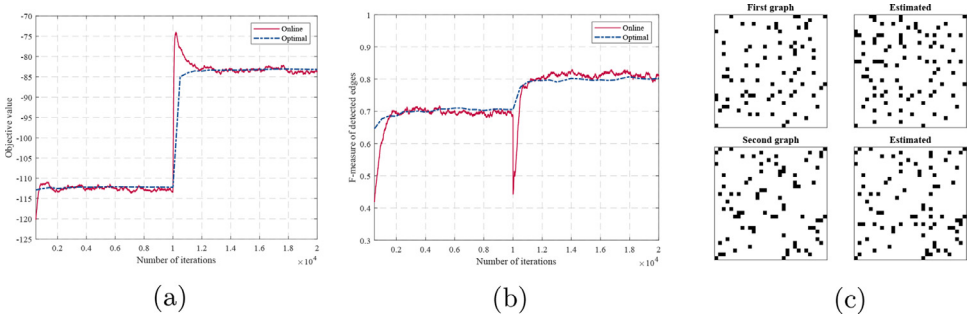

Fig. 4. The mean of (a) optimization objective value, and (b) F-measure of detected edges as a function of acquired time samples (iteration). (c) A snapshot of the estimated graphs using the online algorithm and the ground truth underlying graph. These results indicate that the online graph learning algorithm can track its offline counterpart.

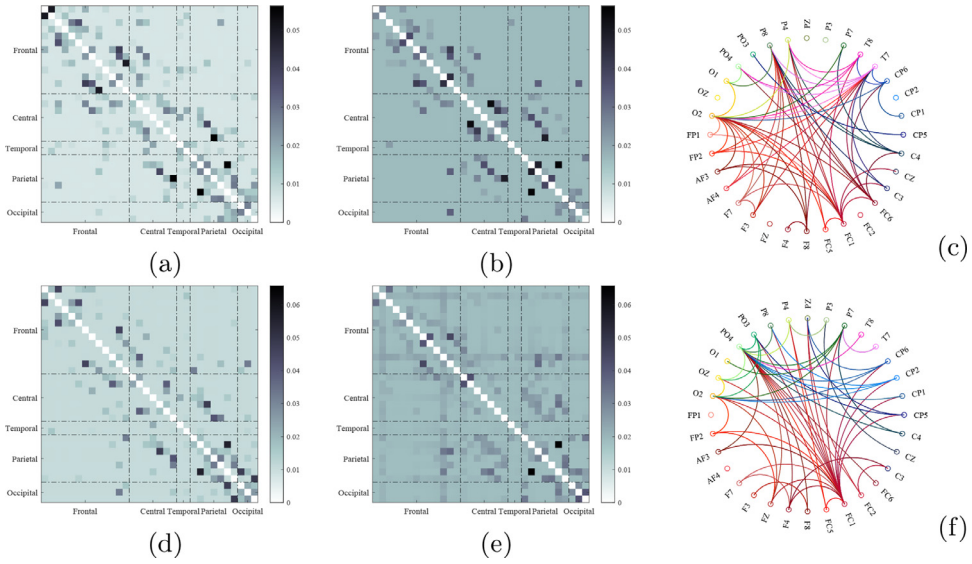

Fig. 5. The average of learned graphs for (a) low, and (b) high valence over all subjects. (c) Significantly different connections between low and high valence with $p \leq 0.002$. The mean of learned graphs for (d) low, and (e) high arousal over all subjects. (f) Significantly different connections between low and high arousal with $p \leq 0.03$. These results suggest the learned graphs for low and high emotions show significant difference with each other. Color shades in (c) and (f) encode connections involving different brain regions.

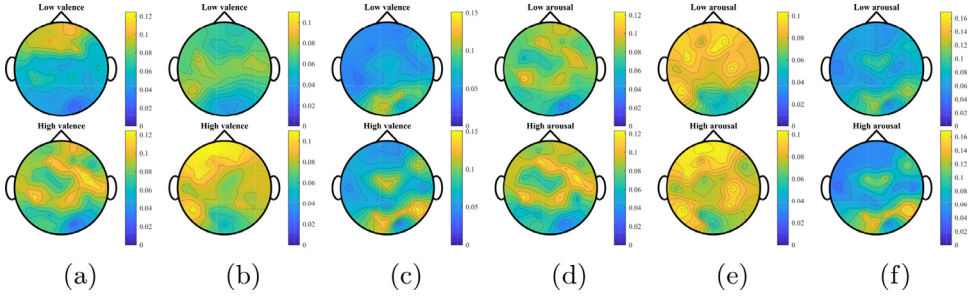

Fig. 6. The mean of the eigenvector magnitudes corresponding to (a) low, (b) mid, and (c) high frequency for valence. The mean of the eigenvector magnitudes corresponding to (d) low, (e) mid, and (f) high frequency for arousal. These results demonstrate the different patterns across low and high emotions, where most of these differences are aligned with the literature.

high valence with $p \leq 0.002$ and between low and high arousal with $p \leq 0.03$, respectively. We find that the significantly different connections across different classes incorporate almost all the channels that were originally mentioned in [37].

Frequency analysis is a common theme in the EEG signal processing literature. Therefore, we conduct a similar analysis by bringing to bear the GFT induced by the learned graphs. Fig. 6(a–c) depicts the average magnitudes of the eigenvector sets associated with low (first 1/4th components), mid, and high (last 1/4th components) frequencies, respectively. The arousal counterparts are shown in Fig. 6(d–f). The asymmetrical pattern of the frontal EEG activity is apparent from Fig. 6(a,b), which is consistent with the findings in [46] about valence. Also, as it is noted in Fig. 5(c) most of the connections are related to the frontal lobe (red-shaded colors). Earlier findings suggest that for classifying positive from negative emotions, the features are generally in the right occipital lobe and parietal lobe for the alpha band, the central lobe for the beta band, and the left frontal and right temporal lobe for the gamma band [47]. Since we carry out the classification focusing on the low graph-frequency components, the relevant features can be visualized in Fig. 6(a,d). Apparently, there are noticeably different pat-

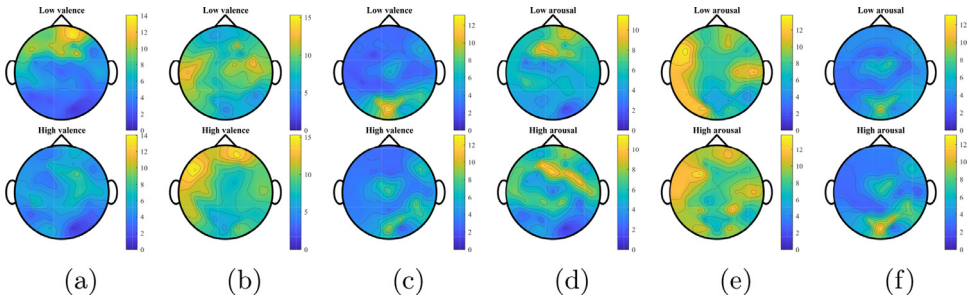

Fig. 7. The average of the absolute value of decomposed signals with respect to (a) low, (b) mid, and (c) high frequency for valence. The average of the absolute value of decomposed signals with respect to (d) low, (e) mid, and (f) high frequency for arousal. Different patterns can be found in these results between low and high emotions.

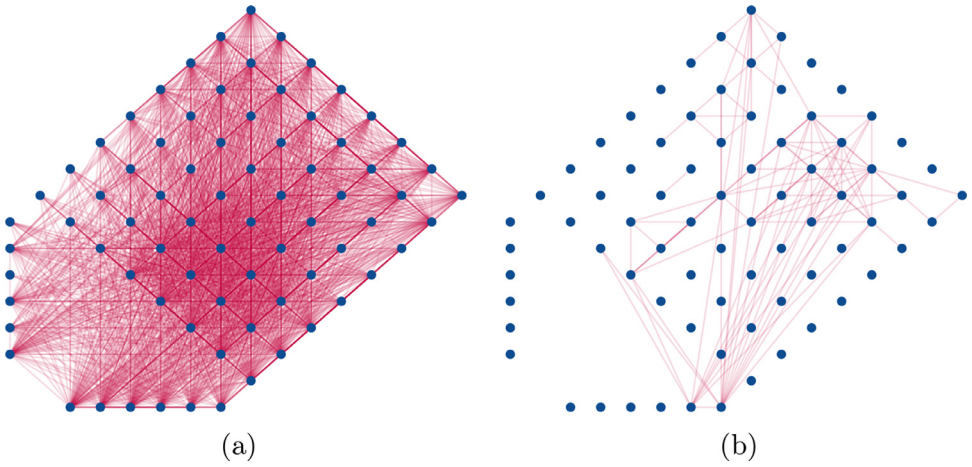

Fig. 8. Learned graph for (a) pre-ictal, and (b) ictal periods. One can see the drop in the number of connections when the seizure happens. This drop of connections is noticeable in the bottom corner of the grid and in the two strips.

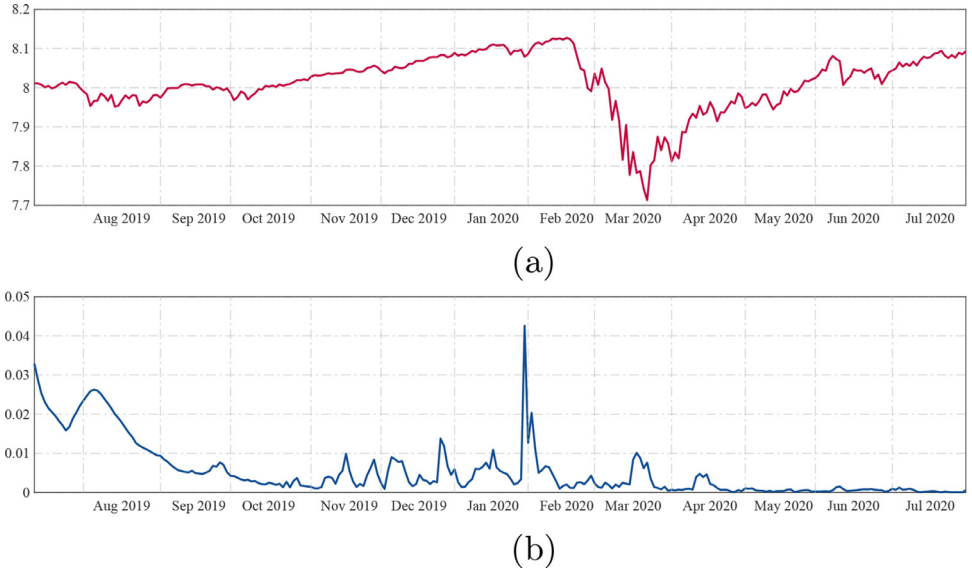

Fig. 9. (a) The S&P500 log-price per day. (b) The relative temporal variation over the days. The temporal variation indicates sudden changes which can be due to some market tensions.

terns in the left frontal, right temporal, central, and parietal lobe between high and low valence/arousal; see Fig. 6(a,d).

Finally, we also we decompose the EEG signals using low-, band-, and high-pass graph filters to visualize spatial activations at different levels of variability with respect to the learned graphs. The mean absolute values of the decomposed signals are once more superimposed to the human scalp in Fig. 7. Different patterns of activation can be identified in low and high valence/arousal. More specifically, different patterns in the frontal lobe are captured in low frequency for both valence and arousal. For mid-frequency, we detect a distinct pattern between high and low valence in the left frontal and right temporal.

5.4. Seizure detection

Next, we test our batch algorithm on electrocorticogram (ECoG) data to detect epileptic seizures [3]. To this end, we use one of

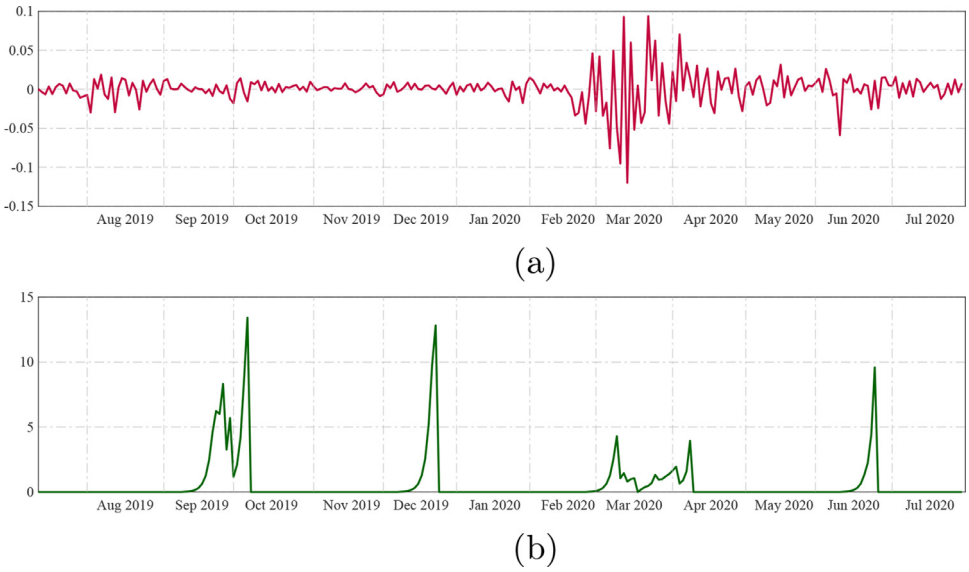

Fig. 10. (a) The relative daily temporal variation (RDTV) of S&P500 price per day. (b) The algebraic connectivity of the estimated graph over the days when the RDTV of stock prices is the graph signal. The algebraic connectivity trends suggest that at the time of crisis the network graph tends to get connected, while it disconnects when the market is more stable.

the publicly available data sets for epileptic seizure detection.[1] This data set contains 8 instances of seizures that are acquired from a human patient with epilepsy. The ECoG data are recorded via 76 electrodes at the sampling rate of 400 Hz. Sixty-four electrodes are in the shape of an 8×8 grid and implanted at the cortical level. In order to record the voltage activity from mesial temporal structures, the other 12 electrodes (two strips of 6 electrodes) are located deeper in the brain [3,48]. Each seizure is annotated by an epileptologist to identify two key segments relating to the seizure i.e., pre-ictal (before the epileptic attack) and ictal (during the epileptic attack) periods.

The task at hand is to classify pre-ictal and ictal periods. In this direction, we use Algorithm 1 to obtain discriminative graph representations of the ECoG signals for the pre-ictal and ictal periods. We perform the classification in a leave-one-trial-out scheme, just like in Section 5.3. Admittedly, this is a simple classification task and we achieve 100% classification accuracy averaged over all trials. Arguably the most valuable information can be gleaned from the learned graphs in Fig. 8. Fig. 8(a) shows the learned graph for the pre-ictal period while Fig. 8(b) depicts the representation for the ictal period. Our findings in Fig. 8 indicate a significant reduction in the overall level of brain connectivity during the seizures. Consistent with the findings in Kramer et al. [48], we observe the edge thinning is more prominent in the bottom corner of the grid and along the two strips. According to [48], localized network sparsification patterns could help identify spatial regions from where the seizures emanate.

5.5. Stock price data analysis

In this test case, we adopt Algorithm 2 to estimate a time-varying graph from real financial data. This is not a classification task, rather the goal is to explore the dynamic structure of a network capturing the relationships among some leading US companies. To this end, we consider the daily stock prices for ten large US companies, namely: Microsoft (MSFT), Apple (AAPL), Amazon (AMZN), Facebook (FB), Alphabet class A shares (GOOGL), Alphabet class C shares (GOOG), Johnson & Johnson (JNJ), Berkshire Hathaway (BRK-B), Visa (V), and Procter and Gamble (PG). We collect their daily stock prices $p_i(t)$ from Yahoo! Finance over the time period from May 1st, 2019 to August 1st, 2020, which overlaps with the COVID-19 pandemic that led to significant market instabilities. Under normal circumstances, we would expect limited variations in the network describing the pairwise relationships between the chosen stock prices, since these large companies are well-established in the market [49]. However, events like COVID-19 can cause abrupt changes in such a network. We run Algorithm 2 to estimate daily graphs in order to monitor the sudden changes in the stock market. To this end, we consider both (i) logarithmic daily stock prices $\log p_i(t)$ and (ii) the relative daily temporal variation (RDTV) of the stock prices $|p_i(t) - p_i(t-1)|/|p_i(t-1)|$ as the input signal. Following studies like [21,49], we quantify the variation of the network via (i) relative temporal deviation $\|\mathbf{W}_t - \mathbf{W}_{t-1}\|_F / \|\mathbf{W}_{t-1}\|_F$, and (ii) algebraic connectivity, i.e. the second smallest eigenvalue of the combinatorial graph Laplacian $\mathbf{L}_t$. Since we have limited amount of data, the discount factor θ and the employed step size in Algorithm 2 are 0.8 and $\mu = 2/\eta$, respectively.

First, we use logarithmic stock prices as the input of our graph learning problem. In Fig. 9(a) we plot the S&P500 log-price as an indicator of the market's condition. The relative temporal variation of the learned graphs is illustrated in Fig. 9(b). The graphs are obtained by setting the regularization parameters as $\alpha = 0.316$, $\beta = 0.05$, and $\gamma = 0$. Since there is no ground-truth dynamic network, we endeavor to indirectly validate the proposed method by commenting on the intuitive structure observed from the learned sequence of graphs. Fig. 9(a) appears to suggest that the COVID-19 impact on the markets started in late January 2020 and it got to its worse situation during March 2020. The following sudden changes are apparent by inspection of the sudden spikes in Fig. 9(b): (i) September 2019, (ii) November 2019, (iii) December 2019, (iv) January 2020, (v) March 2020, and (vi) April 2020. While we can only conjecture on the reason behind these spikes, some are consistent with major events occurring during these time periods. For instance the major spike at the end of January 2020 is probably the result of the World Health Organization (WHO) declaring a global health emergency due to the COVID-19 pandemic.

[1] The data can be found at http://math.bu.edu/people/kolaczyk/datasets.html

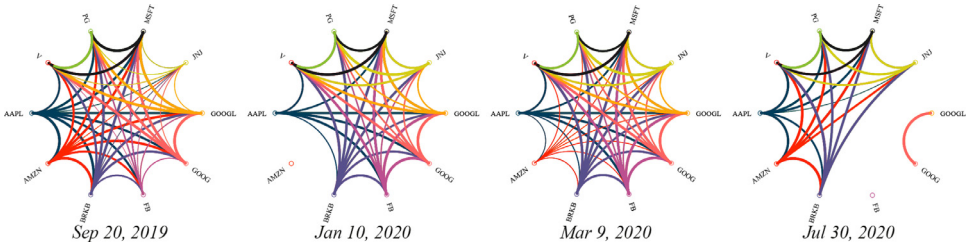

Fig. 11. The estimated network of the market over 4 different days: September 20th, 2019, January 10th, 2020, March 9th, 2020, and July 30th, 2020. The sought graph gets connected at the time of crisis when all the stock prices usually drop. While in stable times, the graph tends to be disconnected.

Next we consider the RDTV of the stock prices as input graph signals, and estimate a time-varying graph by setting the regularization parameters $\alpha = 3.162$, $\beta = 0.177$, and $\gamma = 0$. For reference, the RDTV of the S&P500 index is shown in Fig. 10(a). Moreover, the algebraic connectivity of the sequence of graphs is depicted in Fig. 10(b). The trend appears to suggest that graphs are usually disconnected, except during times of high market volatility (e.g., a financial crisis); see also the networks in Fig. 11. A majority of stock prices usually drop sharply during a crisis, making the RDTV (i.e., discrete price gradients) signals negative. In pursuing graphs for which these signals are smooth, it is thus expected one would find highly connected topologies since most nodal signals exhibit similar trends. On the other hand, during low-volatility times the RDTV gradients are relatively uncorrelated across companies. Hence, the learned graph are expected to be sparser and even possibly disconnected. To illustrate these network connectivity patterns, Fig. 11 illustrates representative graph estimates during stable times as well as during high-volatility periods.

6. Conclusion

In this work, we proposed a novel framework for online task-driven graph learning by leveraging recent advances at the crossroads of GSP and time-varying convex optimization. A particular task at hand is supervised classification of signals assumed to be smooth over latent class-conditional graphs. To recover said graph representations of the data during the training phase, we develop proximal gradient algorithms for discriminative network topology inference given (possibly streaming) smooth signals per class. The intuition behind the optimality criterion guiding the search of the class-conditional graphs is simple: we look for a class c graph over which (i) class c signals are smooth (to capture the data structure); and (ii) all other signals are non-smooth (for discriminability). The GFTs of the optimum graphs can then be used to extract discriminative features of the test signals we wish to classify. We validate the proposed classification pipeline on both synthetic and real data, for instance in an emotion recognition task from EEG data where we outperform state-of-the-art methods developed for the DEAP data-set. In this context, the recovered class-conditional networks can inform significant connections between brain regions to discern among emotional states.

Moreover, results in this paper contribute to the algorithmic foundations of online network topology inference from streaming signals (in a classification setting and beyond). We develop for the first time an online algorithm to track the possibly slowly-varying topology of a dynamic graph, given streaming signals that are smooth over the sought network. The algorithmic framework comes with tracking performance guarantees, is computationally efficient and has a fixed memory storage requirement irrespective of the temporal horizon's span. Numerical tests with synthetic data show the online topology estimates closely track the performance of the batch estimator benchmark, even when the network is subject to abrupt changes in connectivity. We also applied our algorithm to recent stock-prices from leading US companies, which resulted in time-varying graphs that reveal interesting relationships among the firms as well as the financial market's response to the COVID-19 pandemic.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper Section 5.4.

Acknowledgment

The authors would like to thank Mark Kramer and Eric Kolaczyk (Boston University) for providing useful clarifications on the epilepsy dataset studied in.

Appendix A. Lipschitz constant of ∇g

Recall the expression for ∇g in (7). For arbitrary points $\mathbf{x}, \mathbf{y} \in \mathbb{R}_+^{N(N-1)/2}$ representing graph topologies we have

$$\begin{aligned}
\|\nabla g(\mathbf{x}) - \nabla g(\mathbf{y})\| &= \left\| 4\beta(\mathbf{x}-\mathbf{y}) - \alpha \mathbf{S}^\top \left(\frac{\mathbf{1}}{\mathbf{S}\mathbf{x}} - \frac{\mathbf{1}}{\mathbf{S}\mathbf{y}} \right) \right\| \\
&\leq 4\beta \|\mathbf{x}-\mathbf{y}\| + \alpha \|\mathbf{S}^\top\| \left\| \frac{\mathbf{1}}{\mathbf{S}\mathbf{x}} - \frac{\mathbf{1}}{\mathbf{S}\mathbf{y}} \right\| \\
&\leq 4\beta \|\mathbf{x}-\mathbf{y}\| + \frac{\alpha \|\mathbf{S}\|}{\min(\mathbf{S}\mathbf{x}) \min(\mathbf{S}\mathbf{y})} \|\mathbf{S}\mathbf{x} - \mathbf{S}\mathbf{y}\| \\
&\leq 4\beta \|\mathbf{x}-\mathbf{y}\| + \frac{\alpha \|\mathbf{S}\|}{d_{\min}^2} \|\mathbf{S}\mathbf{x} - \mathbf{S}\mathbf{y}\| \\
&= 4\beta \|\mathbf{x}-\mathbf{y}\| + \frac{\alpha \|\mathbf{S}\|^2}{d_{\min}^2} \|\mathbf{x}-\mathbf{y}\| \\
&= \left(4\beta + \frac{2\alpha(N-1)}{d_{\min}^2} \right) \|\mathbf{x}-\mathbf{y}\|,
\end{aligned}$$

establishing the Lipschitz continuity of the gradient. Note that $\mathbf{1}/(\mathbf{S}\mathbf{x})$ stands for element-wise division and nodal degrees are assumed to be larger than some prescribed $d_{\min} > 0$. In deriving the last equality, we used that $\|\mathbf{S}\| = \sqrt{2(N-1)}$ – a simple result we state next for completeness.

Lemma 1. *With reference to (6), let $\mathbf{S} \in \{0,1\}^{N \times N(N-1)/2}$ be such that $\mathbf{W}_c \mathbf{1} = \mathbf{S}\mathbf{w}_c$. Then the spectral norm of $\mathbf{S}$ is $\|\mathbf{S}\| = \sqrt{2(N-1)}$, where N is the number of nodes in $\mathcal{G}_c$.*

Proof. Since $\mathbf{S}$ maps $\mathbf{w}_c$ to nodal degrees $\mathbf{d}_c$, then $\mathbf{S}$ has exactly $N-1$ ones in each row. Hence the diagonal entries of $\mathbf{S}\mathbf{S}^\top$ are all $N-1$ and the off-diagonal entries are equal to 1. Moreover, $\|\mathbf{S}\|$ is equal to the square root of the spectral radius of $\mathbf{S}\mathbf{S}^\top = (N-2)\mathbf{I} + \mathbf{1}\mathbf{1}^\top$. The eigenvalues are solutions to the characteristic equation

$$\det\left(\mathbf{S}\mathbf{S}^\top - \lambda \mathbf{I}\right) = \det\left((N-2)\mathbf{I} + \mathbf{1}\mathbf{1}^\top - \lambda \mathbf{I}\right)$$

$$
\begin{aligned}
&= \det(\underbrace{(N-2-\lambda)\mathbf{I}}_{:=\mathbf{R}} + \mathbf{1}\mathbf{1}^\top) \\
&= \det(\mathbf{R}) + \mathbf{1}^\top \mathrm{adj}(\mathbf{R})\mathbf{1} \\
&= \prod_{i=1}^{N} R_{ii} + \sum_{j=1}^{N} \prod_{i\neq j} R_{ii} \\
&= (N-2-\lambda)^N + N(N-2-\lambda)^{N-1} \\
&= (2N-2-\lambda)(N-2-\lambda)^{N-1} = 0.
\end{aligned}
$$

In arriving at the third equality we used the Sherman-Morrison formula, where adj(**R**) denotes the adjugate matrix of **R**. All in all, the roots are $\lambda_1 = 2(N-1) > \lambda_2 = \ldots = \lambda_N = N-2$ and the result follows. □

Appendix B. Strong convexity of g

To establish that g in (13) is strongly convex with constant $m = 4\beta > 0$, it suffices to note that

$$g(\mathbf{w}) - \frac{m}{2}\|\mathbf{w}\|^2 = -\alpha \mathbf{1}^\top \log(\mathbf{S}\mathbf{w})$$

is a convex function.

References

[1] S.S. Saboksayr, G. Mateos, M. Cetin, EEG-based emotion classification using graph signal processing, in: IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), Toronto, ON, 2021. (to appear)

[2] S.S. Saboksayr, G. Mateos, M. Cetin, Online graph learning under smoothness priors, in: European Signal Process. Conf. (EUSIPCO), Dublin, Ireland, 2021. (submitted; see also arXiv:2103.03762 [cs.LG])

[3] E.D. Kolaczyk, Statistical Analysis of Network Data: Methods and Models, Springer-Verlag, New York, NY, 2009.

[4] A. Ortega, P. Frossard, J. Kovačević, J.M.F. Moura, P. Vandergheynst, Graph signal processing: overview, challenges, and applications, Proc. IEEE 106 (5) (2018) 808–828.

[5] G. Mateos, S. Segarra, A.G. Marques, A. Ribeiro, Connecting the dots: Identifying network structure via graph signal processing, IEEE Signal Process. Mag. 36 (3) (2019) 16–43.

[6] X. Dong, D. Thanou, M. Rabbat, P. Frossard, Learning graphs from data: a signal representation perspective, IEEE Signal Process. Mag. 36 (3) (2019) 44–63.

[7] B. Pasdeloup, V. Gripon, G. Mercier, D. Pastor, M.G. Rabbat, Characterization and inference of graph diffusion processes from observations of stationary signals, IEEE Trans. Signal Inf. Process. Netw. PP (99) (2017) 481–496.

[8] S. Segarra, A. Marques, G. Mateos, A. Ribeiro, Network topology inference from spectral templates, IEEE Trans. Signal Inf. Process. Netw. 3 (3) (2017) 467–483.

[9] X. Dong, D. Thanou, P. Frossard, P. Vandergheynst, Learning Laplacian matrix in smooth graph signal representations, IEEE Trans. Signal Process. 64 (23) (2016) 6160–6173.

[10] V. Kalofolias, How to learn a graph from smooth signals, in: Artif. Intel. and Stat. (AISTATS), 2016, pp. 920–929.

[11] G.B. Giannakis, Y. Shen, G.V. Karanikolas, Topology identification and learning over graphs: accounting for nonlinearities and dynamics, Proc. IEEE 106 (5) (2018) 787–807.

[12] S. Sardellitti, S. Barbarossa, P. Di Lorenzo, Graph topology inference based on sparsifying transform learning, IEEE Trans. Signal Process. 67 (7) (2019) 1712–1727.

[13] Y. Yankelevsky, M. Elad, Dual graph regularized dictionary learning, IEEE Trans. Signal Inf. Process. Netw. 2 (4) (2016) 611–624.

[14] D. Thanou, D.I. Shuman, P. Frossard, Learning parametric dictionaries for signals on graphs, IEEE Trans. Signal Process. 62 (15) (2014) 3849–3862.

[15] J. Kao, D. Tian, H. Mansour, A. Ortega, A. Vetro, Disc-GLasso: discriminative graph learning with sparsity regularization, in: IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), 2017, pp. 2956–2960.

[16] H.P. Maretic, P. Frossard, Graph Laplacian mixture model, IEEE Trans. Signal Inf. Process. Netw. 6 (2020) 261–270.

[17] R. Vidal, Subspace clustering, IEEE Signal Process. Mag. 28 (2) (2011) 52–68.

[18] E. Dall'Anese, A. Simonetto, S. Becker, L. Madden, Optimization and learning with information streams: time-varying algorithms and applications, IEEE Signal Process. Mag. 37 (3) (2020) 71–83.

[19] A. Simonetto, E. Dall'Anese, S. Paternain, G. Leus, G.B. Giannakis, Time-varying convex optimization: time-structured algorithms and applications, Proc. IEEE 108 (2020) 1–17.

[20] L. Madden, S. Becker, E. Dall'Anese, Online sparse subspace clustering, in: IEEE Data Sci. Wrksp. (DSW), 2019, pp. 248–252.

[21] J.V.d. M. Cardoso, D.P. Palomar, Learning undirected graphs in financial markets, arXiv preprint arXiv:2005.09958[stat.ML](2020).

[22] V. Kalofolias, A. Loukas, D. Thanou, P. Frossard, Learning time varying graphs, in: IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), 2017, pp. 2826–2830.

[23] K. Yamada, Y. Tanaka, A. Ortega, Time-varying graph learning with constraints on graph temporal variation, arXiv preprint arXiv:2001.03346[eess.SP](2020).

[24] A.G. Marques, S. Segarra, G. Mateos, Signal processing on directed graphs, IEEE Signal Process. Mag. 37 (6) (2020) 99–116.

[25] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, second ed., Springer, New York, 2009.

[26] N. Komodakis, J. Pesquet, Playing with duality: an overview of recent primal—dual approaches for solving large-scale optimization problems, IEEE Signal Process. Mag. 32 (6) (2015) 31–54.

[27] N. Parikh, S. Boyd, Proximal algorithms, Found. Trends Optim. 1 (3) (2014) 127–239.

[28] R. Shafipour, G. Mateos, Online topology inference from streaming stationary graph signals with partial connectivity information, Algorithms 13 (9) (2020) 1–19.

[29] M. Navarro, Y. Wang, A.G. Marques, C. Uhler, S. Segarra, Joint inference of multiple graphs from matrix polynomials, arXiv preprint arXiv:2010.08120[stat.ML](2020).

[30] A. Beck, M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, SIAM J. Imag. Sci. 2 (2009) 183–202.

[31] A. Natali, M. Coutino, E. Isufi, G. Leus, Online time-varying topology identification via prediction-correction algorithms, in: IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), Toronto, ON, 2021. (submitted; see also arXiv:2010.11634 [eess.SP])

[32] A. Beck, First-order Methods in Optimization, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2018.

[33] C. Manning, P. Raghavan, H. Schütze, Introduction to information retrieval, Nat. Lang. Eng. 16 (1) (2010) 100–103.

[34] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.

[35] P. Erdős, A. Rényi, On the evolution of random graphs, Publ. Math. Inst. Hung. Acad. Sci. 5 (1) (1960) 17–60.

[36] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.

[37] S. Koelstra, C. Muhl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, I. Patras, DEAP: a database for emotion analysis using physiological signals, IEEE Trans. Affect. Comput. 3 (1) (2011) 18–31.

[38] H. Chao, Y. Liu, Emotion recognition from multi-channel EEG signals by exploiting the deep belief-conditional random field framework, IEEE Access 8 (2020) 33002–33012.

[39] S.Y. Chung, H.J. Yoon, Affective classification using Bayesian classifier and supervised learning, in: Intl. Conf. Control, Automation and Systems (ICCAS), 2012, pp. 1768–1771.

[40] V. Rozgić, S.N. Vitaladevuni, R. Prasad, Robust EEG emotion classification using segment level decision fusion, in: IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP), 2013, pp. 1286–1290.

[41] M. Chen, J. Han, L. Guo, J. Wang, I. Patras, Identifying valence and arousal levels via connectivity between EEG channels, in: Intl. Conf. Affect. Comput. and Intel. Interaction (ACII), 2015, pp. 63–69.

[42] S. Tripathi, S. Acharya, R.D. Sharma, S. Mittal, S. Bhattacharya, Using deep and convolutional neural networks for accurate emotion classification on DEAP dataset, in: Proc. of the Thirty-First AAAI Conference on Artificial Intelligence, 2017, pp. 4746–4752.

[43] W. Zheng, J. Zhu, B. Lu, Identifying stable patterns over time for emotion recognition from EEG, IEEE Trans. Affect. Comput. 10 (3) (2019) 417–429.

[44] P. Li, H. Liu, Y. Si, C. Li, F. Li, X. Zhu, X. Huang, Y. Zeng, D. Yao, Y. Zhang, P. Xu, EEG based emotion recognition by combining functional connectivity network and local activations, IEEE Trans. Biomed. Eng. 66 (10) (2019) 2869–2881.

[45] M.P. Fay, M.A. Proschan, Wilcoxon–Mann–Whitney or *t*-test? On assumptions for hypothesis tests and multiple interpretations of decision rules, Stat. Surv. 4 (2010) 1.

[46] L.A. Schmidt, L.J. Trainor, Frontal brain electrical activity (EEG) distinguishes valence and intensity of musical emotions, Cogn. Emot. 15 (4) (2001) 487–500.

[47] D. Nie, X. Wang, L. Shi, B. Lu, EEG-based emotion recognition during watching movies, in: Intl. IEEE/EMBS Conf. Neural Eng., 2011, pp. 667–670.

[48] M.A. Kramer, E.D. Kolaczyk, H.E. Kirsch, Emergent network topology at seizure onset in humans, Epilepsy Res. 79 (2) (2008) 173–186.

[49] D. Hallac, Y. Park, S. Boyd, J. Leskovec, Network inference via the time-varying graphical lasso, in: Proc. of the 23rd ACM SIGKDD Internationalsi Conference on Knowledge Discovery and Data Mining, 2017, pp. 205–213.