Untrained Graph Neural Networks for Denoising

Samuel Rey, *Member, IEEE*, Santiago Segarra, *Senior Member, IEEE*, Reinhard Heckel, *Member, IEEE*, and Antonio G. Marques, *Senior Member, IEEE*

***Abstract*—A fundamental problem in signal processing is to denoise a signal. While there are many well-performing methods for denoising signals defined on regular domains, including images defined on a two-dimensional pixel grid, many important classes of signals are defined over *irregular* domains that can be conveniently represented by a graph. This paper introduces two untrained graph neural network architectures for graph signal denoising, develops theoretical guarantees for their denoising capabilities in a simple setup, and provides empirical evidence in more general scenarios. The two architectures differ on how they incorporate the information encoded in the graph, with one relying on graph convolutions and the other employing graph upsampling operators based on hierarchical clustering. Each architecture implements a different prior over the targeted signals. Finally, we provide numerical experiments with synthetic and real datasets that i) asses the denoising behavior predicted by our theoretical results and ii) compare the denoising performance of our architectures with that of existing alternatives.**

***Index Terms*—Geometric deep learning, graph decoder, graph signal denoising, graph signal processing.**

I. Introduction

VAST amounts of data are generated and stored every day, propelling the deployment of data-driven solutions to address a wide variety of real-world problems. Unfortunately, the input data suffers from imperfections and is corrupted with noise, oftentimes associated with the data-collection process. Noisy signals appear in a gamut of applications, with examples including the processing of voice and images, the measurements in electric, social and transportation networks, or the monitoring of biological signals [1], [2]. The presence of noise entails a detrimental influence on the quality of the data, which may become unusable when the noise power is comparable to that of the signal. As a result, separating the signal from the noise, which is referred to as signal denoising, is a critical and ubiquitous task in contemporary data science applications. While most existing works focus on the denoising of signals defined over regular domains (time and space), signals with irregular supports are becoming pervasive. In particular, signals obtained from sensors deployed across different positions, such as voltage in power networks, temperature in weather stations, or neurological activity on the brain, have largely benefited from graph signal denoising since sensor measurements are typically corrupted by noise [3]. Hence, designing (nonlinear) denoising schemes for signals defined over irregular domains constitutes a relevant problem both from a theoretical and practical point of view.

A versatile and tractable approach to overcome the challenges inherent to data supported on irregular domains consists of representing the underlying structure as a graph, with nodes representing variables and edges encoding levels of similarity, influence, or statistical dependence among nodes. Successful examples of this approach can be found in the subareas of network analytics, machine learning over graphs, and graph signal processing (GSP) [1], [4], [5], with graph neural networks (GNNs) and GSP being particularly relevant for the architectures presented in this paper [6], [7]. Note that traditional data-processing architectures are designed to deal with data defined over regular domains, such as images, and hence, they may incur difficulties when learning and exploiting the more complex structure present in many contemporary applications. Nonetheless, GSP provides a principled approach to handling this issue [2], [5], [6]. Assuming that the structure of the signals can be modeled by a graph, GSP uses the information encoded in the graph topology to analyze, process, and learn from the data. As a result, it is not surprising that GSP has been successfully applied to design and analyze GNNs [7], [8], [9], [10], a class of neural network (NN) architectures that incorporate the graph topology information to enhance their performance when the data is composed of signals defined over a graph.

The importance of leveraging the graph influence when using deep nonlinear architectures is reflected in the wide range of GNNs that co-exist in the literature, including graph convolutional NNs (GCNNs) [11], [12], [13], graph recurrent NNs [14], graph autoencoders [15], [16], [17], graph generative adversarial networks [18], [19], and simplicial NNs [20], [21], to name a few. Incorporating the graph structure into deep nonlinear models involves a wide range of options when designing the architecture. For example, GCNNs can be defined with or without pooling layers and the convolution over a graph can be implemented in

Manuscript received 8 June 2022; revised 26 September 2022 and 4 November 2022; accepted 15 November 2022. Date of publication 23 November 2022; date of current version 7 December 2022. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. David I Shuman. This work was partially supported by the Spanish AEI under Grants SPGRAPH (PID2019-105032GB-I00/AEI/10.13039/501100011033), FPU17/04520, and EST21/00420, F661-MAPPING-UCI CAM-URJC, F663-AAGNCS CAM-URJC, F861 AUTO-BA-GRAPH CAM-URJC, and in part by the USA NSF award CCF-2008555, and in part by the Institute of Advanced Studies at the Technical University of Munich, and in part by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grants 456465471 and 464123524. *(Corresponding author: Antonio G. Marques.)*

Samuel Rey and Antonio G. Marques are with the Department of Signal Theory and Comms., King Juan Carlos University, 28933 Madrid, Spain (e-mail: samuel.rey.escudero@urjc.es; antonio.garcia.marques@urjc.es).

Santiago Segarra is with the ECE Department, Rice University, Houston, TX 77005 USA (e-mail: segarra@rice.edu).

Reinhard Heckel is with the ECE Department, Technical University of Munich, 80333 Munich, Germany (e-mail: reinhard.heckel@tum.de).

Digital Object Identifier 10.1109/TSP.2022.3223552

several ways (vertex vs frequency), each leading to architectures with different properties and performances. In fact, one of the key questions when designing a GNN is to decide the particular way in which the graph is incorporated into the architecture.

Considering the preceding paragraphs, the goal of this work is twofold. First, we explore different ways of incorporating the information encoded in the graph and propose new graph-based NN architectures to denoise graph signals. Second, we provide theoretical guarantees for the denoising capabilities of this approach and show that it is directly influenced by the properties of the graph. The mathematical analysis, performed on particular instances of these architectures, provides guarantees on their denoising performance under specific assumptions for the original signal and its underlying graph. In addition, we provide empirical evidence about the denoising performance of our method for scenarios more general than those strictly covered by our theory, further illustrating the value of our graph-aware untrained architectures.

The proposed architectures are *untrained* NNs, meaning that the parameters of the network are optimized using only the signal observation that we want to denoise, avoiding the dependency on a training set with multiple observed graph signals. The underlying assumption behind this *untrained* denoising architecture is that, due to the graph-specific structure incorporated into the different layers, when tuning the network parameters using stochastic gradient steps, the NNs are capable of learning (matching) the structure of the signal faster than that of the noise. Hence, the denoising process is carried out separately for each individual observation by fitting the weights of the NN and stopping the updates after a few iterations. This same phenomenon has been observed to hold true in non-graph deep learning architectures [22], [23] and constitutes a framework that is closely related to that of zero-shot learning [24], [25]. In the context of signal denoising, the consideration of an overparametrized graph-aware architecture along with early stopping avoids overfitting to the noise.

To incorporate the topology of the graph, the first architecture multiplies the input at each layer by a fixed (non-learnable) graph filter [26], which can be seen as a generalization of the convolutional layer in [12]. The second architecture performs graph upsampling operations that, starting from a low-dimensional latent space, progressively increase the size of the input until it matches the size of the signal to denoise. The sequence of upsampling operators is designed based on hierarchical clustering algorithms [16], [27], [28], [29] so that, in contrast to [30], matrix inversions are not required, avoiding the related numerical issues. Our work is substantially different from [16], [17], which deal with graph encoder-decoder architectures. On top of our theoretical analysis and extensive numerical simulations, additional differences to prior work are that: (a) our graph decoder is an untrained network, and thus, it does not need a training phase; (b) we only require a decoder-like architecture for denoising graph signals, so it is not necessary to jointly design and train two different architectures as carried out in, [16], [17].

Contributions and outline. In summary, the contributions of the paper are the following: (i) we present two new overparametrized and untrained GNNs for solving graph-signal denoising problems; (ii) mathematical analysis is conducted for each architecture offering bounds for their performance, improving our understanding of nonlinear architectures and the influence of incorporating graph structure into NNs; and (iii) the proposed architectures are evaluated and compared to other denoising alternatives through numerical experiments carried out with synthetic and real-world data. These contrast with the contributions of our preliminary work in [31], which only considered a single underparametrized denoising architecture, did not provide mathematical analysis, and focused on synthetic datasets. Moreover, moving to the overparametrized regime not only endows the proposed architectures with a larger learning capacity, but it also opens the door to a more thorough theoretical analysis.

The remainder of the paper is organized as follows. Section I-A reviews related works dealing with graph-signal denoising. Section II explains fundamental concepts leveraged along the paper. Section III formally introduces the problem at hand and presents our general approach. Sections IV and V detail the proposed architectures and provide the mathematical analysis for each of them. Numerical experiments are presented in Section VI and concluding remarks are provided in Section VII.

A. Related Works

Untrained NNs enable the recovery of signals without the need of training over large (or any) datasets by carefully incorporating prior information of the signals [22], [23], [32], [33]. In [22], it is shown that fitting a standard convolutional autoencoder to only one noisy signal using early stopping enables the effective denoising of an image. For this approach to work, it is key that the signal class (images) matches the NN architecture (2D convolutional NN with particular filters).

Previous approaches to the graph-signal denoising task included a graph-regularization term that promoted desired properties on the estimated signals [34]. Some existing works minimize the graph total variation pushing the signal value at neighboring nodes to be close [34], [35]. Later on, total generalized variation extended this idea to promote similar values of higher-order terms [36]. A related approach assumes that the signals are smooth on the graph and add a regularization parameter based on the quadratic form of the graph Laplacian [37]. Also, in [38], the authors propose a spectral graph trilateral filter as a regularizer, based on the prior assumption that the gradient is smooth over the graph. It is worth noting that these alternatives rely on imposing some notion of smoothness on the original graph signal. Furthermore, classical denoising methods typically assume that the signal and the graph are related by a linear or a quadratic mapping. Nonetheless, the actual relation between the signal and the graph may be of a different nature and, in fact, in many relevant applications the actual prior is more complex than that represented by linear and quadratic terms, motivating the development of nonlinear models.

More recently, nonlinear solutions for denoising graph signals have been proposed to tackle the aforementioned issues. In [39], a median graph filter [40] is used to denoise time-varying graph signals defined over dynamic graphs. A different nonlinear

TABLE I
SUMMARY OF MAIN NOTATION

Symbol	Explanation
$\mathcal{G}, N$	Graph, number of nodes.
$\mathcal{V}, \mathcal{E}$	Set of nodes and edges.
$\mathbf{A}, \boldsymbol{\mathcal{A}} \in \mathbb{R}^{N\times N}$	Adjacency matrix, expectation of $\mathbf{A}$.
$\mathbf{V} \in \mathbb{R}^{N\times N}$	Matrix containing the eigenvectors of $\mathbf{A}$.
$\boldsymbol{\Lambda} \in \mathbb{R}^{N\times N}$	Diagonal matrix collecting the eigenvalues of $\mathbf{A}$.
F, $\boldsymbol{\Theta}$	Number of features of the GNN, learnable parameters.
$\mathbf{Z} \in \mathbb{R}^{N^{(0)}\times F^{(0)}}$	Random input of the GNN.
$\mathcal{T}^{(\ell)}_{\boldsymbol{\Theta}^{(\ell)}}\{\cdot\vert\mathcal{G}\}$	Linear graph-dependent transformation of the GNN at layer ℓ.
$f_{\boldsymbol{\Theta}}(\mathbf{Z}\vert\mathcal{G})$	GNN architecture.
$\mathbf{x}, \mathbf{n} \in \mathbb{R}^N$	Noisy signal observation, noise vector.
$\mathbf{x}_0, \hat{\mathbf{x}}_0 \in \mathbb{R}^N$	Original signal, denoised estimate.
$\boldsymbol{\mathcal{X}} \in \mathbb{R}^{N\times N}$	Expected squared Jacobian of $f_{\boldsymbol{\Theta}}(\mathbf{Z}\vert\mathcal{G})$ with respect to $\boldsymbol{\Theta}$.
$\mathbf{W} \in \mathbb{R}^{N\times N}$	Matrix containing the eigenvectors of $\boldsymbol{\mathcal{X}}$.
$\boldsymbol{\Sigma} \in \mathbb{R}^{N\times N}$	Diagonal matrix containing the eigenvalues of $\boldsymbol{\mathcal{X}}$.
$\mathcal{M}(\boldsymbol{\mathcal{A}})$	Set of SBMs with expected adjacency matrix $\boldsymbol{\mathcal{A}}$.
$\mathcal{M}_N(\beta_{min}, \rho)$	Set of SBMs with minimum expected degree increasing with N.
$f_{\boldsymbol{\Theta}}(\mathbf{H})$, $f_{\boldsymbol{\Theta}}(\mathbf{U})$	Two-layer GCG, two-layer GDec.
$\mathbf{H} \in \mathbb{R}^{N\times N}$	Graph filter.
$\mathbf{P}^{(\ell)} \in \mathbb{R}^{N^{(\ell)}\times N^{(\ell-1)}}$	Membership matrix at layer ℓ.
$\mathbf{U}^{(\ell)} \in \mathbb{R}^{N^{(\ell)}\times N^{(\ell-1)}}$	Upsampling matrix at layer ℓ.

approach is followed in [30], where a graph autoencoder is trained to recover the denoised signals. To change the size of the graph, the autoencoder relies on Kron reduction operations [41]. However, since the Kron reduction is based on the inverse of a submatrix of the graph Laplacian, it could fall into numerical issues if the submatrix is singular. Moreover, both architectures need several observations to recover the noiseless signals. Later on, [42] proposes a graph *unrolling* architecture based on GCNNs to approach the denoising task. The architecture is trained in an unsupervised fashion and relies on regularizing the objective function to avoid learning the noise. Differently, our proposed solution implicitly encodes the regularization in the architectures enabling them to learn the signal faster than the noise.

II. Processing Architectures for Graph Signals

This section introduces mathematical notation and the fundamentals of GSP and GNNs. In addition, the main notation is summarized in Table I. Readers familiar with these concepts may give a quick pass and move on to Section III.

A. Fundamentals of GSP

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote an undirected[1] graph, where $\mathcal{V}$ is the set of N nodes, and $\mathcal{E}$ is the set of links such that (i, j) belong to $\mathcal{E}$ if nodes i and j are connected. For a given graph $\mathcal{G}$, the symmetric adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ has non-zero entries A_{ij} only if $(i, j) \in \mathcal{E}$. The value of A_{ij} captures the strength of the link between nodes i and j. Define the degree matrix as $\mathbf{D} = \text{diag}(\mathbf{A1})$, where $\mathbf{1}$ is the vector of all ones and $\text{diag}(\cdot)$ is the diagonal operator that turns a vector into a diagonal matrix. A popular alternative to $\mathbf{A}$ is the degree normalized adjacency matrix $\tilde{\mathbf{A}} := \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$. Indeed, in the subsequent discussions, we assume that the rows and columns of $\mathbf{A}$ are normalized by its degree, so that $\mathbf{A} = \tilde{\mathbf{A}}$. Finally, when the adjacency matrix is symmetric, we can write $\mathbf{A} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$, where $\mathbf{V}$ is an orthonormal $N \times N$ matrix collecting the eigenvectors of $\mathbf{A}$ and $\boldsymbol{\Lambda}$ is diagonal matrix collecting its eigenvalues.

[1]Although our theoretical results assume that the graph is undirected, the proposed architectures can tackle signals defined on directed graphs [43].

Graph signals. In this paper, we focus on the processing of graph signals which are defined on $\mathcal{V}$. Graph signals can be represented as a vector $\mathbf{x} = [x_1, \ldots, x_N]^\top \in \mathbb{R}^N$, where the i-th entry represents the value of the signal at node i. Since the signal $\mathbf{x}$ is defined on $\mathcal{G}$, the core assumption of GSP is that either the values or the properties of $\mathbf{x}$ depend on the topology of $\mathcal{G}$ [44]. For instance, consider a graph that encodes similarity. If the value of A_{ij} is high, then one expects the signal values x_i and x_j to be akin to each other. This rationale helps to explain the success of GNNs since the incorporation of $\mathcal{G}$ into the architectures amounts to including prior information about the signals to process.

Graph filtering. Graph filters, an important tool of GSP, play a fundamental role in the definition of our GNN architectures. Graph filters are linear operators $\mathbb{R}^N \to \mathbb{R}^N$ that can be expressed as a polynomial of the adjacency matrix of the form

$$\mathbf{H} := \sum_{m=0}^{M-1} h_m \mathbf{A}^m, \tag{1}$$

where $\mathbf{H}$ is the graph filter, h_m are the filter coefficients, and $M \leq N$ [26]. Since $\mathbf{A}^m$ encodes the m-hop neighborhoods of the graph, graph filters can be used to diffuse input graph signals $\mathbf{x}$ across the graph as $\mathbf{y} = \sum_{m=0}^{M-1} h_m \mathbf{A}^m \mathbf{x} = \mathbf{H}\mathbf{x}$. Because graph filters diffuse signals across $(M-1)$-hop neighborhoods, they are widely used to generalize the convolution operation to signals defined over graphs.

Frequency representation. The theoretical analysis developed in this paper leverages the notion of bandlimited graph signals, a widely-used definition that links the properties of a signal to those of the (spectrum of) the supporting graph [6]. To be specific, the frequency representation of the signal $\mathbf{x}$ is given by the N-dimensional vector $\tilde{\mathbf{x}} = \mathbf{V}^\top\mathbf{x}$, with $\mathbf{V}^\top$ acting as the graph Fourier transform (GFT) [45]. Then, a graph signal is said to be bandlimited if $\tilde{\mathbf{x}}$ satisfies that $\tilde{x}_k = 0$ for $k > K$, where $K \leq N$ is referred to as the bandwidth of the signal $\mathbf{x}$. If $\mathbf{x}$ is bandlimited with bandwidth K it holds that

$$\mathbf{x} = \mathbf{V}_K \tilde{\mathbf{x}}_K, \tag{2}$$

with $\tilde{\mathbf{x}}_K = [\tilde{x}_1, \ldots, \tilde{x}_K]$ collecting the active frequency components and $\mathbf{V}_K$ collecting the corresponding K eigenvectors. This reduced-dimensionality representation, which can be generalized to graph filters as well, has been shown to bear practical relevance in real-world datasets and it is exploited in denoising and other inverse problems [46].

B. Fundamentals of GNNs

Generically, we represent a GNN using a parametric nonlinear function $f_{\boldsymbol{\Theta}}(\mathbf{Z}\vert\mathcal{G}) : \mathbb{R}^{N^{(0)}\times F^{(0)}} \to \mathbb{R}^N$ that depends on the graph $\mathcal{G}$. The parameters of the architecture are collected in $\boldsymbol{\Theta}$,

and the matrix $\mathbf{Z} \in \mathbb{R}^{N^{(0)} \times F^{(0)}}$ represents the input of the network. Despite the many possibilities for defining a GNN, a broad range of such architectures recursively apply a graph-aware linear transformation followed by an entry-wise nonlinearity. Then, a generic deep architecture $f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G})$ with L layers can be described as

$$\hat{\mathbf{Y}}^{(\ell)} = \mathcal{T}^{(\ell)}_{\boldsymbol{\Theta}^{(\ell)}} \left\{ \mathbf{Y}^{(\ell-1)} | \mathcal{G} \right\}, \quad 1 \leq \ell \leq L, \tag{3}$$

$$Y^{(\ell)}_{ij} = g^{(\ell)} \left(\hat{Y}^{(\ell)}_{ij} \right), \quad 1 \leq \ell \leq L, \tag{4}$$

where $\mathbf{Y}^{(0)} = \mathbf{Z}$ and $\mathbf{y} = \mathbf{Y}^{(L)}$ denote the input and output of the architecture, $\mathcal{T}^{(\ell)}_{\boldsymbol{\Theta}^{(\ell)}} \{\cdot|\mathcal{G}\} : \mathbb{R}^{N^{(\ell-1)} \times F^{(\ell-1)}} \to \mathbb{R}^{N^{(\ell)} \times F^{(\ell)}}$ is a *graph-aware* linear transformation, $\boldsymbol{\Theta}^{(\ell)} \in \mathbb{R}^{F^{(\ell-1)} \times F^{(\ell)}}$ are the parameters that define such a transformation, and $g^{(\ell)} : \mathbb{R} \to \mathbb{R}$ is a *scalar* nonlinear transformation (e.g., the ReLU function), which is oftentimes omitted in the last layer. Moreover, $N^{(\ell)}$ and $F^{(\ell)}$ represent the number of nodes and features at layer ℓ, $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}^{(\ell)}\}_{\ell=1}^{L}$ collects all the parameters of the architecture, and $\mathbf{y}$ is the output of the GNN. Note that although $f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G})$ generates output signals defined in $\mathbb{R}^N$, which is the case of interest for this paper, it can be easily adapted to output graph signals with more than one feature.

III. GNNs for Graph-Signal Denoising

We now formally introduce the problem of graph-signal denoising within the GSP framework, and present our approach to tackle it using untrained GNN architectures. Given the graph $\mathcal{G}$, let us consider the observed graph signal $\mathbf{x} \in \mathbb{R}^N$, which is a noisy version of the original graph signal $\mathbf{x}_0$. With $\mathbf{n} \in \mathbb{R}^N$ being a noise vector, the relation between $\mathbf{x}$ and $\mathbf{x}_0$ is

$$\mathbf{x} = \mathbf{x}_0 + \mathbf{n}. \tag{5}$$

Then, the goal of graph-signal denoising is to remove as much noise as possible from the observed signal $\mathbf{x}$ to estimate the original signal $\mathbf{x}_0$, which is performed by exploiting the information encoded in $\mathcal{G}$.

A traditional approach for the graph-signal denoising task is to solve an optimization problem of the form

$$\hat{\mathbf{x}}_0 = \operatorname{argmin}_{\check{\mathbf{x}}_0} \quad \|\mathbf{x} - \check{\mathbf{x}}_0\|_2^2 + \alpha R(\check{\mathbf{x}}_0|\mathcal{G}). \tag{6}$$

The first term promotes fidelity to the signal observations, the regularizer $R(\cdot|\mathcal{G})$ promotes denoised signals with desirable properties over the given graph $\mathcal{G}$, and $\alpha > 0$ controls the influence of the regularization. Common choices for the regularizer include the quadratic Laplacian $R(\mathbf{x}|\mathcal{G}) = \mathbf{x}^\top \mathbf{L} \mathbf{x}$ [37], or regularizers involving high-pass graph filters $R(\mathbf{x}|\mathcal{G}) = \|\mathbf{H}\mathbf{x}\|_2^2$ that foster smoothness on the estimated signal [34], [45].

While those traditional approaches exhibit a number of advantages (including interpretability, mathematical tractability, and convexity), they may fail to capture more complex relations between $\mathcal{G}$ and $\mathbf{x}_0$, motivating the development of nonlinear graph-denoising approaches.

Algorithm 1: Proposed Graph-Signal Denoising Method.

Inputs : $\mathbf{x}$ and $\mathcal{G}$
Outputs: $\hat{\mathbf{x}}_0$ and $\hat{\boldsymbol{\Theta}}(\mathbf{x})$
Set $f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G})$ as explained in Sec. IV or V
Generate $\mathbf{Z}$ from iid zero-mean Gaussian distribution
Initialize $\boldsymbol{\Theta}_{(0)}$ from iid zero-mean Gaussian
for $t = 1$ **to** T **do**
  Update $\boldsymbol{\Theta}_{(t)}$ minimizing (7) with SGD
end
$\hat{\boldsymbol{\Theta}}(\mathbf{x}) = \boldsymbol{\Theta}_{(T)}$
$\hat{\mathbf{x}}_0 = f_{\hat{\boldsymbol{\Theta}}(\mathbf{x})}(\mathbf{Z}|\mathcal{G})$

As summarized in Algorithm 1, in this paper we advocate handling the graph-signal denoising task by employing an overparametrized GNN (denoted by $f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G})$) as described in (3)–(4). The weights of the architecture, collected in $\boldsymbol{\Theta}$, are learned by minimizing the loss function

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\Theta}) = \frac{1}{2} \|\mathbf{x} - f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G})\|_2^2, \tag{7}$$

applying stochastic gradient descent (SGD) in combination with early stopping to avoid overfitting the noise. The entries of the parameters $\boldsymbol{\Theta}$ and the input matrix $\mathbf{Z}$ are initialized at random using an iid zero-mean Gaussian distributions, and the weights learned after a few iterations of denoising the observation $\mathbf{x}$ are denoted as $\hat{\boldsymbol{\Theta}}(\mathbf{x})$. Note that $\mathbf{Z}$ is fixed to its random initialization. Finally, the denoised graph signal estimate is computed as

$$\hat{\mathbf{x}}_0 = f_{\hat{\boldsymbol{\Theta}}(\mathbf{x})}(\mathbf{Z}|\mathcal{G}). \tag{8}$$

The intuition behind this approach is as follows: since the architecture is overparametrized it can in principle fit any signal, including noise. However, as shown formally later, both empirically and theoretically, the proposed architectures fit graph signals faster than the noise, and therefore with early stopping they fit most of the signal and little of the noise, enabling signal denoising.

Remark 1: The proposed architectures are described as *untrained* NNs because, when minimizing (7), the weights in $\boldsymbol{\Theta}$ are learned to fit *each observation* $\mathbf{x}$, with the denoised signal $\hat{\mathbf{x}}_0$ being the output for those particular weights. This implies that each noisy-denoised signal pair $(\mathbf{x}, \hat{\mathbf{x}}_0)$ is associated with a particular value of the weights $\boldsymbol{\Theta}$, in contrasts with trainable NNs, where the weights $\boldsymbol{\Theta}$ are first learned by fitting the signals in a *training set* and later used (unchanged) to denoise signals that were not in the training set.

Regarding the specific implementation of the untrained network $f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G})$, there are multiple possibilities for selecting the linear and nonlinear transformations $\mathcal{T}^{(\ell)}_{\boldsymbol{\Theta}^{(\ell)}}$ and $g^{(\ell)}$ defined in (3) and (4), respectively. As is customary in NNs dealing with signals defined in $\mathbb{R}^N$, we select the ReLU operator, defined as $\text{ReLU}(x) = \max(0, x)$, to be the entrywise nonlinearity $g^{(\ell)}$. Then, we focus on the design of the linear transformation, which is responsible for incorporating the structure of the graph. The two following sections postulate the implementation of two particular linear transformations $\mathcal{T}^{(\ell)}_{\boldsymbol{\Theta}^{(\ell)}}$ (each giving rise to a different GNN) and analyze the resulting architectures.

IV. Graph Convolutional Generator

Our first architecture to address the graph-signal denoising task is a graph-convolutional generator (GCG) network that incorporates the topology of the graph into the NN pipeline via vertex-based graph convolutions. Then, leveraging the fact that convolutions of a graph signal on the vertex domain can be represented by a graph filter $\mathbf{H} \in \mathbb{R}^{N\times N}$ [26], we define the linear transformation for the convolutional generator as

$$\mathcal{T}^{(\ell)}_{\mathbf{\Theta}^{(\ell)}}\{\mathbf{Y}^{(\ell-1)}|\mathcal{G}\} = \mathbf{H}\mathbf{Y}^{(\ell-1)}\mathbf{\Theta}^{(\ell)}. \tag{9}$$

Remember that the $F^{(\ell-1)} \times F^{(\ell)}$ matrix $\mathbf{\Theta}^{(\ell)}$ collects the learnable weights of the ℓ-th layer, and the graph filter $\mathbf{H}$ is given by (1). The coefficients $\{h_m\}_{m=0}^{M-1}$ are fixed a priori so that $\mathbf{H}$ promotes desired properties on the estimated signal. Using the linear transformation defined in (9), the output of the GCG with L layers is given by the recursion

$$\mathbf{Y}^{(\ell)} = \text{ReLU}\big(\mathbf{H}\mathbf{Y}^{(\ell-1)}\mathbf{\Theta}^{(\ell)}\big), \quad \text{for } \ell = 1, \ldots, L-1, \tag{10}$$

$$\mathbf{y}^{(L)} = \mathbf{H}\mathbf{Y}^{(L-1)}\mathbf{\Theta}^{(L)}, \tag{11}$$

where $\mathbf{Y}^{(0)} = \mathbf{Z}$ denotes the random input and the ReLU is not applied in the last layer of the architecture. With the proposed linear transformation, the GCG learns to combine the features within each node by fitting the weights of the matrices $\mathbf{\Theta}^{(\ell)}$ while the graph filter $\mathbf{H}$ interpolates the signal by mixing features from $M-1$ neighborhoods.

Even though the proposed GCG exploits graph convolutions to incorporate the graph topology into the architecture, it is intrinsically different from other GCNNs. The linear transformation proposed in [12], arguably one of the most popular implementations of GCNNs, is given by

$$\mathcal{T}^{(\ell)}_{\mathbf{\Theta}^{(\ell)}}\{\mathbf{Y}^{(\ell-1)}|\mathcal{G}\} = (\mathbf{A}+\mathbf{I})\mathbf{Y}^{(\ell-1)}\mathbf{\Theta}^{(\ell)}. \tag{12}$$

Recalling the definition of graph filters in (1), it is evident that (12) is a particular case of our proposed linear transformation, obtained by setting the generative graph filter to $\mathbf{H} = \mathbf{A} + \mathbf{I}$, a low-pass graph filter of degree one. In addition to representing a more general scenario, (10) endows the GCG with two main advantages. First, the graph filter $\mathbf{H}$ allows us to incorporate prior information on the signals to denoise, making our GCG architecture more suitable to denoise a (high-) low-frequency signal by employing a (high-) low-pass filter. Second, in (12) there is an equivalence between the depth of the network and the radius of the considered neighborhood, so that gathering information from nodes that are M hops apart requires a GNN with M layers. In contrast, with the architecture considered in (10), the same can be achieved by considering a GCG with L layers and a graph filter $\mathbf{H}$ of degree M/L [26], reducing the number of learnable parameters and bypassing some of the well-known over-smoothing problems associated with (12) [47].

Next, we adopt some simplifying assumptions to provide theoretical guarantees on the denoising capability of the GCG (Section IV-A). Then, we rely on numerical tests to demonstrate that the results also hold in more general settings (Section IV-B).

A. Guaranteed Denoising With the GCG

To formally prove that the proposed architecture can successfully denoise the observed graph signal $\mathbf{x}$, we consider a two-layer GCG given by

$$f_{\mathbf{\Theta}}(\mathbf{Z}|\mathcal{G}) = \text{ReLU}\big(\mathbf{H}\mathbf{Z}\mathbf{\Theta}^{(1)}\big)\boldsymbol{\theta}^{(2)}, \tag{13}$$

where $\mathbf{\Theta}^{(1)} \in \mathbb{R}^{F\times F}$ and $\boldsymbol{\theta}^{(2)} \in \mathbb{R}^{F}$ are the learnable coefficients. With F denoting the number of features, we consider the overparametrized regime where $F \geq 2N$, and analyze the behavior and performance of denoising with the untrained network defined in (13).

We start by noting that scaling the i-th entry of $\boldsymbol{\theta}^{(2)}$ is equivalent to scaling the i-th column of $\mathbf{\Theta}^{(1)}$, so that, without loss of generality, we can set the weights to $\boldsymbol{\theta}^{(2)} = \mathbf{b}$, where $\mathbf{b}$ is a vector of size F with half of its entries set to $1/\sqrt{F}$ and the other half to $-1/\sqrt{F}$. Furthermore, since $\mathbf{Z}$ is a random matrix of dimension $N \times F$, the column space of $\mathbf{Z}$ spans $\mathbb{R}^N$, and hence, minimizing over $\mathbf{Z}\mathbf{\Theta}^{(1)}$ is equivalent to minimizing over $\mathbf{\Theta} \in \mathbb{R}^{N\times F}$. With these considerations in place, the optimization over (7) can be performed replacing the two-layer GCG described in (13) by its simplified form

$$f_{\mathbf{\Theta}}(\mathbf{H}) = f_{\mathbf{\Theta}}(\mathbf{Z}|\mathcal{G}) = \text{ReLU}(\mathbf{H}\mathbf{\Theta})\mathbf{b}. \tag{14}$$

Note that we replaced $f_{\mathbf{\Theta}}(\mathbf{Z}|\mathcal{G})$ with $f_{\mathbf{\Theta}}(\mathbf{H})$ since the graph influence is modeled by the graph filter $\mathbf{H}$, and the influence of the matrix $\mathbf{Z}$ is absorbed by the learnable weights $\mathbf{\Theta}$. Also note that the behavior of the optimization algorithm of (13) and (14) may differ and the upcoming theoretical analysis is focused on the latter case.

The denoising capability of the two-layer architecture is related to the eigendecomposition of its expected squared Jacobian [33]. However, to understand which signals can be effectively denoised with the proposed architecture, we need to connect the spectral domain of the expected squared Jacobian with the spectrum of the graph, given by the eigenvectors of the adjacency matrix.

To that end, we next compute the expected squared Jacobian of the two-layer architecture in (14). Denote as $\mathcal{J}_{\mathbf{\Theta}}(\mathbf{H}) \in \mathbb{R}^{N\times NF}$ the Jacobian matrix of $f_{\mathbf{\Theta}}(\mathbf{H})$ with respect to $\mathbf{\Theta}$, which is given by

$$\mathcal{J}^{\top}_{\mathbf{\Theta}}(\mathbf{H}) = \begin{bmatrix} \mathrm{b}_1\mathbf{H}^{\top}\text{diag}(\text{ReLU}'(\mathbf{H}\boldsymbol{\theta}_1)) \\ \vdots \\ \mathrm{b}_F\mathbf{H}^{\top}\text{diag}(\text{ReLU}'(\mathbf{H}\boldsymbol{\theta}_F)) \end{bmatrix} \in \mathbb{R}^{NF\times N}, \tag{15}$$

where $\boldsymbol{\theta}_i$ represents the i-th column of $\mathbf{\Theta}$, and ReLU$'$ is the derivative of the ReLU, which is the Heaviside step function. Then, define the $N \times N$ expected squared Jacobian matrix as

$$\begin{aligned}\boldsymbol{\mathcal{X}} &:= \mathbb{E}_{\mathbf{\Theta}}\big[\mathcal{J}_{\mathbf{\Theta}}(\mathbf{H})\mathcal{J}^{\top}_{\mathbf{\Theta}}(\mathbf{H})\big] \\ &= \sum_{i=1}^{F} b_i^2\mathbb{E}\left[\text{ReLU}'(\mathbf{H}\boldsymbol{\theta}_i)\text{ReLU}'(\mathbf{H}\boldsymbol{\theta}_i)^{\top}\right] \odot \mathbf{H}\mathbf{H}^{\top}. \end{aligned} \tag{16}$$

Moreover, from the work in [48, Sec. 3.2], we note that $\mathbb{E}[\text{ReLU}'(\mathbf{H}\boldsymbol{\theta}_i)\text{ReLU}'(\mathbf{H}\boldsymbol{\theta}_i)^{\top}]$ is in fact the so-called dual activation of the step function. Therefore, combining the expression

for the dual activation of the step function from [48, Table 1] with (16), we obtain that

$$\boldsymbol{\mathcal{X}} = 0.5\left(\mathbf{1}\mathbf{1}^\top - \pi^{-1}\arccos(\mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1})\right) \odot \mathbf{H}\mathbf{H}^\top, \quad (17)$$

where $\odot$ represents the Hadamard (entry-wise) product, $\arccos(\cdot)$ is computed entry-wise, $\mathbf{h}_i$ represents the i-th column (row) of $\mathbf{H}$, $\mathbf{C} = \text{diag}([\|\mathbf{h}_1\|_2, \ldots, \|\mathbf{h}_N\|_2])$ is a normalization term so that $\mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1}$ is the autocorrelation of the graph filter $\mathbf{H}$.

Since $\boldsymbol{\mathcal{X}}$ is symmetric and positive (semi) definite, it has an eigendecomposition $\boldsymbol{\mathcal{X}} = \mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top$. Here, the columns of the orthonormal matrix $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_N]$ are the N eigenvectors, and the nonnegative eigenvalues in the diagonal matrix $\boldsymbol{\Sigma}$ are assumed to be ordered as $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_N$.

After defining the two-layer GCG $f_{\boldsymbol{\Theta}}(\mathbf{H})$ and its expected square Jacobian $\boldsymbol{\mathcal{X}}$, we formally analyze its performance when denoising bandlimited graph signals. This is particularly relevant given the importance of (approximate) bandlimited graph signals both from analytical and practical points of view [5]. For the sake of clarity, we first introduce the main result (Theorem 1) and then we detail a key intermediate result (Lemma 1) that provides additional insight.

Formally, consider the K-bandlimited graph signal $\mathbf{x}_0$ as described in (2), and let the architecture $f_{\boldsymbol{\Theta}}(\mathbf{H})$ have a sufficiently large number of features F:

$$F \geq \left(\frac{\sigma_1^2}{\sigma_N^2}\right)^{26} \xi^{-8} N, \quad \text{with } \xi \in \left(0, (2\log(2\,N/\phi))^{-\frac{1}{2}}\right) \quad (18)$$

being an error tolerance parameter for some prespecified ϕ. Then, for a specific set of graphs with minimum number of nodes $N_{K,\epsilon,\delta}$ that is introduced later in the section (cf. Assumption 1), if we solve (7) running gradient descent with a step size $\eta \leq \frac{1}{\sigma_1^2}$, the following result holds (see Appendix A).

Theorem 1: Let $f_{\boldsymbol{\Theta}}(\mathbf{H})$ be the network defined in (14), and assume it is sufficiently wide, i.e., it satisfies condition (18) for some error tolerance parameter ξ. Let $\mathbf{x}_0$ be a K-bandlimited graph signal spanned by the eigenvectors $\mathbf{V}_K$, and let $\mathbf{w}_i$ and σ_i be the i-th eigenvector and eigenvalue of $\boldsymbol{\mathcal{X}}$. Let $\mathbf{n}$ be the noise present in $\mathbf{x}$, set ϕ and ϵ to small positive numbers, and let the conditions from Assumption 1 hold. Then, for any ϵ, δ, there exists some $N_{K,\epsilon,\delta}$ such that if $N > N_{K,\epsilon,\delta}$, the error for each iteration t of gradient descent with stepsize η used to fit the architecture is bounded as

$$\|\mathbf{x}_0 - f_{\boldsymbol{\Theta}_{(t)}}(\mathbf{H})\|_2 \leq \left(\left(1-\eta\sigma_K^2\right)^t + \delta\left(1-\eta\sigma_N^2\right)^t\right)\|\mathbf{x}_0\|_2$$
$$+ \xi\|\mathbf{x}\|_2 + \sqrt{\textstyle\sum_{i=1}^N \left(\left(1-\eta\sigma_i^2\right)^t - 1\right)^2 \left(\mathbf{w}_i^\top \mathbf{n}\right)^2}, \quad (19)$$

with probability at least $1 - e^{-F^2} - \phi - \epsilon$.

As explained next, the fitting (denoising) bound provided by the theorem first decreases and then increases with the number of iterations t. To be more precise, let us analyze separately each of the three terms in the right hand side of (19). The first term captures the part of the signal $\mathbf{x}_0$ that is fitted after t iterations while accounting for the misalignment of the eigenvectors $\mathbf{V}_K$ and $\mathbf{W}_K$. This term decreases with t and, since δ can be made arbitrary small (cf. Lemma 1), vanishes for moderately low values of t. The second term is an error term that is negligible if the network is sufficiently wide. Therefore, ξ can be chosen to be sufficiently small by designing the architecture according to the condition in (18). Finally, the third term, which depends on the noise present in each of the spectral components of the squared Jacobian $(\mathbf{w}_i^\top \mathbf{n})^2$, grows with t. More specifically, if the σ_i associated with a spectral component is very small, the term $(1-\eta\sigma_i^2)$ is close to 1 and, hence, the noise power in the i-th frequency will be small. Only when t grows very large the coefficient $(1-\eta\sigma_i^2)^t$ vanishes and the i-th frequency component of the noise is fitted. As a result, if the filter $\mathbf{H}$ is designed such that eigenvalues of the squared Jacobian satisfy that $\sigma_K \gg \sigma_{K+1}$, then there will be a range of moderate-to-high values of t for which: i) the first term is zero and ii) only the K strongest components of the noise have been fitted, so that the third term can be approximated as $\sqrt{\sum_{i=1}^K (\mathbf{w}_i^\top \mathbf{n})^2}$. Clearly, as t grows larger, the coefficient $((1-\eta\sigma_i^2)^t - 1)$ will also be close to one for $i > K$, meaning that additional components of the noise will be fitted as well, deteriorating the performance of the denoising architecture. This implies that if the optimization algorithm is stopped before t grows too large, the original signal is fitted along with the noise that aligns with the signal, but not the noise present in other components.

In other words, Theorem 1 not only characterizes the performance of the two-layer GNN, but also illustrates that, if early stopping is adopted, our overparametrized architecture is able to effectively denoise the bandlimited graph signal. This result is related to the error bound for denoising images presented in [33], where $\mathbf{x}_0$ is assumed to lie in the span of $\mathbf{W}_K$. However, when dealing with graphs, it is unclear which signals would satisfy this requirement. Motivated by this, we assume that $\mathbf{x}_0$ is a bandlimited signal (i.e., lies in the span of $\mathbf{V}_K$), which is a natural condition employed in many applications.

As a consequence, a critical step to attain Theorem 1 is to relate the eigenvectors of $\boldsymbol{\mathcal{X}}$ with those of the adjacency matrix $\mathbf{A}$, denoted as $\mathbf{V}$. To achieve this, we assume that $\mathbf{A}$ is random and provide high-probability bounds between the leading eigenvectors of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$. More specifically, consider a graph $\mathcal{G}$ drawn from a stochastic block model (SBM) [49] with K communities. Also, denote by $\mathcal{M}(\boldsymbol{\mathcal{A}})$ the SBM with expected adjacency matrix $\boldsymbol{\mathcal{A}} = \mathbb{E}[\mathbf{A}]$, and by $\beta_{\min}$ the minimum expected degree $\beta_{\min} := \min_i [\boldsymbol{\mathcal{A}}\mathbf{1}]_i$. Given some $\rho > 0$, we define as $\mathcal{M}_N(\beta_{\min}, \rho)$ the class of SBMs $\mathcal{M}(\boldsymbol{\mathcal{A}})$ with N nodes for which the minimum expected degree is $\beta_{\min}$ or higher. Then, the condition of $\mathcal{G}$ being drawn from this SBM whose expected minimum degree increases with N is formally expressed in the following assumption.

Assumption 1: The model $\mathcal{M}(\boldsymbol{\mathcal{A}})$ from which $\mathbf{A}$ is drawn satisfies $\mathcal{M}(\boldsymbol{\mathcal{A}}) \in \mathcal{M}_N(\beta_{\min}, \rho)$, with $\beta_{\min} = \omega(\ln(N/\rho))$.

Here, $\omega(\cdot)$ denotes the (conventional) asymptotic dominance. We note that, as discussed in [50], the minimal degree condition considered in Assumption 1 ensures that nodes belonging to the same community also belong to the same connected component with high probability, which is helpful to relate $\mathbf{A}$ and $\boldsymbol{\mathcal{A}}$. Under these conditions, the following result holds.

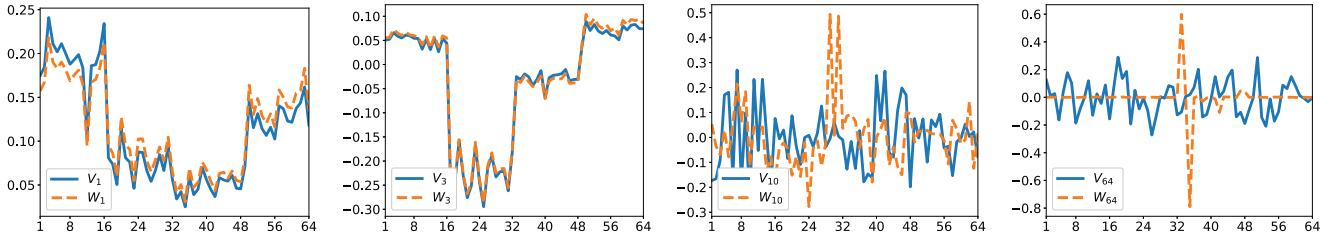


Fig. 1. Comparison between the eigenvectors of the matrices $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$ for an SBM graph with $N = 64$ nodes and $K = 4$ communities, and for a GCG of $L = 5$ layers. From left to right, the figures represent the first, third, tenth, and last eigenvectors.

Lemma 1: Let the matrix $\boldsymbol{\mathcal{X}}$ be defined as in (17), set ϵ and δ to small positive numbers, and denote by $\mathbf{V}_K$ and $\mathbf{W}_K$ the K leading eigenvectors in the respective eigendecompositions of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$. Under Assumption 1, there exists an orthonormal matrix $\mathbf{Q}$ and an integer $N_{K,\epsilon,\delta}$ such that, for $N > N_{K,\epsilon,\delta}$, the bound

$$\|\mathbf{V}_K - \mathbf{W}_K\mathbf{Q}\|_{\mathrm{F}} \leq \delta,$$

holds with probability at least $1 - \epsilon$.

The proof is provided in Appendix B, and it leverages Assumption 1 to relate the eigenvectors $\mathbf{V}_K$ and $\mathbf{W}_K$ based on the eigenvectors of the expected values of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$.

For a given K, Lemma 1 bounds the difference between the subspaces spanned by the K leading eigenvectors of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$ when graphs are big enough, a result that is key in obtaining Theorem 1. Moreover, the lemma shows that if the lower bound $N_{K,\epsilon,\delta}$ increases, then the error encoded δ becomes arbitrary small. Also note that, if a larger value of K is considered, then the minimum required graph size $N_{K,\epsilon,\delta}$ will also be larger. An inspection of (17) reveals that the result in Lemma 1 is not entirely unexpected. Indeed, since $\mathbf{H}$ is a polynomial in $\mathbf{A}$, so is $\mathbf{H}^2$. This implies that $\mathbf{V}$ are also the eigenvectors of $\mathbf{H}^2$, and because $\mathbf{H}^2$ appears twice on the right hand side of (17), a relationship between the eigenvectors of $\boldsymbol{\mathcal{X}}$ and $\mathbf{V}$ can be anticipated. However, the presence of the Hadamard product and the (non Lipschitz continuous) nonlinearity $\arccos$ renders the exact analysis of the eigenvectors a challenging task. Consequently, we resorted to a stochastic framework in deriving Lemma 1.

B. Numerical Inspection of the Deep GCG Spectrum

While for convenience, the previous section focused on analyzing the GCG architecture with $L = 2$ layers, in practice we often work with a larger number of layers. In this section, we provide numerical evidence showing that the relation between matrices $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$ described in Lemma 1 also holds when $L > 2$.

To that end, Fig. 1 shows the pairs of eigenvectors $\mathbf{v}_i$ and $\mathbf{w}_i$ for the indexes $i = \{1, 3, 10, 64\}$, for a given graph $\mathcal{G}$ drawn from an SBM with $N = 64$ nodes and 4 communities. The GCG is composed of $L = 5$ layers and, to obtain the eigenvectors of the squared Jacobian matrix, the Jacobian is computed using the *autograd* functionality of PyTorch. The nodes of the graph are sorted by communities, i.e., the first N_1 nodes belong to the first community and so on. It can be clearly seen that, even for moderately small graphs, the leading eigenvectors of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$ are almost identical, becoming more dissimilar as the eigenvectors are associated with smaller eigenvalues. It can also be observed how leading eigenvectors have similar values for entries associated with nodes within the same community. Moreover, Fig. 2 depicts the matrix product $\mathbf{V}^\top\mathbf{W}$, where it is observed that the $K = 4$ leading eigenvectors of both matrices are orthonormal. The presented numerical results strengthen the argument that the analytical results obtained for the two-layer case can be extrapolated to deeper architectures.

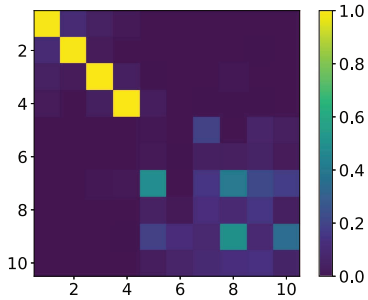


Fig. 2. Heatmap representation of the matrix product $\mathbf{V}_K^\top\mathbf{W}_K$. The low values of the off-diagonal entries illustrate the orthogonality between both sets of eigenvectors. These eigenvectors are the same as those depicted in Fig. 1.

Another key assumption of Lemma 1 is that $\mathcal{G}$ is drawn from the SBM described in $\mathcal{M}_N(\beta_{\min}, \rho)$. This assumption facilitates the derivation of a bound relating the spectra of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$ (i.e., the subspaces spanned by the eigenvectors $\mathbf{V}_K$ and $\mathbf{W}_K$). However, the results reported in Fig. 3 suggest that such a relation exists for other type of graphs, even though its analytical characterization is more challenging. The figure has 12 panels (3 rows and 4 columns). Each of the rows corresponds to a different graph, namely: 1) a realization of a small-world (SW) graph [51] with $N = 150$ nodes, 2) the Zachary's Karate graph [52] with $N = 34$ nodes, and 3) a graph of $N = 316$ weather stations across the United States [53]. Each of the three first columns correspond to an $N \times N$ matrix, namely: 1) the normalized adjacency matrix $\mathbf{A}$, 2) $\mathbf{H}^2$, the squared version of a low pass graph filter and whose coefficients are drawn from a uniform distribution and set to unit ℓ_1 norm, and 3) the squared Jacobian matrix $\boldsymbol{\mathcal{X}}$. Although we may observe some similarity between $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$, the relation between $\boldsymbol{\mathcal{X}}$ and the graph $\mathcal{G}$ becomes apparent when comparing the matrices $\mathbf{H}^2$ and $\boldsymbol{\mathcal{X}}$. The matrix $\mathbf{H}$ is a random graph filter used in the linear transformation of the convolutional generator

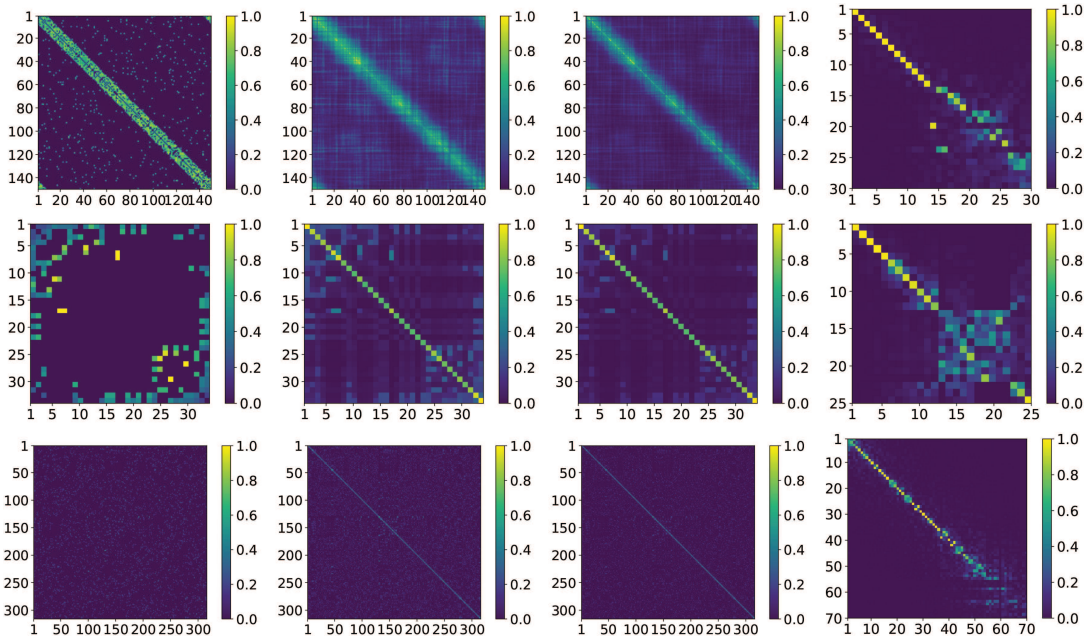


Fig. 3. Illustration of matrices $\mathbf{A}$, $\mathbf{H}^2$, $\boldsymbol{\mathcal{X}}$, and $\mathbf{V}_K^\top \mathbf{W}_K$, shown in columns 1, 2, 3, and 4, respectively, for different types of graphs. The rows 1, 2, and 3 correspond to a small world, the Zachary's Karate, and the weather stations graph. The graph filter $\mathbf{H}^2$ is created as a square graph filter with coefficients drawn from a uniform distribution and with unitary ℓ_1 norm. For each graph (rows), it can be seen that the matrices $\mathbf{A}$, $\mathbf{H}^2$, and $\boldsymbol{\mathcal{X}}$ are related, and that $\mathbf{V}_K$ and $\mathbf{W}_K$ are close to orthogonal.

$f_{\boldsymbol{\Theta}}(\mathbf{H})$, and it is clear that the vertex connectivity pattern of $\boldsymbol{\mathcal{X}}$ is related to that of $\mathbf{H}^2$. Since $\boldsymbol{\mathcal{X}}$ and $\mathbf{H}^2$ are closely related and we know that the eigenvectors of $\mathbf{H}^2$ and those of $\mathbf{A}$ are the same, we expect $\mathbf{W}$ (the eigenvectors of $\boldsymbol{\mathcal{X}}$) and $\mathbf{V}$ (the eigenvectors of $\mathbf{A}$) to be related as well. To verify this, the fourth column of Fig. 3 represents $\mathbf{V}_K^\top \mathbf{W}_K$, i.e., the pairwise inner products of the K leading eigenvectors of $\mathbf{A}$ and those of $\boldsymbol{\mathcal{X}}$. It can be observed that the K leading eigenvectors are close to orthogonal, which means that the relation observed in the vertex domain carries over to the spectral domain and $\mathbf{V}_K$ and $\mathbf{W}_K$ expand the same subspace. These results suggest that a deep GCG could be able to denoise signals living in the subspace spanned by $\mathbf{V}_K$. However, because the bound in Theorem 1 assumed a 2-layer GCG, we address this hypothesis numerically in Section VI.

To summarize, the presented results illustrate that the analytical characterization provided in Section IV-A, which considered a 2-layer GCG operating over SBM graphs, carries over to more general setups.

V. Graph Upsampling Decoder

The GCG architecture presented in Section IV incorporated the topology of $\mathcal{G}$ via the vertex-based convolutions implemented by the graph filter $\mathbf{H}$. In this section, we introduce the graph decoder (GDec) architecture. In contrast to the GCG and other GCNNs, this novel graph-aware denoising NN incorporates the topology of $\mathcal{G}$ via a (nested) collection of graph upsampling operators [31]. Specifically, we propose the linear transformation for the GDec denoiser to be given by

$$\mathcal{T}^{(\ell)}_{\boldsymbol{\Theta}^{(\ell)}}\{\mathbf{Y}^{(\ell-1)}|\mathcal{G}\} = \mathbf{U}^{(\ell)}\mathbf{Y}^{(\ell-1)}\boldsymbol{\Theta}^{(\ell)}, \tag{20}$$

where $\mathbf{U}^{(\ell)} \in \mathbb{R}^{N^{(\ell)} \times N^{(\ell-1)}}$, with $N^{(\ell)} \geq N^{(\ell-1)}$, are graph upsampling matrices to be defined soon. Note that, compared to (9), the graph filter $\mathbf{H}$ is replaced with the upsampling operator $\mathbf{U}^{(\ell)}$ that *depends* on ℓ. Adopting the proposed linear transformation, the output of the GDec with L layers is given by the recursion

$$\mathbf{Y}^{(\ell)} = \text{ReLU}\big(\mathbf{U}^{(\ell)}\mathbf{Y}^{(\ell-1)}\boldsymbol{\Theta}^{(\ell)}\big), \quad \text{for } \ell = 1, \ldots, L-1, \tag{21}$$

$$\mathbf{y}^{(L)} = \mathbf{U}^{(L)}\mathbf{Y}^{(L-1)}\boldsymbol{\Theta}^{(L)}, \tag{22}$$

where the ReLU is also removed from the last layer.

Similar to the GCG, the proposed GDec learns to combine the features within each node. However, the interpolation of the signals in this case is determined by the graph upsampling operators $\{\mathbf{U}^{(\ell)}\}_{\ell=1}^{L}$, rather than by employing convolutions. The size of the input $N^{(0)}$ is now a design parameter that will determine the implicit degrees of freedom of the architecture. Note that, from the GSP perspective, the input feature matrix $\mathbf{Y}^{(\ell-1)} \in \mathbb{R}^{N^{(\ell-1)} \times F^{(\ell-1)}}$ represents $F^{(\ell-1)}$ graph signals, each of them defined over a graph $\mathcal{G}^{(\ell-1)}$ with $N^{(\ell-1)}$ nodes. Therefore, even though the input $\mathbf{Y}^{(0)} = \mathbf{Z}$ is still a random white matrix across rows and columns, since $N^{(\ell)} \geq N^{(\ell-1)}$, the dimensionality of the input is progressively increasing.

A closer comparison with the GCG reveals that the smaller dimensionality of the input $\mathbf{Z}$ endows the GDec architecture with fewer degrees of freedom, rendering the architecture more robust to noise. Not only that, but the graph information is now included via the graph upsampling operators $\mathbf{U}^{(\ell)}$ instead of relying on graph filters. Clearly, the method used to design the graph upsampling matrices, which is the subject of the next section, will have an impact on the type of graph signals that can be efficiently denoised using the GDec architecture.

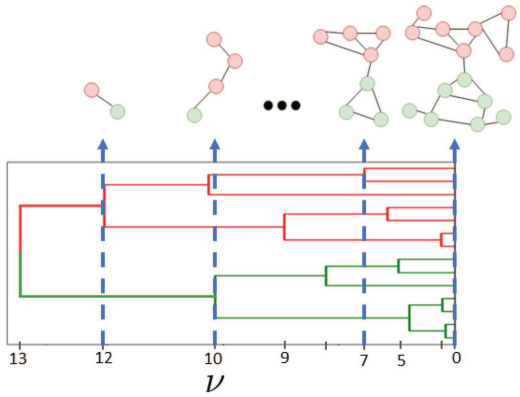


Fig. 4. Dendrogram of an agglomerative hierarchical clustering algorithm and the resulting graphs with 2, 4, 7 and 14 nodes.

A. Graph Upsampling Operator From Hierarchical Clustering

Regular upsampling operators have been successfully used in NN architectures to denoise signals defined on regular domains [33]. While the design of upsampling operators in regular grids is straightforward, when the signals are defined on irregular domains the problem becomes substantially more challenging. The approach that we put forth in this paper is to use agglomerative hierarchical clustering methods [27], [28], [29] to design a graph upsampling operator that leverages the graph topology. These methods take a graph as an input and return a dendrogram; see Fig. 4. A dendrogram can be interpreted as a rooted-tree structure that shows different clusters at the different levels of resolution ν. At the finest resolution ($\nu = 0$) each node forms a cluster of its own. Then, as ν increases, nodes start to group together (agglomerate) in bigger clusters and, when the resolution becomes large (coarse) enough, all nodes end up being grouped in the same cluster.

By cutting the dendrogram at $L+1$ resolutions, including $\nu = 0$, we obtain a collection of node sets with parent-child relationships inherited by the refinement of clusters. Since we are interested in performing graph upsampling, note that the dendrogram is interpreted from left to right. This can be observed in the example shown in Fig. 4, where the three red nodes in the second graph ($\nu = 10$, layer $\ell = 1$) are children of the red parent in the coarsest graph ($\nu = 12$, layer $\ell = 0$). In this sense, the graph upsampling operator is given by the inverse operation of the clustering algorithm. We leverage these parent-children relations to define the membership matrices $\mathbf{P}^{(\ell)} \in \{0,1\}^{N^{(\ell)} \times N^{(\ell-1)}}$, where the entry $P_{ij}^{(\ell)} = 1$ only if the i-th node in layer ℓ is the child of the j-th node in layer $\ell - 1$. Moreover, we can further exploit the dendrogram to obtain coarser-resolution versions of the original graph $\mathcal{G}$. To that end, note that the clusters at layer ℓ can be interpreted as nodes of a graph $\mathcal{G}^{(\ell)}$ with $N^{(\ell)}$ nodes and adjacency matrix $\mathbf{A}^{(\ell)}$. There are several ways of defining $\mathbf{A}^{(\ell)}$ based on the original adjacency matrix $\mathbf{A}$. While our architecture does not focus on a particular form, in the simulations we set $A_{ij}^{(\ell)} \neq 0$ only if, in the original graph $\mathcal{G}$, there is at least one edge between nodes belonging to the cluster i and nodes from cluster j. In addition, the weight of the edge depends on the number of existing edges between the two clusters.

With the definition of the membership matrix $\mathbf{P}^{(\ell)}$ and the adjacency matrix $\mathbf{A}^{(\ell)}$, the upsampling operator of the ℓ-th layer is given by

$$\mathbf{U}^{(\ell)} = \left(\gamma \mathbf{I} + (1-\gamma)\, \mathbf{A}^{(\ell)}\right) \mathbf{P}^{(\ell)}, \tag{23}$$

where $\gamma \in [0,1]$ is a pre-specified constant. Notice that $\mathbf{U}^{(\ell)}$ first copies the signal value from the parents to the children by applying the matrix $\mathbf{P}^{(\ell)}$, and then every child performs a convex combination between this value and the average signal value of its neighbors. This design promotes that nodes descending from the same parent have similar (related) values, which conveys a notion (prior) of smoothness on the targeted graph signals. As we show in Section VI, the implicit smoothness prior results in a better performance when denoising smooth signals but, on the other hand, makes the architecture more sensitive to model mismatch. Therefore, when dealing with high-frequency signals, a worth-looking approach left as a future research direction is to rely on algorithms that cluster the nodes considering not only the topology of $\mathcal{G}$ but also the properties of the graph signals.

Because the membership matrices $\mathbf{P}^{(\ell)}$ are designed using a clustering algorithm over $\mathcal{G}$, and the matrices $\mathbf{A}^{(\ell)}$ capture how strongly connected the clusters of layer ℓ are in the original graph, these two matrices are responsible for incorporating the information of $\mathcal{G}$ into the upsampling operators $\mathbf{U}^{(\ell)}$. Furthermore, we remark that the upsampling operator $\mathbf{U}^{(\ell)}$ can be reinterpreted as the application of $\mathbf{P}^{(\ell)}$ followed by the application of a graph filter

$$\tilde{\mathbf{H}}^{(\ell)} = \gamma \mathbf{I} + (1-\gamma)\mathbf{A}^{(\ell)}, \tag{24}$$

which sets the filter coefficients as $h_0 = \gamma$ and $h_1 = 1 - \gamma$.

B. Guaranteed Denoising With the GDec

As we did for the GCG, our goal is to theoretically characterize the denoising performance of the GNN architecture defined by (21)–(23). To achieve that goal, we replicate the approach implemented in Section IV-A. We first derive the matrix $\boldsymbol{\mathcal{X}}$ and provide theoretical guarantees when denoising a K-bandlimited graph signal with the GDec. Then, to gain additional insight, we detail the relation between the subspace spanned by the eigenvectors $\mathbf{W}$ and the spectral domain of $\mathbf{A}$. This relation is key in deriving the theoretical analysis.

We start by introducing the 2-layer GDec

$$f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G}) = \mathrm{ReLU}\big(\mathbf{U}\mathbf{Z}\boldsymbol{\Theta}^{(1)}\big)\boldsymbol{\theta}^{(2)}. \tag{25}$$

Then, following a similar reasoning to that provided after (14), instead of employing the architecture in (25) we can optimize (7) over its simplifying version

$$f_{\boldsymbol{\Theta}}(\mathbf{U}) = f_{\boldsymbol{\Theta}}(\mathbf{Z}|\mathcal{G}) = \mathrm{ReLU}\big(\mathbf{U}\boldsymbol{\Theta}\big)\mathbf{b}. \tag{26}$$

An important difference with respect to the GCG presented in (14) is that the matrix $\boldsymbol{\Theta}$ has a dimension of $N^{(0)} \times F$, so it spans $\mathbb{R}^{N^{(0)}}$ instead of $\mathbb{R}^{N}$. Since $N^{(0)} < N$, the smaller subspace spanned by the weights of the GDec renders the architecture more robust to fitting noise, but, on the other hand, the number of degrees of freedom to learn the graph signal of interest are

reduced. As a result, the alignment between the targeted graph signals and the low-pass vertex-clustering architecture becomes more important.

The expected squared Jacobian $\boldsymbol{\mathcal{X}} = \mathbb{E}_{\boldsymbol{\Theta}}[\mathcal{J}_{\boldsymbol{\Theta}}(\mathbf{U})\mathcal{J}_{\boldsymbol{\Theta}}^{\top}(\mathbf{U})]$ is obtained following the procedure used to derive (17), arriving at the expression

$$\boldsymbol{\mathcal{X}} = 0.5\left(\mathbf{1}\mathbf{1}^{\top} - \frac{1}{\pi}\arccos\left(\tilde{\mathbf{C}}^{-1}\mathbf{U}\mathbf{U}^{\top}\tilde{\mathbf{C}}^{-1}\right)\right) \odot \mathbf{U}\mathbf{U}^{\top}, \quad (27)$$

where $\mathbf{u}_i$ represents the i-th row of $\mathbf{U}$, and $\tilde{\mathbf{C}} = \text{diag}([\|\mathbf{u}_1\|_2, \ldots, \|\mathbf{u}_N\|_2])$ is a normalization matrix.

Then, let $\mathbf{x}_0$ be a K-bandlimited graph signal and let $f_{\boldsymbol{\Theta}}(\mathbf{U})$ have a number of features F satisfying (18). If we solve (7) running gradient descent with a step size $\eta \leq \frac{1}{\sigma_1^2}$, the following result holds.

Theorem 2: Let $f_{\boldsymbol{\Theta}}(\mathbf{U})$ be the network defined in (26). Consider the conditions described in Theorem 1 and let $N^{(0)}$ match the number of communities K (see Assumption 1). Then, for any ϵ, δ, there exists some $N_{K,\epsilon,\delta}$ such that if $N > N_{K,\epsilon,\delta}$, then the error for each iteration t of gradient descent with stepsize η used to fit the architecture is bounded as (19), with probability at least $1 - e^{-F^2} - \phi - \epsilon$.

The proof is analogous to the one provided in Appendix A but exploiting Lemma 2 instead of Lemma 1. Lemma 2 is fundamental in attaining Theorem 2 and is presented later in the section.

Theorem 2 formally establishes the denoising capability of the GDec when $\mathbf{x}_0$ is a K-bandlimited graph signal and $K = N^{(0)}$ matches the number of communities in the SBM graph. When compared with the GCG, the smaller dimensionality of the input $\mathbf{Z}$, and thus the smaller rank of the matrix $\boldsymbol{\Theta}$, constrains the learning capacity of the architecture, making it more robust to the presence of noise. However, this additional robustness also implies that the architecture is more sensitive to model mismatch, since its capacity to learn arbitrary signals is smaller. Intuitively, the GDec represents an architecture tailored for a more specific family of graph signals than the GCG. Moreover, employing the GDec instead of the GCG has a significant impact on the relation between the subspaces spanned by $\mathbf{V}_K$ and $\mathbf{W}_K$.

To establish the new relation between $\mathbf{V}_K$ and $\mathbf{W}_K$, assume that the adjacency matrix is drawn from an SBM $\mathcal{M}(\boldsymbol{\mathcal{A}})$ with K communities such that $\mathcal{M}(\boldsymbol{\mathcal{A}}) \in \mathcal{M}_N(\beta_{\min}, \rho)$, so that the SBM follows Assumption 1. In addition, set the size of the latent space to the number of communities so $N^{(0)} = K$. Under this setting, the counterpart to Lemma 1 for the case where $f_{\boldsymbol{\Theta}}(\mathbf{U})$ is a GDec architecture follows.

Lemma 2: Let the matrix $\boldsymbol{\mathcal{X}}$ be defined as in (27), set ϵ and δ to small positive numbers, and denote by $\mathbf{V}_K$ and $\mathbf{W}_K$ the K leading eigenvectors in the respective eigendecompositions of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$. Under Assumption 1, there exist an orthonormal matrix $\mathbf{Q}$ and an integer $N_{K,\epsilon,\delta}$ such that for $N > N_{K,\epsilon,\delta}$ the bound

$$\|\mathbf{V}_K - \mathbf{W}_K\mathbf{Q}\|_{\text{F}} \leq \delta,$$

holds with probability at least $1 - \epsilon$.

Lemma 2 asserts that the difference between the subspaces spanned by $\mathbf{V}_K$ and $\mathbf{W}_K$ becomes arbitrarily small as the size of the graph increases. The proof is provided in Appendix C and the intuition behind it arises from the fact that the upsampling operator can be understood as $\mathbf{U} = \tilde{\mathbf{H}}\mathbf{P}$, where $\tilde{\mathbf{H}}$ is a graph filter of the specific form described in (24). Remember that $\mathbf{P}$ is a binary matrix encoding the cluster in the layer $\ell - 1$ to which the nodes in the layer ℓ belong. Since we are only considering two layers, and we have that $N^{(0)} = K$, the matrix $\mathbf{P}$ is encoding the node-community membership of the SBM graph and, hence, the product $\mathbf{P}\mathbf{P}^{\top}$ is a block matrix with constant entries matching the block pattern of $\boldsymbol{\mathcal{A}}$. As shown in the proof, this property can be leveraged to bound the eigendecomposition of $\mathbf{A}$ and $\boldsymbol{\mathcal{X}}$.

C. Analyzing the Deep GDec

The deep GDec composed of $L > 2$ layers can be constructed following the recursion presented in (21) and (22). In this case, by stacking more layers we perform the upsampling of the input signal in a progressive manner and, at the same time, we add more nonlinearities, which helps alleviating the rank constraint related to the input size $N^{(0)}$. In the absence of nonlinear functions, the maximum rank of the weights would be $N^{(0)}$, and thus, only signals in a subspace of size $N^{(0)}$ could be learned. By properly selecting the number of layers and the input size when constructing the network, we can obtain a trade-off between the robustness of the architecture and its learning capability.

In addition, the effect of adding more layers is also reflected on the smoothness assumption inherited from the construction of the upsampling operator. Adding more layers is related to less smooth signals, since the number of nodes in $\mathcal{G}$ with a common parent, and thus, with similar values, is smaller.

We note that numerically illustrating that the bound between $\mathbf{V}_K$ and $\mathbf{W}_K$ holds true for the deep GDec, and that its denoising capability is not limited to signals defined over SBM graphs provide results similar to those in Section IV-B. Therefore, instead of replicating the previous section, we directly illustrate the performance of the deep GDec under more general settings in the following section, where we present the numerical evaluation of the proposed architectures.

VI. Numerical Results

This section presents different experiments to numerically validate the theoretical claims introduced in the paper, and to illustrate the denoising performance of the GCG and the GDec. The experiments are carried out using synthetic and real-world data, and the proposed architectures are compared to other graph-signal denoising alternatives. The code for the experiments and the architectures is available on GitHub[2]. For hyper-parameter settings and implementation details the interested reader is referred to the online available code.

[2]https://github.com/reysam93/Graph_Deep_Decoder

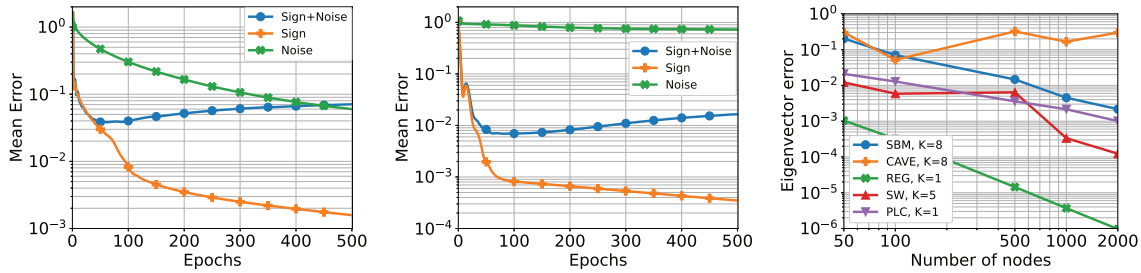


Fig. 5. (a) Error of the 2-layer GCG when fitting a piece-wise constant signal, noise, and a noisy signal, as a function of the number of epochs. The graph is drawn from an SBM with 64 nodes and 4 communities, and the normalized noise power is $P_n = 0.1$. (b) Counterpart of (a) but for the 2-layer GDec architecture. (c) Mean distance between the K leading eigenvectors of the adjacency matrix and $\boldsymbol{\mathcal{X}}$ as a function of the graph size for several graph models.

A. Denoising Capability of Graph Untrained Architectures

The goal of the experiment shown in Fig. 5(a) and (b) is to illustrate that the proposed graph untrained architectures are capable of learning the structured original signal $\mathbf{x}_0$ faster than the noise, which is one of the core claims of the paper. To that end, we generate an SBM graph with $N = 64$ nodes and $K = 4$ communities, and define 3 different signals: (i) "Signal": a piece-wise constant signal $\mathbf{x}_0$ with the value of each node being the label of its community; (ii) "Noise": zero-mean white Gaussian noise $\mathbf{n}$ with unit variance; and (iii) "Signal + Noise": a noisy observation $\mathbf{x} = \mathbf{x}_0 + \mathbf{n}$ where the noise has a normalized power of 0.1. Fig. 5(a) and (b) show the normalized mean squared error (NMSE), with the error for each realization being $\|\mathbf{x}_0 - \hat{\mathbf{x}}_0\|_2^2/\|\mathbf{x}_0\|_2^2$. The mean is computed for 100 realizations of the noise as the number of epochs increases when the different signals are fitted by the 2-layer GCG and the 2-layer GDec, respectively. It can be seen how, in both cases, the error when fitting the noisy signal $\mathbf{x}$ decreases for a few epochs until it reaches a minimum, and then starts to increase. This is because the proposed untrained architectures learn the signal $\mathbf{x}_0$ faster than the noise, but if they fit the observation for too many epochs, they start learning the noise as well and, hence, the MSE increases. As stated by Theorems 1 and 2, this result illustrates that, if early stopping is applied, both architectures are capable of denoising the observed graph signals without a training step. It can also be noted that, under this setting, the GDec learns the signal $\mathbf{x}_0$ faster than the GCG and, at the same time, is more robust to the presence of noise. This can be seen as a consequence of GDec implicitly making stronger assumptions about the smoothness of the targeted signal.

The goal of the second test case is two-fold. First, it illustrates that the result presented in Lemma 1 is not constrained to the family of SBM (as specified by Assumption 1), but can be generalized to other families of random graphs as well. In addition, it measures the influence of the number of nodes in the discrepancies between $\mathbf{V}_K$ and $\mathbf{W}_K$. To that end, Fig. 5(c) contains the mean eigenvector similarity measured as $\frac{1}{K}\|\mathbf{V}_K - \mathbf{W}_K\mathbf{Q}\|_F$ as a function of the number of nodes in the graph. The eigenvector similarity is computed for 50 realizations of random graphs and the presented error is the median of all the realizations. The random graph models considered are: the SBM ("SBM"), the connected caveman graph ("CAVE") [54], the regular graph whose fixed degree increases with its size ("REG"), the small world graph ("SW") [51], and the power law cluster graph model ("PLC") [55]. The second term in the legend denotes the number of leading eigenvectors taken into account in each case, which depends on the number of active frequency components of the specific random graph model. We can clearly observe that for most of the random graph models, the eigenvector error goes to 0 as N increases and, furthermore, the error is below 10^{-1} even for moderately small graphs. This illustrates that, although the conditions assumed for Lemmas 1 and 2 focus on the specific setting of the SBM, the results can be applied to a wider class of graphs. Here, the regular graphs are particularly interesting since most classical signals may be interpreted as signals defined over regular graphs. As a result, this empirical evidence motivates the extension of the proposed theorems to more general settings as a future line of work.

B. Denoising Synthetic Data

We now proceed to comment on the denoising performance of the proposed architectures with synthetic data. The usage of synthetic signals allows us to study how the properties of the noiseless signal influence the quality of the denoised estimate.

The first experiment, shown in Fig. 6(a), studies the error of the denoised estimate obtained with the 2-layer GCG as the number of epochs increases. The reported error is the NMSE of the estimated signal $\hat{\mathbf{x}}_0$, and the figure shows the mean values of 100 realizations of graphs and graph signals. The normalized power of the noise present in the data is 0.1. Graphs are drawn from an SBM with $N = 64$ nodes and 4 communities, and the graph signals are generated as: (i) a zero-mean white Gaussian noise with unit variance ("Rand"); (ii) a bandlimited graph signal (cf. 2) using the K leading eigenvectors of $\mathbf{A}$ as base ("BL"); and (iii) a diffused white ("DW") signal created as $\mathbf{y} = \text{med}(\mathbf{H}\mathbf{w}|\mathcal{G})$, where $\mathbf{w}$ is a white vector whose entries are sampled from $\mathcal{N}(0, 1)$, $\mathbf{H}$ is a low-pass graph filter, and $\text{med}(\cdot|\mathcal{G})$ represents the graph-aware median operator such that the value of the node i is the median of its neighborhood [39], [40]. The results in Fig. 6(a) show that the best denoising error is obtained when the signal is composed of just a small number of eigenvectors, and the performance deteriorates as the bandwidth (i.e., the number of eigenvectors that span the signal subspace)

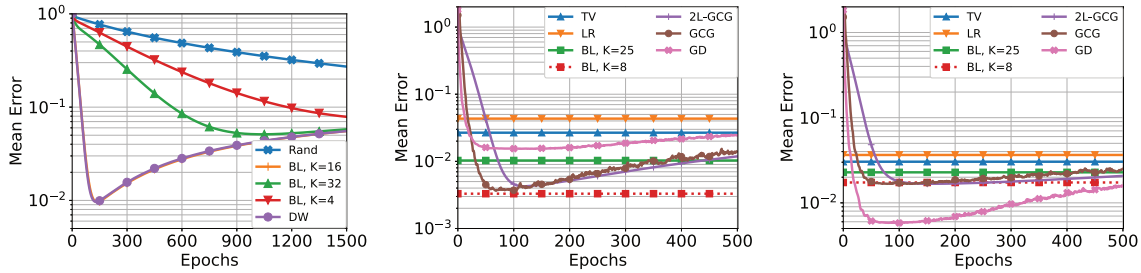


Fig. 6. Median MSE when denoising a graph signal as a function of the number of epochs. (a) The 2-layer GCG is used to denoise different families of signals. (b) Performance comparison between total variation, Laplacian regularization, bandlimited models, the 2-layer GCG, the deep GCG, and the deep GDec, when the signals are bandlimited. (c) Counterpart of b) for the case where signals are diffused white.

increases, obtaining the worst result when the signal is generated at random. This result is aligned with the theoretical claims since it is assumed that the signal $\mathbf{x}_0$ is bandlimited. It is also worth noting that the architecture also achieves a good denoising error with the "DW" model, showcasing that the GCG is also capable of denoising other types of smooth graph signals.

Next, Fig. 6(b) compares the performance of the 2-layer GCG ("2L-GCG"), the deep GCG ("GCG") and the deep GDec ("GDec") with the baseline models introduced in Section III, which are the total variation ("TV") [34], Laplacian regularization ("LR") [37], and bandlimited model ("BL") [46]. In this setting, the graphs are SBM with 256 nodes and 8 communities, and the signals are bandlimited with a bandwidth of 8. Since the "BL" model with $K = 8$ captures the actual generative model of the signal $\mathbf{x}_0$, it achieves the best denoising performance. However, it is worth noting that the GCG obtains a similar result, outperforming the other alternatives. On the other hand, the "LR" obtains an error noticeably larger than that of "BL" and "GCG," highlighting that, even though "BL" and "LR" are related models their different assumptions lead to different performances. Moreover, the benefits of using the deep GCG instead of the 2-layer architecture are apparent, since it achieves a better performance in fewer epochs.

On the other hand, Fig. 6(c) illustrates a similar experiment but with the graph signals generated as "DW". Under this setting, it is clear that the GDec outperforms the other alternatives. These results showcase the benefits of employing a nonlinear architecture relative to classical denoising approaches. Furthermore, this experiment corroborates that the GDec is more robust to the presence of noise when the signals are aligned with the prior implicitly captured by the architecture.

C. Denoising Real-World Signals

Finally, we assess the performance of the proposed architectures in several real-world datasets. To the baselines considered in the previous experiments, we add the following competitive denoising algorithms: graph trend filtering ("GTF") [35], a graph-aware median operator ("MED") [39], a GCNN ("GCNN") implemented as in [12], a graph attention network ("GAT") [56], a Kron reduction-based autoencoder ("K-GAE") [41], and the graph unrolling sparse coding architecture ("GUSC") in [42]. Moreover, we consider the following noise distributions: (i) zero-mean Gaussian distribution, which is the noise model typically assumed for sensor measurements in signal processing; (ii) uniform distribution on some interval $[0, a]$, where $a \in \mathbb{R}_+$ is chosen accordingly to the desired noise power; and (iii) Bernoulli distribution to model errors in binary signals. Next, we describe the selected datasets and analyze the achieved results, which are summarized in Table II.

Temperature. We consider a network of 316 weather stations distributed across the United States [45]. Graph signals represent daily temperature measurements in the first three months of the year 2003. The graph $\mathcal{G}$ represents the geographical distance between weather stations and is given by the 8-nearest neighbors graph. The first and second rows of Table II list the NMSE when the noise is drawn from a Gaussian and a uniform distribution, respectively. In both cases, the noise has a normalized power of 0.3. It is clear that the GDec architecture outperforms the alternatives in both scenarios. Furthermore, we can observe that the GCG achieves a better performance than GCNN, showcasing the benefits of being able to use a more general graph filter.

S&P 500. In this experiment, we have 189 nodes representing stocks belonging to 6 different sectors of the S&P 500 with the graph signals representing the prices of those stocks at particular time instants. We follow [57] to estimate the graph $\mathcal{G}$ assuming that the signals are drawn from a multivariate Gaussian distribution and are smooth on $\mathcal{G}$. We consider the noise specifications described in the previous dataset and provide the NMSE in the third and fourth rows of Table II. It is worth noting that considering Gaussian noise in this dataset constitutes a more challenging denoising problem than using uniform noise. A plausible explanation is that the graph is estimated assuming that the data follows a Gaussian distribution, and hence, it is harder to separate the Gaussian noise from the true signals. In the presence of Gaussian noise, the GCG and the GDec outperform the other 8 alternatives. However, when the noise follows a uniform distribution, the best performance is obtained by the GCG and the GCNN, with GDec being the third best. In addition, we observe that traditional methods yield an error that is considerably larger than that incurred by the proposed architectures. This is aligned with our initial intuition about linear and quadratic methods being more limited when the actual relation between $\mathbf{x}_0$ and $\mathcal{G}$ is more intricate, as is the case for financial data.

TABLE II
DENOISING ERROR OF SEVERAL DATASETS WITH DIFFERENT TYPES OF RANDOM NOISE

DATASET (METRIC)	METHOD	BL	TV	LR	GTF	MED	GCNN	GAT	K-GAE	GUSC	GCG	GDec
TEMPERATURE (NMSE)	Gaussian	0.062	0.117	0.095	0.066	0.053	0.123	0.045	0.134	0.044	0.056	**0.035**
	Uniform	0.063	0.117	0.094	0.064	0.053	0.118	0.047	0.136	0.049	0.057	**0.036**
S&P 500 (NMSE)	Gaussian	0.350	0.238	0.231	0.239	0.319	0.252	0.199	0.354	0.203	**0.188**	**0.188**
	Uniform	0.216	0.246	0.161	0.298	0.340	**0.091**	0.222	0.273	0.127	**0.094**	0.121
CORA (ERROR RATE)	Whole $\mathcal{G}$	0.154	0.142	0.115	0.126	0.167	0.099	0.141	0.135	0.099	**0.093**	0.121
	Conn. comp.	0.151	0.141	0.105	0.116	0.165	0.093	0.139	0.135	0.094	**0.088**	0.125

The font in bold highlights the architectures with the best errors.

Cora. Lastly, we consider the Cora citation network dataset [12]. Nodes represent different scientific documents and edges capture citations among them. Like in [42], we consider the 7 class labels as binary graph signals encoding if the particular node belongs to that class. For each signal, we consider 25 realizations of Bernoulli noise that randomly flips 30% of the binary values of the signals, resulting in a total of 175 noisy graph signals. With the error rate denoting the proportion of labels correctly recovered after the denoising process, Table II shows the error metric averaged over all the signals. Moreover, since the graph is formed by several connected components, we report two results: the error rate when the whole graph is considered (fifth row) and the error rate when only the largest connected component is considered (sixth row). It can be seen that the GCG yields the best performance in both cases.

VII. CONCLUSION

In this paper, we faced the relevant task of graph-signal denoising. To approach this problem, we presented two over-parametrized and untrained GNNs and provided theoretical guarantees on the denoising performance of both architectures when denoising K-bandlimited graph signals under some simplifying assumptions. Moreover, we numerically illustrated that the proposed architectures are also capable of denoising graph signals in more general settings. The key difference between the two architectures resided in the linear transformation that incorporates the information encoded in the graph. The GCG employs fixed (non-learnable) low-pass graph filters to model convolutions in the vertex domain, promoting smooth estimates. On the other hand, the GDec relies on a nested collection of graph upsampling operators to progressively increase the input size, limiting the degrees of freedom of the architecture, and providing more robustness to noise. In addition to the aforementioned analysis, we tested the validity of the proposed theorems and evaluated the performance of both architectures with real and synthetic datasets, showcasing a better performance than other classical and nonlinear methods for graph-signal denoising. Finally, we consider extending the results from Theorems 1 and 2 to more general scenarios as an interesting future line of work.

APPENDIX A
PROOF OF THEOREM 1

Let $\mathbf{x}_0$ be a K bandlimited graph signal as described in (2), which is spanned by the K leading eigenvectors of the graph $\mathbf{V}_K$, with $\tilde{\mathbf{x}}_0$ denoting its frequency representation. Let $\mathbf{Q}$ be an orthonormal matrix that aligns the subspaces spanned by $\mathbf{V}_K$ and $\mathbf{W}_K$, and denote as $\bar{\mathbf{x}}_0 = \mathbf{W}_K\mathbf{Q}\tilde{\mathbf{x}}_0$ the bandlimited signal using $\mathbf{W}_K$ as basis and whose frequency response is also $\tilde{\mathbf{x}}_0$. Note that $\bar{\mathbf{x}}_0$ can be interpreted as recovering $\mathbf{x}_0$ from its frequency response using $\mathbf{W}_K$ in lieu of $\mathbf{V}_K$. Also, note that $\mathbf{x}_0 - \bar{\mathbf{x}}_0 = (\mathbf{V}_K - \mathbf{W}_K\mathbf{Q})\tilde{\mathbf{x}}_0$ represents the error between the signal $\mathbf{x}_0$ and its approximation inside the subspace spanned by $\mathbf{W}_K$. With these definitions in place, in [33, Th. 3] the authors showed that error when denoising a signal $\mathbf{x} = \mathbf{x}_0 + \mathbf{n}$ is bounded with probability at least $1 - e^{-F^2} - \phi$ by

$$\|\mathbf{x}_0 - f_{\boldsymbol{\Theta}_{(t)}}(\mathbf{Z}|\mathcal{G})\|_2 \leq \|\boldsymbol{\Psi}\mathbf{x}_0\|_2 + \xi\|\mathbf{x}\|_2 + \sqrt{\textstyle\sum_{i=1}^{N}((1-\eta\sigma_i^2)^t - 1)^2(\mathbf{w}_i^\top \mathbf{n})^2}, \tag{28}$$

with $\boldsymbol{\Psi} := \mathbf{W}(\mathbf{I}_N - \eta\boldsymbol{\Sigma}^2)^t\mathbf{W}^\top$, and $\mathbf{I}_N$ the $N \times N$ identity matrix. However, note that the bound provided for $\|\boldsymbol{\Psi}\mathbf{x}_0\|_2$ in [33] requires $\mathbf{x}_0$ lying in the subspace spanned by $\mathbf{W}_K$, which is not the case. As a result, we further bound this term as

$$\begin{aligned}
\|\boldsymbol{\Psi}\mathbf{x}_0\|_2 &= \|\boldsymbol{\Psi}(\mathbf{x}_0 + \bar{\mathbf{x}}_0 - \bar{\mathbf{x}}_0)\|_2 \\
&\overset{(i)}{=} \|\boldsymbol{\Psi}_K\bar{\mathbf{x}}_0 + \boldsymbol{\Psi}(\mathbf{V}_K - \mathbf{W}_K\mathbf{Q})\tilde{\mathbf{x}}_0\|_2 \\
&\overset{(ii)}{\leq} \|\boldsymbol{\Psi}_K\bar{\mathbf{x}}_0\|_2 + \|\boldsymbol{\Psi}(\mathbf{V}_K - \mathbf{W}_K\mathbf{Q})\tilde{\mathbf{x}}_0\|_2 \\
&\overset{(iii)}{\leq} \|\boldsymbol{\Psi}_K\|_2\|\bar{\mathbf{x}}_0\|_2 + \|\boldsymbol{\Psi}\|_2\|\mathbf{V}_K - \mathbf{W}_K\mathbf{Q}\|_F\|\tilde{\mathbf{x}}_0\|_2 \\
&\overset{(iv)}{\leq} \left(\|\boldsymbol{\Psi}_K\|_2 + \delta\|\boldsymbol{\Psi}\|_2\right)\|\mathbf{x}_0\|_2 \\
&\overset{(v)}{=} \left((1-\eta\sigma_K^2)^t + \delta(1-\eta\sigma_N^2)^t\right)\|\mathbf{x}_0\|_2.
\end{aligned} \tag{29}$$

Here, $\boldsymbol{\Psi}_K := \mathbf{W}_K(\mathbf{I}_K - \eta\boldsymbol{\Sigma}_K^2)^t\mathbf{W}_K^\top$, and $\boldsymbol{\Sigma}_K$ represents a diagonal matrix containing the first K leading eigenvalues σ_k. We have that (i) follows from $\bar{\mathbf{x}}_0$ being bandlimited in $\mathbf{W}_K$, so $\boldsymbol{\Psi}\bar{\mathbf{x}}_0 = \boldsymbol{\Psi}_K\bar{\mathbf{x}}_0$. Then, (ii) follows from the triangle inequality, and (iii) from the ℓ_2 norm being submultiplicative and using the Frobenius norm as an upper bound for the ℓ_2 norm. In (iv) we apply the result of Lemma 1, which holds with probability at least $1-\epsilon$ because $N > N_{K,\epsilon,\delta}$, and the fact that, since both $\mathbf{W}_K$ and $\mathbf{V}_K$ are orthonormal matrices, we have that $\|\mathbf{x}_0\|_2 = \|\bar{\mathbf{x}}_0\|_2 = \|\tilde{\mathbf{x}}_0\|_2$. We obtain (v) from the largest eigenvalues present in $\boldsymbol{\Psi}_K$ and $\boldsymbol{\Psi}$.

Finally, the proof concludes by combining (29) and (28).

APPENDIX B
PROOF OF LEMMA 1

Define $\tilde{\boldsymbol{\mathcal{A}}}$ as $\tilde{\boldsymbol{\mathcal{A}}} := \mathbb{E}[\tilde{\mathbf{A}}] = \mathbb{E}[\mathbf{D}]^{-\frac{1}{2}}\boldsymbol{\mathcal{A}}\mathbb{E}[\mathbf{D}]^{-\frac{1}{2}}$ and let $\boldsymbol{\mathcal{X}}$ be given by (17). Denote by $\boldsymbol{\mathcal{H}}$ a graph filter defined as a polynomial of the expected adjacency matrix $\tilde{\boldsymbol{\mathcal{A}}}$, and let $\bar{\boldsymbol{\mathcal{X}}}$ be the expected squared Jacobian using the graph filter $\boldsymbol{\mathcal{H}}$, i.e.,

$$\bar{\boldsymbol{\mathcal{X}}} = 0.5\left(\mathbf{1}\mathbf{1}^\top - \frac{1}{\pi}\arccos\left(\boldsymbol{\mathcal{C}}^{-1}\boldsymbol{\mathcal{H}}^2\boldsymbol{\mathcal{C}}^{-1}\right)\right)\odot\boldsymbol{\mathcal{H}}^2, \tag{30}$$

where $\boldsymbol{\mathcal{C}}$ is the counterpart of $\mathbf{C}$ in (17), but using $\boldsymbol{\mathcal{H}}$ instead of $\mathbf{H}$. Given the following eigendecompositions $\tilde{\mathbf{A}} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$, $\boldsymbol{\mathcal{X}} = \mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top$, $\tilde{\boldsymbol{\mathcal{A}}} = \bar{\mathbf{V}}\bar{\boldsymbol{\Lambda}}\bar{\mathbf{V}}^\top$, and $\bar{\boldsymbol{\mathcal{X}}} = \bar{\mathbf{W}}\bar{\boldsymbol{\Sigma}}\bar{\mathbf{W}}^\top$, for arbitrary orthonormal matrices $\mathbf{T}$ and $\mathbf{R}$, we have that

$$\begin{aligned}\|\mathbf{V}_K - \mathbf{W}_K\mathbf{Q}\|_{\mathrm{F}} &\le \|\mathbf{V}_K - \bar{\mathbf{V}}_K\mathbf{T}\|_{\mathrm{F}} \\ &+ \|\bar{\mathbf{V}}_K\mathbf{T} - \bar{\mathbf{W}}_K\mathbf{R}\|_{\mathrm{F}} + \|\bar{\mathbf{W}}_K\mathbf{R} - \mathbf{W}_K\mathbf{Q}\|_{\mathrm{F}}.\end{aligned} \tag{31}$$

To prove the theorem, we bound the three terms on the right hand side of (31).

Bounding $\|\bar{\mathbf{V}}_K\mathbf{T} - \bar{\mathbf{W}}_K\mathbf{R}\|_F$. From the definition of an SBM, it follows that $\boldsymbol{\mathcal{A}} = \mathbb{E}[\mathbf{A}] = \mathbf{B}\boldsymbol{\Omega}\mathbf{B}^\top$, where $\mathbf{B} \in \{0,1\}^{N\times K}$ is an indicator matrix encoding the community to which each node belongs, and $\boldsymbol{\Omega}$ is a $K \times K$ matrix encoding the link probability between the communities of the graph. Therefore, $\tilde{\boldsymbol{\mathcal{A}}}$ and $\bar{\boldsymbol{\mathcal{X}}}$ are both block matrices whose blocks coincide with the communities in the SBM. This implies that the eigenvectors associated with non-zero eigenvalues must span the columns of $\mathbf{B}$. Hence, the leading eigenvectors must be related by an orthonormal transformation, from where it follows that, given $\mathbf{T}$, we can always find $\mathbf{R}$ such that

$$\|\bar{\mathbf{V}}_K\mathbf{T} - \bar{\mathbf{W}}_K\mathbf{R}\|_{\mathrm{F}} = 0. \tag{32}$$

Bounding $\|\mathbf{V}_K - \bar{\mathbf{V}}_K\mathbf{T}\|_F$. Under Assumption 1, as it is shown in [50], with probability at least $1-\rho$ we have that

$$\|\tilde{\mathbf{A}} - \tilde{\boldsymbol{\mathcal{A}}}\| \le 3\sqrt{\frac{3\ln(4\,N/\rho)}{\beta_{\min}}}. \tag{33}$$

Then, we combine the concentration (33) with the Davis-Kahan results [58, Th. 2], which bound the distance between the subspaces spanned by the population eigenvectors ($\bar{\mathbf{V}}_K$) and their sample version ($\mathbf{V}_K$). Denoting as $\bar{\lambda}_i$ the i-th eigenvalue collected in $\bar{\boldsymbol{\Lambda}}$, i.e. $\bar{\lambda}_i = \bar{\Lambda}_{ii}$, we obtain that there exists an orthonormal matrix $\mathbf{T}$ such that

$$\begin{aligned}\|\mathbf{V}_K - \bar{\mathbf{V}}_K\mathbf{T}\|_F &\le \frac{\sqrt{8\,K}}{\bar{\lambda}_K - \bar{\lambda}_{K+1}}\|\tilde{\mathbf{A}} - \tilde{\boldsymbol{\mathcal{A}}}\|_F \\ &\le \frac{3\sqrt{8\,K}}{\bar{\lambda}_K}\sqrt{\frac{3\ln(4\,N/\rho)}{\beta_{\min}}},\end{aligned} \tag{34}$$

where we note that, since $\tilde{\boldsymbol{\mathcal{A}}}$ follows an SBM, then $\bar{\lambda}_i = 0$ for all $i > K$.

Since $\beta_{\min} = \omega(\ln(N/\rho))$, we obtain that

$$\|\mathbf{V}_K - \bar{\mathbf{V}}_K\mathbf{T}\|_{\mathrm{F}} \to 0, \qquad \text{as } N \to \infty. \tag{35}$$

Bounding $\|\bar{\mathbf{W}}_K\mathbf{R} - \mathbf{W}_K\mathbf{Q}\|_F$. If we show that $\|\boldsymbol{\mathcal{X}} - \bar{\boldsymbol{\mathcal{X}}}\| \to 0$ as $N \to \infty$, we can then mimic the procedure in (33) and (34) to show that the difference between the leading K eigenvectors of $\boldsymbol{\mathcal{X}}$ and $\bar{\boldsymbol{\mathcal{X}}}$ also vanishes. Hence, we are left to show that $\|\boldsymbol{\mathcal{X}} - \bar{\boldsymbol{\mathcal{X}}}\| \to 0$ as $N \to \infty$. From the definitions of $\boldsymbol{\mathcal{X}}$ and $\bar{\boldsymbol{\mathcal{X}}}$, it follows that

$$\begin{aligned}\|\boldsymbol{\mathcal{X}} - \bar{\boldsymbol{\mathcal{X}}}\| &\le 0.5\|\mathbf{H}^2 - \boldsymbol{\mathcal{H}}^2\| \\ &+ \frac{1}{2\pi}\|\arccos\left(\boldsymbol{\mathcal{C}}^{-1}\boldsymbol{\mathcal{H}}^2\boldsymbol{\mathcal{C}}^{-1}\right)\odot\boldsymbol{\mathcal{H}}^2 \\ &- \arccos\left(\mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1}\right)\odot\mathbf{H}^2\|.\end{aligned} \tag{36}$$

To bound the difference between the sampled and expected filters, we have that

$$\begin{aligned}\|\mathbf{H}^2 - \boldsymbol{\mathcal{H}}^2\| &= \left\|\left(\sum_{\ell=0}^{L} h_\ell\tilde{\mathbf{A}}^\ell\right)^2 - \left(\sum_{\ell=0}^{L} h_\ell\tilde{\boldsymbol{\mathcal{A}}}^\ell\right)^2\right\| \\ &= \left\|\sum_{\ell=0}^{2\,L}\alpha_\ell\left(\tilde{\mathbf{A}}^\ell - \tilde{\boldsymbol{\mathcal{A}}}^\ell\right)\right\| \le \sum_{\ell=0}^{2\,L}\alpha_\ell\left\|\tilde{\mathbf{A}}^\ell - \tilde{\boldsymbol{\mathcal{A}}}^\ell\right\|,\end{aligned} \tag{37}$$

for suitable coefficients α_ℓ and recalling that $L = 2$. Then, we can then leverage the fact that $\|\tilde{\mathbf{A}}\| = \|\tilde{\boldsymbol{\mathcal{A}}}\| = 1$ to see that $\|\tilde{\mathbf{A}}^\ell - \tilde{\boldsymbol{\mathcal{A}}}^\ell\| \le \ell\|\tilde{\mathbf{A}} - \tilde{\boldsymbol{\mathcal{A}}}\|$. We thus get that

$$\|\mathbf{H}^2 - \boldsymbol{\mathcal{H}}^2\| \le \sum_{\ell=0}^{2\,L}\ell\alpha_\ell\left\|\tilde{\mathbf{A}} - \tilde{\boldsymbol{\mathcal{A}}}\right\| \to 0, \quad \text{as } N \to \infty, \tag{38}$$

where the limiting behavior follows from (33). Finally, to bound the second term in (36), we first note that the argument of the norm can be re-written as $\arccos(\mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1}) \odot (\boldsymbol{\mathcal{H}}^2 - \mathbf{H}^2) + (\arccos(\boldsymbol{\mathcal{C}}^{-1}\boldsymbol{\mathcal{H}}^2\boldsymbol{\mathcal{C}}^{-1}) - \arccos(\mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1})) \odot \boldsymbol{\mathcal{H}}^2$. The limit in (38) ensures that the first of these two terms vanishes. Similarly, it follows that $\|\boldsymbol{\mathcal{C}}^{-1}\boldsymbol{\mathcal{H}}^2\boldsymbol{\mathcal{C}}^{-1} - \mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1}\| \to 0$ which, combined with the fact that arccos is a uniformly continuous function, we can always find an $N_{\delta'}$ such that $\|\arccos(\boldsymbol{\mathcal{C}}^{-1}\boldsymbol{\mathcal{H}}^2\boldsymbol{\mathcal{C}}^{-1}) - \arccos(\mathbf{C}^{-1}\mathbf{H}^2\mathbf{C}^{-1})\| \le \delta'$ with high probability. Combining this result with (38) and applying the Davis-Kahan Theorem as done to obtain (34) we get that

$$\|\bar{\mathbf{W}}_K\mathbf{R} - \mathbf{W}_K\mathbf{Q}\|_{\mathrm{F}} \to 0, \qquad \text{as } N \to \infty. \tag{39}$$

Replacing (32), (35), and (39) into (31) our result follows.

APPENDIX C
PROOF OF LEMMA 2

Recall that $\tilde{\boldsymbol{\mathcal{A}}} = \mathbb{E}[\tilde{\mathbf{A}}]$, and define $\tilde{\boldsymbol{\mathcal{H}}} := \gamma\mathbf{I} + (1-\gamma)\tilde{\boldsymbol{\mathcal{A}}}$ as the specific graph filter introduced in Section V-A as a polynomial of $\tilde{\boldsymbol{\mathcal{A}}}$. Let $\boldsymbol{\mathcal{X}}$ be given by (27), and denote by $\bar{\boldsymbol{\mathcal{X}}}$ the expected squared Jacobian using the graph filter $\boldsymbol{\mathcal{H}}$, i.e.,

$$\bar{\boldsymbol{\mathcal{X}}} = 0.5\left(\mathbf{1}\mathbf{1}^\top - \frac{1}{\pi}\arccos\left(\tilde{\boldsymbol{\mathcal{C}}}^{-1}\boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^\top\tilde{\boldsymbol{\mathcal{C}}}^{-1}\right)\right)\odot\boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^\top \tag{40}$$

with $\boldsymbol{\mathcal{U}} = \tilde{\boldsymbol{\mathcal{H}}}\mathbf{P}$ and where the matrix $\tilde{\boldsymbol{\mathcal{C}}}$ is the counterpart of $\tilde{\mathbf{C}}$ in (27), but using $\boldsymbol{\mathcal{U}}$ in lieu of $\mathbf{U}$. Given the eigendecompositions $\tilde{\mathbf{A}} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top$, $\boldsymbol{\mathcal{X}} = \mathbf{W}\boldsymbol{\Sigma}\mathbf{W}^\top$, $\tilde{\boldsymbol{\mathcal{A}}} = \bar{\mathbf{V}}\bar{\boldsymbol{\Lambda}}\bar{\mathbf{V}}^\top$, and $\bar{\boldsymbol{\mathcal{X}}} = \bar{\mathbf{W}}\bar{\boldsymbol{\Sigma}}\bar{\mathbf{W}}^\top$, analogously to Lemma 1, we bound the difference between $\mathbf{V}_K$ and $\mathbf{W}_K$ by bounding the three terms in the right hand side of (31).

Bounding $\|\bar{\mathbf{V}}_K\mathbf{T} - \bar{\mathbf{W}}_K\mathbf{R}\|$. We have that $\boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^\top = \tilde{\boldsymbol{\mathcal{H}}}\mathbf{P}\mathbf{P}^\top\tilde{\boldsymbol{\mathcal{H}}}^\top$. Since $\mathbf{P}$ is a binary matrix indicating to which community belongs each node, $\mathbf{P}\mathbf{P}^\top$ is a block diagonal matrix that captures the structure of the communities of the SBM. Then, because $\tilde{\boldsymbol{\mathcal{H}}}$ is also block matrix with the same block pattern that the SBM, it turns out that the matrix $\bar{\boldsymbol{\mathcal{X}}}$ is also a block matrix whose blocks coincide with the communities in the SBM graph. Therefore, the rest of the bound is analogous to that in Lemma 1.

Bounding $\|\mathbf{V}_K - \bar{\mathbf{V}}_K\mathbf{T}\|$. The relation between $\mathbf{A}$ and $\boldsymbol{\mathcal{A}}$ is the same as in Lemma 1 so the bound provided in (35) holds.

Bounding $\|\bar{\mathbf{W}}_K\mathbf{R} - \mathbf{W}_K\mathbf{Q}\|$. To derive this bound we show that $\|\mathbf{U}\mathbf{U}^\top - \boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^\top\| = \|\tilde{\mathbf{H}}\mathbf{P}\mathbf{P}^\top\tilde{\mathbf{H}}^\top - \tilde{\boldsymbol{\mathcal{H}}}\mathbf{P}\mathbf{P}^\top\tilde{\boldsymbol{\mathcal{H}}}^\top\|$ goes to 0 as N grows. From (38) we have that $\|\mathbf{H} - \boldsymbol{\mathcal{H}}\| \to 0$, as $N \to \infty$, and hence, $\|\tilde{\mathbf{H}} - \tilde{\boldsymbol{\mathcal{H}}}\| \to 0$, as $N \to \infty$. Therefore, it can be seen that

$$\|\mathbf{U}\mathbf{U}^\top - \boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^\top\| \to 0, \quad \text{as } N \to \infty, \tag{41}$$

with $\|\mathbf{U}\mathbf{U}^\top - \boldsymbol{\mathcal{U}}\boldsymbol{\mathcal{U}}^\top\|$ vanishing as N grows. The remainder of the derivation of the bound is analogous to that for (39).

REFERENCES

[1] E. D. Kolaczyk and G. Csárdi, *Statistical Analysis of Network Data With R*, Berlin, Germany: Springer, 2014.

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

[3] J. Wu, C. Ma, L. Li, W. Dong, and G. Shi, "Probabilistic undirected graph based denoising method for dynamic vision sensor," *IEEE Trans. Multimedia*, vol. 23, no. 23, pp. 1148–1159, May 2020.

[4] M. I. Jordan, *Learning in Graphical Models*, Berlin, Germany: Springer, 1998.

[5] P. Djuric and C. Richard, *Cooperative and Graph Signal Processing: Principles and Applications*. Cambridge, MA, USA: Academic Press, 2018.

[6] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.

[7] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Process. Mag.*, vol. 34, no. 4, pp. 18–42, Jul. 2017.

[8] F. Scarselli, M. Gori, A. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

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

[10] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[11] S. Sakhavi, C. Guan, and S. Yan, "Learning temporal information for brain-computer interface using convolutional neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5619–5629, Nov. 2018.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.

[13] Q. Li, Z. Han, and X. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 3538–3545.

[14] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 11, pp. 4883–4894, Nov. 2020.

[15] C. Wang, S. Pan, G. Long, X. Zhu, and J. Jiang, "MGAE: Marginalized graph autoencoder for graph clustering," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 889–898.

[16] S. Rey, V. Tenorio, S. Rozada, L. Martino, and A. G. Marques, "Deep encoder-decoder neural network architectures for graph output signals," in *Proc. IEEE Conf. Signals, Syst., Comput.*, 2019, pp. 225–229.

[17] S. Rey, V. Tenorio, S. Rozada, L. Martino, and A. G. Marques, "Over-parametrized deep encoder-decoder schemes for inputs and outputs defined over graphs," in *Proc. IEEE Eur. Signal Process. Conf.*, 2021, pp. 855–859.

[18] H. Wang et al., "Graphgan: Graph representation learning with generative adversarial nets," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2508–2515.

[19] W. Liu, P. Chen, F. Yu, T. Suzumura, and G. Hu, "Learning graph topological features via GAN," *IEEE Access*, vol. 7, pp. 21834–21843, 2019.

[20] M. T. Schaub, Y. Zhu, J.-B. Seby, T. M. Roddenberry, and S. Segarra, "Signal processing on higher-order networks: Livin' on the edge... and beyond," *Signal Process.*, vol. 187, 2021, Art. no. 108149.

[21] T. M. Roddenberry, N. Glaze, and S. Segarra, "Principled simplicial neural networks for trajectory prediction," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 9020–9029.

[22] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9446–9454.

[23] R. Heckel and P. Hand, "Deep decoder: Concise image representations from untrained non-convolutional networks," in *Proc. Int. Conf. Learn. Repr.*, 2018, pp. 3538–3545.

[24] S. Liu, M. Long, J. Wang, and M. I. Jordan, "Generalized zero-shot learning with deep calibration network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2005–2015.

[25] B. Yaman, S. A. H. Hosseini, and M. Akçakaya, "Zero-shot self-supervised learning for mri reconstruction," in *Proc. Int. Conf. Learn. Representations*, 2021.

[26] S. Segarra, A. G. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, Aug. 2017.

[27] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Englewood Cliffs, NJ, USA: Prentice Halls, 1988.

[28] G. Carlsson, F. Memoli, A. Ribeiro, and S. Segarra, "Hierarchical clustering of asymmetric networks," *Adv. Data Anal. Classification*, vol. 12, no. 1, pp. 65–105, Mar. 2018.

[29] G. Carlsson, F. Mémoli, A. Ribeiro, and S. Segarra, "Axiomatic construction of hierarchical clustering in asymmetric networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 5219–5223.

[30] T. H. Do, D. M. Nguyen, and N. Deligiannis, "Graph auto-encoder for graph signal denoising," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 3322–3326.

[31] S. Rey, A. G. Marques, and S. Segarra, "An underparametrized deep decoder architecture for graph signals," in *Proc. IEEE Int. Workshop Comput. Adv. Multi-Sensor Adaptive Process.*, 2019, pp. 231–235.

[32] G. Mataev, P. Milanfar, and M. Elad, "Deepred: Deep image prior powered by red," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop*, 2019.

[33] R. Heckel and M. Soltanolkotabi, "Denoising and regularization via exploiting the structural bias of convolutional generators," in *Proc. Int. Conf. Learn. Representations*, 2020.

[34] S. Chen, A. Sandryhaila, J. M. F. Moura, and J. Kovacevic, "Signal denoising on graphs via graph filtering," in *Proc. IEEE Global Conf. Signal Info. Process.*, 2014, pp. 872–876.

[35] Y. Wang, J. Sharpnack, A. Smola, and R. Tibshirani, "Trend filtering on graphs," in *Proc. Artif. Intell. Statist.*, 2015, pp. 1042–1050.

[36] S. Ono, I. Yamada, and I. Kumazawa, "Total generalized variation for graph signals," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2015, pp. 5456–5460.

[37] J. Pang and G. Cheung, "Graph laplacian regularization for image denoising: Analysis in the continuous domain," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 26, no. 4, pp. 1770–1785, Apr. 2017.

[38] M. Onuki, S. Ono, M. Yamagishi, and Y. Tanaka, "Graph signal denoising via trilateral filter on graph spectral domain," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 2, pp. 137–148, Jun. 2016.

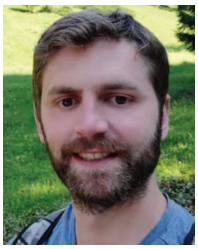
[39] D. Tay and J. Jiang, "Time-varying graph signal denoising via median filters," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 68, no. 3, pp. 1053–1057, Mar. 2021.

[40] S. Segarra, A. G. Marques, G. R. Arce, and A. Ribeiro, "Design of weighted median graph filters," in *Proc. IEEE Int. Workshop. Computat. Adv. Multi-Sensor Adaptive Process.*, 2017, pp. 1–5.

[41] F. Dorfler and F. Bullo, "Kron reduction of graphs with applications to electrical networks," *IEEE Trans. Circuits Syst. I., Reg. Papers*, vol. 60, no. 1, pp. 150–163, Jan. 2013.

[42] S. Chen, Y. C. Eldar, and L. Zhao, "Graph unrolling networks: Interpretable neural networks for graph signal denoising," *IEEE Trans. Signal Process.*, vol. 69, pp. 3699–3713, Jun. 2021, doi: 10.1109/TSP.2021.3087905.

[43] A. G. Marques, S. Segarra, and G. Mateos, "Signal processing on directed graphs: The role of edge directionality when processing and learning from network data," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 99–116, Nov. 2020.
[44] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.
[45] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, Jun. 2014.
[46] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Trans. Signal Process.*, vol. 63, no. 24, pp. 6510–6523, Dec. 2015.
[47] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3438–3445.
[48] A. Daniely, R. Frostig, and Y. Singer, "Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity," in *Adv. Neural Inf. Proc. Syst*., 2016, pp. 2253–2261.
[49] M. Newman, *Networks*. London, U.K.: Oxford Univ. Press, 2018.
[50] M. T. Schaub, S. Segarra, and J. N. Tsitsiklis, "Blind identification of stochastic block models from dynamical observations," *SIAM J. Math. Data Sc.*, vol. 2, no. 2, pp. 335–367, 2020.
[51] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[52] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthrop. Res.*, vol. 33, no. 4, pp. 452–473, 1977.
[53] "National centers for environmental information," 2020. [Online]. Available: https://www.ncei.noaa.gov/data/global-summary-of-the-day
[54] D. J. Watts, "Networks, dynamics, and the small-world phenomenon," *Amer. J. Sociol.*, vol. 105, no. 2, pp. 493–527, 1999.
[55] P. Holme and B. J. Kim, "Growing scale-free networks with tunable clustering," *Phys. Rev. E*, vol. 65, no. 2, 2002, Art. no. 0 26107.
[56] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.
[57] J. V. D. M. Cardoso, J. Ying, and D. P. Palomar, "Algorithms for learning graphs in financial markets," 2020, *arXiv:2012.15410*.
[58] Y. Yu, T. Wang, and R. J. Samworth, "A useful variant of the Davis–Kahan theorem for statisticians," *Biometrika*, vol. 102, no. 2, pp. 315–323, 2015.



Samuel Rey (Student Member, IEEE) received the B.Sc. and M.Sc. degrees in telecommunication engineering, both with highest honors, from King Juan Carlos University (URJC), Madrid, Spain, in 2018. He is currently working towards the Ph.D. thesis with the Department of Signal Theory and Communications, King Juan Carlos University. His current research intersests include graph signal processing, graph neural networks, nonconvex optimization, and data science over networks. He received the "Best Young Investigator Award" across all M.Sc. students at URJC in 2018. He was the recipient of the Spanish Federal FPU Scholarship for Ph.D. studies in 2018, and with the Mobility Grant for Ph.D. FPU students in 2021.

Santiago Segarra (Senior Member, IEEE) received the B.Sc. degree in industrial engineering with highest honors (Valedictorian) from the Instituto Tecnológico de Buenos Aires (ITBA), Buenos Aires, Argentina, in 2011, the M.Sc. in electrical engineering from the University of Pennsylvania (Penn), Philadelphia, PA, USA, in 2014, and the Ph.D. degree in electrical and systems engineering from Penn in 2016. From September 2016 to June 2018 he was a Postdoctoral Research Associate with the Institute for Data, Systems, and Society, Massachusetts Institute of Technology Cambridge, MA, USA. Since July 2018, he has been a W. M. Rice Trustee Assistant Professor with the Department of Electrical and Computer Engineering, Rice University, Houston, TX, USA. His research interests include network theory, data analysis, machine learning, and graph signal processing. Dr. Segarra was the recipient of the ITBA's 2011 Best Undergraduate Thesis Award in Industrial Engineering, the 2011 Outstanding Graduate Award granted by the National Academy of Engineering of Argentina, the 2017 Penn's Joseph and Rosaline Wolf Award for Best Doctoral Dissertation in Electrical and Systems Engineering, the 2020 IEEE Signal Processing Society Young Author Best Paper Award, the 2021 Rice's School of Engineering Research and Teaching Excellence Award, and five best conference paper awards.

Reinhard Heckel (Member, IEEE) received the Ph.D. degree in electrical engineering from ETH Zurich, Zurich, Switzerland, in 2014. He was a Visiting Ph.D. Student with the Department of Statistics, Stanford University, Stanford, CA, USA. He is currently a Rudolf Moessbauer Assistant Professor with the Department of Electrical and Computer Engineering (ECE), Technical University of Munich, Munich, Switzerland, and an Adjunct Assistant Professor with the Department ECE, Rice University, Houston, TX, USA, where he was an Assistant Professor, from 2017 to 2019. Before that, he was a Postdoctoral Scholar with UC Berkeley, Berkeley, CA, USA–sharing an office with Ilan Shomorony–and a Researcher with the Cognitive Computing and Computational Sciences Department, IBM Research Zurich, Ruschlikon, Switzerland. His research interests include the intersection of machine learning and signal/information processing with a current focus on deep networks for solving inverse problems, learning from few and noisy samples, and DNA data storage.

Antonio G. Marques (Senior Member, IEEE) received the joint B.Sc.&M.Sc. (5-year) degree in telecommunications engineering and the doctorate degree, both with highest honors, from the Carlos III University of Madrid, Getafe, Spain, in 2002 and 2007, respectively. In 2007, he became a Faculty with the Department of Signal Theory and Communications, King Juan Carlos University, Madrid, Spain, where he currently develops his research and teaching activities as a Full Professor. From 2005 to 2015, he held different visiting positions with the University of Minnesota, Minneapolis, Minnesota. In 2015, 2016, and 2017 he was a Visitor Scholar with the University of Pennsylvania, Philadelphia, PA, USA. His current research intersets include on signal processing, machine learning, data science and artificial intelligence over graphs, and nonlinear and stochastic optimization of wireless, power and transportation networks. Dr. Marques has served the IEEE in a number of posts, including as an Associate Editor and the Technical/General Chair of different conferences, and, currently, he is a Senior Area Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESS. He is a member of the IEEE Signal Process, Theory and Methods Tech. Comm. His work has been awarded in several journals, conferences, and workshops, with recent ones including IEEE SSP 2016, IEEE SAM 2016, IEEE SPS IEEE Y.A. Best Paper Award 2020, and CIT 2021. He is the recipient of the "2020 EURASIP Early Career Award" and a member of IEEE, EURASIP, and the ELLIS society.