Stability Properties of Graph Neural Networks
Fernando Gama, Joan Bruna, and Alejandro Ribeiro

Abstract—Graph neural networks (GNNs) have emerged as a powerful tool for nonlinear processing of graph signals, exhibiting success in traffic forecasting, recommender systems, power outage prediction, and motion planning, among others. GNNs consist of a cascade of layers, each of which applies a graph convolution, followed by a pointwise nonlinearity. In this work, we study the impact that changes in the underlying topology have on the output of the GNN. First, we show that GNNs are permutation equivariant, which implies that they effectively exploit internal symmetries of the underlying topology. Then, we prove that graph convolutions with integral Lipschitz filters, in combination with the frequency mixing effect of the corresponding nonlinearities, yields an architecture that is both stable to small changes in the underlying topology and discriminative of information located at high frequencies. These are two properties that cannot simultaneously hold when using only linear graph filters, which are either discriminative or stable, thus explaining the superior performance of GNNs.

Index Terms—graph neural networks, graph signal processing, network data, stability, graph filters, graph convolutions, scattering transform

I. INTRODUCTION

Convolutional neural networks (CNNs) [1] are the tool of choice for machine learning in Euclidean space. CNNs consist of nonlinear maps in which the output follows from sending the input through a cascade of layers, each of which computes a convolution with a bank of filters followed by a pointwise nonlinearity [2] Ch. 9. The value of the filter taps in the convolution is obtained by minimizing some cost function over a training set. The success of CNNs is simultaneously predictable and surprising. If we were to restrict attention to linear machine learning, a century of empirical and theoretical evidence would prescribe the use of convolutional filters. It is then predictable that the addition of a pointwise nonlinearity, which on the face of it is a pretty minor modification, is a sensible choice for a nonlinear processing architecture. But at the same time it is surprising that such a minor modification produces so much of an effect on empirically observed performance.

An answer to this question was put forth in [3] in the form of stability to diffeomorphisms. This seminal work considers smooth deformations of space and shows that CNNs are Lipschitz stable with respect to the gradient of this deformation. This is a property that linear convolutional filters can have only if their frequency responses are flat at high frequencies [4]. One can restrict attention to this class of filters but only at the cost of losing the ability to discriminate high frequency features. The nonlinear operation in a CNN functions as a frequency mixer that brings part of the high frequency energy towards low frequencies where it can be discriminated with stable filters. Thus, CNNs can be, both, stable and discriminative, but linear filters cannot be simultaneously stable and discriminative.

Parallel to the development of CNNs, the field of graph signal processing (GSP) has emerged as a generalization of Euclidean signal processing to signals whose components are related by arbitrary pairwise relationships described by an underlying graph support [5]–[8]. Central to GSP is the generalization of linear convolutional filters as polynomials of some matrix representation of the graph [9]. Having a valid convolution operation the notion of a graph neural network (GNN) emerges naturally as a cascade of layers, where each layer is made up of a graph convolution filter bank composed with a pointwise nonlinearity [10]–[13]. GNNs have, predictably, proved useful in a variety of problems where they, surprisingly, outperform linear graph filters [14]–[17].

The main contribution of this paper is to show that the advantage of GNNs relative to linear graph filters is their stability to graph deformations (Thm. 1). Our analysis utilizes graph Fourier transforms to provide a representation of the filter on the spectrum of the matrix representation of the graph. This representation shows that graph filters cannot be stable if they are designed to isolate features associated with large eigenvalues of the graph (Thm. 3). This is the equivalent of Euclidean convolutional filters being unable to be stable if they discriminate high frequencies. The graph filter banks that are used by GNNs cannot discriminate features associated with large eigenvalues either. But pointwise nonlinearities perform frequency mixing that brings part of the energy associated with large eigenvalues towards low eigenvalues where it can be discriminated with stable graph filters. Thus, GNNs can be, both, stable and discriminative, but linear graph filters cannot be simultaneously stable and discriminative. This is the exact same reason that explains the advantage of CNNs with respect to linear Euclidean filters.

The analysis of stability properties for the case of non-trainable GNNs built with graph wavelet filter banks has been carried out by [18], [19], in analogy to [3], [4]. More specifically, [18] studies the stability of these GNNs to perturbations, as well as to perturbations on the eigenvalues and eigenvectors of the underlying graph support. Alternatively, [19] considers the specific case of using diffusion wavelets [20] and proves permutation invariance as well as stability to perturbations measured by the diffusion distance [21], [22]. Both of these works consider an absolute perturbation model, where changes in the underlying graph support do not take into account the particularities of the topology. This leads to results that either depend on the size of the graph (i.e. larger...
graphs admit smaller edge weight changes \cite{19} or on the spectral gap \cite{19}, tying the applicability of the results to the specific graph under consideration. Stability of GNNs with arbitrary (trainable) filter banks have been studied in \cite{23} by leveraging a bound on the powers of the graph shift operator \cite{19} eq. (23). The resulting bound also depends on the spectral gap.

We begin the paper by proposing a relative perturbation model (Def. 2), and stating that GNNs built with integral Lipschitz filters (Def. 3) are stable under this model (Thm. 1). Since the proof of Thm. 1 builds upon the stability of the linear graph filters used in the GNN, we devote Sec. III to analyze these properties. First, we show that linear graph filters are equivariant to permutations (Prop. 1). Second, we discuss the model of absolute perturbations modulo permutation (analogous to that in \cite{18}, \cite{19}) and show that a linear filter whose frequency response is Lipschitz continuous is stable (Thm. 2), with a constant that depends on the Lipschitz condition of the filters as well as the intrinsic topological characteristics of the graph and its perturbation. Next, we show that, under the relative perturbation model, graph filters need to satisfy the integral Lipschitz condition to be stable (Thm. 3), and determine a family of perturbations under which the stability can be entirely controlled by the integral Lipschitz constant of the filters, for any graph (Thm. 4). In Sec. IV we show how the stability results for graph filters carry over to GNN architectures (Thm. 5). Sec. V offers an insightful discussion of the filters, for any graph (Thm. 4). In Sec. IV we show how far from a permutation operators that are impervious to permutations. To that end we define an operator distance modulo permutation as follows.

**Definition 1** (Operator Distance Modulo Permutation). Given operators \( \Psi : \mathbb{R}^N \rightarrow \mathbb{R}^N \) and \( \hat{\Psi} : \mathbb{R}^N \rightarrow \mathbb{R}^N \) we define their operator distance modulo permutation as

\[
\| \Psi - \hat{\Psi} \|_F = \min_{P \in \mathcal{P}} \max_{x \in \{x \mid \|x\|_1 = 1\}} \|P^T \Psi(x) - \hat{\Psi}(P^T x)\|_2 \quad (5)
\]

where \( \mathcal{P} \) is the set of \( N \times N \) permutation matrices (cf. \cite{16}) and where \( \| \cdot \|_2 \) stands for the \( \ell_2 \)-norm.

The operator distance in (5) compares the operators \( \Psi \) and \( \hat{\Psi} \) when the same permutations are applied at their respective inputs and outputs. If an operator is insensitive to permutations, as we expect GNNs to be, we must have \( \| \Psi - \hat{\Psi} \|_F = 0 \). The distance in (5) is therefore a measure of how far from a permutation operators \( \Psi \) and \( \hat{\Psi} \) are.

Asides from operator perturbations we also want to measure perturbations of the graph \( S \). As we explain in Sec. III-C, we tie the perturbation of edges to their magnitudes by considering a relative perturbation model such that the relationship between a graph \( S \) and its perturbed version is

\[
\tilde{S} = S + (ES + SE).
\]

If matrices \( S \) and \( \tilde{S} \) are given, we can find a symmetric matrix \( E \) satisfying (6) to define a relative dissimilarity between \( S \) and \( \tilde{S} \). As in Def. 1, we want to incorporate invariance to relabeling. We therefore present a permutation-insensitive perturbation model in the following definition.

**Definition 2** (Relative Perturbation Modulo Permutation). Given shift operators \( S \) and \( \tilde{S} \) we define the set of relative perturbation matrices modulo permutation as

\[
\mathcal{E}(S, \tilde{S}) = \left\{ E : P^T \tilde{S} P = S + (ES + SE), P \in \mathcal{P} \right\} \quad (7)
\]

The input to the GNN is the 0th layer signal \( x = x_0 \) and the output of the GNN is the \( L \)th layer feature \( x_L \). We assume that at each layer each feature is processed by \( F \) filters. This means that the first and last layer contain \( F \) filters whereas the remaining intermediate layers contain \( F^2 \) filters. It is elementary to consider intermediate layers with varying number of features but this complicates notation unnecessarily. We emphasize that the nonlinear operation in (2) is applied to each entry of \( z^f \) individually. We further assume that the nonlinearity is normalized Lipschitz so that for all \( a, b \in \mathbb{R} \),

\[
|\sigma(b) - \sigma(a)| \leq |b - a| \quad (3)
\]

Asides from the input \( x \), the GNN’s output depends on the filters \( h^f \) and the graph \( S \). We interpret a GNN as a transform defined by the filter coefficients that we can apply on any graph to any signal defined on the graph. Define then the map

\[
\Phi(S, x) = x_L^f \quad (4)
\]

to represent the outcome of applying (1)-(2) on graph \( S \) to input signal \( x = x_0 \). Our goal is to study the stability of the operator \( \Phi(S, \cdot) \) with respect to perturbations of the graph \( S \).

**II. STABILITY OF GRAPH NEURAL NETWORKS WITH RESPECT TO RELATIVE GRAPH PERTURBATIONS**

Consider undirected graphs with symmetric matrix representation \( S \) and GNNs made up of layers, each of which is the composition of graph convolutional filter banks with pointwise nonlinearities. At each layer we have input features \( x_{l-1}^g \) and a graph filter bank in which filters are specified by a set of coefficients \( h^f_{lk} = \{h^f_{lk}\}_{k=0}^\infty \). Filter \( h^f_{lk} \) processes input feature \( x_{l-1}^g \) to produce the intermediate feature

\[
z^f_{lk} = \sum_{k=0}^\infty h^f_{lk} S^k x_{l-1}^g. \quad (1)
\]

All of the intermediate features \( z^f_{lk} \) for a given index \( f \) are summed together and passed through a pointwise nonlinear function \( \sigma : \mathbb{R} \rightarrow \mathbb{R} \) to produce the output feature

\[
x^f_l = \sigma \left( \sum_g z^f_{lg} \right) \quad (2)
\]

The input to the GNN is the 0th layer signal \( x = x_0 \) and the output of the GNN is the \( L \)th layer feature \( x_L \). We
The smallest operator norm $\|E\|$ across all perturbations $E \in \mathcal{E}(S, \hat{S})$ provides a measure $d(S, \hat{S})$ of how far $S$ and $\hat{S}$ are from being permutations of each other in relative terms – see Sec. III-C. We will prove that a small perturbation of the graph $\hat{S}$ in the sense of Def. 2 produces a small perturbation of the GNN operator in the sense of Def. 3. This is not true for arbitrary graph filters but it is true for the class of integral Lipschitz filters of the next section.

**B. Integral Lipschitz filters**

Our perturbation analysis relies on the graph frequency response of graph filters – see Sec. III-A. For filter coefficients $h = \{h_k\}_{k=0}^\infty$, the graph frequency response $h(\lambda)$ is the analytic function $h : \mathbb{R} \rightarrow \mathbb{R}$ defined by the series

$$h(\lambda) = \sum_{k=0}^{\infty} h_k \lambda^k. \quad (8)$$

Integral Lipschitz filters satisfy a condition on the rate of change of their frequency responses that we introduce next.

**Definition 3 (Integral Lipschitz Filter).** Given a filter $h = \{h_k\}_{k=0}^\infty$ its frequency response $h(\lambda)$ is given by (8) and satisfies $|h(\lambda)| \leq 1$. We say the filter is integral Lipschitz if there exists a constant $C > 0$ such that for all $\lambda_1$ and $\lambda_2$,

$$|h(\lambda_2) - h(\lambda_1)| \leq C \frac{|\lambda_2 - \lambda_1|}{|\lambda_1 + \lambda_2|/2}. \quad (9)$$

The condition in (9) can be read as requiring the filter’s frequency response to be Lipschitz in any interval $(\lambda_1, \lambda_2)$ with a Lipschitz constant that is inversely proportional to the interval’s midpoint $(|\lambda_1 + \lambda_2|)/2$. To see this better, observe that (9) restricts the frequency response’s derivative to satisfy,

$$|h'(\lambda)| \leq C. \quad (10)$$

Thus, filters that are integral Lipschitz must have frequency responses that have to be flat for large $\lambda$ but can vary very rapidly around $\lambda = 0$, see Fig. 1. This is, not coincidentally, a condition reminiscent of the scale invariance of wavelet transforms [24, Ch. 7], [25], [26]. The condition $|h(\lambda)| \leq 1$ is not strictly necessary but it facilitates interpretations since it prevents the filter from amplifying energy – see Appendix A.

For relative perturbation models (Def. 2) and integral Lipschitz filters (Def. 3) the following stability result holds.

**Theorem 1.** Let $S = \mathbf{VAV}^H$ and $\hat{S}$ represent graphs with $N$ nodes. Let $E = \mathbf{UUM}^H \in \mathcal{E}(S, \hat{S})$ be a relative perturbation matrix (cf. Def. 2) whose norm is such that

$$d(S, \hat{S}) \leq \|E\| \leq \epsilon. \quad (11)$$

Consider graph neural network $\Phi$ [cf. (1)–(4)] with $L$ layers, $F$ features per layer, and normalized Lipschitz nonlinearity $\sigma$ [cf. (3)]. If the filters $h^\beta$ are integral Lipschitz (cf. Def. 3) with constant $C$, the distance modulo permutation (cf. Def. 1) between operators $\Phi(S, \cdot)$ and $\Phi(\hat{S}, \cdot)$ satisfies

$$\|\Phi(S, \cdot) - \Phi(\hat{S}, \cdot)\|_F \leq 2CLF^{L-1}(1+\delta\sqrt{N})\epsilon + O(\epsilon^2) \quad (12)$$

with $\delta := (\|U - V\|_2 + 1)^2 - 1$ standing for the eigenvector misalignment between shift operator $S$ and error matrix $E$.

**Proof.** See Appendix B.

To a first order approximation, Thm. 1 shows that GNNs are Lipschitz stable with respect to relative graph perturbations with stability constant $2CLF^{L-1}(1+\delta\sqrt{N})$ and that this is true uniformly for all graphs with a given number of nodes $N$.

The stability constant in (12) consists of the product of three terms. One given by the filter’s integral Lipschitz constant, $C$, one given by the number of filters and layers in the GNN, $L$, and one containing the eigenvector misalignment (1 + $\delta\sqrt{N}$). The latter is not a property of the filter, but a property of the perturbation $E$. The important point is that, regardless of $\delta$, judicious choice of filter coefficients $h$, affects the Lipschitz constant $C$ and allows control of the stability of the graph filters that define a GNN.

Thm. 1 provides insights that explain why GNNs work better than linear graph filters. These are presented in Sec. III-A. The proof of Thm. 1 is presented in Appendix A but the proof requires building on similar results for linear graph filters. The stability properties of linear graph filters are explored in Sec. III-A. A result that is stronger than Thm. 1 can be derived by imposing restrictions on the structure of the perturbation matrix $E$. This is presented in Sec. III-B.

III. Stability Properties of Graph Filters

We work with graphs $G = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ described by a set of $N$ nodes $\mathcal{V}$, a set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and a weight function $\mathcal{W} : \mathcal{E} \rightarrow \mathbb{R}$. The matrix representation $S$ is extended here to be a graph shift operator (GSO) [5], [7] which we
formally define as a matrix $S$ that respects the sparsity of the graph, namely, $s_{ij} = |S|_{ij} = 0$ whenever $i \neq j$ and $(j, i) \notin E$. This is a condition that is verified by, e.g., adjacency matrices, Laplacians, random walk Laplacians, and their normalized counterparts. We assume the shift operator is symmetric with eigenvector basis $V = \{v_1, \ldots, v_N\}$ and eigenvalue matrix $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_N)$ so that we can write

$$S = VA\hat{V}^H.$$  

(13)

It is assumed that eigenvalues are ordered from smallest to largest so that $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$.

The graph acts as a support for the data vector $x \in \mathbb{R}^N$ which we henceforth say to be a graph signal $x = [x_1, \ldots, x_N]^T$ that assigns the value $x_i$ to node $i$. The shift operator $S$ defines a linear map $y = Sx$ between graph signals that represents the local exchange of information between a node and its neighbors. Repeated application of $S$ accesses information from nodes located farther away since the shift operator that for a set of coefficients $h$ and the filter outputs $z := H(S)x$ and $\hat{z} := H(S)\hat{x}$ satisfy

$$\hat{z} := H(S)\hat{x} := H(S)(P^T x) = P^T(H(S)x) := P^Tz.$$  

(17)

Proof. See Appendix A.

Prop. 1 states the permutation equivariance of graph filters. Namely, a permutation of the input $x$ to $\hat{x} = P^T x$ – along with a permutation of the shift operator $S$ to $\hat{S} = P^T SP$ – results in a permutation of the output $z$ to $\hat{z} = P^Tz$. Notice that if we are given vectors $x$ and $\hat{x} = P^T x$ and we know the permutation matrix $P$ it is elementary to design linear operators that are permutation equivariant by simply applying the permutation to the linear operator. The permutation equivariance in (17) is more subtle in that it holds without having access to the permutation $P$.

Permutation equivariance of graph filters implies their usefulness in applications where graph relabeling is inconsequential. This motivates consideration of the space of linear operators modulo permutation along with operator distances between these equivalence classes which we introduce next.

Definition 4 (Perturbation Modulo Permutation). Given linear operators $A$ and $\hat{A}$ we define the operator distance modulo permutation as

$$\|A - \hat{A}\|_P = \min_{P \in \mathcal{P}} \max_{\|x\|=1} \|P^T(Ax) - \hat{A}(P^T x)\|.$$  

(18)

We denote as $P_0$ a matrix that attains the minimum in (18) and define the perturbation modulo permutation as the matrix

$$E = A - P_0^T\hat{A}P_0.$$  

(19)

The distance in (18) is a particularization of (5) to linear operators. It compares the effect of applying $A$ and $\hat{A}$ to a vector $x$ with a permutation $P$ applied before or after application of the linear operators. It measures this difference at the unit norm vector for which it is largest and at the permutation matrix that makes it smallest. The perturbation matrix $E$ is the difference between operators $A$ and $P_0^T\hat{A}P_0$ for the permutation matrix $P_0$ that achieves the minimum norm difference in (18). If there is more than one matrix that achieves the minimum, an arbitrary choice is acceptable for the results we will derive to hold. Further notice that it follows from (18) and (19) that the operator distance between $A$ and $\hat{A}$ is simply the operator norm of the error matrix $E$.

$$\|A - \hat{A}\|_P = \|E\|.$$  

(20)

It is ready to see that Def. 3 is a proper distance in the space of linear operators modulo permutation. In particular, it holds that $\|A - \hat{A}\|_P = 0$ if and only there exists a permutation matrix for which $\hat{A} = P^T AP$. From this latter fact it follows
that we can rewrite Prop. 1 to say that the distance modulo permutation between graph filters \( H(S) \) and \( H(\hat{S}) \) is null if the distance modulo permutation between the shift operators \( S \) and \( \hat{S} \) is null. We formally state and prove this fact in the following corollary.

**Corollary 1** (Permutation equivariance). For shift operators \( S \) and \( \hat{S} \) whose distance modulo permutation is \( \|S-\hat{S}\|_p=0 \), the distance modulo permutation between graph filters \( H(S) \) and \( H(\hat{S}) \) satisfies

\[
\|H(S) - H(\hat{S})\|_p = 0. \tag{21}
\]

**Proof.** If \( \|S-\hat{S}\|_p = 0 \) there exists permutation \( P \) such that \( \hat{S} = P^T SP \) and Proposition 1 holds. Thus, for this same permutation we have \( P^T (H(\hat{S})x) = H(S)(P^T x) \) for any vector \( x \). The result in (21) then follows from Def. 4.

Corollary 1 says that if two graphs are the same, the graph filters are pointwise operators in the graph frequency domain because

\[
\|S-P\|_p = 0 \implies \text{Graph frequency response are instantiated \[\text{cf. (24)}\] – through its eigenvalues} \Lambda \text{ – and which values of the frequency response are instantiated \[\text{cf. (24)}\] – through its eigenvalues } \Lambda .
\]

The remarkable observation to be made at this point is that the frequency response of a filter is completely characterized by the filter coefficients \( h \) [cf. (23)]. The effect of a specific graph is to determine the components of the GFT \( \hat{x} = V^H x \) – through its eigenvectors \( V \) – and which values of the frequency response are instantiated [cf. (23)] – through its eigenvalues \( \Lambda \). Fig. 2 shows an illustration of this latter fact. We have a filter with frequency response \( h(\lambda) \) represented as a continuous function. For a graph with eigenvalues \( \lambda_1 \) only the values at frequencies \( h(\lambda_n) \) affect the response of the filter. For a different graph with eigenvalues \( \hat{\lambda}_1 \) the values \( h(\hat{\lambda}_1) \) are the ones that determine the effect of the filter in the given graph.

Since graph perturbations alter the spectrum of a graph it seems apparent that the variability of the frequency response \( h(\lambda) \) has a direct effect on a filter’s stability to perturbations.

**Definition 5** (Lipschitz Filter). Given a filter \( h = \{h_k\}_{k=0}^\infty \) its frequency response \( h(\lambda) \) is given by (23) and satisfies \( |h(\lambda)| \leq 1 \). We say the filter is Lipschitz if there exists a constant \( C > 0 \) such that for all \( \lambda_1 \) and \( \lambda_2 \) the frequency response is such that

\[
|h(\lambda_2) - h(\lambda_1)| \leq C|\lambda_2 - \lambda_1|. \tag{25}
\]

As its name suggests, a filter is Lipschitz if its frequency response is Lipschitz. This means a Lipschitz filter is one whose frequency response does not change faster than linear. For filters that are Lipschitz, the following stability result relative to perturbations that are close to permutations holds.

**Theorem 2.** Let \( S = VAV^H \) and \( \hat{S} \) be graph shift operators. Let \( E = UM^H \) be the perturbation modulo permutation between \( S \) and \( \hat{S} \) (cf. Def. 4) and assume their operator distance modulo permutation (cf. Def. 4) satisfies

\[
\|S-\hat{S}\|_p = \|E\| \leq \varepsilon. \tag{26}
\]

For a Lipschitz filter (cf. Def. 5) with Lipschitz constant \( C \) the operator distance modulo permutation between filters \( H(S) \) and \( H(\hat{S}) \) satisfies

\[
\|H(S) - H(\hat{S})\|_p \leq C \left(1 + \delta \sqrt{N}\right) \varepsilon + O(\varepsilon^2) \tag{27}
\]

with \( \delta := \|U - V\|^2 + 1 \), standing for the eigenvector misalignment between shift operator \( S \) and error matrix \( E \).

**Proof.** See appendix B

Thm. 2 shows that filters are Lipschitz stable with respect to perturbations of the graph with stability constant \( C(1 + \delta \sqrt{N}) \).
Since the stability constant has the same relevant factors of the bound in Thm. 1 the same comments apply: (i) The bound holds uniformly for all graphs with \( N \) nodes. (ii) Stability is affected by a term, the Lipschitz constant \( C \), that is controllable through filter design (iii) Stability is further affected by a term, the eigenvector misalignment \((1 + \delta \sqrt{N})\), that depends on the structure of the perturbations that are expected in a particular problem but that cannot be affected by judicious filter choice.

Although Thm. 2 shows filter stability with respect to graph perturbations, the stability claim may be misleading given that the perturbation’s norm is not tied to the norm of the graph shift. To do so we replace (26) with the hypothesis \( \| S - \hat{S} \|_F = \| E \| \leq \varepsilon \| S \|, \) under which (27) becomes

\[
\| H(S) - H(\hat{S}) \|_F \leq C \left( 1 + \delta \sqrt{N} \right) \| S \| \varepsilon + O(\varepsilon^2). \tag{28}
\]

The bound in (28) is the result of a relative perturbation model. It is still a stability result but, in contrast to (27), not one that is uniform for all graphs with a given number of nodes. Making \( \| S \| \) arbitrarily large, makes the constant \((1 + \delta \sqrt{N})\| S \|\) arbitrarily large. One could think that it is reasonable to allow for larger filter perturbations when graphs have larger norm but this is not true. Filter perturbations determine feature perturbations whose magnitude need not be related to the graph’s norm. On closer inspection the problem with making \( \| E \| \leq \varepsilon \| S \| \) is that the norms of \( E \) and \( S \) are global properties of the error and the graph. In particular, this may imply that parts of the graph with small weights have large relative modifications because some other parts of the graph have large weights. This observation prompts the relative perturbation model we introduced in (7)-(9) in Sec. II-A which ties local properties of the shift operator and error matrices. We discuss these relative perturbations in the context of linear graph filters in the next section.

C. Effect of Relative Graph Perturbations on Graph Filters

We consider relative perturbations between \( S \) and \( \hat{S} \) as determined by the set \( \mathcal{E}(S, \hat{S}) \) in (7) of Def. 2. This set considers all the matrices \( E \) that allow us to write different permutations of \( S \) through relative perturbations of \( S \) of the form in (6). The norm \( \| E \| \) of a given error matrix \( E \) associated to a given permutation \( P \) is a measure of relative dissimilarity between \( P^T S P \) and \( S \). To measure dissimilarity between \( S \) and \( \hat{S} \) modulo permutation we evaluate the error norm at the permutation that affords the smallest error norm

\[
d(S, \hat{S}) = \min_{P \in \mathcal{E}} \| E \|
\]

s. t. \( P^T S P = S + (ES + SE) \). \tag{29}

The dissimilarity \( d(S, \hat{S}) \) measures how different \( S \) and \( \hat{S} \) are from being permutations of each other as measured by a multiplicative factor \( E \). This multiplicative model ties weight changes to the local structure of the graph – something that is not necessarily true of the absolute perturbation model considered in Sec. III-B. Specifically, if we write the entries of the perturbation matrix \( ES + SE \) we see that the difference between the \((i, j)\) edges of \( S \) and \( P_0^T S P_0 \) is proportional to the sum of the degrees of nodes \( i \) and \( j \). Thus, individual graph edges are being perturbed by amounts that are proportional to the degrees of their boundary nodes.

To study the effect of relative perturbations we can rely on the consideration of Lipschitz filters (cf. Def. 3) but more illuminating results are possible with the use of integral Lipschitz filters (cf. Def. 5). For relative perturbation models and integral Lipschitz filters the following stability result holds.

**Theorem 3.** Let \( S = VAV^H \) and \( \hat{S} \) be graph shift operators. Let \( E = UMU^H \in \mathcal{E}(S, \hat{S}) \) be a relative perturbation matrix (cf. Def. 2) whose norm is such that

\[
d(S, \hat{S}) \leq \| E \| \leq \varepsilon. \tag{30}
\]

For an integral Lipschitz filter (cf. Def. 3) with integral Lipschitz constant \( C \) the operator distance modulo permutation between filters \( H(S) \) and \( H(\hat{S}) \) satisfies

\[
\| H(S) - H(\hat{S}) \|_F \leq 2C \left( 1 + \delta \sqrt{N} \right) \varepsilon + O(\varepsilon^2) \tag{31}
\]

with \( \delta = (\| U - V \|_2 + 1)^2 - 1 \) standing for the eigenvector misalignment between shift operator \( S \) and error matrix \( E \).

**Proof.** See appendix C

Thm. 3 establishes stability with respect to relative perturbations of the form introduced in Def. 2. If a matrix \( E \) exists that makes \( S \) and \( \hat{S} \) close to permutations of each other in terms of this relative perturbations, the filters are stable with respect to the norm of the perturbation with stability constant \( 2C(1 + \delta \sqrt{N}) \). The constant is the same that appears in Thm. 1 and has the same shape as the one in Thm. 2. The former holds because Thm. 1 leverages Thm. 2 in its proof. That the latter holds is a coincidence. In any event, the same comments hold: The bound is uniform for all graphs and stability is affected by the Lipschitz constant \( C \) which depends on the filter and the eigenvector misalignment \((1 + \delta \sqrt{N})\), which depends on the structure of the perturbation. The important difference between Thms. 2 and 3 is that the class of filters that are admissible for stability with respect to relative perturbations is that of integral Lipschitz filters – whereas Lipschitz filters are required for stability with respect to absolute perturbations.

Before elaborating on the implications of allowing for integral Lipschitz filters we consider a variation of Thm. 3 in which we impose a structural constraint on the perturbation matrix \( E \).

**Theorem 4.** With the same hypotheses and definitions of Thm. 3 assume that there exists a matrix \( E \in \mathcal{E}(S, \hat{S}) \) that satisfies (30) and, furthermore, is such that

\[
\min \left[ \left\| E \frac{E}{\| E \|} - I \right\|, \left\| E \frac{E}{\| E \|} + I \right\| \right] \leq \varepsilon. \tag{32}
\]

Then, the operator distance modulo permutation between filters \( H(S) \) and \( H(\hat{S}) \) satisfies

\[
\| H(S) - H(\hat{S}) \|_F \leq 2C \varepsilon + O(\varepsilon^2). \tag{33}
\]

**Proof.** See appendix C

The structural constraint in (32) requires the error matrix \( E \) to be a scaled identity to within a first order approximation.
With this restriction on the set of admissible perturbations we can bound the eigenvector misalignment between $S$ and $E$ and remove the dependency that the bound in Thm. [3] has on the number of nodes in the graph. The bound in [3] holds uniformly for all graphs independently of their number of nodes.

The integral Lipschitz filters in Thms. [3] and [4] are of interest because they can be made finely discriminative at low-eigenvalue frequencies without affecting stability. Indeed, to control stability in [3] we need to limit the value of the integral Lipschitz constant $C$. This requires filter that change more slowly. In particular, for large $\lambda$ the filters must be constant and cannot discriminate nearby spectral features. But at values of $\lambda$ close to $\lambda = 0$ the filters can change rapidly and can therefore be designed to discriminate (arbitrarily) close spectral features. To the extent that relative perturbations are admissible – which, as we already pointed out, are arguably more sensible than absolute perturbations – Thm. [3] shows two fundamental properties of linear graph filters: (i) They cannot be stable and discriminative of spectral features associated with large $\lambda$. (ii) They can be stable and discriminative of spectral features associated with $\lambda \approx 0$.

Thus, if we are interested in discriminating features associated with $\lambda \approx 0$, linear graph filters are sufficient. However, if we are interested in discriminating features associated with large $\lambda$, linear graph filters will fail because of their sensitivity to graph perturbations. We will see in the following section that this is an issue we can resolve with the introduction of pointwise nonlinearities to produce graph neural networks.

IV. STABILITY PROPERTIES OF GRAPH NEURAL NETWORKS

The superior performance of graph neural networks (GNNs) can be explained by the use of filter banks and nonlinearities to successfully process high-eigenvalue frequencies in a stable manner. An arbitrary GNN $\Phi(S, \cdot)$ with $L$ layers, over a graph representation $S$, is defined by $\Phi(S, \cdot)$ for $\ell = 1, \ldots, L$. GNNs retain the two fundamental properties of linear filters. Namely, that they are permutation equivariant, and that they are stable.

**Proposition 2** (GNN permutation equivalence). Consider graph shifts $S$ and $S = P^TSP$ for some permutation matrix $P \in \mathcal{P}$ [cf. (16)]. Given a bank of filters $\{h_{\ell}^S\}$ for each layer $\ell = 1, \ldots, L$ and a pointwise nonlinearity $\sigma$, define a GNN $\Phi$ [cf. (14–2)]. Then, for any pair of corresponding signal graphs $x$ and $\hat{x} = P^T x$ used as input to the GNN it holds that

$$\Phi(S, x) = P^T \Phi(S, \hat{x}).$$  \hspace{1cm} (34)

**Proof.** See Appendix [D].

The stability of GNNs is inherited from the stability of the graph filters that conform the filter bank used in (1). The details of the GNN architecture control how this stability propagates, as specified next.

**Theorem 5** (GNN Stability). Let $S$ and $\hat{S}$ be GSOs related by perturbation matrix $E$ [cf. (19) or (29)] such that $\|E\| \leq \varepsilon$. Given a bank of filters $\{h_{\ell}^S\}$ such that $|h_{\ell}^S(\lambda)| \leq 1$ [cf. (1)] and a pointwise nonlinearity $\sigma$ that satisfies (3), define GNNs $\Phi(S, \cdot)$ and $\Phi(\hat{S}, \cdot)$ [cf. (1–2)]. If the corresponding filter banks satisfy $\|H_{\ell}^S(S) - H_{\ell}^S(\hat{S})\|_F \leq \Delta \varepsilon$, then it holds that

$$\|\Phi(S, \cdot) - \Phi(\hat{S}, \cdot)\|_F \leq \Delta LF^{L-1} \varepsilon + O(\varepsilon^2)$$  \hspace{1cm} (35)

for a GNN with a single input feature, a single output feature and $F$ features in each hidden layer.

**Proof.** See Appendix [E].

Thm. [5] establishes how the stability of the filters $\Delta$ is affected by the hyperparameters of the GNN architecture. More specifically, we see that the stability gets degraded linearly with the number of layers $L$, and exponentially with the number of features $F$ (with an exponent controlled by $L$). In essence, the deeper a GNN is, the less stable it is. However, we see that the result is still linear in the size of the perturbation $\varepsilon$ and in the stability constant $\Delta$ of the filters. This stability constant depends on the perturbation model under consideration (either absolute –Sec. [III-B]– or relative –Sec. [III-C]–) and on the Lipschitz condition on the graph filters (either Lipschitz –Def. [2]– or integral Lipschitz –Def. [3]–), as determined next.

**Proposition 3.** Under the conditions of Thm. [5] with $S = \mathbf{UAV}^{H}$, consider the following models.

(i) If matrix $E = \mathbf{UMU}^H$ models absolute perturbations [cf. (19)], and the filters are Lipschitz (Def. [5]) we have

$$\Delta = C(1 + \delta \sqrt{N})$$  \hspace{1cm} (36)

with $\delta = (\|\mathbf{U} - \mathbf{V}\| + 1)^2 - 1$.

(ii) If matrix $E = \mathbf{UMU}^H$ models relative perturbations [cf. Def. [2]], and the filters are integral Lipschitz (Def. [3]) we have

$$\Delta = 2C(1 + \delta \sqrt{N})$$  \hspace{1cm} (37)

with $\delta = (\|\mathbf{U} - \mathbf{V}\| + 1)^2 - 1$.

(iii) If matrix $E$ models relative perturbations [cf. Def. [2]] and satisfies (32), and the filters are integral Lipschitz (Def. [3]) we have

$$\Delta = 2C.$$  \hspace{1cm} (38)

**Proof.** Follows directly from Thm. [5] in combination with Thms. [2, 3] and [4]. These theorems establish the conditions and the corresponding values of $\Delta$.

The results of Thm. [5] in combination with Prop. [3] show that: (i) the use of Lipschitz filters lead to stable GNNs under absolute perturbations, and (ii) the use of integral Lipschitz filters lead to stable GNNs under relative perturbations.

The value of $\varepsilon$ in model (i) of Prop. [3] represents the absolute perturbation distance (cf. Def [2]) between shifts $S$ and $\hat{S}$ and as such, is independent of the actual particularities of the graph under study (a fixed value of $\varepsilon$ would mean a different perturbation level for graphs that have very different edge weights). Likewise, the value of $C$ is given by the Lipschitz constant of the filters. The higher the value of $C$, the more selective the filters can be (the more narrow they can be), but the more unstable the GNNs become. Finally, the value
of $\delta$ accounts for the eigenvector misalignment between the absolute error matrix $E$ and the shift $S$, which indicates the impact on the spectrum basis by the perturbation, and affects the stability bound by a value dependent on the number of nodes (the larger the graph, the more a change in the spectrum basis will affect stability of GNNs).

With respect to model (ii) of Prop. 3 we observe that now $\varepsilon$ represents the relative distance between $S$ and its perturbation $\hat{S}$ [cf. (29)], meaning that a fixed $\varepsilon$ represents the same level of perturbation for any possible reweighing of the difference $\alpha(S - \hat{S})$, $\alpha \in \mathbb{R}$. The value of $C$, in this case, represents the integral Lipschitz constant of the filters (cf. Def. 3). Integral Lipschitz filters, however, can be made arbitrarily selective near $\lambda \approx 0$, irrespective of the value of $C$, allowing for perfect discrimination of features around it, without affecting the overall stability. In integral Lipschitz filters, the value of $C$ determines the smallest eigenvalue for which the filter response becomes (approximately) flat, and hence loses discriminative power. A high value of $C$ would allow for greater selectivity in higher-eigenvalue frequencies, but at the expense of a deteriorated stability. With respect to $\delta$, the same analysis as for model (i) holds, except that in this case, the eigenvectors $U$ correspond to the relative error matrix $E$.

To overcome the degradation of the stability with the size of the graph, we propose model (iii) of Prop. 3. In this model, where $\varepsilon$ measures the relative perturbation distance and $C$ the integral Lipschitz constant, the family of admissible perturbations has been restricted to those that satisfy the structural constraint (32). Admissible perturbations are now those that are either dilations or contractions of the edge weights of $S$. Dilations and contractions can be different for different nodes but cannot be a mix of dilation and contraction in different parts of the graph. We remark that if the structural constraint is satisfied, then the stability can be controlled by determining the integral Lipschitz constant of the filters, for any graph. However, for some specific families of graphs, where we have information on how the eigenvectors change with a given perturbation size, we can improve on the result by relaxing the structural constraint. This is the case of (3), where extraneous geometric information (Euclidean space) is leveraged to quantify the impact of the perturbation (diffeomorphism) on the spectrum basis.

### V. Discussions

From the analysis of model (i) in Prop. 3 we concluded that Lipschitz filters are stable under absolute perturbations, but the stability presents a trade-off with the selectivity of the filters (the more stable the GNN is, the less selective are the filters that compose it). Moreover, we commented that the absolute perturbation model presents certain limitations, in that it does not take into account the particularities of the actual graph that is being perturbed.

Under a relative perturbation model, integral Lipschitz filters can be made arbitrarily selective near $\lambda \approx 0$ without sacrificing stability. Therefore, in order to discriminate among signals with frequency content in high values of $\lambda$ we need to spill the information into lower-eigenvalue frequencies, which is easily achieved by the mixing effect of the nonlinearities employed. This is illustrated in the following example that we use to shed light on the intricacies of the stability results put forward in Thm. 3 and Prop. 3.

Suppose that we have shift operators $S$ and $\hat{S}$ where the latter is a simple scaling of the former by a factor $(1 + \varepsilon)$

$$S = (1 + \varepsilon)S.$$  \hfill (39)

The graph dilation in (39) produces a graph in which all edges are scaled by a $(1 + \varepsilon)$ factor. This is a perturbation model of the form in (7) with $E = (\varepsilon/2)I$. We consider that $\varepsilon \approx 0$ in which case the graph dilation produces a minimal modification of the graph. Note that, for such a perturbation, we have $\delta = 0$ in model (ii) and it also satisfies the structural constraint (32) of model (iii), so that both models are applicable here.

Suppose now that we are given a set of filter coefficients $h$ and that we consider the filter $H(S)$ implemented on GSO $S$ vis-à-vis the filter $H(\hat{S})$ implemented on another GSO $\hat{S}$ [cf. (14)-(15)]. Given that the graph perturbation is inconsequential we would expect the filter differences to be inconsequential as well. Thm. 3 states that if the filters are integral Lipschitz this is true but if they are simply Lipschitz this need not be true.

![Figure 3. Stability of graph filters.](image-url)
Figure 4. High frequency feature extraction. We illustrate two sharp filters designed to successfully extract high frequency features located at $\lambda_{N-1}$ and $\lambda_N$. However, when the graph is slightly perturbed, which results in large changes in high frequency eigenvalues, the designed filters are no longer able to extract these features, now located at $\hat{\lambda}_{N-1}$ and $\hat{\lambda}_N$, since they have moved out of the narrow pass band of the sharp filter.

To understand why this happens we look at the differences between the spectra of $\mathbf{S}$ and $\hat{\mathbf{S}}$.

Given that $\mathbf{S}$ and $\hat{\mathbf{S}}$ are related by a scaling, they share the same set of eigenvectors and the scaling is translated to the eigenvalues. Thus, if $\mathbf{S} = \mathbf{V} \Lambda \mathbf{V}^H$ is the eigenvector decomposition of $\mathbf{S}$ [cf. (13)], the eigenvector decomposition of $\hat{\mathbf{S}}$ is

$$
\hat{\mathbf{S}} = \mathbf{V} [(1 + \varepsilon) \Lambda] \mathbf{V}^H. 
$$

(40)

As per (40), the eigenvalues of $\hat{\mathbf{S}}$ are the eigenvalues of $\mathbf{S}$ scaled by a factor $(1 + \varepsilon)$. Thus, the effect of the dilation in (39) on a filter with frequency response $h(\lambda)$ is that instead of instantiating the response at eigenvalues $\lambda_i$ we instantiate it at eigenvalues $(1 + \varepsilon)\lambda_i$. Consequently the response values that we expect to be $h(\lambda_i)$ if the filter is run on $\mathbf{S}$ actually turn out to be $h((1 + \varepsilon)\lambda_i)$ if the filter is run on $\hat{\mathbf{S}}$. This observation is the core argument in the proof of Thm. 3 and motivates the important observations that we discuss next.

**Graph perturbations and filter perturbations.** Fig. 3 illustrates the effect of the dilation in (39) on a Lipschitz (top) and integral Lipschitz filter (bottom). The difference in the positions between eigenvalues is given by $\hat{\lambda}_i - \lambda_i = \varepsilon \lambda_i$, and as such, depends on the value of the specific eigenvalue $\lambda_i$. For low-eigenvalue frequencies $\lambda_i$ the dilation results in a small perturbation of the eigenvalues. If the change in eigenvalues is small the change in the filter’s response from $h(\lambda_i)$ to $h(\hat{\lambda}_i)$ is small for both filters. For large eigenvalues the difference $\hat{\lambda}_i - \lambda_i = \varepsilon \lambda_i$ grows large. For Lipschitz filters a large difference in the arguments may translate into a large difference in the instantiated values of frequency responses $h(\lambda_i)$ and $h(\hat{\lambda}_i)$ since we can have

$$
|h(\hat{\lambda}_i) - h(\lambda_i)| \approx |\hat{\lambda}_i - \lambda_i| = \varepsilon \lambda_i. 
$$

(41)

This explains the filter’s instability. A small graph perturbation may result in a large filter perturbation at high-eigenvalue frequencies. For integral Lipschitz filters, on the other hand, changes in the frequency response must taper off as $\lambda$ grows. Thus, even though there may be a large variation in the eigenvalues the instances of the frequency responses are close since we must have

$$
|h(\hat{\lambda}_i) - h(\lambda_i)| \approx \frac{|\hat{\lambda}_i - \lambda_i|}{|\lambda_i + |\lambda_i|/2} = \frac{2\varepsilon}{2 - \varepsilon} \approx \varepsilon. 
$$

(42)

This explains the filter’s instability. No matter how large the eigenvalues are, a small perturbation of the graph results in a small perturbation of the graph filter. Thm. 3 shows that this is true for arbitrary relative perturbations.

**Graph perturbations and feature identification.** There is an obvious cost we pay for the stability of integral Lipschitz filters: they are unable to discriminate high-eigenvalue frequencies. The graph dilation example shows that this is not a limitation of the analysis. It is impossible to have a filter that is both stable and able to isolate high-eigenvalue features because small graph perturbations can result in large eigenvalue perturbations. This is a major drawback of linear graph filters in the extraction of features from graph signals. To illustrate this drawback suppose we have graph signals $\mathbf{x}_1 = \mathbf{v}_N$ and $\mathbf{x}_2 = \mathbf{v}_{N-1}$ and we want to design graph filters to discriminate between the two. The graph frequency domain representation of these two signals on the graph $\mathbf{S}$ are shown in Fig. 4. For us to discriminate between $\mathbf{x}_1 = \mathbf{v}_N$
and $x_2 = v_{N-1}$ we need filters centered at frequencies $\lambda_N$ and $\lambda_{N-1}$. These filters must have sharp transitions so that the filter isolating $x_1 = v_N$ does not let the signal $x_2 = v_{N-1}$ pass and, conversely, the filter isolating $x_2 = v_{N-1}$ does not let the signal $x_1 = v_N$. Yet, if these filters are sharp in high-eigenvalue frequencies, they will be unstable. More specifically, let $\lambda_N = (1+\varepsilon)\lambda_N$ be the eigenvalue associated to $x_1 = v_N$ in the perturbed graph, and $\lambda_{N-1} = (1+\varepsilon)\lambda_{N-1}$ be the one associated to $x_2 = v_{N-1}$. Now, since the filters were designed to be sharp around $\lambda_N$ and $\lambda_{N-1}$, but the perturbed eigenvalues $\lambda_N$ and $\lambda_{N-1}$ are far from these (at points where the filter response is virtually zero) the filter fails to adequately recover $x_1$ and $x_2$ in the perturbed graph. See Fig. [4] for an illustration of the instability effect at large eigenvalues.

**Pointwise nonlinearities.** So far, we have observed that stable filters require a flat response on high-eigenvalue frequencies, but that this inevitably prevents them from discriminating between features located at these frequencies. This illustrates an inherent, insurmountable limitation of linear information processing schemes. Neural networks introduce pointwise nonlinearities to the processing pipeline, as a computationally straightforward means of discriminating information located at large eigenvalues. The basic effect of these nonlinearities is to cause a spillage of information throughout the frequency band, see Fig. [5] This spillage of information into smaller eigenvalues allows for a stable filter to accurately discriminate between them, since information at these frequencies does not get severely affected by perturbations. However, since the energy in smaller eigenvalues is usually less than the energy still found at larger ones, and since it is also spread through a wide band of frequencies, the use of a bank of linear filters becomes a sensitive idea to better capture this spillage. Therefore, the use of banks of linear filters in combination with pointwise nonlinearities allows for information processing architectures that are able to capture high-eigenvalue frequency content in a stable fashion.

**Permutation Equivariance.** The permutation equivariance stated in Prop. [2] shows that the features that are learned by a GNN are independent of the labeling of the graph. But permutation equivariance is also important because it means that GNNs exploit internal signal symmetries as we illustrate in Fig. [6]. The graphs in Figs. [6a] and [6b] are the same, as indicated by the integer labels. The signals in Figs. [6a] and [6b] are different, as indicated by different colors. However, it is possible to permute the graph onto itself to make the signals match – rotate $180^\circ$ degrees and pull it inside out (Fig. [6c]). It then follows from Prop. [2] that the output of a GNN applied to the signal on the left (6a) is a corresponding permutation of the output of the same GNN applied to the signal on the right (6b). This is beneficial because we can learn to process the signal on (6a) from seeing examples of the signal on (6b). We note that, while most graphs do not exhibit perfect symmetries, they might have (sub)structures that are close to permutations. Therefore, stability shows the ability of GNNs to exploit these similarities.

**Eigenvalue Normalization.** The shift operator $S$ can be normalized so that all eigenvalues fall in a finite interval, i.e. $\lambda \in [0,1]$. We note that, for this case, the same conceptual analysis carries over. For model (i) in Prop. [3] we note that, since all eigenvalues are now crammed into $[0,1]$, the Lipschitz constant $C$ would need to be large to discriminate between contiguous frequency components, leading to an unstable architecture. For models (ii) and (iii), we see that stability is obtained by decreasing the integral Lipschitz constant, which causes the flat band to get closer to 0, and thus unable to discriminate frequencies located around the higher extreme of the interval.

VI. NUMERICAL EXPERIMENTS

In this last section, we illustrate the impact of the stability of GNNs in a movie recommendation problem [27], [28]. We use the MovieLens-100k dataset [29], which has a list of 100,000 ratings given by 943 users to some of the 1,582 movies available. The problem we are interested in is that of estimating the rating a user might give to some specific movie that they have not watched yet.

We consider a movie-based graph to support the data. Each movie is a node in the graph and we compute a similarity score between movies by taking into account the correlation among...
ratings given to each pair of movies by the same set of users, see [28] eq. (6) for details. We use this score as edge weights, and we build a graph consisting of the 40-nearest neighbors. To build the training and test sets, we consider all users that have rated the specific movie we are interested in, and we split them 90% for training and 10% for testing. Then, each user represents a graph signal, where the value at each node is the rating given to that movie. Movies not rated are given a value of 0. The value of the rating of the specific movie we are interested in is extracted as a label, and zeroed out in the graph signal. The objective is to use the ratings given by the user to other movies (nonzero elements in the graph signal) to estimate the rating they would give to the specific movie of interest. In particular, we focus on the movie Toy Story which has the largest amount of ratings, 452 in total.

We analyze the performance of three architectures: a bank of linear graph filters [28], and two one-layer GNNs (1)-(2), with filters that exhibit different values of integral Lipschitz constant $C$. In all cases, the filters are of length $K = 5$ and build $F = 32$ features. Then, we take the 32 features obtained at the node representing the movie of interest, and feed them to a $32 \times 5$ linear readout layer to obtain a one-hot vector representing the 5 possible ratings. The GNNs have a ReLU nonlinearity after the first graph convolutional layer. All the architectures are trained by minimizing the cross entropy loss over a set of 366 user ratings (graph signals), randomly split in batches of size 5, for 40 epochs, using an ADAM optimizer [30] with learning rate 0.005 and forgetting factors $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The evaluation measure is the root mean squared error (RMSE) between the estimated rating and the true rating. We run 10 different, random train/test dataset splits, and report the average results. The baseline obtained, together with the integral Lipschitz constants, when testing on the same graph that the architectures were trained, are as follows: (i) Linear, $1.12(\pm 0.22)$ ($C = 2.40$); (ii) GNN.1, $1.01(\pm 0.08)$ ($C = 2.91$); and (iii) GNN.2, $1.08(\pm 0.26)$ ($C = 0.04$). We note that the use of nonlinearities increases performance, since both GNNs outperform the linear graph filters, and we also observe a trade-off between the value of $C$ and the performance, since GNN.1 exhibits better results than GNN.2.

We are interested in analyzing the robustness of the trained architecture to changes in the underlying graph support. As a first case, we consider synthetic changes, where we generate diagonal matrices $E$ with each element being drawn uniformly at random from the interval $[-\varepsilon, -1 + \varepsilon]$ so that both $\|E\| \leq \varepsilon$ and the structural constraint (32) are satisfied. We then build matrix $\hat{S}$ according to (6). We vary $\varepsilon$ from $10^{-3}$ to 1, and for each train/test split and each $\varepsilon$, we run 10 different realizations of matrix $E$. Fig. 7a shows the relative RMSE difference as a function of the perturbation size $\varepsilon$. It becomes evident that GNN.2, which has the lowest value of $C$, and therefore is more stable, indeed has a lower variability of performance with increasing size of perturbation. Fig. 7b shows the output of the graph convolutional layer for each architecture. Again, GNN.2, which is the most stable architecture, exhibits the lowest difference in the output of the graph convolutional layer. It is interesting to note that, even though the linear graph filter has a slightly slower value of $C$ than GNN.1, the latter is actually more stable, which could be traced to the frequency mixing effect of nonlinearities. Also, we see that the RMSE difference for lower perturbation is virtually zero for all architectures (Fig. 7a) but the difference in the output of the GNNs is considerable (Fig. 7b). This could be explained by the effect of the readout layer. Finally, we consider perturbations that arise from different train/test partitions, which generate different underlying graph supports. In particular we consider different ratios of the train/test split, ranging from 30% to 90% of samples allocated to the training set. We observe from Fig. 7c that the relative difference in RMSE is quite noticeable for the linear graph filter bank and for GNN.1, whereas GNN.2 is quite stable. This shows that using filters with a smaller integral Lipschitz constant $C$ might lead to slightly worse performance for the particular case of the given graph, but is considerably more robust when considering perturbations of this underlying support.

**VII. Conclusions**

We focused on the impact that changes in the underlying topology have on the output of a GNN. First, we studied changes brought by permutations. We proved that GNNs are permutation equivariant, and that this implies that they effectively exploit the topological symmetries present in the underlying graph. Then, we discussed the absolute perturbation model existing in the literature, and proved that GNNs composed of Lipschitz filters are stable. However, not only
the absolute perturbation model ignores the particularities of the underlying graph, but also the stability comes at the expense of the discriminative power of the filters (i.e. the more stable, the less discriminative). We thus proposed a relative perturbation model and proved that filters used in GNNs need be integral Lipschitz for the resulting architecture to be stable. Integral Lipschitz filters can be made arbitrarily selective around low-eigenvalue frequencies, but need to have a flat response in high-eigenvalue frequencies, precluding accurate discrimination of information located in this band. We show that the frequency mixing effect of nonlinearities succeeds in spreading the information throughout the frequency spectrum, and thus allowing for accurate discrimination of information located at all frequencies. In essence, superior performance of GNNs can be explained by the fact that they are both stable and discriminative architectures, whereas linear graph filters can only satisfy one of these properties. We illustrated the discriminability and stability properties of both GNNs and graph filters in a movie recommendation problem.

APPENDIX A

PERMUTATION EQUIVALENCE OF GRAPH FILTERS

Proof of Prop. 7. A permutation matrix \( P \in \mathcal{P} \) is an orthogonal matrix, \( P^T P = P P^T = I \), from where it follows that powers \( \hat{S}^k \) of a permuted shift operator are permutations of the respective shift operator powers \( S^k \)

\[
S^k = (P^T S P)^k = P^T S^k P.
\]

Substituting this fact in the definition of the permuted graph filter \( H(S) \) in (13) yields

\[
H(\hat{S}) = \sum_k h_k (P^T S P)^k = P^T \left( \sum_k h_k S^k \right) P.
\]

In the last equality the sum is the filter \( H(S) = \sum_k h_k S^k \) as defined in (14). We can then write \( H(\hat{S}) = P^T H(S) P \) and use this fact to express application of the permuted filter \( H(\hat{S}) \) to the permuted signal \( \hat{x} = P^T x \) as

\[
\hat{z} = H(\hat{S}) \hat{x} = P^T H(S) P P^T x \tag{45}
\]

Since \( P \) is orthogonal, as already argued, we have that \( P P^T = I \). Substituting this into the right hand side of (45), the result in (17) follows.

APPENDIX B

STABILITY UNDER ABSOLUTE PERTURBATIONS

Lemma 1. Let \( S = V A V^H \) and \( E = U M U^H \) such that \( \|E\| \leq \varepsilon \). For any eigenvector \( v_i \) of \( S \) it holds that

\[
E v_i = m_i v_i + E_U v_i \tag{46}
\]

with \( \|E_U\| \leq \varepsilon \delta \), where \( \delta = (\|U - V\|^2 + 1)^2 - 1 \).

Proof. Start by writing the error matrix \( E \) as

\[
E = E_V + E_U \tag{47}
\]

\[
E_V = VMV^H \tag{48}
\]

\[
E_U = (U - V) M (U - V)^H + VM (U - V)^H + (U - V) MV^H \tag{49}
\]

We see that \( E_V v_i = m_i v_i \), since \( v_i \) is an eigenvector of \( E_V \). Next, note that, since \( \|E\| \leq \varepsilon \), then \( \|M\| \leq \varepsilon \), so that

\[
\|E_U\| \leq \|U - V\| \|M (U - V)^H\| + \|VM (U - V)^H\| + \|M\| \|U - V\| \|V\| \|M\|
\]

\[
\leq \varepsilon \|U - V\|^2 + 2 \varepsilon \|U - V\| \|V\| \|M\|
\]

\[
\leq \varepsilon (\|U - V\|^2 + 1)^2 - 1 = \varepsilon \delta
\]

which completes the proof.

Proof of Thm. 2. Without loss of generality, assume \( P_0 = I \) in (19) and write \( S = S + E \). Let us start by computing the first order expansion of \( (S + E)^k \)

\[
(S + E)^k = S^k + \sum_{r=0}^{k-1} S^r E S^{k-r-1} + C \tag{51}
\]

with \( C \) such that \( \|C\| \leq \sum_{r=2}^{k-1} \|E\|^r \|S\|^{k-r} \). Using this first-order approximation back in (13), we get

\[
H(\hat{S}) - H(S) = \sum_{k=0}^{\infty} h_k \sum_{r=0}^{k-1} S^r E S^{k-r-1} + D \tag{52}
\]

with \( D \) such that \( \|D\| = O(\|E\|^2) \) since the coefficients \( \{h_k\}_{k=0}^{\infty} \) of the filter \( H(S) \) are defined in terms of the power series expansion of the analytic function \( h \) which has bounded derivatives.

Next, consider an arbitrary graph signal \( x \) with finite energy \( \|x\| < \infty \) that has a GFT given by \( \hat{x} = [\hat{x}_1, \ldots, \hat{x}_N]^T \) so that \( x = \sum_{i=1}^N \hat{x}_i v_i \) for \( \{v_i\}_{i=1}^N \) the eigenvector basis of the GSO \( S \). Then, we can compute

\[
[H(\hat{S}) - H(S)] x = \sum_{i=1}^N \hat{x}_i D v_i \tag{53}
\]

\[
+ \sum_{i=1}^N \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} \hat{x}_i h_k S^r E S^{k-r-1} v_i.
\]

Let us focus on the second term of the sum in (53). It is immediate that \( S^{k-r-1} v_i = \lambda_i^{k-r-1} v_i \), so that

\[
\sum_{i=1}^N \hat{x}_i \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} h_k S^r E S^{k-r-1} v_i \tag{54}
\]

\[
= \sum_{i=1}^N \hat{x}_i \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} \lambda_i^{k-r-1} S^r E v_i
\]

Now, using Lemma 1 in (54) yields two terms

\[
\sum_{i=1}^N \hat{x}_i \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} \lambda_i^{k-r-1} S^r E v_i \tag{55}
\]

\[
= \sum_{i=1}^N \hat{x}_i \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} \lambda_i^{k-r-1} S^r m_i v_i \tag{56}
\]

\[
+ \sum_{i=1}^N \hat{x}_i \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} \lambda_i^{k-r-1} V A^r V^H E_U v_i. \tag{57}
\]
For $\lambda_i^{k-1}$, we note that $S^r v_i = \lambda_i^r v_i$, leading to the product $\lambda_i^{k-1} = \lambda_i^{k-1}$ being independent of $r$, so that

$$\sum_{i=1}^{N} \tilde{x}_i m_i \sum_{k=1}^{\infty} h_k \lambda_i^{k-1} v_i = \sum_{i=1}^{N} \tilde{x}_i m_i h'(\lambda_i) v_i$$

(58)

where $h'(\lambda_i) = \sum_{k=1}^{\infty} h_k \lambda_i^{k-1}$ is the derivative $h'(\lambda)$ of $h(\lambda)$ evaluated at $\lambda = \lambda_i$. In the case of (22), we note that

$$\sum_{i=1}^{N} \tilde{x}_i V \sum_{k=0}^{\infty} \sum_{r=0}^{k-1} \lambda_i^{-r-1} \lambda_j^{k-1} E_U v_i$$

(59)

$$= \sum_{i=1}^{N} \tilde{x}_i V \text{diag}(g_i) V^H E_U v_i$$

where $g_i \in \mathbb{R}^N$ is such that

$$[g_i]_j = \sum_{k=0}^{\infty} h_k \sum_{r=0}^{k-1} \lambda_i^{-r-1} \lambda_j^{k-1}.$$ (60)

For $j = i$ we have $[g_i]_i = h'(\lambda_i)$, while, for $j \neq i$, recall that $\sum_{r=0}^{k-1} \lambda_i^{-r-1} \lambda_j^{k-1} = (\lambda_j^{k} - \lambda_j^{k-1})/(\lambda_j - \lambda_i)$ so that

$$[g_i]_j = \begin{cases} h'(\lambda_i) & \text{if } j = i \\ \frac{h'(\lambda_i)}{h(\lambda_i) - h(\lambda_j)} \left( \frac{1}{\lambda_i - \lambda_j} \right) & \text{if } j \neq i \end{cases}$$ (61)

Note $\max_j \| [g_i]_j \| \leq C$ due to hypothesis (25), for all $i = 1, \ldots, N$.

Using (58) and (59) back in (53), and computing the norm, we get

$$\left\| \begin{bmatrix} H(S) - H(S) \end{bmatrix} x \right\| \leq \|D\tilde{x}\|$$ (62)

$$+ \left\| \sum_{i=1}^{N} \tilde{x}_i m_i h'(\lambda_i) v_i \right\|$$ (63)

$$+ \left\| \sum_{i=1}^{N} \tilde{x}_i V \text{diag}(g_i) V^H E_U v_i \right\|.$$ (64)

For (63) we have

$$\left\| \sum_{i=1}^{N} \tilde{x}_i m_i h'(\lambda_i) v_i \right\|^2 = \sum_{i=1}^{N} |\tilde{x}_i|^2 |m_i|^2 |h'(\lambda_i)|^2 \| v_i \|^2$$ (65)

since $\{v_i\}$ conform an orthonormal basis. Then, we recall that $\|v_i\|^2 = 1$ and, from hypothesis (26), we have $|m_i| \leq \varepsilon$ and from hypothesis (25), $|h'(\lambda_i)| \leq C$, so that

$$\left\| \sum_{i=1}^{N} \tilde{x}_i m_i h'(\lambda_i) v_i \right\|^2 \leq \varepsilon^2 C^2 \sum_{i=1}^{N} |\tilde{x}_i|^2.$$ (66)

Recalling that $\sum_{i=1}^{N} |\tilde{x}_i|^2 = \| \tilde{x} \|^2 = \| x \|^2$ and applying square root, we finally bound (63) by

$$\left\| \sum_{i=1}^{N} \tilde{x}_i m_i h'(\lambda_i) v_i \right\| \leq \varepsilon C \| x \|.$$ (67)

Now, moving on to (64) and using triangle inequality together with submultiplicativity of the operator norm, we have

$$\left\| \sum_{i=1}^{N} \tilde{x}_i V \text{diag}(g_i) V^H E_U v_i \right\|$$ (68)

$$\leq \sum_{i=1}^{N} |\tilde{x}_i| \left\| V \text{diag}(g_i) V^H \right\| \| E_U \| \| v_i \|.$$

We have $\| V \text{diag}(g_i) V^H \| \leq C$ for all $i = 1, \ldots, N$ from (61) in combination with hypothesis (25), and also $\| v_i \| = 1$. As for $\| E_U \|$, we know from Lemma 1 that $\| E_U \| \leq \varepsilon \delta$. Then (68) yields

$$\left\| \sum_{i=1}^{N} V \text{diag}(g_i) V^H E_U (\tilde{x}_i v_i) \right\| \leq C \varepsilon \delta \sqrt{N} \| x \|.$$ (69)

where we used the fact that $\sum_{i=1}^{N} |\tilde{x}_i| = \| \tilde{x} \| \leq \sqrt{N} \| x \|$.

Finally, for the second order term (62) stemming from the expansion of $S^k$, we obtain

$$\| D \tilde{x} \| \leq O(\| E \|^2) \| x \|_2 \leq O(\varepsilon^2) \| x \|_2.$$ (70)

Using bound (67) in (63) and bound (69) in (64), together with the bound (70) we just obtained for (62), we obtain

$$\left\| \begin{bmatrix} H(S) - H(S) \end{bmatrix} x \right\| \leq \varepsilon C \| x \| + \varepsilon C \delta \sqrt{N} \| x \| + O(\varepsilon^2) \| x \|.$$ (71)

We complete the proof by using that $|x| = 1$ as per Def. 4 and recalling that we have assumed that $I$ is the permutation that achieves the minimum norm of all $P \in \mathcal{P}$. $\square$

APPENDIX C
STABILITY UNDER RELATIVE PERTURBATIONS

Proof of Thm. 7. Mirroring the start of the proof of Thm. 2, we assume, without loss of generality, that $P^* = I$ solves (29). From a first order expansion analogous to (51), where we use $E S + S E$ instead of just $E$ as the second term, we obtain

$$H(\tilde{S}) - H(S)$$ (71)

$$\begin{aligned}
&= \sum_{k=0}^{\infty} h_k \sum_{r=0}^{k-1} (S^r E^H S^{k-r} + S^{k+1} E^H S^{k-r-1}) + D
\end{aligned}$$

with $D$ such that $\| D \| = O(\| E \|^2)$, in analogy to (52).

Next, we consider the difference in the effects of the filter on an arbitrary graph signal $x$ with finite energy $\| x \| < \infty$ that has a GFT given by $\bar{x} = [\tilde{x}_1, \ldots, \tilde{x}_N]^T$ so that $x = \sum_{i=1}^{N} \tilde{x}_i v_i$ for $\{v_i\}_{i=1}^{N}$ the eigenvector basis of the GSO $S$. Then, we can compute

$$\begin{aligned}
H(\tilde{S}) - H(S) x &= \sum_{i=1}^{N} \tilde{x}_i D v_i
\end{aligned}$$ (72)

$$+ \sum_{i=1}^{N} \tilde{x}_i \sum_{k=0}^{\infty} h_k \sum_{r=0}^{k-1} (S^r E^H S^{k-r} + S^{k+1} E^H S^{k-r-1}) v_i.$$ (73)

Let us consider first the product $S^{r+1} E^H S^{k-r-1} v_i$ in (72). It is immediate that $S^{k-r-1} v_i = \lambda_i^{k-r-1} v_i$, and, in combination with Lemma 1 we get

$$S^{r+1} E^H S^{k-r-1} v_i = \lambda_i^{k-r-1} S^{r+1} (m_i v_i + E_U v_i)$$

$$= \lambda_i^{k-r-1} m_i v_i + \lambda_i^{k-r-1} S^{r+1} E_U v_i.$$ (73)
Analogously, for the second product, we get $S'ES^{k-r}v_i = m_i \lambda_i^k v_i + \lambda_i^{k-r}S' E_U v_i$. Then, using these results, we can write

$$\sum_{i=1}^{N} \sum_{k=0}^{N} h_k \sum_{r=0}^{k-1} (S'ES^{k-r} + S'^{r+1}ES^{k-r-1})v_i$$

(74)

$$= 2 \sum_{i=1}^{N} \tilde{x}_i m_i \lambda_i h'(\lambda_i)v_i + \sum_{i=1}^{N} \tilde{x}_i V\text{diag}(g_i)V^H E_U v_i$$

where, for the first term, we gathered the two equal terms $m_i \lambda_i^k v_i$ and used the fact that $\sum_{k=0}^{N} \lambda_i \lambda_k = \lambda_i h'(\lambda_i)$; and for the second term, we defined $g_i \in \mathbb{R}^N$ as

$$[g_i]_j = \sum_{k=0}^{N} h_k \sum_{r=0}^{k-1} (\lambda_i^{k-r}[A^{r+1}] + \lambda_i^{k-r}[A^r])$$

(75)

Finally, we proceed to bound $\|(H(S) - H(S))x\|$ for the first term in (72), we simply have $\|Dx\| \leq O(\varepsilon^2)\|x\|$ by definition of operator norm and the error of the first order approximation (71). For the second term in (72) we need to bound the two terms in (74). The first of the terms in (74) is bounded analogously to (67), noting that, in this case, $|m_i| \leq \varepsilon$ by means of (30), and $|\lambda_i h'(\lambda_i)| \leq C$ due to (9). For the second term in (74), we proceed analogously to (69), where now $\|E_{\mathcal{G}}\| \leq \varepsilon$ and $\|V\text{diag}(g_i)V^H\| \leq 2C$, following the condition imposed by integral Lipschitz filters (9). All of these results together yield

$$\|(H(S) - H(S))x\| \leq 2C\varepsilon\|x\| + 2C\varepsilon \delta \sqrt{N}\|x\| + O(\varepsilon^2)\|x\|$$

We complete the proof by using that $\|x\| = 1$ as per Def. 2 and recalling that we have assumed that $I$ is the permutation that achieves the minimum norm of all $P \in \mathcal{P}$. \hfill \Box

Proof of Thm. 2 The proof is analogous to that of Thm. 3 with the following main difference. Denote by $m_i$, $i = 1, \ldots, N$, the eigenvalues of $E = \text{UMU}^H$. If we order these eigenvalues as $|m_1| \leq \cdots \leq |m_N|$, we know that $\|E\| = |m_N|$ and condition (32) becomes equivalent to $\|E/m_N - I\| \leq \varepsilon$. This can be used to write $Ev_i$, not as in Lemma 1 but as

$$Ev_i = \sum_{n=1}^{N} m_n u_n u_n^H v_i = m_N \sum_{n=1}^{N} (1 + \delta_n) u_n u_n^H v_i$$

(76)

where $m_n/m_N = 1 + \delta_n$ for all $n = 1, \ldots, N$ with $|\delta_n| \leq \varepsilon$ in virtue of (32), yielding

$$Ev_i = m_N v_i + m_N w_i \quad w_i = \sum_{n=1}^{N} \delta_n u_n u_n^H v_i$$

(77)

Using this expression in (74), it becomes

$$\sum_{i=1}^{N} \sum_{k=0}^{N} h_k \sum_{r=0}^{k-1} (S'ES^{k-r} + S'^{r+1}ES^{k-r-1})v_i$$

(78)

$$= 2m_N \sum_{i=1}^{N} \tilde{x}_i \lambda_i h'(\lambda_i)v_i + m_N \sum_{i=1}^{N} \tilde{x}_i V\text{diag}(g_i)V^H w_i.$$
Adding and subtracting $\hat{H}_L^g x_{L-1}^g$ from the terms in the sum, and using the triangular inequality once more, we get
\[
\left\| H_L^g x_{L-1}^g - \hat{H}_L^g x_{L-1}^g \right\| \leq \left\| \left( H_L^g - \hat{H}_L^g \right) x_{L-1}^g \right\| + \left\| \hat{H}_L^g (x_{L-1}^g - \hat{x}_{L-1}^g) \right\|.
\]
(85)

The definition of operator norm, implies that
\[
\left\| H_L^g x_{L-1}^g - \hat{H}_L^g x_{L-1}^g \right\| \leq \left\| H_L^g - \hat{H}_L^g \right\| \left\| x_{L-1}^g \right\| + \left\| \hat{H}_L^g \right\| \left\| x_{L-1}^g - \hat{x}_{L-1}^g \right\|.
\]
(86)

For the first term in the inequality (85), we can use the hypothesis that $\left\| H_L^g - \hat{H}_L^g \right\| \leq \Delta \varepsilon$ for all layers $l = 1, \ldots, L$, while for the second term, we can use that $\left\| \hat{H}_L^g \right\| \leq B = 1$ for all layers. Using these two facts in (86) and substituting back in (84), we get
\[
\left\| x_L^g - \hat{x}_L^g \right\| \leq C_{\sigma} \sum_{g=1}^{F_L-1} (\Delta \varepsilon \left\| x_{L-1}^g \right\| + B \left\| x_{L-1}^g - \hat{x}_{L-1}^g \right\|) .
\]
(87)

We observe that (87) shows a recursion, where the bound at layer $L$ depends on the bound at layer $L - 1$ as well as the norm of the features at layer $L - 1$, summed over all features. That is, for an arbitrary layer $l \in \{1, \ldots, L\}$, we have
\[
\left\| x_l^g - \hat{x}_l^g \right\| \leq C_{\sigma} \sum_{g=1}^{F_l-1} (\Delta \varepsilon \left\| x_{l-1}^g \right\| + B \left\| x_{l-1}^g - \hat{x}_{l-1}^g \right\|).
\]
(88)

with initial conditions given by the input features $x_0^g = x^g$ for $g = 1, \ldots, F_0$, so that $\left\| x_0^g - \hat{x}_0^g \right\| = \left\| x^g - \hat{x}^g \right\| = 0$. For the first step to solve the recursion (88), we compute the norm $\left\| x_L^g \right\|$.
\[
\left\| x_L^g \right\| = \left\| \sum_{g=1}^{F_l-1} H_l^g x_{l-1}^g \right\| \leq B \sum_{g=1}^{F_l-1} \left\| x_{l-1}^g \right\|.
\]
(89)

where we used the triangle inequality, followed by the bound on the filters. Solving recursion (89) with initial condition $\left\| x_0^g \right\| = \left\| x^g \right\|$ yields
\[
\left\| x_L^g \right\| \leq B L \prod_{l=1}^{L-1} F_l \sum_{g=1}^{F_0} \left\| x^g \right\|.
\]
(90)

Using (90) back in recursion (88) and solving it with the corresponding initial conditions, we get
\[
\left\| x_L^g - \hat{x}_L^g \right\| \leq \Delta \varepsilon B^{L-1} \left( \sum_{l=1}^{L} C_{\sigma} \prod_{l'=1}^{l-1} F_{l'} \right) \sum_{g=1}^{F_0} \left\| x^g \right\|.
\]
(91)

Evaluating (91) for $l = L$ and using it back in (87), we get that (82) yields
\[
\left\| \Phi(S, x) - \Phi(S, \hat{x}) \right\|^2 \leq \sum_{f=1}^{F_L} \left( \Delta \varepsilon B^{L-1} \sum_{l=1}^{L} C_{\sigma} \prod_{l'=1}^{l-1} F_{l'} \sum_{g=1}^{F_0} \left\| x^g \right\|^2 \right) .
\]
(92)

Noting that no term in the sum of (92) depends on $f$, and subsequently applying a square root, we get
\[
\left\| \Phi(S, x) - \Phi(S, \hat{x}) \right\| \leq \sqrt{F_L \Delta \varepsilon B^{L-1} \sum_{l=1}^{L} C_{\sigma} \prod_{l'=1}^{l-1} F_{l'} \sum_{g=1}^{F_0} \left\| x^g \right\|^2}.
\]
(93)

Finally, setting $F_L = F_0 = 1$ yields $\sqrt{F_L} = 1$ and $\sum_{g=1}^{F_0} \left\| x^g \right\| = \left\| x \right\|$, setting $F_1 = \cdots = F_l = F$, $B = 1$ and $C_{\sigma} = 1$ yields $B^{L-1} = 1$ and $\sum_{l=1}^{L} C_{\sigma} = L$, respectively. This completes the proof.

Proof of Thm. 7. The proof follows from Thm. 5 in combination with model (ii) of Prop. 3.
[18] D. Zou and G. Lerman, “Graph convolutional neural networks via scattering,” arXiv:1804.00099v2 [cs.IT], 18 Nov. 2018. [Online]. Available: [http://arxiv.org/abs/1804.00099](http://arxiv.org/abs/1804.00099)

[19] F. Gama, A. Ribeiro, and J. Bruna, “Diffusion scattering transforms on graphs,” in Int. Conf. Learning Representations 2019. New Orleans, LA: Assoc. Comput. Linguistics, 6-9 May 2019.

[20] R. R. Coifman and M. Maggioni, “Diffusion wavelets,” Appl. Comput. Harmonic Anal., vol. 21, no. 1, pp. 53–94, July 2006.

[21] B. Nadler, S. Lafon, I. Kevrekidis, and R. R. Coifman, “Diffusion maps, spectral clustering and eigenfunctions of Fokker-Planck operators,” in 19th Annu. Conf. Neural Inform. Process. Syst. Vancouver, BC: Neural Inform. Process. Syst. Foundation, 5-8 Dec. 2005, pp. 1–8.

[22] R. R. Coifman and S. Lafon, “Diffusion maps,” Appl. Comput. Harmonic Anal., vol. 21, no. 1, pp. 5–30, July 2006.

[23] R. Levie, E. Isufi, and G. Kutyniok, “On the transferability of spectral graph filters,” arXiv:1901.10524v1 [cs.LG], 29 Jan. 2019. [Online]. Available: [http://arxiv.org/abs/1901.10524](http://arxiv.org/abs/1901.10524)

[24] I. Daubechies, *Ten Lectures on Wavelets*, ser. CBMS-NSF Regional Conf. Series Appl. Math. Philadelphia, PA: SIAM, 1992, vol. 61.

[25] D. K. Hammond, P. Vandergheynst, and R. Gribonval, “Wavelets on graphs via spectral graph theory,” Appl. Comput. Harmonic Anal., vol. 30, no. 2, pp. 129–150, March 2011.

[26] D. I. Shuman, C. Wiesmeyr, N. Holighaus, and P. Vandergheynst, “Spectrum-adapted tight graph wavelet and vertex-frequency frames,” IEEE Trans. Signal Process., vol. 63, no. 16, pp. 4223–4235, Aug. 2015.

[27] L. Ruiz, F. Gama, A. G. Marques, and A. Ribeiro, “Invariance-preserving localized activation functions for graph neural networks,” arXiv:1903.12575v1 [eess.SP], 29 March 2019. [Online]. Available: [http://arxiv.org/abs/1903.12575](http://arxiv.org/abs/1903.12575)

[28] W. Huang, A. G. Marques, and A. Ribeiro, “Rating prediction via graph signal processing,” IEEE Trans. Signal Process., vol. 66, no. 19, pp. 5066–5081, Oct. 2018.

[29] F. M. Harper and J. A. Konstan, “The MovieLens datasets: History and context,” ACM Trans. Interactive Intell. Syst., vol. 5, no. 4, pp. 19:(1–19), Jan. 2016.

[30] D. P. Kingma and J. L. Ba, “ADAM: A method for stochastic optimization,” in Int. Conf. Learning Representations 2015. San Diego, CA: Assoc. Comput. Linguistics, 7-9 May 2015, pp. 1–15.