Robust Graph Neural Networks via Probabilistic Lipschitz Constraints

Raghu Arghal[*] RARGHAL@SEAS.UPENN.EDU
Eric Lei[*] ELEI@SEAS.UPENN.EDU
Shirin Saeedi Bidokhti SAEEDI@SEAS.UPENN.EDU
Dept. of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

Graph neural networks (GNNs) have recently been demonstrated to perform well on a variety of network-based tasks such as decentralized control and resource allocation, providing computationally efficient methods for tasks which have traditionally been challenging in that regard. However, like many neural-network based systems, GNNs are susceptible to shifts and perturbations on their inputs, which can include both node attributes and graph structure. In order to make them more useful for real-world applications, it is important to ensure their robustness post-deployment. Motivated by controlling the Lipschitz constant of GNN filters with respect to the node attributes, we propose to constrain the frequency response of the GNN's filter banks. We extend this formulation to the dynamic graph setting using a continuous frequency response constraint, and solve a relaxed variant of the problem via the scenario approach. This allows for the use of the same computationally efficient algorithm on sampled constraints, which provides PAC-style guarantees on the stability of the GNN using results in scenario optimization. We also highlight an important connection between this setup and GNN stability to graph perturbations, and provide experimental results which demonstrate the efficacy and breadth of our approach.

Keywords: graph neural networks, constrained optimization, robust learning

1. Introduction

Graph neural networks (GNNs) have proven to be a powerful method for network-based learning tasks, achieving state-of-the-art performance in many applications such as epidemic spread prediction (Kapoor et al. (2020); Ruiz et al. (2020b)), resource allocation (Gao et al. (2020)), and decentralized control (Tolstaya et al. (2019); Yang and Matni (2021)). The success of GNNs can be largely attributed to the graph convolution operation, which yields many desirable properties, such as permutation invariance and equivariance (Keriven and Peyré (2019)) and transferability (Ruiz et al. (2020a)) to graphs of varying size. However, like many other neural network models, GNNs have been shown to be particularly vulnerable to data shifts, perturbations, noise, and many other forms of attacks. This is of critical importance for control-based applications, where input data, such as sensor inputs or infection counts, can be inherently noisy.

An important property of GNNs that makes a distinction between GNN robustness and traditional neural network robustness is the fact that GNNs are models with two inputs: the graph signals (i.e. node attributes), and the graph adjacency matrix itself. Thus, there are three separate ways in

[*] Authors contributed equally

which a GNN can experience shifts on the data; either through (i) shifts of the graph signals they operate on (Zügner and Günnemann (2019)), (ii) shifts of the graph adjacency matrix (Bojchevski and Günnemann (2019); Tang et al. (2020); Gama et al. (2020); Cerviño et al. (2021); Dai et al. (2018)), or (iii) both (Zügner et al. (2018); Zhu et al. (2019)). Many of the aforementioned works enforce robustness to such attacks using variants of learning under distributional shifts (Biggio et al. (2013); Szegedy et al. (2013); Carlini and Wagner (2017); Madry et al. (2017); Hendrycks et al. (2019); Duchi and Namkoong (2018); Robey et al. (2020, 2021)) and applying them to GNN settings. In these works, one typically assumes some sort of model of how the data might be shifted (e.g. ℓ^∞ attacks) and aims to ensure robustness against attacks in line with these specified models.

In many safety-critical applications that use GNNs, however, it is likely that one does not know the sort of noise to be encountered after the GNN has been deployed. Therefore, it would be useful to have a method that is *agnostic* to the data shift model of the system. In this paper, we take an approach that follows Lipschitz-training methods in robust machine learning (Cisse et al. (2017); Fazlyab et al. (2019); Pauli et al. (2022); Zhao et al. (2021)), which enforce stability and robustness by constraining the Lipschitz constant of the neural network during training, and assume no knowledge of a data shift model. Under this paradigm for GNNs, we demonstrate how both forms of GNN stability (shifts on node attributes and shifts on graph structure) have an inherent connection to the frequency responses of the GNN filters, which are simply polynomials with coefficients as the filter coefficients. Therefore, both forms of robustness for GNNs can be achieved via a constraint on the graph filter frequency response.

In what follows, we first consider shifts of graph signals on a fixed graph, and motivate a frequency response constraint by demonstrating that the Lipschitz constant of a graph filter (w.r.t. graph signals) is given by the ℓ^∞ norm of the frequency response evaluated on the spectrum of the graph shift operator. This finite constrained problem is easily solvable via ℓ^∞ projection. To be universally stable to graph signal shifts across a class of graphs, we extend the discrete frequency response constraint to a continuous constraint. We propose a semi-infinite problem formulation to enforce the continuous constraints, which we relax to a chance-constrained problem for computational tractability. The fact that frequency response is given by polynomials on the filter weights provides inherent structure to the problem. Therefore, efficient methods can be used such as the scenario approach, which samples the constraints and provides sample complexity guarantees via VC theory. We then show how GNN stability with respect to shifts on the graph structure, which can be enforced by a constraint on the frequency response's derivative (Gama et al. (2020)), can also be easily performed using our framework. We provide experiments that demonstrate the efficacy of our approach, demonstrating stability to various noise distributions, as well as adversarial attacks, in several application settings.

2. Background

We approach graph neural networks from the graph signal processing point of view (Gama et al. (2019)). Specifically, we consider an n-node graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and use it to define linear, shift-invariant filters that operate on graph signals $x \in \mathbb{R}^n$, where the i-th entry x_i corresponds to the value of the i-th node. Let $S \in \mathbb{R}^{n \times n}$ denote a graph shift operator (e.g. adjacency or Laplacian of $\mathcal{G}$), which we assume to be symmetric (i.e. $\mathcal{G}$ is undirected). We can define a K-order graph filter as a vector of coefficients $h \in \mathbb{R}^K$. In order to filter some graph signal x with h with respect to the graph shift operator S, we use the graph convolution $h *_S x = \sum_{k=1}^{K} h_k S^{k-1} x$. Intuitively, the graph convolution takes shifted copies of x and weights them by h_k. Here, the "shift", which in linear time-invariant filters is simply a time delay, corresponds to aggregations of the $k-1$-th hop neighborhood $S^{k-1}x$.

Defining graph convolutions this way allows one to define the frequency response of filter h as $H(\lambda) = \sum_{k=1}^{K} h_k \lambda^{k-1}$ which is a $K-1$-degree polynomial with coefficients as the filter coefficients, evaluated at some frequency $\lambda \in \mathbb{R}$. If we let $S = VDV^H$ be the graph Fourier transform (eigendecomposition) of S, where $D = \text{diag}(\{\lambda_1, \dots, \lambda_n\})$ contains the spectrum of S, then the graph convolution can be written as $h *_S x = V\left(\sum_{k=1}^{K} h_k D^{k-1}\right) V^H x = VH(D)V^H x$, where $H(D) = \text{diag}(H(\lambda_1), \dots, H(\lambda_n))$ contains the frequency response of h evaluated on the eigenvalues of S. The frequency response (and the eigenvalues on which it is evaluated) is a key tool with many implications for GNN stability properties, as will be described in Sec. 3, 4.

A graph neural network (GNN) can then be built from graph filters and described as a cascade of Q layers, where each layer is given by a graph filter bank followed by a pointwise nonlinearity (Gama et al. (2019)). Concretely, at the q-th layer, let $\mathcal{H}^{(q)} \in \mathbb{R}^{G_{q-1} \times G_q \times K}$ denote the filter tensor, which is simply G_{q-1} filter banks containing G_q filters each, where each filter contains K taps. We define $G_0 = d$ to be the number of input feature dimensions of a sample of our data $X \in \mathbb{R}^{n \times d}$, which is a graph signal with d features. Then, the output of layer q is simply

$$X_q = \sigma\left(\sum_{k=1}^{K} S^{k-1} X_{q-1} \mathcal{H}_k^{(q)}\right) \tag{1}$$

where $\sigma(\cdot)$ is a pointwise nonlinearity, and $X_0 := X$ is the input to the GNN. We denote the final output after all Q layers on input X as $\Phi(X; S, \mathcal{H}^Q)$ where $\mathcal{H}^Q = \{\mathcal{H}^{(1)}, \dots, \mathcal{H}^{(Q)}\}$ is the collection of all filter tensors across all layers. These are the weights that parametrize the GNN.

2.1. Notions of GNN Stability

We now formalize the two notions of GNN stability corresponding to perturbations on the two inputs of a GNN: the underlying graph and the signal supported upon it. The first notion is provided in (Gama et al. (2020)).

Definition 1 (GNN Stability to Graph Perturbations) *A GNN Φ is C_1-stable to graph perturbations with respect to a set of node attributes $\mathcal{X}$ if*

$$\sup_{x \in \mathcal{X}} \|\Phi(x; S, \mathcal{H}) - \Phi(x; S', \mathcal{H})\| \leq C_1 d(S, S') \quad \forall S, S' \in \mathbb{S} \tag{2}$$

for distance d on graph shift operators.

Definition 2 (GNN Stability to Signal Perturbations) *A GNN Φ is C_2-stable to signal perturbations with respect to a set of graphs $\mathbb{S}$ if*

$$\sup_{S \in \mathbb{S}} ||\Phi(x; S, \mathcal{H}) - \Phi(x'; S, \mathcal{H})|| \leq C_2 ||x - x'|| \quad \forall x, x' \in \mathcal{X} \tag{3}$$

While (Gama et al. (2020)) provides conditions and bounds pertaining to the former, the latter has not been explored in depth. Moreover, the latter is of particular importance in control settings where knowledge of the graph (e.g. communication network, proximity, etc.) is well known, but graph signals often originate from sensors that may be noisy and/or miscalibrated. In Sec. 3, 4, we will first focus on the stability notion in Def. 2, and then connect back to Def. 1 in Sec. 5.

3. Problem Formulation

In this section, we motivate our approach, which is to constrain the frequency response of the GNN filters during training. We first discuss the case in which there is a fixed graph. Later, in Sec. 4, we generalize to the case where we may have set of graphs, which are applicable in time-varying problem settings.

3.1. Lipschitz Filters in the Graph Signal Domain

If we want to impose the property that signals which are close in some distance will result in similar graph filter outputs, i.e. that Def. 2 holds, we can try to restrict the Lipschitz constant of the filter, which is defined as

$$\mathrm{Lip}_p(h) \triangleq \sup_{x_1 \neq x_2 \in \mathbb{R}^n} \frac{\|h *_S x_1 - h *_S x_2\|_p}{\|x_1 - x_2\|_p} \tag{4}$$

with respect to the p-norm, $\|x\|_p = (\sum_{i=1}^n |x_i|^p)^{1/p}$. The following lemma establishes the connection between the Lipschitz constant of h and the frequency response of h, $H(\lambda)$.

Lemma 1 *Let $h \in \mathbb{R}^K$ be a vector of filter coefficients of length K. The Lipschitz constant of h, which is taken with respect to changes in input graph signals on some graph shift operator S, is given by $\mathrm{Lip}_p(h) = \max_{\lambda \in \Lambda(S)} |H(\lambda)|$, where $\Lambda(S)$ is the spectrum of S.*

Proof Recall that $h *_S x = VH(D)V^H$, where V are the eigenvectors of S, and $H(D)$ is a diagonal matrix containing the frequency response of h evaluated at $\Lambda(S) := \{\lambda_1, \ldots, \lambda_n\}$, which is the spectrum of S. Then

$$\begin{aligned} \|h *_S x_1 - h *_S x_2\|_p &= \|VH(D)V^H(x_1 - x_2)\|_p = \|H(D)V^H(x_1 - x_2)\|_p \\ &\leq \|H(D)V^H\|_p \|x_1 - x_2\|_p = \|H(D)\|_p \|x_1 - x_2\|_p \end{aligned} \tag{5}$$

Since this upper bound on $\|h *_S x_1 - h *_S x_2\|_p$ can be achieved when $x_1 - x_2$ is equal to the vector achieving the max in the induced p-norm of $H(D)V^H$, we have that $\mathrm{Lip}_p(h) = \|H(D)\|_p = \max_{i \in \{1,\ldots,n\}} |H(\lambda_i)|$, where the second equality holds since $H(D)$ is diagonal. ■

Thus, to enforce stability to graph signal perturbations, our objective is to *constrain the maximum absolute value of $H(\lambda)$*, where the max is taken over the eigenvalues of the graph shift operator S. When this is applied to each layer of a GNN, we arrive at the following statistical risk minimization problem, where $(X, y) \sim \mathcal{D}$ is the data distribution and ℓ is some loss function:

$$\begin{aligned} \underset{\mathcal{H}^{(1)},\ldots,\mathcal{H}^{(Q)}}{\text{minimize}} \quad & \mathbb{E}_{(X,y)\sim\mathcal{D}}[\ell(\Phi(X; S, \mathcal{H}^Q), y)] \\ \text{subject to} \quad & |H_{f,g}^{(q)}(\lambda)| \leq c \quad \forall q \in [Q], f \in [G_{q-1}], g \in [G_q], \forall \lambda \in \Lambda(S) \end{aligned} \tag{6}$$

where $\Lambda(S)$ refers to the spectrum of matrix S, and $H_{f,g}^{(q)}(\lambda)$ is the frequency response of filter g in the f-th bank of the q-th layer. We can simplify notation by defining

$$H^*(\lambda) \triangleq \max_{q \in [Q]} \max_{f \in [G_{q-1}]} \max_{g \in [G_q]} |H_{f,g}^{(q)}(\lambda)| \tag{7}$$

which is simply the max absolute frequency response of all filters in the GNN evaluated at λ. This leads us to the following constrained learning problem:

$$\begin{aligned} \underset{\mathcal{H}^{(1)},\ldots,\mathcal{H}^{(Q)}}{\text{minimize}} \quad & \mathbb{E}_{(X,y)\sim\mathcal{D}}[\ell(\Phi(X;S,\mathcal{H}^Q),y)] \\ \text{subject to} \quad & H^*(\lambda) \leq c \quad \forall \lambda \in \Lambda(S) \end{aligned} \tag{8}$$

Note (see Fig. 1) that the constraint in (8) requires constraining $H^*(\lambda)$ only on $\lambda \in \Lambda(S)$.

Remark 1 (Multiplicative Lipschitz Constant of the GNN) *Solving* (8) *allows one to guarantee a bound on the Lipschitz constant that is multiplicative in the number of layers of the GNN with respect to the input graph signals. This allows one to ensure stability in the sense of Def.* 2 *on a singleton graph, i.e. $\mathbb{S} = \{S\}$. Sec.* 4 *will generalize this to general graph sets $\mathbb{S}$. The constraint c can also differ across layers depending on the problem and desired outcome.*

3.2. Problem Realization for Static Graphs

To solve (8), we would like to write the constraint directly in terms of the GNN filter weights. To do so, let $\mathcal{V}_{\Lambda(S)}$ be the Vandermonde matrix evaluated on the values of $\Lambda(S)$, and truncated to the length of the filters (we assume $K < |\Lambda(S)|$), i.e.

$$\mathcal{V}_{\Lambda(S)} \triangleq \begin{bmatrix} 1 & \lambda_1 & \lambda_1^2 & \ldots & \lambda_1^{K-1} \\ 1 & \lambda_2 & \lambda_2^2 & \ldots & \lambda_2^{K-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \lambda_m & \lambda_m^2 & \ldots & \lambda_m^{K-1} \end{bmatrix} \tag{9}$$

The Lipschitz constant can be expressed directly in terms of the filter coefficients via the Vandermonde matrix: $\text{Lip}_p(h) = \max_{i\in\{1,\ldots,n\}} \left|\sum_{k=1}^{K} h_k \lambda_i^{k-1}\right| = \|\mathcal{V}_{\Lambda(S)} h\|_\infty$. Hence we simply need to constrain $\|\mathcal{V}_{\Lambda(S)} h\|_\infty \leq L$ in order to ensure L-Lipschitzness of filter h.

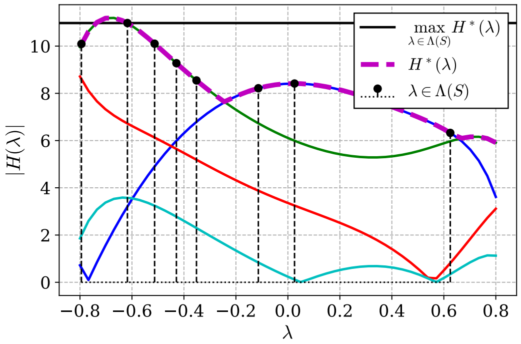


Figure 1: Various graph filter frequency responses and its maximum (7) evaluated on eigenvalues of a specific graph shift operator S.

This implies that in order to solve (8), we should constrain the the ∞-norm of $\mathcal{V}_{\Lambda(S)} h$ for each filter h in the GNN, and solve the following problem for a Q-layer GNN:

$$\begin{aligned} \underset{\mathcal{H}^{(1)},\ldots,\mathcal{H}^{(Q)}}{\text{minimize}} \quad & \underset{(x,y)\sim\mathcal{D}}{\mathbb{E}}[\ell(\Phi(x;S,\mathcal{H}^Q),y)] \\ \text{subject to} \quad & \|\mathcal{V}_{\Lambda(S)} \mathcal{H}^{(q)}_{(f,g,:)}\|_\infty \leq c \quad \forall q \in [Q], f \in [G_{q-1}], g \in [G_q] \end{aligned} \tag{10}$$

where $\mathcal{H}^{(q)}_{(f,g,:)} \in \mathbb{R}^K$ is filter g in the f-th bank in the q-th layer. This is now an optimization over the weight tensors with $\sum_{q=1}^{Q} G_q G_{q-1}$ constraints (G_0 is the number of input channels of X, and layer q has G_{q-1} inputs and G_q outputs). Assuming that the nonlinearities are 1-Lipschitz, the above optimization problem is equivalent to (8) and guarantees a bounded Lipschitz constant.

Algorithm 1: Discrete frequency response constraints via projected SGD

```
Input: Set of eigenvalues $\Lambda$, step size $\eta_t$, batch size $B$
Output: Trained weight tensors $\mathcal{H}^Q \in \mathbb{R}^{Q \times G_{q-1} \times K \times G_q}$
Randomly initialize GNN filter weights $\mathcal{H}^Q \in \mathbb{R}^{Q \times G_{q-1} \times K \times G_q}$
Generate Vandermonde matrix $\mathcal{V}_\Lambda$ evaluated on $\Lambda$
Compute the SVD $\mathcal{V}_\Lambda = U \Sigma V^\top$, where $\Sigma_{ii} = \sigma_i$
while not converged do
    Sample a batch $\{(X_i, y_i)\}_{i=1}^B \sim \mathcal{D}$
    $\mathcal{H}^Q \leftarrow \mathcal{H}^Q - \eta_t \frac{1}{B} \sum_{i=1}^B \nabla_{\mathcal{H}^Q} \ell(\Phi(X_i; S, \mathcal{H}^Q), y_i)$
    for $q = 1, \dots, Q$ do
        for each filter $h$ in $\mathcal{H}^{(q)}$ do
            Solve $h^{\text{proj}} = \text{Proj}_{\{h : \|\Sigma h\|_\infty \leq c\}}(V^\top h)$ using (12)
            $h \leftarrow V h^{\text{proj}}$
return $\mathcal{H}^Q$
```

3.3. Enforcing Frequency Response Constraints on a Finite Set of Eigenvalues

In order to solve (10), we would like to use projected gradient descent, which guarantees that the filters we learn satisfy the constraints. While primal-dual algorithms have been used before, e.g. in (Cerviño et al. (2021)), they do not guarantee the solution lies in the feasible set, due to the lack of strong duality. To solve the projection

$$\text{Proj}_{\{h : \|\mathcal{V}_{\Lambda(S)} h\|_\infty \leq c\}}(g) = \underset{h : \left\|\mathcal{V}_{\Lambda(S)} h\right\|_\infty \leq c}{\arg\min} \|h - g\|_2 \tag{11}$$

we may first change the basis to V where $\mathcal{V}_{\Lambda(S)} = U \Sigma V^\top$ is the SVD of $\mathcal{V}_{\Lambda(S)}$, and $\Sigma_{ii} = \sigma_i$, for $i \leq K$. Since the set $\{h \in \mathbb{R}^K : \|\Sigma h\|_\infty \leq c\} = \{h \in \mathbb{R}^K : |h_i| \leq \frac{c}{|\sigma_i|}, i \in [K]\}$ is a box, the solution in the original basis (11) is given by $V h^{\text{proj}}$, where

$$h_i^{\text{proj}} = [\text{Proj}_{\{h : \|\Sigma h\|_\infty \leq c\}}(V^\top g)]_i = \begin{cases} \text{sign}([V^\top g]_i) \frac{c}{|\sigma_i|} & |[V^\top g]_i| > \frac{c}{|\sigma_i|} \\ [V^\top g]_i & |[V^\top g]_i| \leq \frac{c}{|\sigma_i|} \end{cases} \tag{12}$$

The procedure in (12) is easily tensorized for each weight tensor $\mathcal{H}^{(q)}$, since $\mathcal{H}^{(q)}$ is just $G_{q-1} \times G_q$ filter vectors of length K. This yields the algorithm described in Alg. 1.

While (10) was formulated in the setting of learning problems involving only a single static graph S, we will see in Sec. 4 that this setup extends to the dynamic graph setting as well as methods enforcing GNN stability to graph perturbations, which we connect in Sec. 5. Therefore, Alg. 1 can be used as the workhorse of many Lipschitz-based learning methods for GNNs.

4. Extension to Dynamic Graphs

In control applications, it is common for the underlying network of interest to be changing in time. This is especially prevalent in time-series applications where the graph changes depending node locality at each time step. For instance, in decentralized control settings, agents may only share information with local neighbors on a dynamic graph (see Sec. 6.2). Thus, it is important that networks be stable to inputs on a broad set of possible graphs that they may encounter.

4.1. Formulating a Semi-Infinite Optimization Problem

More concretely, we wish now to extend the bound outlined in Sec. 3.1, which was with respect to a single graph, to the broader sense of stability defined in Def. 2, which is universally stable on some set of graphs $\mathbb{S}$. Under that definition, it follows that to guarantee input stability on a (potentially infinite) set of graphs $\mathbb{S}$, the same constraint must be applied to all $\lambda \in \Lambda(\mathbb{S}) \triangleq \bigcup_{S \in \mathbb{S}} \{\lambda \in \Lambda(S)\}$. Note that $\Lambda(\mathbb{S})$ is now a set function containing the union of eigenvalues of all graphs in $\mathbb{S}$. The meaning of $\Lambda(\cdot)$ should be understood depending on its argument.

To consider all graph shift operators $S \in \mathbb{S}$, it might be the case that the set of eigenvalues $\Lambda(\mathbb{S})$ might be very large, which might make (13) difficult or intractable; however, one can easily obtain simple bounds on $\Lambda(\mathbb{S})$. Indeed, by the Gershgorin circle theorem (Gershgorin (1931)), we have a guarantee that $\Lambda(\mathbb{S}) \subseteq [-n+1, n]$ for any $\mathbb{S} \subseteq \{0,1\}^{n \times n}$, i.e. $\mathbb{S}$ contains graph shift operators that represent adjacency matrices. In later sections, we will see that in real-world settings, we can constrain $\Lambda(\mathbb{S})$ to much smaller intervals than the one given via Gershgorin's theorem.

To enforce input stability on a GNN, we can solve the semi-infinite constrained problem:

$$\begin{aligned} \min_{\mathcal{H}_1, \ldots, \mathcal{H}_Q} \quad & \mathop{\mathbb{E}}_{(X,y)\sim\mathcal{D}} [\ell(\Phi(X; S, \mathcal{H}^Q), y)] \\ \text{s.t.} \quad & H^*(\lambda) \leq c \quad \forall \lambda \in \Lambda(\mathbb{S}) \end{aligned} \tag{13}$$

where $H^*(\lambda)$ is as defined in Sec. 3 and $\Lambda(\mathbb{S}) \subseteq [-n+1, n]$.

Note that, due to the reduction of Lipschitzness to a condition on the eigenvalues of the graph shift operator, we are able to easily extend our formulation to dynamic graph settings. Constraining our problem over eignevalues rather than families of graphs significantly reduces the dimensionality and computational difficulty of ensuring stability. This is also particularly salient in decentralized control applications where the underlying network can change dramatically, but eigenvalues can be reasonably bounded.

4.2. Scenario Optimization

Rather than constraining the worst-case eigenvalues, we follow the scenario approach introduced in Calafiore and Campi (2004). By relaxing the semi-infinite constraint to a chance constraint which is then sampled, we identify a solution that, with high probability, satisfies our constraints while maintaining computational tractability and sacrificing less performance.

Formally, we introduce a random variable $\lambda \in \Lambda(\mathbb{S}) \subseteq \mathbb{R}$ defined on a probability space $(\Lambda(\mathbb{S}), \mathcal{F}, \mathbb{P})$ and relax the semi-infinite constrained problem to a chance constrained problem (CCP):

$$\begin{aligned} \underset{\mathcal{H}_1, \ldots, \mathcal{H}_Q}{\text{minimize}} \quad & \mathop{\mathbb{E}}_{(X,y)\sim\mathcal{D}} [\ell(\Phi(X; S, \mathcal{H}^Q), y)] \\ \text{subject to} \quad & \mathbb{P}\left(\{\lambda \in \Lambda(\mathbb{S}) \big| \, |H^{(q)}_{(f,g,:)}(\lambda)| \leq c\} \right) \geq 1 - \epsilon \quad \forall q \in [Q], f \in [G_{q-1}], g \in [G_q] \end{aligned} \tag{14}$$

We then draw a sample of m eigenvalues $\bar{\sigma} := \{\bar{\lambda}_1, \ldots, \bar{\lambda}_m\}$ according to $\mathbb{P}$ on which to enforce the constraint to arrive at the following scenario program:

$$\begin{aligned} \underset{\mathcal{H}_1, \ldots, \mathcal{H}_Q}{\text{minimize}} \quad & \mathop{\mathbb{E}}_{(X,y)\sim\mathcal{D}} [\ell(\Phi(X; S, \mathcal{H}^Q), y)] \\ \text{subject to} \quad & |H^{(q)}_{(f,g,:)}(\lambda)| \leq c \quad \forall q \in [Q], f \in [G_{q-1}], g \in [G_q], \lambda \in \bar{\sigma} \end{aligned} \tag{15}$$

The following proposition, given by classical VC theory (Anthony and Biggs (1992)), provides a sample complexity guarantee on the generalization of the scenario approach to solve (14).

Proposition 1 (Sample Complexity Bound via VC Theory)
For any $\delta, \epsilon \in (0, 1)$ and $\bar{\sigma}$ drawn according to $\mathbb{P}^{\otimes m}$ such that

$$m \geq \left\lceil \frac{4}{\epsilon} \left(K \ln \left(\frac{12}{\epsilon} \right) + \ln \left(\frac{2}{\delta} \right) \right) \right\rceil \tag{16}$$

the solution of (15) satisfies $\mathbb{P}\left(\{\lambda \in \Lambda(\mathbb{S}) \mid H^{(q)}_{(f,g,:)}(\lambda) \leq c\}\right) \geq 1 - \epsilon$ with probability at least $1 - \delta$ for all $q \in [Q], f \in [G_{q-1}], g \in [G_q]$.

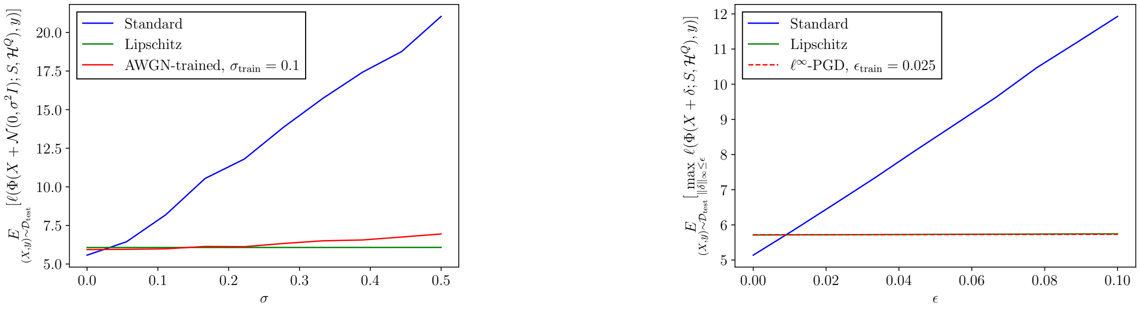
Proof First note that the constraint on the absolute value of $H(\lambda)$ can be broken into two constraints, each on a polynomial of λ of the same degree as in the original constraint. Next, observe that each of the $2\sum_{q=1}^{Q} G_{q-1}G_q$ constraints is described by a $(K-1)$-degree polynomial in λ. Thus, the family of functions describing the constraint has VC dimension K. The result then follows from Theorem 8.4.1 in (Anthony and Biggs (1992)) . ■

This result implies that in order to solve (14), we can simply sample a (large enough) number of eigenvalues according to $\mathbb{P}$, enforce the frequency response constraints on those eigenvalues, and guarantee that there is at least a $1 - \epsilon$ fraction of constraints in $\Lambda(\mathbb{S})$ that are satisfied, with high probability. Thus, since this procedure has a finite number of constraints, we can again use Alg. 1 to solve problems in the dynamic graph setting, where the set of eigenvalues Λ is no longer those contained in the spectrum of a particular graph shift operator, but rather the random draw of eigenvalues $\bar{\sigma}$. An appealing property of this approach is that since our Lipschitz constraints have reduced to enforcing a constraint on a single-variable polynomial (the frequency response), the sample complexity is linear in the number of filter taps, which is generally small. Note that one need not know the distribution $\mathbb{P}$ to follow this procedure. Simply being able to sample eigenvalues on which to enforce constraints is sufficient and can be done using the networks in training data.

5. Connections to GNN Stability Under Graph Shifts

The ability to enforce Lipschitzness of a GNN through conditions on its frequency response on specific eigenvalues also engenders connections between the various notions of stability. In (Gama et al. (2020)), the authors investigate conditions on which a GNN is stable with respect to graph shifts, i.e. the stability notion given in Def. 1. They show that one can guarantee stability under relative graph shifts if the network's filters are *integral Lipschitz* and the distance on graph shift operators d is taken to be operator distance modulo permutations, i.e. $d(S, S') = \min_{P \in \mathcal{P}} \|SP^\top - P^\top S'\|$, where $\mathcal{P}$ is the set of permutation matrices.

The integral Lipschitz condition implies that one simply needs to ensure that the function $\lambda \mapsto \lambda \frac{dH(\lambda)}{d\lambda}$ is bounded. Since $H(\lambda)$ is a $K-1$ degree polynomial, $\lambda \frac{dH(\lambda)}{d\lambda} = \sum_{k=1}^{K} h_k(k-1)\lambda^{k-1}$ is also a $K-1$ degree polynomial. If we define $h' = [0, h_1, 2h_2, \ldots, (K-1)h_{K-1}]^\top$, we can use the exact same setup in the previous two sections to enforce this constraint on the modified filter coefficients h'. Moreover, this connection implies that if one enforces stability to graph shifts, one can obtain a bound on the stability under graph signal shifts, and vice versa.



(a) (b)

Figure 2: Source localization. In (a), AWGN. In (b), ℓ^∞ adversary.

6. Experimental Results

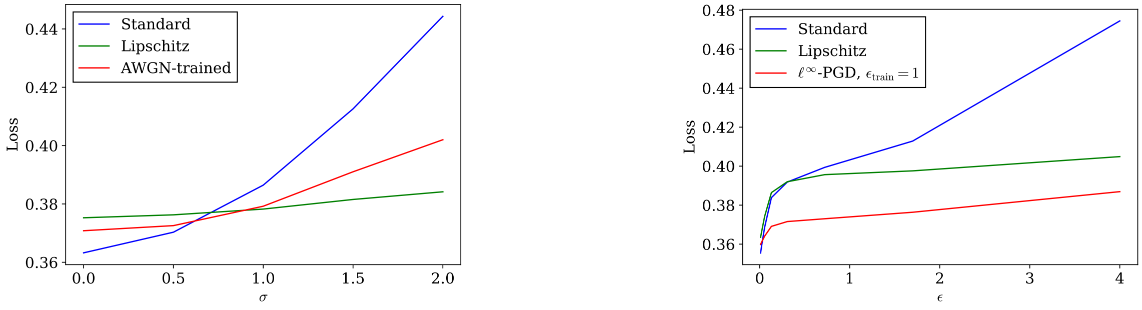
6.1. Static Graphs

In the static graph setting, we evaluate source localization, where there is a static graph shift operator S, and some information is allowed to propagate along the graph. Given the time series of propagated information, the GNN's task is to predict the original information source at a previous time step. We train a GNN with no constraints, and one with frequency response constraints via Alg. 1 ('Lipschitz'). To evaluate these models, we apply two types of noise at test time: Gaussian noise, and adversarial noise. For the former, if X, y is our input graph signal and target, then we input $X_\sigma^{\text{AWGN}} = X + \mathcal{N}(0, \sigma I)$ to the GNN. For the latter, we use $X_\epsilon^{\text{adv}} = \arg\max_{X' : \|X' - X\|_\infty \leq \epsilon} \ell(\Phi(X'; S, \mathcal{H}^Q), y)$. For both noises, we also compare with a GNN trained against each noise model (i.e. AWGN data augmentation and ℓ^∞-PGD training).

Shown in Fig. 2, the performance of the standard GNN degrades quickly with increasing noise on the input signal. Data augmentation is able to slow this degradation and improve performance overall, but it too struggles (particularly in the AWGN case) once the power of the noise applied during evaluation surpasses that of its noisy training data. The Lipschitz GNN, however, maintains high performance with increasing noise while sacrificing very little performance on clean data. In the adversarial case, our method performs as well as the defense designed against the adversary, without any knowledge of the adversary. This corroborates the analytical results and shows that frequency response constraints are an effective method of ensuring robustness to input signal noise while being *agnostic* to the data shift model.

6.2. Dynamic Graphs

In the dynamic graph setting, we use an example involving agents flocking together in a decentralized manner (Gama et al. (2021)). In this example, there are agents that seek to move in the same direction at some velocity without hitting each other. If each agent is aware of all the other agents' positions and velocities at each time step, then each agent can apply the optimal *centralized* policy. However, in practice, these agents are constrained by communication, and cannot ascertain information of far-away agents instantaneously. The objective is to learn a decentralized, communication-constrained policy that mimics the optimal centralized policy, using a GNN that respects the communication constraints. Due to the movement of the agents, the graphs change at



(*a*) (*b*)

Figure 4: Decentralized control of flocking agents. Loss is measured as the error between the GNN's policy and the optimal flocking policy, averaged over the horizon. In (*a*), evaluate on AWGN-perturbed data. In (*b*), ℓ^∞ adversarial noise.

each time step. Furthermore, the agents' sensor inputs of neighboring positions and velocities may be noisy, which we hope to combat using our stability framework over dynamic graphs (Sec. 4).

In order to apply the scenario approach, we need to ascertain $\Lambda(\mathbb{S})$, where $\mathbb{S}$ contains all communication-constrained graphs of the flocking agents. In practice, we set $\Lambda(\mathbb{S}) = [a, b]$, where a and b are the min and max of the all the eigenvalues of all the graph shift operators in the training set, and sample m constraints according to $\mathbb{P} = \text{Unif}([a, b])$. In the flocking example, we set $[a, b] = [-0.75, 1.25]$, and $m = 1000$. We enforce the constraints on these points using Alg. 1, and evaluate on the same setup as in the previous section, where both types of noise are added to the GNN controller's inputs for each agent. As shown in Fig. 4, we see that the Lipschitz constrained GNN is again able to maintain stability in a model-agnostic fashion and outperform AWGN data augmentation when evaluated on noise that differs from its training set, and perform nearly as well as ℓ^∞-PGD when evaluated on ℓ^∞ adversarial attacks. In Fig. 3, enforcing the frequency response constraints to each of the filters in the GNN via scenario approach does generalize to the constraints over the continuous range $[a, b]$ in practice, while data augmentation does not provide any additional constraints.

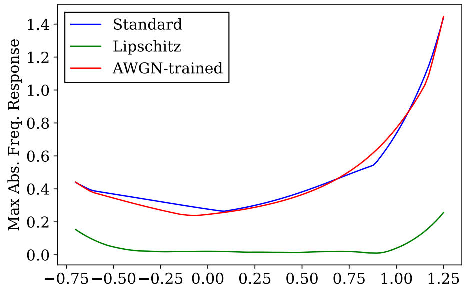


Figure 3: Max absolute frequency response $H^*(\lambda)$ (7) over filters of GNNs.

7. Conclusion

In this paper, we propose a simple constrained optimization framework for enforcing stability to GNNs by controlling the Lipschitz constants. We show that this framework encompasses several different notions of GNN stability, and how scenario optimization allows for efficient computation with reasonable PAC-style guarantees. Experiments on noisy networked control settings demonstrate the efficacy of our approach.

Acknowledgments

The work of Eric Lei is supported by a NSF Graduate Research Fellowship. The work of Raghu Arghal and Shirin Saeedi Bidokhti is supported by NSF CAREER Award 2047482 and NSF Grant 1910594. We thank Hamed Hassani, Alejandro Ribeiro, and Alexander Robey for helpful discussions.

References

Martin Anthony and Norman L. Biggs. *Computational learning theory: An introduction*. Cambridge Univ. Press, 1992.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

Aleksandar Bojchevski and Stephan Günnemann. Adversarial attacks on node embeddings via graph poisoning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 695–704. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/bojchevski19a.html.

Giuseppe Calafiore and M.C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, Feb 2004. doi: 10.1007/s10107-003-0499-y. URL http://dx.doi.org/10.1007/s10107-003-0499-y.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

Juan Cerviño, Luana Ruiz, and Alejandro Ribeiro. Training stable graph neural networks through constrained learning. *ArXiv*, abs/2110.03576, 2021.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 854–863. JMLR.org, 2017.

Hanjun Dai, Hui Li, Tian Tian, Xin Huang, L. Wang, Jun Zhu, and Le Song. Adversarial attack on graph structured data. In *ICML*, 2018.

John C. Duchi and Hongseok Namkoong. Learning models with uniform performance via distributionally robust optimization. *ArXiv*, abs/1810.08750, 2018.

Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George J Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Fernando Gama, Antonio G. Marques, Geert Leus, and Alejandro Ribeiro. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing*, 67(4):1034–1049, 2019. doi: 10.1109/TSP.2018.2887403.

Fernando Gama, Joan Bruna, and Alejandro Ribeiro. Stability properties of graph neural networks. *IEEE Transactions on Signal Processing*, 68:5680–5695, 2020. doi: 10.1109/TSP.2020.3026980.

Fernando Gama, Ekaterina V. Tolstaya, and Alejandro Ribeiro. Graph neural networks for decentralized controllers. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5260–5264, 2021.

Zhan Gao, Mark Eisen, and Alejandro Ribeiro. Resource allocation via graph neural networks in free space optical fronthaul networks. *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pages 1–6, 2020.

S. Gershgorin. Uber die abgrenzung der eigenwerte einer matrix. *Izvestija Akademii Nauk SSSR, Serija Matematika*, 7(3):749–754, 1931.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

Amol Kapoor, Xue Ben, Luyang Liu, Bryan Perozzi, Matt Barnes, Martin J. Blais, and Shawn O'Banion. Examining covid-19 forecasting using spatio-temporal graph neural networks. *ArXiv*, abs/2007.03113, 2020.

Nicolas Keriven and Gabriel Peyré. Universal invariant and equivariant graph neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/ea9268cb43f55d1d12380fb6ea5bf572-Paper.pdf.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Patricia Pauli, Anne Koch, Julian Berberich, Paul Kohler, and Frank Allgower. Training robust neural networks using lipschitz bounds. *IEEE Control Systems Letters*, 6:121–126, 2022. ISSN 2475-1456. doi: 10.1109/lcsys.2021.3050444. URL http://dx.doi.org/10.1109/LCSYS.2021.3050444.

Alexander Robey, H. Hassani, and George J. Pappas. Model-based robust deep learning. *ArXiv*, abs/2005.10247, 2020.

Alexander Robey, Luiz F. O. Chamon, George J. Pappas, Hamed Hassani, and Alejandro Ribeiro. Adversarial robustness with semi-infinite constrained learning. *ArXiv*, abs/2110.15767, 2021.

Luana Ruiz, Luiz Chamon, and Alejandro Ribeiro. Graphon neural networks and the transferability of graph neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1702–1712. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper/2020/file/12bcd658ef0a540cabc36cdf2b1046fd-Paper.pdf.

Luana Ruiz, Fernando Gama, and Alejandro Ribeiro. Gated graph recurrent neural networks. *IEEE Transactions on Signal Processing*, 68:6303–6318, 2020b. ISSN 1941-0476. doi: 10.1109/tsp.2020.3033962. URL http://dx.doi.org/10.1109/TSP.2020.3033962.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Xianfeng Tang, Yandong Li, Yiwei Sun, Huaxiu Yao, Prasenjit Mitra, and Suhang Wang. Transferring robustness for graph neural network against poisoning attacks. *Proceedings of the 13th International Conference on Web Search and Data Mining*, Jan 2020. doi: 10.1145/3336191.3371851. URL http://dx.doi.org/10.1145/3336191.3371851.

Ekaterina V. Tolstaya, Fernando Gama, James Paulos, George J. Pappas, Vijay R. Kumar, and Alejandro Ribeiro. Learning decentralized controllers for robot swarms with graph neural networks. In *CoRL*, 2019.

Feng Yang and N. Matni. Communication topology co-design in graph recurrent neural network based distributed control. *ArXiv*, abs/2104.13868, 2021.

Xin Zhao, Zeru Zhang, Zijie Zhang, Lingfei Wu, Jiayin Jin, Yang Zhou, Ruoming Jin, Dejing Dou, and Da Yan. Expressive 1-lipschitz neural networks for robust multiple graph learning against adversarial attacks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12719–12735. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/zhao21e.html.

Dingyuan Zhu, Ziwei Zhang, Peng Cui, and Wenwu Zhu. Robust graph convolutional networks against adversarial attacks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 1399–1407, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330851. URL https://doi.org/10.1145/3292500.3330851.

Daniel Zügner and Stephan Günnemann. Certifiable robustness and robust training for graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 246–256, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362016. doi: 10.1145/3292500.3330905. URL https://doi.org/10.1145/3292500.3330905.

Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Jul 2018. doi: 10.1145/3219819.3220078. URL http://dx.doi.org/10.1145/3219819.3220078.