Understanding the Effect of GCN Convolutions in Regression Tasks

Juntong Chen University of Twente
Johannes Schmidt-Hieber University of Twente
Claire Donnat University of Chicago
Olga Klopp ESSEC Business School

Abstract

Graph Convolutional Networks (GCNs) have become a pivotal method in machine learning for modeling functions over graphs. Despite their widespread success across various applications, their statistical properties (e.g., consistency, convergence rates) remain ill-characterized. To begin addressing this knowledge gap, we consider networks for which the graph structure implies that neighboring nodes exhibit similar signals and provide statistical theory for the impact of convolution operators. Focusing on estimators based solely on neighborhood aggregation, we examine how two common convolutions—the original GCN and GraphSAGE convolutions—affect the learning error as a function of the neighborhood topology and the number of convolutional layers. We explicitly characterize the bias-variance type trade-off incurred by GCNs as a function of the neighborhood size and identify specific graph topologies where convolution operators are less effective. Our theoretical findings are corroborated by synthetic experiments, and provide a start to a deeper quantitative understanding of convolutional effects in GCNs for offering rigorous guidelines for practitioners.

1 INTRODUCTION

Graph Convolutional Networks (GCNs) have become one of the preferred tools for modeling, analyzing, and predicting signals on graphs (Hamilton et al., 2017; Kipf and Welling, 2017). Despite their impressive success on academic benchmarks, several fundamental issues limit their broader applicability and reliability in real-world scenarios. In particular, while graphs are highly diverse in their properties (e.g., sparsity, degree distribution, node feature types) and the relationships they encode, GCNs are often suggested as a one-size-fits-all approach. This leaves practitioners with the challenge of selecting an appropriate convolution and architecture for their specific task. Without a clear understanding of the inductive biases these convolutions encode and their expected performance in relation to dataset properties, choosing among the sixty-six available convolution options in pytorch geometric can become a daunting task (Paszke et al., 2019). To bridge this knowledge gap, this paper examines the properties of graph convolutions and their relationship to the graph's neighborhood characteristics, with a particular focus on regression tasks over networks.

Statistical Setting. We consider the problem of network regression under fixed design. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$. Without loss of generality, we index the vertices from 1 to n, so $\mathcal{V} = \{1, \dots, n\}$ and $|\mathcal{V}| = n$. Assume that at each node $i \in \{1, \dots, n\}$, we observe a corresponding real-valued response variable Y_i, which is generated according to the following mechanism:

$$Y_i = f_i^* + \varepsilon_i, \tag{1}$$

where the noise variables ε_i are assumed to be uncorrelated, centered, and have variance equal to one. Let $\mathbf{Y} := (Y_1, \dots, Y_n)^\top \in \mathbb{R}^n$ denote the vector of responses from all nodes in the graph $\mathcal{G}$, $\boldsymbol{\varepsilon} := (\varepsilon_1, \dots, \varepsilon_n)^\top \in \mathbb{R}^n$ the noise vector, and $\mathbf{f}^* := (f_1^*, \dots, f_n^*)^\top \in \mathbb{R}^n$ the regression vector, which we also refer to as the signals. Throughout the paper, we assume that neighboring nodes in $\mathcal{G}$ admit similar signals: $f_i^* \approx f_j^*$, if $(i, j) \in \mathcal{E}$. Our goal is to recover $\mathbf{f}^*$ from the observations $\mathbf{Y}$.

To reconstruct $\mathbf{f}^*$, a possible solution is to minimize the least squares objective:

$$\hat{\mathbf{f}}_{\mathcal{F}} := \operatorname*{argmin}_{g \in \mathcal{F}} \left\| \mathbf{Y} - g(\mathbf{Y}) \right\|_2^2$$

with $\mathcal{F}$ being a prespecified function class. To incorporate information about the graph, graph-based reg-

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

ularization methods consider instead a penalized objective of the form:

$$\hat{\mathbf{f}}_\lambda := \underset{g\in\mathcal{F}}{\operatorname{argmin}} \left\|\mathbf{Y} - g(\mathbf{Y})\right\|_2^2 + \lambda \sum_{(i,j)\in\mathcal{E}} \operatorname{Pen}\big(g(\mathbf{Y})_i - g(\mathbf{Y})_j\big)$$

where Pen denotes a function penalizing differences along edges of the graph, and $g(\mathbf{Y})_i$ represents the i-th component of $g(\mathbf{Y}) \in \mathbb{R}^n$.

When the graph is endowed with a node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times p}$, and the estimator $\hat{\mathbf{f}}$ is chosen to be a function of these features (e.g. $\hat{\mathbf{f}} = \hat{\mathbf{f}}(\mathbf{X})$), the task is usually known as *node regression*. When the estimator is constructed using information solely from the observed values of $\mathbf{Y} \in \mathbb{R}^n$, the problem boils down to a denoising task usually known as *graph-trend filtering* (Hütter and Rigollet, 2016; Wang et al., 2016), an extension of the classical Gaussian sequence model (Rigollet, 2015; Wasserman, 2006) to accommodate graph structure. In this paper, we aim to study the performance of graph convolutions in providing reliable estimates for $\mathbf{f}^*$ under the denoising setting. Specifically, the feature X_i associated with each node i is simply given by $X_i = i$.

Contributions. The main contributions of this paper are summarized as follows:

- We establish a bias-variance type trade-off for the resulting denoising estimator in terms of the number of convolutional layers. This trade-off highlights how increasing the number of layers can reduce variance while potentially introducing greater bias, thereby affecting the overall estimation accuracy.
- By showing that the variance of the GCN is a weighted sum over paths, we relate the variance of GCNs to that of the uniformly local averaging estimator and introduce the novel walk analysis approach to investigate the problem. This method may pave the way for further statistical studies of GCNs.
- Based on the proposed walk analysis, we further quantitatively demonstrate the different variance decay behaviors under distinct local topologies of the graph. Specifically, we show that when high edge degree nodes are connected to low edge degree nodes this can dominate the variance and result in much slower overall variance decay.
- Additionally, we provide a concrete example of the graph structure to illustrate the over-smoothing phenomenon, which serves as a prototype to explain why simply stacking layers leads to poor performance in practice. Whereas previous work has provided evidence of this phenomenon for asymptotically large graphs and classification settings (Oono and Suzuki, 2020; Cai and Wang, 2020), we identified a non-asymptotic regression setting.

2 RELATED WORK

The theory for node regression has been derived from different viewpoints. Asymptotic viewpoint assumes that the number of vertices n tends to infinity, so that the Laplacian matrix is used to approximate the Laplace-Beltrami operator on an underlying manifold (Belkin and Niyogi, 2003; Lafon, 2004; Singer, 2006) on which the function $\mathbf{f}^*$ is assumed to be smooth (Hein, 2006; Von Luxburg et al., 2008). Instead of assuming that the number of vertices n tends to infinity, one can alternatively work with a fixed graph where, for a randomly selected subset of nodes, the response vectors are masked/unobserved (Kipf and Welling, 2017; Donnat et al., 2024). A typical assumption consists in assuming that neighboring nodes will have the same response. In this setting, several strategies leveraging the graph structure to guide the inference can be deployed: (a) Laplacian regularization, (b) ℓ_1 regularization and (c) graph convolutions.

Laplacian regularization (also known as Laplacian smoothing (Smola and Kondor, 2003), a special instance of Tikhonov regularization) is perhaps one of the better studied version of the problem. As described in a previous paragraph, these methods usually consider a smooth function (where the smoothness of the function $\mathbf{f}^*$ is consequently defined via the eigenvalue decay of the graph Laplacian), and use the convergence of the graph Laplacian to the Laplace-Beltrami operator to establish the consistency and convergence rates of the proposed estimator (Hein, 2006; Von Luxburg et al., 2008). Several research groups have explored similar concepts. Belkin et al. (2004) provides bounds on the generalization error for Tikhonov regularization and interpolated regularization. Work by Kirichenko and van Zanten (2017) derives posterior contraction rates for $\mathbf{f}^*$ using a Bayesian formalism. The rates depend on the eigenvalue decay of the graph Laplacian and a smoothness index that is defined via a Sobolev-type space based on the graph Laplacian. More recently, work by Green and coauthors (Green et al., 2021, 2023) derives optimal convergence rates for Laplacian-based regularization in the fixed design setting.

Graph-trend filtering (also known as "graph total-variation" (Hütter and Rigollet, 2016) or "fused lasso" (Tibshirani et al., 2005; Tibshirani and Taylor, 2011)) estimators use an ℓ_1 penalty to guide the inference, so that estimators are derived as the solution of the

following optimization problem:

$$\hat{\mathbf{f}} \in \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^n} \|\mathbf{Y}-\mathbf{x}\|_2^2 + \lambda \sum_{(i,j)\in\mathcal{E}} |x_i - x_j|.$$

Recent work (e.g., see Donnat et al. (2024); Hütter and Rigollet (2016); Padilla et al. (2018)) has characterized the convergence rate of these estimators as a function of the underlying graph smoothness, defined by the amount of variation along the graph edges: $\sum_{(i,j)\in\mathcal{E}} |f_i^* - f_j^*|$. While this approach is attractive for approximating near piecewise-constant functions over graphs, the estimators are often more computationally intensive than Laplacian-based estimators.

Finally, *Graph convolutional networks* (Kipf and Welling, 2017) are a more recent addition to the set of methods. While typically used for classification, this method is increasingly used for node regression tasks. The theoretical understanding of GCN models remains quite limited. For example, Verma and Zhang (2019) analyzes the stability of single-layer GCN models and derives their generalization guarantees, showing that, in regression problems, the generalization gap between training and testing errors decreases at a sub-linear rate as the number of training samples increases. The work in Henaff et al. (2015) and Defferrard et al. (2016) demonstrates that GCNs can be interpreted as a simplification of spectral-type Graph Neural Networks (GNNs) that makes use of the graph Laplacian. In Li et al. (2018), it is shown that graph convolution is essentially a form of Laplacian smoothing. Vinas and Amini (2024) investigates the classification performance of GNNs with graph-polynomial features under a general contextual stochastic block model.

Contrary to earlier work, in this paper we aim to provide statistical insight into the regression setting. We analyze a widely used class of GNNs, namely GCNs, which we discuss in detail in the next section.

Notation. Let $A = (A_{ij})_{i,j\in\mathcal{V}} \in \mathbb{R}^{n\times n}$ denote the binary adjacency matrix, which encodes the edge structure of the graph $\mathcal{G}$ as follows:

$$A_{ij} := \begin{cases} 1, & \text{if } i \text{ and } j \text{ are connected by an edge,} \\ 0, & \text{otherwise.} \end{cases}$$

For any $k = 1, 2, \ldots$, let $\mathcal{N}^k(i)$ represent the nodes within k hops of node i. The neighborhood of node i is denoted by $\mathcal{N}(i)$, with $\mathcal{N}^1(i) = \mathcal{N}(i) \cup \{i\}$. For $i \in \mathcal{V}$, let $d_i := |\mathcal{N}(i)|$ be the edge degree of node i. The degree matrix D is the $n \times n$ diagonal matrix with diagonal entries $D_{ii} = d_i$, $i = 1, \ldots, n$. For a vector $\mathbf{v} = (v_1, \ldots, v_n) \in \mathbb{R}^n$, $\|\mathbf{v}\|_2^2 = \sum_{i=1}^n v_i^2$ denotes the squared Euclidean norm and the notation $(\mathbf{v})_i = v_i$ represents the i-th component of the vector. For a square matrix M, let $\|M\|_F$ denote the Frobenius norm and $\mathrm{Tr}(M)$ denote the trace.

3 GRAPH CONVOLUTIONAL NETWORKS

While GNNs is a broad term encompassing various network structures for graph learning, GCNs (Kipf and Welling, 2017) form a specific class of GNNs. GCNs take the response vector $\mathbf{Y} \in \mathbb{R}^n$ and the adjacency matrix $A \in \mathbb{R}^{n\times n}$ as the input. Let L be the number of GCN layers. For each $\ell \in \{0, \ldots, L-1\}$, the output of the layers can be defined recursively via

$$H_{\ell+1} = g(H_\ell, A),$$

where g is a non-linear function and $H_0 = \mathbf{Y}$. In layer ℓ, the network has performed a ℓ-fold composition of the map $g(\cdot, A)$ and we will set $g(\mathbf{Y}, A)^{\circ\ell} := H_\ell$. A GCN with L layers is then a function $\mathbf{Y} \mapsto g(\mathbf{Y}, A)^{\circ L}$.

Specific GCN models differ in the choice and parametrization of the function g. An important case is the layer-wise propagation rule with g of the form

$$\mathbf{y} \mapsto g_\sigma(\mathbf{y}, A) := \sigma(T\mathbf{y}W). \tag{2}$$

Here, σ denotes the component-wise application of the ReLU activation function, $\mathbf{y} \in \mathbb{R}^n$ represents a realization of the random vector $\mathbf{Y}$, and $W \in \mathbb{R}$ is a weight parameter. The convolutional operator T is defined as

$$T := \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}, \tag{3}$$

where $\tilde{A} := A + I_n$ is the adjacency matrix with added self-loops, and $\tilde{D}$ is the diagonal $n \times n$ matrix with entries $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij} = d_i + 1$.

The above definition can be extended to the case where, at each node, we observe not just one number but a feature vector of length p. In this case, $\mathbf{Y}$ becomes an $n \times p$ matrix, and W is a $p \times p$ matrix. For simplicity, we focus on the $p = 1$ case for the rest of the paper.

The formulation of (2) originates from Kipf and Welling (2017). Its motivation is to provide a first-order approximation for the localized spectral filters introduced in Defferrard et al. (2016) for signal processing, suggesting that GCNs should be applied in the fixed design framework.

In addition to the widely used propagation operator T, this paper also discusses an alternative normalization approach (known as the GraphSAGE convolution (Hamilton et al., 2017)), that replaces T by

$$S := \tilde{D}^{-1}\tilde{A}. \tag{4}$$

If the activation function σ in (2) is set to the identity function, then the GCN of depth L takes on a specific form

$$g_{\mathrm{id}}(\mathbf{y}, A)^{\circ L} = P^L \mathbf{y} W_{L-1} \dots W_0 \tag{5}$$

with $P = T$, or $P = S$, and $W_0, \dots, W_{L-1} \in \mathbb{R}$ are the weight parameters in each layer. Empirical studies demonstrate that, compared to nonlinear activation functions (2), linear GCNs as defined in (5) do not have a detrimental effect on accuracy in many downstream applications (Wu et al., 2019). In Oono and Suzuki (2020), the authors provide theoretical evidence that non-linearity does not enhance the expressive power of GCNs. Further discussion comparing linear GCNs to the original GCNs can be found in NT and Maehara (2019) and NT et al. (2021).

Rewriting (5) yields

$$g_{\mathrm{id}}(\mathbf{y}, A)^{\circ L} = P^L \mathbf{y} W \tag{6}$$

with some $W \in \mathbb{R}$. We denote the class of functions described in (6) as $\mathcal{F}_{\mathrm{GCN}}(L)$ and base our analysis later on this class.

The Theory of GCNs. Several studies have begun investigating the capabilities and limitations of Graph Neural Networks (GNNs), an umbrella term that includes Graph Convolutional Networks (GCNs) as an important example. Most analyses focus on comparing different classes of GNN architectures by using the Weisfeiler-Lemann test to distinguish between graph structures (Xu et al., 2018) or on performance in classification tasks (Garg et al., 2020). While issues such as over-smoothing in deeper GNNs have been explored (Li et al., 2018; Oono and Suzuki, 2020), there is a noticeable gap in evaluating the consistency of GNN-based estimators. In particular, little has been done to analyze the bias-variance trade-off concerning neighborhood size and topology. This gap highlights the need for a deeper understanding of how graph convolution operations affect the statistical properties of GCNs, motivating our investigation.

4 THEORETICAL RESULTS FOR DEEP LINEAR GCNS

Although many variants of GCNs include some form of regularization, the original design incorporates graph information within the layers and deliberately avoids penalization (Kipf and Welling, 2017) suggesting that minimizing the least-squares objective over the class of GCNs is sufficient:

$$\hat{\mathbf{f}}_L := \operatorname*{argmin}_{g \in \mathcal{F}_{\mathrm{GCN}}(L)} \left\| \mathbf{Y} - g(\mathbf{Y}) \right\|_2^2 = \widehat{W} P^L \mathbf{Y} \tag{7}$$

with

$$\widehat{W} = \frac{\mathbf{Y}^\top P^L \mathbf{Y}}{\|P^L \mathbf{Y}\|_2^2} = \frac{\mathbf{Y}^\top P^L \mathbf{Y}}{\mathbf{Y}^\top (P^L)^\top P^L \mathbf{Y}},$$

where the equality follows by minimizing over the GCN parameter W in (6). This estimator denoises Y_i by regressing it on itself and neighboring nodes. The size of the neighborhood is controlled by the depth L.

To understand the role of the parameter in the GCN, we disregard its randomness and focus on a fixed parameter $W \in \mathbb{R}$. For any matrix $P \in \mathbb{R}^{n \times n}$, using that the measurement noise ε_i in Model (1) is centered and the triangle inequality, we then obtain the bias-variance type decomposition

$$\begin{aligned}
&\mathbb{E}\left[\|WP\mathbf{Y} - \mathbf{f}^*\|_2^2\right] \\
&= \mathbb{E}\left[\|WP(\mathbf{f}^* + \boldsymbol{\varepsilon}) - \mathbf{f}^*\|_2^2\right] \\
&= \|(WP - I_n)\mathbf{f}^*\|_2^2 + W^2 \mathbb{E}\left[\|P\boldsymbol{\varepsilon}\|_2^2\right] \\
&\le \Big(|W| \cdot \left\|(P - I_n)\mathbf{f}^*\right\|_2 + |1 - W| \cdot \|\mathbf{f}^*\|_2\Big)^2 \\
&\quad + W^2 \mathbb{E}\left[\|P\boldsymbol{\varepsilon}\|_2^2\right].
\end{aligned} \tag{8}$$

Optimizing the parameter W balances the squared bias and variance. In the case of a small signal $\mathbf{f}^*$, the optimal W will be in $[0, 1)$, thereby inducing shrinkage. An extreme case occurs when $\mathbf{f}^* = \mathbf{0}$. Choosing $W = 0$ then leads to a vanishing mean squared error.

In nonparametric statistics, the convergence rate of an estimator depends on the smoothness of the regression function. In the context of graphs, this smoothness is naturally described in terms of the neighborhood structure. As mentioned, we consider graphs $\mathcal{G}$ with the property that neighboring nodes are expected to have similar signals, that is, $f_i^* \approx f_j^*$, whenever $(i, j) \in \mathcal{E}$. However, the specific similarity assumption concerning neighboring nodes depends on the graph aggregation operator P, which incorporates different normalization approaches. We present these formally as follows.

Assumption 1. *There exists a $\Delta > 0$ such that for any i and any $j \in \mathcal{N}(i)$,*

$$\left|f_i^* - f_j^*\right| \le \Delta, \quad \text{if } P = S, \tag{9}$$

and

$$\left|\sqrt{d_i + 1}\, f_i^* - \sqrt{d_j + 1}\, f_j^*\right| \le \Delta, \quad \text{if } P = T. \tag{10}$$

Recall that $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. The subsequent result bounds the mean squared error as the sum of the squared bias and variance.

Theorem 1. *If (9) holds, then*

$$\frac{1}{n}\mathbb{E}\left[\|WS^L\mathbf{Y} - \mathbf{f}^*\|_2^2\right] \le \mathrm{Bias}_S^2(L, W) + \underbrace{W^2 \frac{\|S^L\|_F^2}{n}}_{\text{variance}},$$

and if (10) *holds, then*

$$\frac{1}{n}\mathbb{E}\left[\|WT^L\mathbf{Y}-\mathbf{f}^*\|_2^2\right] \le \mathrm{Bias}_T^2(L,W) + \underbrace{W^2\frac{\|T^L\|_F^2}{n}}_{variance},$$

where

$$\mathrm{Bias}_S(L,W) = |W|\cdot L\Delta + \frac{|1-W|}{\sqrt{n}}\cdot\|\mathbf{f}^*\|_2,$$

$$\mathrm{Bias}_T(L,W) = |W|\cdot L\Delta\sqrt{\frac{2|\mathcal{E}|}{n}+1} + \frac{|1-W|}{\sqrt{n}}\cdot\|\mathbf{f}^*\|_2.$$

While the bias terms $\mathrm{Bias}_S(L,W)$ and $\mathrm{Bias}_T(L,W)$ grow with L, Section 4.1 demonstrates that the variance terms, specifically $\|S^L\|_F^2$ and $\|T^L\|_F^2$, can decrease rapidly as L increases. Overall, Theorem 1 indicates a bias-variance type trade-off mediated by the number of layers L. It implies the existence of an optimal value of L that balances these two terms, providing theoretical evidence for the over-smoothing phenomenon in the regression setting. We will further validate this theoretical finding through an experimental study in Section 5, see also Figure 1.

As shown in (7), GCNs with linear layers denoise the response variables by averaging over neighboring nodes. A natural competitor is the estimator that estimates the i-th signal f_i^* by uniformly averaging over the set $\mathcal{N}^L(i)$, which represents the neighboring nodes within a distance of at most L from node i. Specifically, the local average estimator of $\mathbf{f}^*$ is given by

$$\overline{\mathbf{f}}_L := \left(\overline{f}_{L,1},\dots,\overline{f}_{L,n}\right)^\top, \tag{11}$$

with

$$\overline{f}_{L,i} := \frac{1}{|\mathcal{N}^L(i)|}\sum_{j\in\mathcal{N}^L(i)} Y_j, \quad i=1,\dots,n.$$

The following result bounds the mean squared error of $\overline{\mathbf{f}}_L$ using a similar bias-variance type decomposition.

Theorem 2. *For $i=1,\dots,n$, the variance of each $\overline{f}_{L,i}$ is given by $\mathrm{Var}\left(\overline{f}_{L,i}\right) = |\mathcal{N}^L(i)|^{-1}$. If (9) holds, then*

$$\frac{1}{n}\mathbb{E}\left[\|\overline{\mathbf{f}}_L-\mathbf{f}^*\|_2^2\right] \le \underbrace{L^2\Delta^2}_{bias} + \underbrace{\frac{1}{n}\sum_{i=1}^n\frac{1}{|\mathcal{N}^L(i)|}}_{variance}.$$

By comparing the bounds in Theorem 1 and 2, under the same smoothness assumption, the bias term $\mathrm{Bias}_S(L,W)$ may be smaller than that of $\overline{\mathbf{f}}_L$, depending on the choice of W. We analyze the variance term in details in the next session.

4.1 Bounds for the Variance via Weighted Local Walks on the Graph

We examine in this section how the network depth L and the local network topology influence the variance decay for the aggregation operators S and T.

To this end, we introduce walk analysis as a novel approach. Recall that the definitions of T in (3) and S in (4) rely on the adjacency matrix with added self-loops, $\tilde{A}$. While the original graph is denoted by $\mathcal{G}$, we write $\mathcal{G}'$ for the graph augmented with self-loops. A path from node i to node j via nodes $\ell_1,\dots,\ell_{L-1}\in\{1,\dots,n\}$ is denoted by

$$(i\to\ell_1\to\ldots\to\ell_{L-1}\to j).$$

The length of the path is L. The weights associated with this path under the aggregation operator S are defined as

$$\omega_i(\ell_1,\dots,\ell_{L-1}) := \frac{1}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-1}}+1)}$$

and under T as

$$\widetilde{\omega}_{ij}(\ell_1,\dots,\ell_{L-1}) := \left(\frac{d_i+1}{d_j+1}\right)^{1/2}\omega_i(\ell_1,\dots,\ell_{L-1}),$$

where d_i is the edge degree of node i. Let $\mathcal{P}_L(i\to j)$ denote all paths from node i to node j of length L in the self-loop augmented graph $\mathcal{G}'$. Each path of length L in the augmented graph corresponds to a path of length $\le L$ in the original graph $\mathcal{G}$. The edge degrees and the length of the paths L decrease the weights but increase the number of paths. The key observation is that for any vector $\mathbf{v}=(v_1,\dots,v_n)^\top$,

$$\left(S^L\mathbf{v}\right)_i = \sum_{j=1}^n\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} v_j\omega_i\left(\ell_1,\dots,\ell_{L-1}\right)$$

and

$$\left(T^L\mathbf{v}\right)_i = \sum_{j=1}^n\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} v_j\widetilde{\omega}_{ij}\left(\ell_1,\dots,\ell_{L-1}\right),$$

with a detailed derivation provided in the appendix.

Since the noise vector $\boldsymbol{\varepsilon}=(\varepsilon_1,\dots,\varepsilon_n)^\top$ consists of centered and uncorrelated random variables with variance one, we find

$$\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] = \sum_{j=1}^n\left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)}\omega_i(\ell_1,\dots,\ell_{L-1})\right]^2$$

and

$$\mathbb{E}\left[\left(T^L\boldsymbol{\varepsilon}\right)_i^2\right] = \sum_{j=1}^n\left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)}\widetilde{\omega}_{ij}(\ell_1,\dots,\ell_{L-1})\right]^2.$$

If the components of $\boldsymbol{\varepsilon}$ are uncorrelated and have variance one, then for $P \in \{S, T\}$, we have

$$\|P^L\|_F^2 = \mathbb{E}\left[\|P^L\boldsymbol{\varepsilon}\|_2^2\right] = \sum_{i=1}^{n} \mathbb{E}\left[\left(P^L\boldsymbol{\varepsilon}\right)_i^2\right].$$

The quantity $\mathbb{E}[(P^L\boldsymbol{\varepsilon})_i^2]$ thus represents the contribution of node i to the variance term in Theorem 1 after L applications of the propagation operator P.

We first compare $\mathbb{E}[(S^L\boldsymbol{\varepsilon})_i^2]$ with the variance $\mathrm{Var}(\overline{f}_{L,i})$ of the uniform average estimator $\overline{f}_{L,i}$ defined in (11). Since the row sums of the matrix $S = \tilde{D}^{-1}\tilde{A}$ are all one, S is a transition matrix of a Markov chain. Thus, S^L is a transition matrix with row sums equal to one and for any node $i \in \{1, \ldots, n\}$,

$$\sum_{j=1}^{n} \sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} \omega_i(\ell_1, \ldots, \ell_{L-1}) = 1.$$

In fact, in the previous display, $\sum_{j=1}^n$ can be replaced by $\sum_{j\in\mathcal{N}^L(i)}$. Applying the Cauchy-Schwarz inequality (AM-QM) and Theorem 2, we deduce

$$\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] \geq \frac{1}{|\mathcal{N}^L(i)|} = \mathrm{Var}\left(\overline{f}_{L,i}\right). \tag{12}$$

For $L = 1$, equality holds in (12). The inequality shows that, in general, the local averaging estimator $\overline{\mathbf{f}}_L$ has smaller variance if we take the same L in the GCN estimator and the local averaging estimator. On the contrary, the GCN estimator is expected to have a smaller bias, as it computes a weighted sum over the local neighborhood $\mathcal{N}^L(i)$, assigning more mass to nodes that are closer to i in path distance. Although the gain in the bias is hard to quantify theoretically, we see that it outweighs the loss in the variance in our empirical studies. For instance, in all the scenarios in Figure 4 the GCN estimator has smaller test error.

In the remainder of this section, we study the decay rate of $\mathbb{E}[(S^L\boldsymbol{\varepsilon})_i^2]$ with respect to the depth L. Notably, the decay can vary significantly depending on the local graph structure around node i. Since S and T behave similarly, we focus on the results for $\mathbb{E}[(S^L\boldsymbol{\varepsilon})_i^2]$ and delay the analysis of $\mathbb{E}[(T^L\boldsymbol{\varepsilon})_i^2]$ to the appendix.

We first provide an upper bound assuming that node i is the root of a locally-rooted tree. In this scenario, the variance term decays exponentially in the number of layers L.

Proposition 1. *Assume the local graph around node i is a rooted tree, in the sense that the subgraph induced by the vertices $\mathcal{N}^L(i)$ forms a rooted tree with root i and all nodes having edge degree d. Then, we have*

$$\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] \leq 4(d+1)^{-L}(L+1)3^{2L}.$$

In this case, the variance decay in d is $(d+1)^{-L}$. This is the sharp rate, since the lower bound in (12) gives the same rate, namely $1/|\mathcal{N}^L(i)| = (d-1)/(d^{L+1}-1) \approx d^{-L}$ for large d.

A slow rate of decay for the variance occurs if node i has edge degree d but is connected to nodes with small edge degrees.

Proposition 2. *If there exists a path from node i with degree d to node j via nodes $\ell_1, \ldots, \ell_{L-1} \in \{1, \ldots, n\}$ such that $d_{\ell_1}, \ldots, d_{\ell_{L-1}} \leq 3$, then*

$$\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] \geq (d+1)^{-2}4^{2-2L}.$$

This shows that for large d and large L, the variance decay is necessarily much slower than the rate $(d+1)^{-L}$ obtained in Proposition 1.

The next result shows that adding a cycle to node i can immediately slow variance decay.

Proposition 3. *Assume that the graph decomposes into a cycle $\mathcal{C}$ of length r and a graph $\mathcal{H}$ in the sense that $\mathcal{C}$ and $\mathcal{H}$ share exactly one node $i := \mathcal{C} \cap \mathcal{H}$ and there are no edges connecting $\mathcal{C}\setminus\{i\}$ and $\mathcal{H}\setminus\{i\}$. Then, for any $L = 1, 2, \ldots$,*

$$\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] \geq \left[\frac{3}{(d_i+1)(r-1)}\right]^2 1.5^{-2L}. \tag{13}$$

The variance term in GCNs is sensitive to small perturbations in the graph. To see this, suppose node i admits a local tree structure as described in Proposition 1. Attaching an additional cycle to node i, (13) demonstrates that regardless of the edge degree d_i of node i, the variance decay rate in (d_i, L) will immediately drop from $\lesssim (d_i+1)^{-L}$ to $\gtrsim (d_i+1)^{-2}2.25^{-L}$. On the contrary, the local average estimator (11) will hardly be affected by an additional cycle.

Proposition 3 serves as a prototype illustrating why stacking many GCN layers often results in poor performance. In practical applications like social network analysis, nodes frequently exhibit additional cycles. Proposition 3 suggests that in such cases, more GCN layers are needed to balance the bias-variance trade-off. However, as shown in Theorem 1, increasing the number of layers also raises the variance term, ultimately leading to the well-known over-smoothing phenomenon (Oono and Suzuki, 2020; Cai and Wang, 2020; Rusch et al., 2023; Li et al., 2018). This phenomenon contrasts sharply with Fully Connected Neural Networks (FNNs), where both theory and empirical evidence suggest that greater depth increases the network expressiveness (Telgarsky, 2016; Schmidt-Hieber, 2020).

5 NUMERICAL SIMULATIONS

We validate our results through a series of numerical experiments[1]. Specifically, to evaluate the link between neighborhood topology – particularly degree distribution and degree skewness – and optimal neighborhood size, we generate graphs with $n = 100$ nodes using the following topologies:

- *Latent variable graphs*: For each node i, we generate a latent variable vector $U_{i\cdot} \in \mathbb{R}^{1\times 2}$ by sampling from a two-dimensional uniform distribution on $[0,1]^2$. The edges of the network are then sampled from a Bernoulli distribution with $p_{ij} = 1/(1+e^{-5\|U_{i\cdot}^\top - U_{j\cdot}^\top\|_2})$, where the scaling factor (here, 5) is chosen empirically to control the edge density. To further measure the effect of the edge density on the optimal neighborhood size, we sparsify the induced network as follows. We first sample a Minimum Spanning Tree (MST) as the graph's "backbone" to ensure the sparsified graph remains connected. We then delete edges that are not in the MST with probability p, so that a higher p results in a sparser graph.
- *Preferential attachment graphs:* the graphs are generated using the Holme and Kim algorithm (Holme and Kim, 2002), varying their local clustering probability p. In this case, higher values of p yield graphs with higher clustering coefficients.
- *k-regular trees:* where k denotes the degree of each node in the tree.
- *Barbell graphs:* consisting of two cliques of m fully connected nodes, joined by a chain of $n - 2m$ nodes.

For the last three graph topologies, we create a latent variable $U_{i\cdot} \in \mathbb{R}^{1\times 2}$ for each node by taking the coordinates of the first two eigenvectors of the graph Laplacian. Denoting the eigenvalue decomposition of the graph Laplacian $\mathbf{L} = D - A$ as $\mathbf{L} = \mathbf{V}\Lambda\mathbf{V}^\top$, we define $\mathbf{U}$ as the matrix formed by the first two columns of $\mathbf{V}$, i.e., $\mathbf{U} = \mathbf{V}[,1:2]$.

A node signal $\mathbf{f}^*$ is then generated as a smooth function of the latent variable vector $U_{i\cdot} \in \mathbb{R}^{1\times 2}$:

$$f_i^* = 2\cos(U_{i\cdot}\beta), \tag{14}$$

where $U_{i\cdot}$ denotes the i-th row of $\mathbf{U}$ and β is a vector of coefficients of the form $\beta = (-\alpha, \alpha)^\top$, with α adjusted to control different levels of smoothness across the graph. As highlighted in Figures 7-10, lower values of α tend to create smoother functions, while higher values of α induce faster varying signals.

[1]The code used for the experiments in the following two sections is available at the following link: https://github.com/donnate/GNN_regression.

The left subplots of Figures 7-10 present examples of the generated graphs, with node colors indicating the true $\mathbf{f}^*$. The middle and right subplots provide examples of the graph's spectral embeddings (or latent variables) $\mathbf{U}$, for different values of α, showcasing signals with various levels of smoothness. Finally, the observed response vector $\mathbf{Y} = (Y_1, \ldots, Y_n)$ is simply generated as:

$$Y_i = f_i^* + \varepsilon_i$$

where $\varepsilon_i \sim N(0, \sigma^2)$ is independent Gaussian noise. Unless otherwise specified, σ is set to 1. In all subsequent experiments, consistent with the GNN literature on transductive learning (e.g., Kipf and Welling (2017)), we partition the dataset, using 20% of the nodes for training and a separate 20% for testing. This emulates the standard GNN setting, where *only a fraction of the nodes* is available for training. We compare the results of the estimators obtained by (a) smoothing the signal on neighborhoods of size L (where L varies), using either the GCN (T) or GraphSAGE (S). These estimators have no learned parameters and solely assess the effect of neighborhood smoothing, which we compare to the local averaging estimator $\bar{\mathbf{f}}_L$; and (b) trained GNNs (either the GCN or GraphSAGE) with a hidden dimension of size 8 and non-linearities. The purpose of this second set of methods is to show that our results extends to trained, non-linear GNNs.

Result 1: The optimal convolution size L decreases sharply as the signal's roughness increases. Figure 1 displays the optimal neighborhood size, averaged over 20 experiments, as a function of the graph's roughness (simply defined as $\sqrt{[\sum_{(i,j)\in\mathcal{E}}(f_i^* - f_j^*)^2]/|\mathcal{E}|}$) for the latent variable graph. We observe that the optimal number of convolutions decreases as a function of the graph roughness, as long as the neighborhood is informative of the underlying signal. Similar figures for the other three graph topologies (and for various levels of noise) are presented in Figures 11-13 in the appendix. The result confirms the theoretical findings presented in Theorem 1, which illustrate the bias-variance trade-off phenomenon. Interestingly, the GraphSAGE convolution S results in optimal neighborhood sizes that tend to be slightly larger than those of the usual GCN convolution, particularly for very smooth signals.

Result 2: The variance appears to be generally lower for the GCN convolution (T) compared to GraphSAGE (S). Figures 14-19 in the appendix illustrate the bias-variance trade-off across different graph topologies as the number of convolutions increases. As expected, and as predicted by (12), the local averaging estimator consistently exhibits lower variance than the GNN convolutions (S

and T). Conversely, the bias of the local averaging operator increases more substantially as a function of L than that of the GNN convolutions, particularly in cases where the signal is not extremely smooth. In the Barbell graph, for instance, local averaging shows three times the bias of GNNs for $L = 4$ at $\alpha = 5$. GNNs with learned weights and non-linearities behave similarly for sufficiently smooth signals. Across topologies, the signal $f_i^* = 2\cos(U_i.\beta)$ with $\beta = (-0.1, 0.1)^\top$ yields smooth graphs that display a clear bias-variance trade-off. Moreover, non-linear GCNs tend to exhibit lower variance than their GraphSAGE counterparts.

Result 3: The graph topology does affect the rate of variance decay. The derived theory (Proposition 1) predicts exponential variance decay in rooted trees with respect to the number of layers L. Moreover, the larger the degree d of the root node, the faster the variance decay. Figure 2 shows the decay of the variance as a function of L (logged on the y-axis), for the signal f_i^* defined in (14). As expected, the log of the variance decays linearly with the number of convolutions L, with higher degrees resulting in steeper decay slopes.

Our theory also predicts that, for instance, nodes with a high degree but connected to low-degree nodes will exhibit a lower variance decay than nodes connected to high-degree nodes. To this end, we compare the same signal in settings where either a complete graph or a star is added to a tree. In both settings, the local average estimator is expected to behave similarly. We find (Figure 20) that the variance for the GCN convolution on the star graph does not decay effectively.

Next, we investigate the impact of adding local cycles, as discussed in Proposition 3. We construct a binary tree and introduce edges between nodes at levels 2 and 3. The results in Figure 3 confirm that variance decays much more slowly with cycles.

6 REAL-DATA ANALYSIS

We validate our framework on 6 real-world examples. More specifically, we used the publicly available regression datasets used as validation by Jia and Benson (2020) and Huang et al. (2024). These datasets include 3 spatial datasets on election results, income as well as education levels across the counties in the United States. We also include 2 transportation networks, based on traffic data in the cities of Anaheim and Chicago. Each vertex represents a directed lane, and two lanes that meet at the same intersection are connected. In these datasets, the response Y_i represents the traffic flow in the lane. Finally, the Twitch dataset captures friendships among Twitch streamers in Portugal, while the node signal corresponds to the logarithm of the number of viewers for each streamer. Figure 5 provides details on the datasets and their properties, emphasizing their distinct topological characteristics. In particular, spatial datasets typically have a regular degree distribution, whereas the Twitch dataset exhibits a highly skewed degree distribution.

We use our estimator in the prediction setting, where the value at the node of interest is unobserved. In this setting, we predict $\hat{Y}_i$ using different convolutions: $\hat{Y}_i = \sum_{j\in\mathcal{N}(i)} w_{ij}Y_j$, where the weights in the sum depend on the convolution. To emulate a real-life training scenario, we randomly select a set of 500 nodes for training and another 500 for validation. Figure 4 shows the mean squared error on the validation set as a function of the neighborhood size. Interestingly, we observe that the optimal neighborhood size is reached at $L = 2$ for all spatial graphs. In these cases, the degree distribution is relatively homogeneous across nodes (see statistics in the appendix), and the convolutions S and T produce fairly similar results. However, in the social network example, different aggregation operators lead to drastically different effects, with the GraphSAGE convolution outperforming the other two.

We also test our estimator as a denoiser. In this setting, we denoise the observed value Y_i as $\hat{Y}_i = \sum_{j\in\mathcal{N}^1(i)} w_{ij}Y_j$, with the weights depending on the convolution. To avoid data snooping between the training and test sets, we replace the values of the test nodes with those of one of their direct neighbors. Assuming that direct neighbors have roughly the same distribution, this modification ensures that no information from the training set permeates the test set while creating another version of the same graph. The results are presented in the appendix in Figure 6.

7 CONCLUSION

In this work, we lay the groundwork for a statistical understanding of GCNs. GCNs perform weighted local averaging over neighboring nodes, with the neighborhood size determined by the depth L of the network. In the context of denoising, we characterize a bias-variance type trade-off risk bound in terms of L for both the original GCN and GraphSAGE convolution. We showed that the variance term of the GraphSAGE convolution is lower-bounded by its counterpart from the local averaging estimator. We propose investigating the variance term from a new perspective: walk analysis, which enables the derivation of both lower and upper bounds of the variance based on local graph topologies. Our analysis reveals a lack of robustness in GCNs, in that small perturbations in the graph structure can lead to substantial changes in the variance.

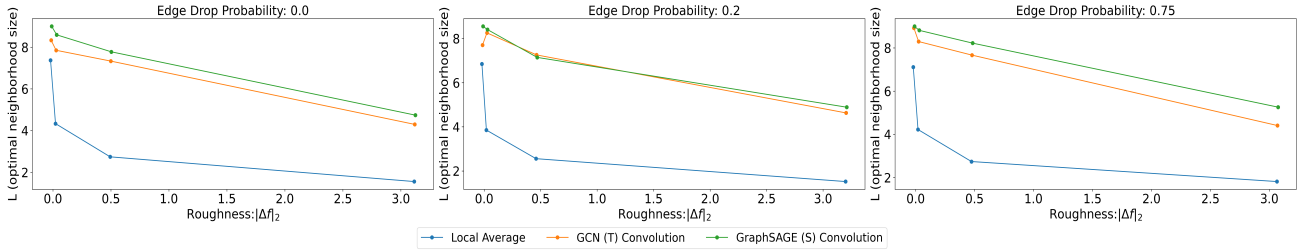


Figure 1: Optimal number of convolutions as a function of the roughness $\|\Delta \mathbf{f}^*\|_2 = \sqrt{[\sum_{(i,j)\in\mathcal{E}}(f_i^* - f_j^*)^2]/|\mathcal{E}|}$ of $\mathbf{f}^*$ on the latent variable graph with $\sigma^2 = 2$. Each subplot corresponds to a different sparsification level, with 0 representing the original graph and 0.75 representing a graph with 75% of its edges removed.

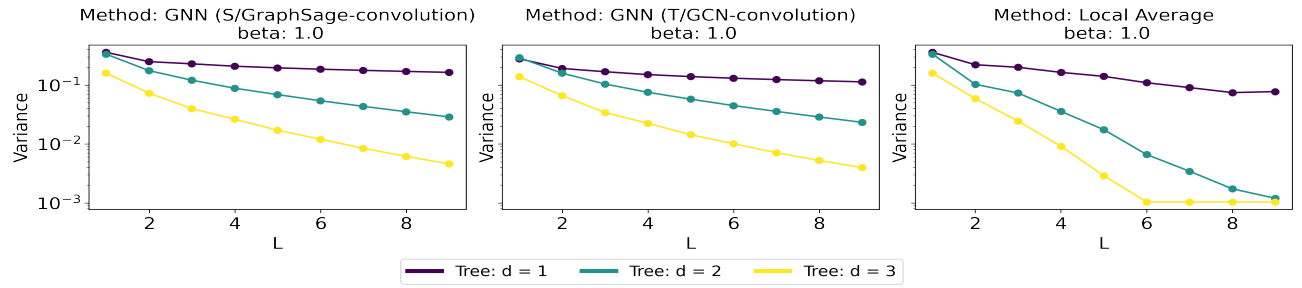


Figure 2: The variance decay at the root of the tree as a function of L under the signal $f_i^* = 2\cos(U_{i\cdot}\beta)$, with $\beta = (-1, 1)^\top$, colored by degree value. The y-axis is shared across all 3 plots.

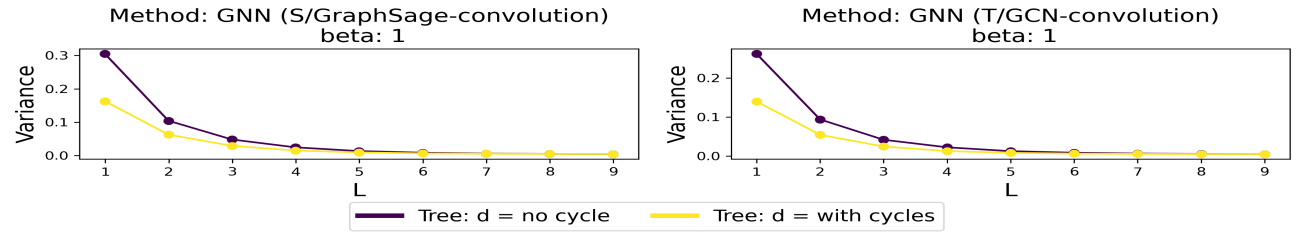


Figure 3: The variance decay at the root of the tree as a function of L under the signal $f_i^* = 2\cos(U_{i\cdot}\beta)$, with $\beta = (-1, 1)^\top$, and its behavior after adding cycles.

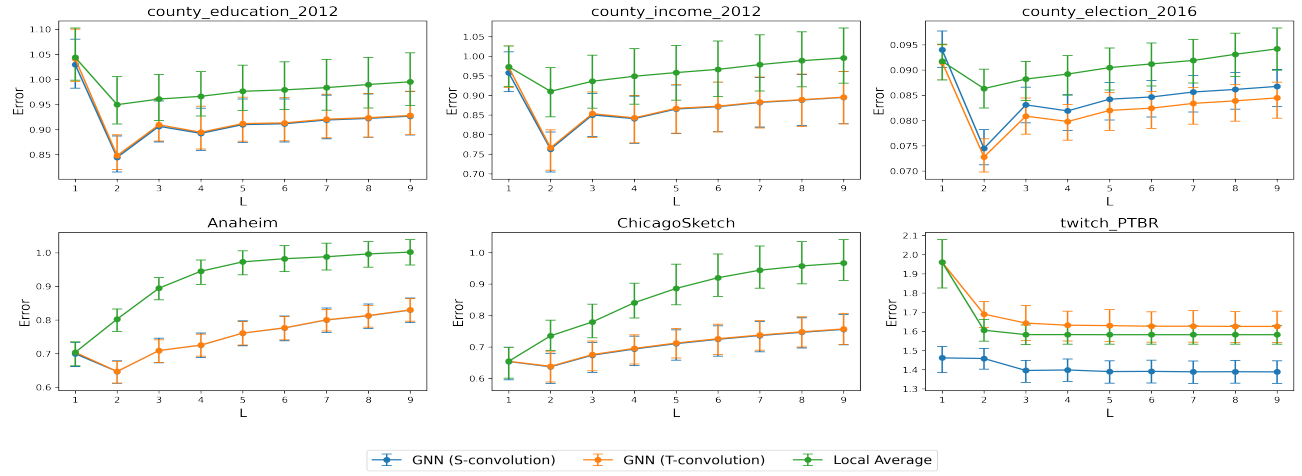


Figure 4: Mean Squared Error (MSE) as a function of the neighborhood size. The nodes denote the mean MSE over 50 random splits of the data into training and validation sets, and the error bars denote interquartile ranges.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. The research of J. S.-H. and J. C. was supported by the NWO Vidi grant VI.Vidi.192.021. The research of C. D. was funded by the National Science Foundation (Award Number 2238616), as well as the resources provided by the University of Chicago's Research Computing Center. The work of O. K. was funded by CY Initiative (grant "Investissements d'Avenir" ANR-16-IDEX0008) and Labex MME-DII(ANR11-LBX-0023-01). Part of this work was carried out while J. S.-H. and J. C. were visiting the Simons Institute for the Theory of Computing in Berkeley.

References

Belkin, M., Matveeva, I., and Niyogi, P. (2004). Regularization and semi-supervised learning on large graphs. In *Conference on Learning Theory (COLT)*, pages 624–638.

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396.

Cai, C. and Wang, Y. (2020). A note on over-smoothing for graph neural networks. *arXiv preprint arXiv:2006.13318*.

Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Neural Information Processing Systems (NeurIPS)*, pages 3844–3852.

Donnat, C., Klopp, O., and Verzelen, N. (2024). One-bit total variation denoising over networks with applications to partially observed epidemics. *arXiv preprint arXiv:2405.00619*.

Garg, V., Jegelka, S., and Jaakkola, T. (2020). Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning (ICML)*, pages 3419–3430.

Green, A., Balakrishnan, S., and Tibshirani, R. (2021). Minimax optimal regression over Sobolev spaces via Laplacian regularization on neighborhood graphs. In *International Conference on Artificial Intelligence and Statistics (AIStats)*, pages 2602–2610.

Green, A., Balakrishnan, S., and Tibshirani, R. J. (2023). Minimax optimal regression over Sobolev spaces via Laplacian Eigenmaps on neighbourhood graphs. *Information and Inference: A Journal of the IMA*, 12(3):2423–2502.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Inductive representation learning on large graphs. *arXiv preprint arXiv:1706.02216*.

Hein, M. (2006). Uniform convergence of adaptive graph-based regularization. In *Conference on Learning Theory (COLT)*, pages 50–64.

Henaff, M., Bruna, J., and LeCun, Y. (2015). Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*.

Holme, P. and Kim, B. J. (2002). Growing scale-free networks with tunable clustering. *Physical review E*, 65(2):026107.

Huang, K., Jin, Y., Candes, E., and Leskovec, J. (2024). Uncertainty quantification over graph with conformalized graph neural networks. In *Neural Information Processing Systems (NeurIPS)*, volume 36.

Hütter, J.-C. and Rigollet, P. (2016). Optimal rates for total variation denoising. In *Conference on Learning Theory (COLT)*, pages 1115–1146.

Jia, J. and Benson, A. R. (2020). Residual correlation in graph neural network regression. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 588–598.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Kirichenko, A. and van Zanten, H. (2017). Estimating a smooth function on a large graph by Bayesian Laplacian regularisation. *Electronic Journal of Statistics*, 11(1):891–915.

Lafon, S. S. (2004). *Diffusion maps and geometric harmonics*. Yale University.

Li, Q., Han, Z., and Wu, X.-M. (2018). Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI Conference on Artificial Intelligence*.

NT, H. and Maehara, T. (2019). Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*.

NT, H., Maehara, T., and Murata, T. (2021). Revisiting graph neural networks: Graph filtering perspective. In *International Conference on Pattern Recognition (ICPR)*, pages 8376–8383.

Oono, K. and Suzuki, T. (2020). Graph neural networks exponentially lose expressive power for node classification. In *International Conference on Learning Representations (ICLR)*.

Padilla, O. H. M., Sharpnack, J., Scott, J. G., and Tibshirani, R. J. (2018). The DFS Fused Lasso: Linear-time denoising over general graphs. *Journal of Machine Learning Research*, 18(176):1–36.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Neural Information Processing Systems (NeurIPS)*.

Rigollet, P. (2015). High-dimensional statistics. *Lecture Notes, Cambridge, MA, USA: MIT OpenCourseWare*.

Rusch, T. K., Bronstein, M. M., and Mishra, S. (2023). A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*.

Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics*, 48(4):1875 – 1897.

Singer, A. (2006). From graph to manifold Laplacian: The convergence rate. *Applied and Computational Harmonic Analysis*, 21(1):128–134.

Smola, A. J. and Kondor, R. (2003). Kernels and regularization on graphs. In *Conference on Learning Theory (COLT)*, pages 144–158.

Telgarsky, M. (2016). Benefits of depth in neural networks. In *Conference on Learning Theory (COLT)*, pages 1517–1539.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108.

Tibshirani, R. J. and Taylor, J. (2011). The solution path of the generalized lasso. *The annals of statistics*, 39(3):1335–1371.

Verma, S. and Zhang, Z.-L. (2019). Stability and generalization of graph convolutional neural networks. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1539–1548.

Vinas, L. and Amini, A. A. (2024). Sharp bounds for Poly-GNNs and the effect of graph noise. *arXiv preprint arXiv:2407.19567*.

Von Luxburg, U., Belkin, M., and Bousquet, O. (2008). Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586.

Wang, Y.-X., Sharpnack, J., Smola, A. J., and Tibshirani, R. J. (2016). Trend filtering on graphs. *Journal of Machine Learning Research*, 17(105):1–41.

Wasserman, L. (2006). *All of nonparametric statistics*. Springer Science & Business Media.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. (2019). Simplifying graph convolutional networks. In *International Conference on Machine Learning (ICML)*, pages 6861–6871.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2018). How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826*.

Checklist

1. For all models and algorithms presented, check if you include:
   (i) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (ii) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (iii) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes/No/Not Applicable]

2. For any theoretical claim, check if you include:
   (i) Statements of the full set of assumptions of all theoretical results. [Yes]
   (ii) Complete proofs of all theoretical results. [Yes]
   (iii) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (i) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (ii) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (iii) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (iv) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (i) Citations of the creator If your work uses existing assets. [Yes]
   (ii) The license information of the assets, if applicable. [Yes]

(iii) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]

(iv) Information about consent from data providers/curators. [Not Applicable]

(v) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(i) The full text of instructions given to participants and screenshots. [Not Applicable]

(ii) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

(iii) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A PROOFS OF MAIN THEOREMS

A.1 Proof of Theorem 1

Recall that as the noise ε_i is centered, with inequality (8) in the paper, we have for any fixed parameter $W \in \mathbb{R}$ and matrix $P \in \mathbb{R}^{n\times n}$,

$$\mathbb{E}\left[\|WP\mathbf{Y} - \mathbf{f}^*\|_2^2\right] \leq \left(|W| \cdot \left\|P\mathbf{f}^* - \mathbf{f}^*\right\|_2 + |1-W| \cdot \|\mathbf{f}^*\|_2\right)^2 + W^2\mathbb{E}\left[\|P\boldsymbol{\varepsilon}\|_2^2\right]. \tag{A.1}$$

We first consider the case $P = S^L$ for $L = 1, 2, \ldots$. To further bound (A.1) for $P = S^L$, we now prove by induction on L that

$$\|S^L\mathbf{f}^* - \mathbf{f}^*\|_\infty \leq L\Delta, \tag{A.2}$$

where $\|\mathbf{v}\|_\infty := \max_{i=1,\ldots,n} |v_i|$ for any vector $\mathbf{v} = (v_1, \ldots, v_n)^\top$.

Write

$$\tilde{A} = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}, \tag{A.3}$$

where $a_{ii} = 1$, for all $i \in \{1, \ldots, n\}$ and $a_{ij} = a_{ji} \in \{0, 1\}$, for $i \neq j$, indicating whether nodes i and j are connected by an edge belonging to $\mathcal{E}$. The matrix $\tilde{A}$ represents the connection structure of the augmented graph $\mathcal{G}'$, where self-loops are included by setting $a_{ii} = 1$, for $i = 1, \ldots, n$. Thus,

$$S = \tilde{D}^{-1}\tilde{A} = \begin{pmatrix} \frac{a_{11}}{d_1+1} & \frac{a_{12}}{d_1+1} & \cdots & \frac{a_{1n}}{d_1+1} \\ \frac{a_{21}}{d_2+1} & \frac{a_{22}}{d_2+1} & \cdots & \frac{a_{2n}}{d_2+1} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{a_{n1}}{d_n+1} & \frac{a_{n2}}{d_n+1} & \cdots & \frac{a_{nn}}{d_n+1} \end{pmatrix}, \tag{A.4}$$

where d_i is the edge degree of node i.

For $L = 1$, we can employ the smoothness condition (9) in Assumption 1 and the fact that the edge degree d_i is the same as the number of neighbors of i, to show

$$\begin{aligned} \|S\mathbf{f}^* - \mathbf{f}^*\|_\infty &= \|\tilde{D}^{-1}\tilde{A}\mathbf{f}^* - \mathbf{f}^*\|_\infty \\ &= \max_{i=1,\ldots,n} \left| \sum_{j=1}^n \frac{a_{ij}}{d_i+1} f_j^* - f_i^* \right| \\ &= \max_{i=1,\ldots,n} \left| \sum_{j\in\mathcal{N}(i)} \frac{1}{d_i+1} (f_j^* - f_i^*) \right| \\ &\leq \Delta. \end{aligned}$$

For the induction step, assume the claim holds until $L-1$, in the sense that, for any $1 \leq \ell \leq L-1$, $\|S^\ell\mathbf{f}^* - \mathbf{f}^*\|_\infty \leq \ell\Delta$. Then,

$$\begin{aligned} \|S^L\mathbf{f}^* - \mathbf{f}^*\|_\infty &= \left\|S(S^{L-1}\mathbf{f}^* - \mathbf{f}^*) + S\mathbf{f}^* - \mathbf{f}^*\right\|_\infty \\ &\leq \left\|S(S^{L-1}\mathbf{f}^* - \mathbf{f}^*)\right\|_\infty + \|S\mathbf{f}^* - \mathbf{f}^*\|_\infty \\ &\leq \max_{i=1,\ldots,n} \left[\sum_{j=1}^n \frac{a_{ij}}{d_i+1} (L-1)\Delta \right] + \Delta \\ &= (L-1)\Delta \max_{i=1,\ldots,n} \left[\sum_{j\in\mathcal{N}(i)} \frac{1}{d_i+1} \right] + \Delta \\ &\leq L\Delta, \end{aligned}$$

completing the induction step. Thus, we conclude that $\|S^L\mathbf{f}^* - \mathbf{f}^*\|_\infty \leq L\Delta$, for all $L = 1, 2, \ldots$. This implies that

$$\|S^L\mathbf{f}^* - \mathbf{f}^*\|_2 \leq \sqrt{n}\|S^L\mathbf{f}^* - \mathbf{f}^*\|_\infty \leq \sqrt{n}L\Delta. \tag{A.5}$$

Moreover, for the variance part, we use the fact that the components of $\boldsymbol{\varepsilon}$ are uncorrelated and have a variance of one to obtain

$$\mathbb{E}\left[\|S^L\boldsymbol{\varepsilon}\|_2^2\right] = \mathrm{Tr}((S^L)^\top S^L) = \|S^L\|_F^2. \tag{A.6}$$

Plugging (A.5) and (A.6) into (A.1) with $P = S^L$, we deduce that

$$\begin{aligned}
\frac{1}{n}\mathbb{E}\left[\|WS^L\mathbf{Y} - \mathbf{f}^*\|_2^2\right] &\leq \frac{1}{n}\Big(|W| \cdot \left\|S^L\mathbf{f}^* - \mathbf{f}^*\right\|_2 + |1-W| \cdot \|\mathbf{f}^*\|_2\Big)^2 + \frac{W^2}{n}\mathbb{E}\left[\|S^L\boldsymbol{\varepsilon}\|_2^2\right] \\
&\leq \left(|W| \cdot L\Delta + \frac{|1-W|}{\sqrt{n}} \cdot \|\mathbf{f}^*\|_2\right)^2 + W^2\frac{\|S^L\|_F^2}{n} \\
&= \mathrm{Bias}_S^2(L, W) + W^2\frac{\|S^L\|_F^2}{n},
\end{aligned}$$

which concludes the proof of the first upper bound in Theorem 1 of the paper.

Next, we address the case $P = T^L$ with $T = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$. Observe that for $L = 1, 2, \ldots,$

$$T^L = \tilde{D}^{1/2}S^L\tilde{D}^{-1/2},$$

which implies

$$T^L\mathbf{f}^* - \mathbf{f}^* = \tilde{D}^{1/2}S^L\tilde{D}^{-1/2}\mathbf{f}^* - \mathbf{f}^* = \tilde{D}^{1/2}\big(S^L - I_n\big)\tilde{D}^{-1/2}\mathbf{f}^*. \tag{A.7}$$

Since $\mathbf{f}^*$ satisfies (10) in Assumption 1, the vector $\tilde{D}^{-1/2}f^*$ satisfies (9) in Assumption 1. Using (A.2) and (A.7),

$$\begin{aligned}
\|T^L\mathbf{f}^* - \mathbf{f}^*\|_2 &= \|\tilde{D}^{1/2}\big(S^L - I_n\big)\tilde{D}^{-1/2}\mathbf{f}^*\|_2 \\
&\leq \sqrt{\sum_{i=1}^n (d_i+1)} \cdot \|\big(S^L - I_n\big)\tilde{D}^{-1/2}\mathbf{f}^*\|_\infty \\
&\leq \sqrt{\sum_{i=1}^n (d_i+1)} \cdot L\Delta \\
&= \sqrt{2|\mathcal{E}| + n} \cdot L\Delta,
\end{aligned} \tag{A.8}$$

where the last equality follows from $|\mathcal{E}| = (\sum_{i=1}^n d_i)/2$.

For the variance term, we can proceed similarly as in (A.6) and obtain

$$\mathbb{E}\left[\|T^L\boldsymbol{\varepsilon}\|_2^2\right] = \mathrm{Tr}((T^L)^\top T^L) = \|T^L\|_F^2. \tag{A.9}$$

Plugging (A.8) and (A.9) into (A.1) with $P = T^L$, we deduce

$$\begin{aligned}
\frac{1}{n}\mathbb{E}\left[\|WT^L\mathbf{Y} - \mathbf{f}^*\|_2^2\right] &\leq \frac{1}{n}\Big(|W| \cdot \left\|T^L\mathbf{f}^* - \mathbf{f}^*\right\|_2 + |1-W| \cdot \|\mathbf{f}^*\|_2\Big)^2 + \frac{W^2}{n}\mathbb{E}\left[\|T^L\boldsymbol{\varepsilon}\|_2^2\right] \\
&\leq \left(|W| \cdot L\Delta\sqrt{\frac{2|\mathcal{E}|}{n} + 1} + \frac{|1-W|}{\sqrt{n}} \cdot \|\mathbf{f}^*\|_2\right)^2 + W^2\frac{\|T^L\|_F^2}{n} \\
&= \mathrm{Bias}_T^2(L, W) + W^2\frac{\|T^L\|_F^2}{n},
\end{aligned}$$

proving the second upper bound in Theorem 1.

A.2 Proof of Theorem 2

By definition,

$$\overline{f}_{L,i} = \frac{1}{|\mathcal{N}^L(i)|}\left(\sum_{j\in\mathcal{N}^L(i)} Y_j\right) = \frac{1}{|\mathcal{N}^L(i)|}\left[\sum_{j\in\mathcal{N}^L(i)} (f_j^* + \varepsilon_j)\right].$$

Since the components of $\boldsymbol{\varepsilon}$ are uncorrelated and have a variance of one, we obtain

$$\mathrm{Var}\left(\overline{f}_{L,i}\right) = \frac{1}{|\mathcal{N}^L(i)|^2}\sum_{j\in\mathcal{N}^L(i)} \mathrm{Var}\left(f_j^* + \varepsilon_j\right) = \frac{1}{|\mathcal{N}^L(i)|}.$$

For any $j \in \mathcal{N}^L(i)$, there exists a path in the augmented graph $\mathcal{G}'$, where self-loops are allowed, given by

$$(i \to \ell_1 \to \ldots \to \ell_{L-1} \to j), \tag{A.10}$$

which connects node i to node j through nodes $\ell_1, \ldots, \ell_{L-1} \in \mathcal{V}$ and has a length of L. The path in (A.10) also implies that $\ell_1 \in \mathcal{N}^1(i), \ell_2 \in \mathcal{N}^1(\ell_1), \ldots, j \in \mathcal{N}^1(\ell_{L-1})$. Therefore, if condition (9) in Assumption 1 holds, we have, for any $j \in \mathcal{N}^L(i)$,

$$|f_i^* - f_j^*| \le |f_i^* - f_{\ell_1}^*| + |f_{\ell_1}^* - f_{\ell_2}^*| + \cdots + |f_{\ell_{L-1}}^* - f_j^*| \le L\Delta. \tag{A.11}$$

Since the noise variables ε_i are assumed to be uncorrelated, centered, and have a variance of one, we can derive using (A.11) that for any $i = 1, \ldots, n$,

$$\begin{aligned}
\mathbb{E}\left[|\overline{f}_{L,i} - f_i^*|^2\right] &= \mathbb{E}\left[\left|\frac{1}{|\mathcal{N}^L(i)|}\left(\sum_{j\in\mathcal{N}^L(i)} Y_j\right) - f_i^*\right|^2\right] \\
&= \mathbb{E}\left[\left|\frac{1}{|\mathcal{N}^L(i)|}\left(\sum_{j\in\mathcal{N}^L(i)} (f_j^* + \varepsilon_j)\right) - f_i^*\right|^2\right] \\
&\le \frac{\sum_{j\in\mathcal{N}^L(i)} |f_j^* - f_i^*|^2}{|\mathcal{N}^L(i)|} + \mathbb{E}\left[\left|\frac{1}{|\mathcal{N}^L(i)|}\left(\sum_{j\in\mathcal{N}^L(i)} \varepsilon_j\right)\right|^2\right] \\
&\le L^2\Delta^2 + \frac{1}{|\mathcal{N}^L(i)|}.
\end{aligned} \tag{A.12}$$

Summing (A.12) for all $i = 1, \ldots, n$ finally yields

$$\frac{1}{n}\mathbb{E}\left[\|\overline{\mathbf{f}}_L - \mathbf{f}^*\|_2^2\right] = \frac{1}{n}\mathbb{E}\left[\sum_{i=1}^n |\overline{f}_{L,i} - f_i^*|^2\right] \le (L\Delta)^2 + \frac{1}{n}\sum_{i=1}^n \frac{1}{|\mathcal{N}^L(i)|}.$$

B DERIVATION OF ASSOCIATED WEIGHTS

B.1 Associated Weights for S^L

Recall that we can represent S as shown in (A.4), where a_{ij} for $i \neq j$ indicates whether node i is connected to node j by an edge in $\mathcal{E}$, and $a_{ii} = 1$ since self-loops are included. Similarly, we define $a_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)} := 1$, if there exists a path with self-loops of length L from node i to node j via nodes $\ell_1, \ldots \ell_{L-1} \in \{1, \ldots, n\}$; otherwise $a_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)} := 0$. In particular, when $L = 1$, we have $a_{ij} = a_{(i\to j)}$.

Let s_{ij}^L be the entry of the matrix S^L in the i-th row and j-th column. In what follows, we show by induction that for any $i, j = 1, \ldots, n$,

$$s_{ij}^L = \sum_{\ell_1,\ldots,\ell_{L-1}=1}^n \frac{a_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)}}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-1}}+1)}, \tag{A.13}$$

where $d_i, d_{\ell_1}, \ldots, d_{\ell_{L-1}}$ denote the edge degrees of nodes $i, \ell_1, \ldots, \ell_{L-1}$, respectively. The equality (A.13) then implies that the associated weight for each path is given by

$$\omega_i(\ell_1, \ldots, \ell_{L-1}) = \frac{1}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-1}}+1)}.$$

One can observe from (A.4) that the conclusion holds for $L = 1$. For the induction step, assume the claim holds until $L-1$, in the sense that, for any $1 \le k \le L-1$, and $i, j = 1, \ldots, n$,

$$s_{ij}^k = \sum_{\ell_1,\ldots,\ell_{k-1}=1}^{n} \frac{a_{(i\to\ell_1\to\ldots\to\ell_{k-1}\to j)}}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{k-1}}+1)}.$$

Then

$$\begin{aligned} s_{ij}^L &= \sum_{\ell_{L-1}=1}^{n} s_{i\ell_{L-1}}^{L-1} \frac{a_{\ell_{L-1}j}}{d_{\ell_{L-1}}+1} \\ &= \sum_{\ell_{L-1}=1}^{n} \left[\left(\sum_{\ell_1,\ldots,\ell_{L-2}=1}^{n} \frac{a_{(i\to\ell_1\to\ldots\to\ell_{L-2}\to\ell_{L-1})}}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-2}}+1)} \right) \frac{a_{(\ell_{L-1}\to j)}}{d_{\ell_{L-1}}+1} \right] \\ &= \sum_{\ell_1,\ldots,\ell_{L-1}=1}^{n} \frac{a_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)}}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-1}}+1)}, \end{aligned}$$

which completes the argument.

B.2 Associated Weights for T^L

By definition, $T = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ and $S = \tilde{D}^{-1}\tilde{A}$. Thus, for any $L = 1, 2, \ldots,$

$$T^L = \tilde{D}^{1/2} S^L \tilde{D}^{-1/2}. \tag{A.14}$$

For any $i, j = 1, \ldots, n$, let t_{ij}^L represent the entry of the matrix T^L in the i-th row and j-th column. Building on (A.14) and the formula (A.13) for the entries of S^L, we can derive that

$$\begin{aligned} t_{ij}^L = \frac{\sqrt{d_i+1}}{\sqrt{d_j+1}} s_{ij}^L &= \sum_{\ell_1,\ldots,\ell_{L-1}=1}^{n} \frac{\sqrt{d_i+1}}{\sqrt{d_j+1}} \frac{a_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)}}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-1}}+1)}, \\ &= \sum_{\ell_1,\ldots,\ell_{L-1}=1}^{n} a_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)} \widetilde{\omega}_{ij}(\ell_1, \ldots, \ell_{L-1}), \end{aligned}$$

which implies that the associated weight for each path $(i \to \ell_1 \to \ldots \to \ell_{L-1} \to j)$ is $\widetilde{\omega}_{ij}(\ell_1, \ldots, \ell_{L-1})$.

C PROOFS OF PROPOSITIONS

C.1 Proof of Proposition 1

We examine walks on the augmented graph $\mathcal{G}'$ that includes self-loops. Specifically, for an integer r satisfying $0 \le r \le L$, we consider the paths on $\mathcal{G}'$ that start at node i and arrive after L moves at node $j \in \mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)$. For any $j \in \mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)$, we denote the set of paths by $\mathcal{P}_L^r(i \to j)$.

To derive an upper bound on the cardinality of $\mathcal{P}_L^r(i \to j)$, we refine the analysis as follows. For each r, define the set

$$\mathcal{S}(r) := \{s \in \mathbb{Z} : \ r \le s \le L \text{ and } s \equiv r \pmod 2\},$$

which represents the set of integers s satisfying $r \le s \le L$ and having the same parity as r. Each path in $\mathcal{P}_L^r(i \to j)$ can then be expressed as starting at node i, moving $(k+r)/2$ times toward j, $(k-r)/2$ times away from j, and remaining at the same node for $L-k$ steps, for some $k \in \mathcal{S}(r)$. For any r with $0 \le r \le L$ and

$k \in \mathcal{S}(r)$, let $\mathcal{P}_L^{k,r}(i \to j)$ denote all paths from node i to node j of length L in the self-loop augmented graph $\mathcal{G}'$, characterized by $(k+r)/2$ movements toward j, $(k-r)/2$ movements away from j, and $L-k$ steps remaining at the same node. Since $\mathcal{P}_L^r(i \to j) \subseteq \cup_{k \in \mathcal{S}(r)} \mathcal{P}_L^{k,r}(i \to j)$, we now focus our attention on the set $\mathcal{P}_L^{k,r}(i \to j)$.

We will use F to represent a movement towards j, B to represent a movement away from j, and write R if the walk remains at the same node. Then each path in $\mathcal{P}_L^{k,r}(i \to j)$ corresponds to a sequence of length L composed of F, B, and R, with F, B, and R occurring $(k+r)/2$, $(k-r)/2$, and $L-k$ times, respectively. An example of such a sequence is

$$\underbrace{R, \cdots, R}_{(L-k)\text{-times}}, \underbrace{F, \cdots, F}_{r\text{-times}}, \underbrace{B, \cdots, B}_{(k-r)/2\text{-times}}, \underbrace{F, \cdots, F}_{(k-r)/2\text{-times}} . \tag{A.15}$$

Under the locally rooted tree structure assumption on $\mathcal{G}$, there is a unique sequence of edges $(i \to \ell_1 \to \ldots \to \ell_{r-1} \to j)$ connecting node i to node $j \in \mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)$ in the original graph. All paths in $\mathcal{P}_L^{k,r}(i \to j)$ in $\mathcal{G}'$ are derived from variations of this path $(i \to \ell_1 \to \ldots \to \ell_{r-1} \to j)$ in $\mathcal{G}$. For the F and R steps, the resulting node after one movement is completely determined, ensuring that multiple branches will not arise based on the existing path before employing either F or R in $\mathcal{G}'$. In contrast, in every B step, we have $d-1$ choices to move to another node. Focusing on the $(k-r)/2$ occurrences of B, we can write the walk in the form

$$\ldots, \underbrace{B, \cdots, B}_{q_1\text{-times}}, \ldots, \underbrace{B, \cdots, B}_{q_2\text{-times}}, \ldots, \underbrace{B, \cdots, B}_{q_3\text{-times}}, \ldots, \underbrace{B, \cdots, B}_{q_m\text{-times}}, \ldots, \tag{A.16}$$

with $q_1, \ldots, q_m \geq 1$ and $q_1 + q_2 + \ldots + q_m = (k-r)/2$. The number of paths of the form (A.16) in the augmented graph is hence bounded by $d^{q_1} d^{q_2} \ldots d^{q_m} = d^{(k-r)/2}$. In a path we can in each step choose one of the three options F, B, R. Thus there are at most 3^L different ways in (A.16) and for any $0 \leq r \leq L$, $j \in \mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)$, and $k \in \mathcal{S}(r)$,

$$\left| \mathcal{P}_L^{k,r}(i \to j) \right| \leq 3^L d^{(k-r)/2}. \tag{A.17}$$

Moreover, observe that for any $0 \leq r \leq L$, due to the locally rooted tree structure of $\mathcal{G}$, we have

$$|\mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)| = d^r, \tag{A.18}$$

with the convention that $\mathcal{N}^0(i) = \{i\}$ and $\mathcal{N}^{-1}(i) = \emptyset$. Therefore, using (A.17) and (A.18),

$$\begin{aligned}
\mathbb{E}\left[\left(S^L \boldsymbol{\varepsilon}\right)_i^2 \right] &= \sum_{j=1}^n \left[\sum_{(i \to \ell_1 \to \ldots \to \ell_{L-1} \to j) \in \mathcal{P}_L(i \to j)} \omega_i(\ell_1, \ldots, \ell_{L-1}) \right]^2 \\
&\leq \sum_{r=0}^L \sum_{j \in \mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)} \left(\sum_{(i \to \ell_1 \to \ldots \to \ell_{L-1} \to j) \in \mathcal{P}_L^r(i \to j)} (d+1)^{-L} \right)^2 \\
&\leq \sum_{r=0}^L \sum_{j \in \mathcal{N}^r(i) \setminus \mathcal{N}^{r-1}(i)} \left(\sum_{k \in \mathcal{S}(r)} \sum_{(i \to \ell_1 \to \ldots \to \ell_{L-1} \to j) \in \mathcal{P}_L^{k,r}(i \to j)} (d+1)^{-L} \right)^2 \\
&\leq \sum_{r=0}^L d^r \left[\sum_{k \in \mathcal{S}(r)} 3^L d^{\frac{k-r}{2}} (d+1)^{-L} \right]^2 \\
&\leq 3^{2L} (d+1)^{-2L} \sum_{r=0}^L \left(\sum_{k \in \mathcal{S}(r)} d^{\frac{k}{2}} \right)^2 .
\end{aligned} \tag{A.19}$$

For $d > 1$,

$$\begin{aligned}\sum_{r=0}^{L}\left(\sum_{k\in\mathcal{S}(r)} d^{\frac{k}{2}}\right)^2 &= \sum_{r=0}^{L}\left(\sum_{\ell=0}^{\lfloor\frac{L-r}{2}\rfloor} d^{\frac{r}{2}+\ell}\right)^2 \\ &= \sum_{r=0}^{L}\left[d^{\frac{r}{2}}\frac{(1-d^{\lfloor\frac{L-r}{2}\rfloor+1})}{1-d}\right]^2 \\ &\leq \sum_{r=0}^{L}\frac{(d^{\frac{L}{2}+1}-d^{\frac{r}{2}})^2}{(d-1)^2} \\ &\leq (L+1)d^L\frac{d^2}{(d-1)^2}.\end{aligned}$$

Combining the previous inequalities with $d/(d-1) \leq 2$, we finally obtain

$$\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] \leq 4\cdot 3^{2L}(d+1)^{-L}(L+1).$$

C.2 Proof of Proposition 2

In this case, we only need to focus on the single path $(i \to \ell_1 \to \ldots \to \ell_{L-1} \to j)$, where $d_{\ell_1}, \ldots, d_{\ell_{L-1}} \leq 3$ and $d_i = d$ under the given condition. For this single path, the associated weight is

$$\omega_i(\ell_1,\ldots,\ell_{L-1}) = \frac{1}{(d_i+1)(d_{\ell_1}+1)\cdot\ldots\cdot(d_{\ell_{L-1}}+1)} \geq \frac{4^{1-L}}{d+1}. \tag{A.20}$$

A direct calculation using (A.20) gives the result,

$$\begin{aligned}\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_i^2\right] &= \sum_{j=1}^{n}\left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)}\omega_i(\ell_1,\ldots,\ell_{L-1})\right]^2 \\ &\geq \left[\frac{1}{(d+1)\cdot 4^{L-1}}\right]^2 \\ &= (d+1)^{-2}4^{2-2L}.\end{aligned}$$

C.3 Proof of Proposition 3

Consider all the nodes on the cycle $\mathcal{C}$, where the nodes are reindexed from 1 to r, starting from the node $i = \mathcal{C}\cap\mathcal{H}$. It is sufficient to consider only paths that remain in the cycle $\mathcal{C}$, move in the first step to either node 2 or node r, and do not visit node 1 again. Since self-loops are allowed, walking along these paths gives in every step at least two possible nodes to move to in the next step. The total number of paths is thus $\geq 2^L$. Then by the pigeonhole principle, there exists at least one node $j \in \{2,\ldots,r\}$ such that $|\mathcal{P}_L(1\to j)| \geq 2^L/(r-1)$. Thus,

$$\begin{aligned}\mathbb{E}\left[\left(S^L\boldsymbol{\varepsilon}\right)_1^2\right] &\geq \left(\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(1\to j)}\frac{1}{(d_i+1)3^{L-1}}\right)^2 \\ &\geq \left[\frac{2^L}{r-1}\frac{1}{(d_i+1)3^{L-1}}\right]^2 \\ &= \left[\frac{3}{(d_i+1)(r-1)}\right]^2 1.5^{-2L}.\end{aligned}$$

D ADDITIONAL PROPOSITIONS AND PROOFS FOR $P = T^L$

In this section, we will repeatedly use the equality

$$\widetilde{\omega}_{ij}(\ell_1,\ldots,\ell_{L-1}) := \left(\frac{d_i+1}{d_j+1}\right)^{1/2}\omega_i(\ell_1,\ldots,\ell_{L-1}) \tag{A.21}$$

derived in the main part of the paper.

Proposition 1'. Assume the local graph around node i is a rooted tree, in the sense that the subgraph induced by the vertices $\mathcal{N}^L(i)$ forms a rooted tree with root i and all nodes having edge degree d. Then, we have

$$\mathbb{E}\big[\big(T^L\boldsymbol{\varepsilon}\big)_i^2\big] \leq 4(d+1)^{-L}(L+1)3^{2L}.$$

Proof. Using (A.21),

$$\begin{aligned}
\mathbb{E}\left[\big(T^L\boldsymbol{\varepsilon}\big)_i^2\right] &= \sum_{j=1}^n \left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} \widetilde{\omega}_{ij}(\ell_1,\ldots,\ell_{L-1})\right]^2 \\
&= \sum_{j=1}^n \left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} \Big(\frac{d_i+1}{d_j+1}\Big)^{1/2}\omega_i(\ell_1,\ldots,\ell_{L-1})\right]^2 \\
&= \sum_{j=1}^n \left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} \omega_i(\ell_1,\ldots,\ell_{L-1})\right]^2 \\
&= \mathbb{E}\left[\big(S^L\boldsymbol{\varepsilon}\big)_i^2\right] \\
&\leq 4(d+1)^{-L}(L+1)3^{2L},
\end{aligned}$$

where the second-to-last equality is due to $d_i = d_j = d$, and the last inequality follows from Proposition 1 in the paper. □

Proposition 2'. If there exists a path from node i to node j via nodes $\ell_1,\ldots,\ell_{L-1} \in \{1,\ldots,n\}$ such that $d_{\ell_1},\ldots,d_{\ell_{L-1}} \leq 3$, then

$$\mathbb{E}\big[\big(T^L\boldsymbol{\varepsilon}\big)_i^2\big] \geq \frac{4^{2-2L}}{(d+1)(d_j+1)},$$

where d_j denotes the edge degree of node j.

Proof. Considering the single path $(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)$ with $d_{\ell_1},\ldots,d_{\ell_{L-1}}\leq 3$ and $d_i = d$, we obtain by applying (A.21) that

$$\begin{aligned}
\mathbb{E}\big[\big(T^L\boldsymbol{\varepsilon}\big)_i^2\big] &= \sum_{j=1}^n \left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)} \Big(\frac{d_i+1}{d_j+1}\Big)^{1/2}\omega_i(\ell_1,\ldots,\ell_{L-1})\right]^2 \\
&\geq \left[\Big(\frac{d+1}{d_j+1}\Big)^{1/2}\frac{1}{(d+1)\cdot 4^{L-1}}\right]^2 \\
&= \frac{4^{2-2L}}{(d+1)(d_j+1)}.
\end{aligned}$$

□

Proposition 3'. Assume that the graph decomposes into a cycle $\mathcal{C}$ of length r and a graph $\mathcal{H}$ in the sense that $\mathcal{C}$ and $\mathcal{H}$ share exactly one node $i := \mathcal{C}\cap\mathcal{H}$ and there are no edges connecting $\mathcal{C}\setminus\{i\}$ and $\mathcal{H}\setminus\{i\}$. Then, for any $L = 1,2,\ldots,$

$$\mathbb{E}\big[\big(T^L\boldsymbol{\varepsilon}\big)_i^2\big] \geq \frac{3}{(d_i+1)(r-1)^2}1.5^{-2L}.$$

Proof. Following the proof of Proposition 3, in which the nodes on the cycle $\mathcal{C}$ have been reindexed from 1 to r, starting from $i = \mathcal{C}\cap\mathcal{H}$, there exists at least one node $j\in\{2,\ldots,r\}$ such that $|\mathcal{P}_L(1\to j)| \geq 2^L/(r-1)$. Using

(A.21), we can derive that

$$\begin{aligned}
\mathbb{E}\big[\big(T^L\varepsilon\big)_i^2\big] &= \sum_{j=1}^{n}\left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)}\left(\frac{d_i+1}{d_j+1}\right)^{1/2}\omega_i(\ell_1,\ldots,\ell_{L-1})\right]^2 \\
&\geq \left[\sum_{(i\to\ell_1\to\ldots\to\ell_{L-1}\to j)\in\mathcal{P}_L(i\to j)}\left(\frac{d_i+1}{d_j+1}\right)^{1/2}\omega_i(\ell_1,\ldots,\ell_{L-1})\right]^2 \\
&\geq \left[\frac{2^L}{r-1}\left(\frac{d_i+1}{3}\right)^{1/2}\frac{1}{(d_i+1)3^{L-1}}\right]^2 \\
&= \frac{3}{(d_i+1)(r-1)^2}1.5^{-2L}.
\end{aligned}$$

□

E EXPERIMENTS

E.1 Real Data

In this subsection, we present some of the properties of the datasets used in the main text of this paper. Figure 5 shows the differences in terms of graph topologies and graph signal (assessed in terms of their "roughness"). In particular, we note that the county graphs are extremely regular in structure, while the Twitch dataset exhibits a heavier tail in terms of degree distribution. Among the county graphs, the county election data is the smoothest. Interestingly, in this setting, the GCN convolution outperforms the GraphSAGE convolution in both denoising (see Figure 6) and prediction.

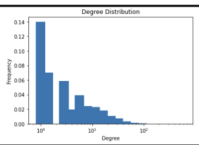
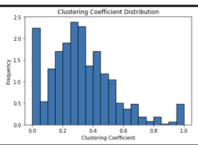
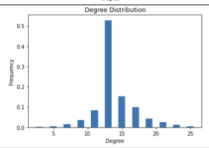
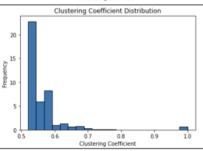
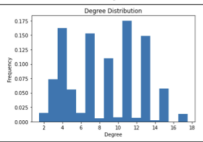
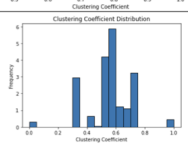
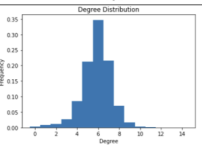
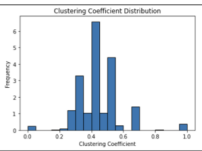
Dataset Type	# Nodes	# Edges	Degree Distribution	Clustering Coefficient	$\frac{1}{\lvert\mathcal{E}\rvert}\lVert\Delta Y\rVert_2^2$	$\lVert\Delta Y\rVert_{\max}$	$\lVert\Delta(SY)\rVert_{\max}$
Twitch	1,912	31,299			5.01	6.13	3.87
Chicago Sketch	2,176	15,104			0.74	8.51	2.19
Anaheim	914	3,881			0.52	3.93	1.84
County Education	3,234	9,483			1.14	5.49	3.25
County Election					0.08	1.61	0.69
County Income					0.80	6.70	2.73

Figure 5: Properties of the datasets used in the real-data experiment section. The last 3 columns measure the signal roughness, defined either in terms of the ℓ_2 norm ($\|\Delta\mathbf{Y}\|_2^2/|\mathcal{E}| = [\sum_{(i,j)\in\mathcal{E}}(Y_i - Y_j)^2]/|\mathcal{E}|$) or the ℓ_∞ norm ($\|\Delta\mathbf{Y}\|_{\max} = \max_{(i,j)\in\mathcal{E}}|Y_i - Y_j|$). The last column represent the graph smoothness over the neighborhood (after one convolution S).

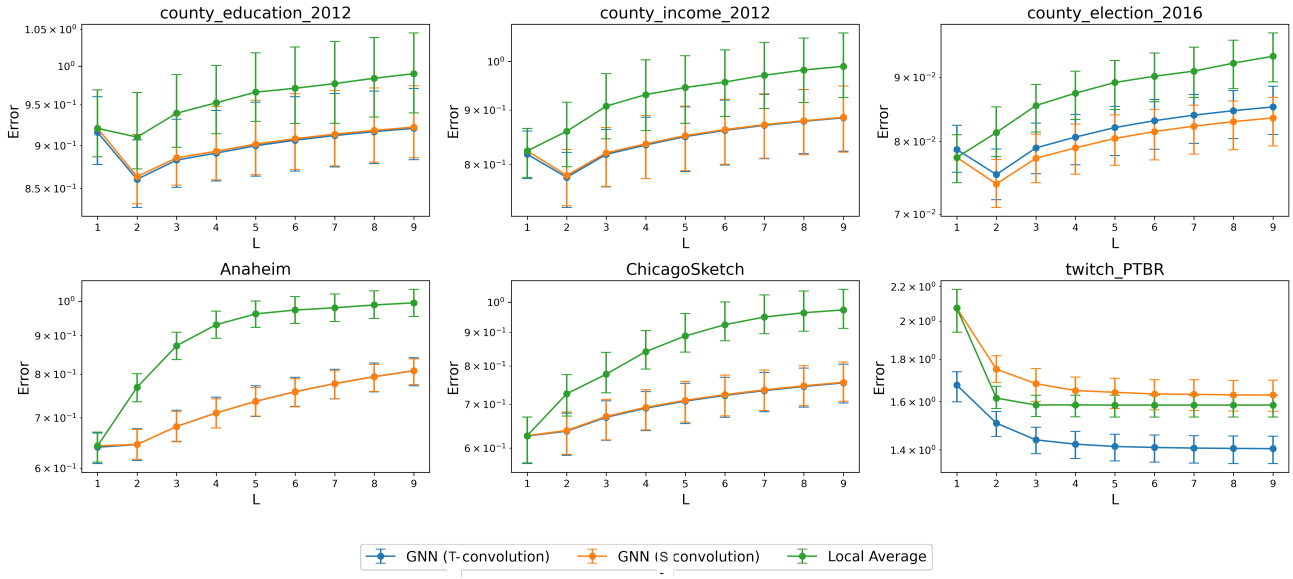


Figure 6: Results of the denoising experiment on real data across different regression datasets.

E.2 Simulated Data

In this subsection, we provide examples of the different graphs used in our synthetic experiments. The following four figures (Figures 7, 8, 9, and 10) illustrate the various graph topologies and the corresponding graph signals.

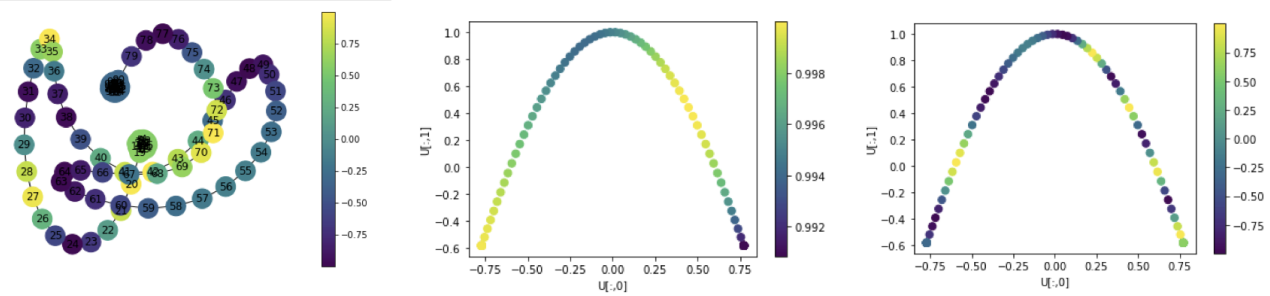
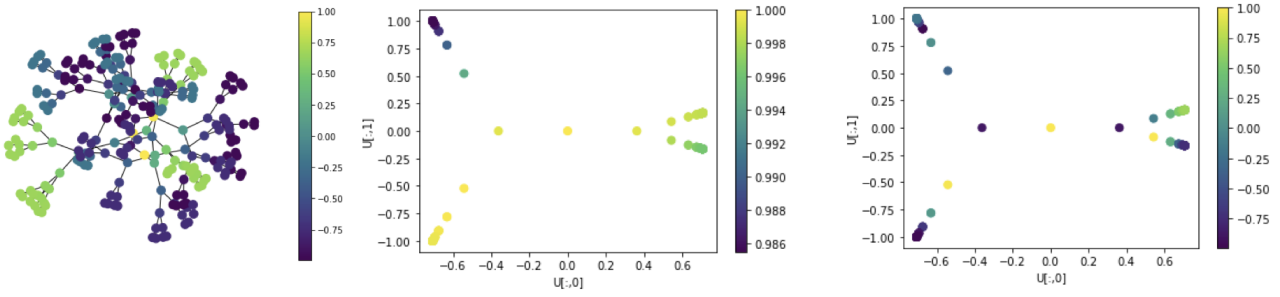


Figure 7: Barbell graph. Figure (a) shows the graph with corresponding graph signal $f_i^* = 2\cos(U_i.\beta), \beta = (-\alpha, \alpha)^\top$ with $\alpha = 10$. Here $U_i. \in \mathbb{R}^{1\times 2}$ represents the two dimensional spectral embedding of the graph. Figure (b) shows the spectral embedding, colored by signal for $\alpha = 0.001$ and figure (c) for $\alpha = 10$.

The next set of plots (Figures 11, 12 and 13) highlights the optimal number of convolutions as a function of the graph roughness. Overall, we observe the same phenomenon on the GCN and GraphSAGE convolutions as for the latent variable graph described in the main text: as the roughness of the graph increases, the optimal number of convolutions decreases. In all cases, we observe that when the neighborhood becomes too uninformative, the optimal number of layers increases again. This, however, needs to be understood alongside with the error, which increases substantially.

The next sets of plots (Figures 14, 15, 16, 17, 18 and 19) highlight the bias-variance trade-off across different types of topologies. Overall, we observe a slower decay of the variance for the GraphSAGE convolution compared to the GCN convolution, for similar levels of bias.

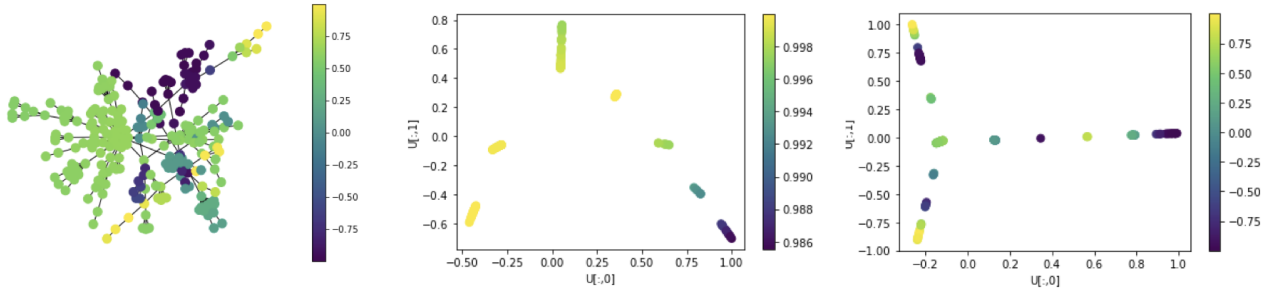


a. Binary Tree on 255 nodes

b. Spectral representation, $\alpha = 0.001$
$\|\Delta \mathbf{f}^\star\|_\infty = 0.005$

c. Spectral representation, $\alpha = 10$
$\|\Delta \mathbf{f}^\star\|_\infty = 1.89$

Figure 8: Binary Tree. Figure (a) shows the graph with corresponding graph signal $f_i^* = 2\cos(U_i.\beta), \beta = (-\alpha, \alpha)^\top$ with $\alpha = 10$. Here $U_i. \in \mathbb{R}^{1\times 2}$ represents the two dimensional spectral embedding of the graph. Figure (b) shows the spectral embedding, colored by signal for $\alpha = 0.001$ and figure (c) for $\alpha = 10$.

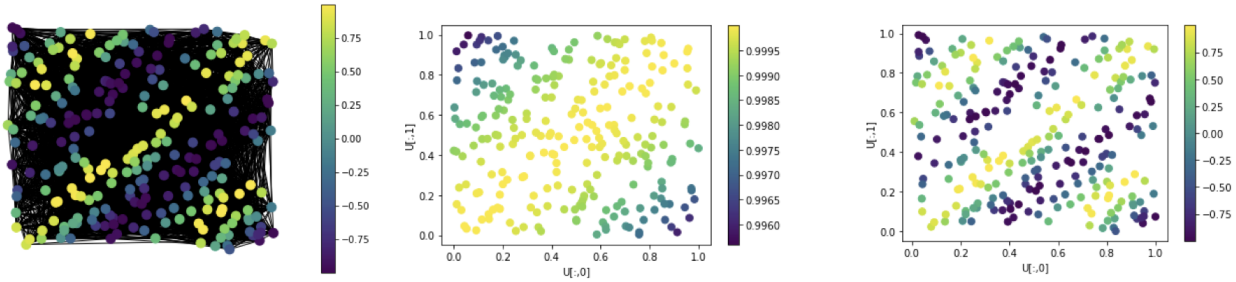


a. Power law cluster graph
on 100 nodes, m=1, p=0.5

b. Spectral representation, $\alpha = 0.001$
$\|\Delta \mathbf{f}^\star\|_\infty = 0.005$

c. Spectral representation, $\alpha = 10$
$\|\Delta \mathbf{f}^\star\|_\infty = 1.69$

Figure 9: Power Law Cluster graph. Figure (a) shows the graph with corresponding graph signal $f_i^* = 2\cos(U_i.\beta), \beta = (-\alpha, \alpha)^\top$ with $\alpha = 10$. Here $U_i. \in \mathbb{R}^{1\times 2}$ represents the two dimensional spectral embedding of the graph. Figure (b) shows the spectral embedding, colored by signal for $\alpha = 0.001$ and figure (c) for $\alpha = 10$.



a. Latent Variable Graph

b. Spectral representation, $\alpha = 0.001$
$\|\Delta \mathbf{f}^\star\|_\infty = 0.0043$

c. Spectral representation, $\alpha = 10$
$\|\Delta \mathbf{f}^\star\|_\infty = 1.99$

Figure 10: Latent Variable graph. Figure (a) shows the graph with corresponding graph signal $f_i^* = 2\cos(U_i.\beta), \beta = (-\alpha, \alpha)^\top$ with $\alpha = 10$. Here $U_i. \in \mathbb{R}^{1\times 2}$ represents the two dimensional spectral embedding of the graph. Figure (b) shows the spectral embedding, colored by signal for $\alpha = 0.001$ and figure (c) for $\alpha = 10$.

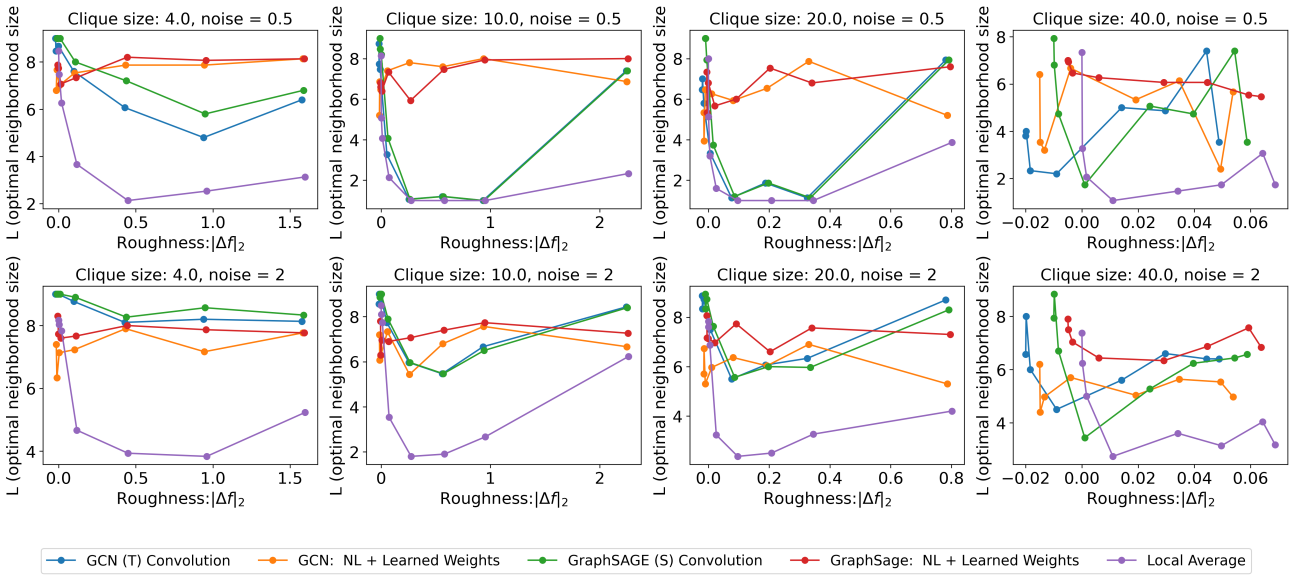


Figure 11: Optimal number of convolutions as a function of the roughness of $\mathbf{f}^*$ (here defined as $\|\Delta \mathbf{f}^*\|_2 = \sqrt{[\sum_{(i,j)\in\mathcal{E}}(f_i^* - f_j^*)^2]/|\mathcal{E}|}$) on the Barbell graph. Each column corresponds to a different clique size m, (with, for instance, 10 meaning that the two cliques on either side of the Barbell graph are of size 10) and each row to a value of the noise σ^2.

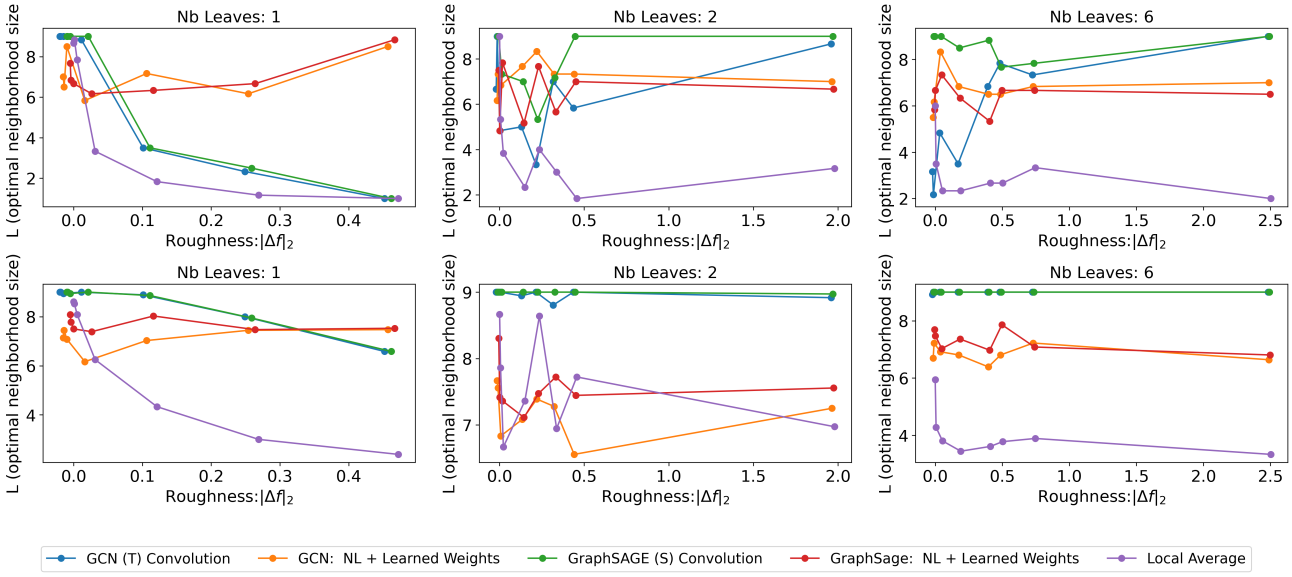


Figure 12: Optimal number of convolutions as a function of the roughness of $\mathbf{f}^*$ (here defined as $\|\Delta \mathbf{f}^*\|_2 = \sqrt{[\sum_{(i,j)\in\mathcal{E}}(f_i^* - f_j^*)^2]/|\mathcal{E}|}$) on a tree graph. Each column corresponds to a different branching number (or number of leaves) k, and each row to a value of the noise σ^2: the top row corresponds to a value of $\sigma^2 = 0.5$, while the bottom row corresponds to $\sigma^2 = 2$.

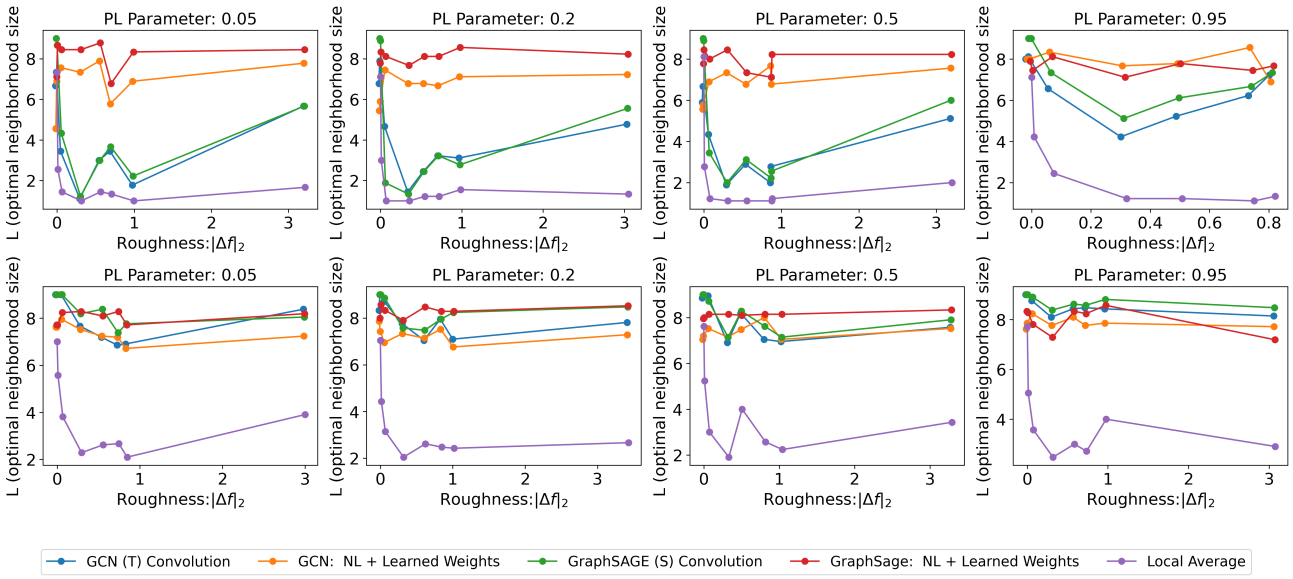


Figure 13: Optimal number of convolutions as a function of the roughness of $\mathbf{f}^*$ (here defined as $\|\Delta\mathbf{f}^*\|_2 = \sqrt{[\sum_{(i,j)\in\mathcal{E}}(f_i^* - f_j^*)^2]/|\mathcal{E}|}$) on the power-law cluster graph. Each column corresponds to a different clustering parameter p, and each row to a value of the noise σ^2: the top row corresponds to a value of $\sigma^2 = 0.5$, while the bottom row corresponds to $\sigma^2 = 2$.

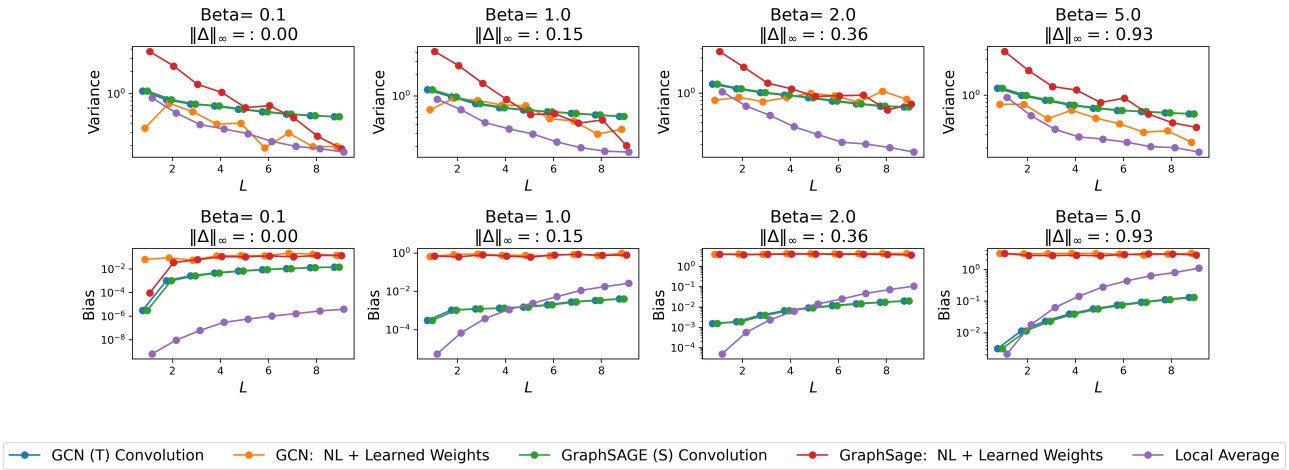


Figure 14: Bias-Variance as a function of L for the Barbell graph, clique size $m = 20$, $\sigma^2 = 2$.

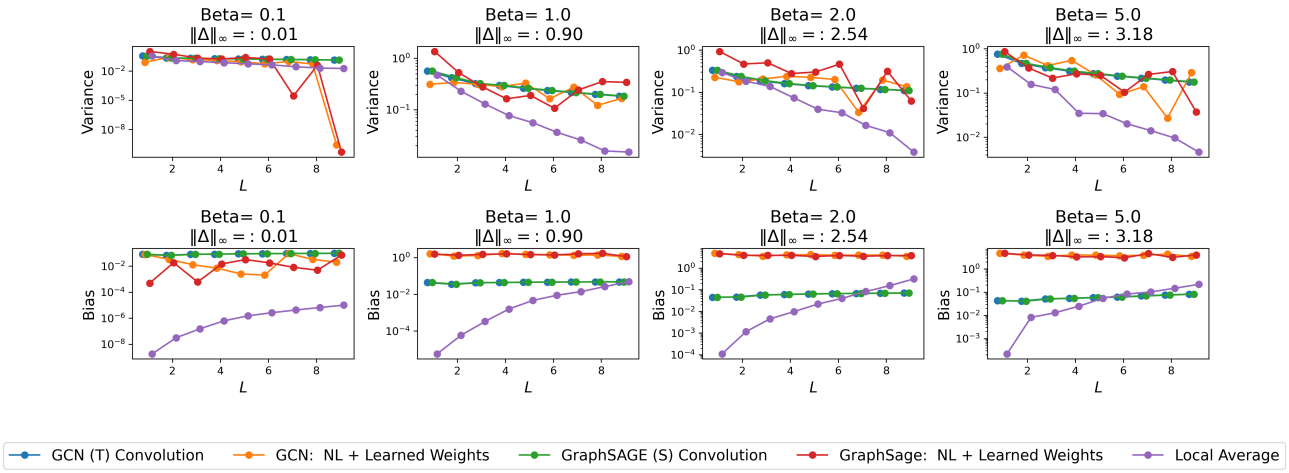


Figure 15: Bias-Variance as a function of L for the tree (with branching number equal to 2), $\sigma^2 = 1$.

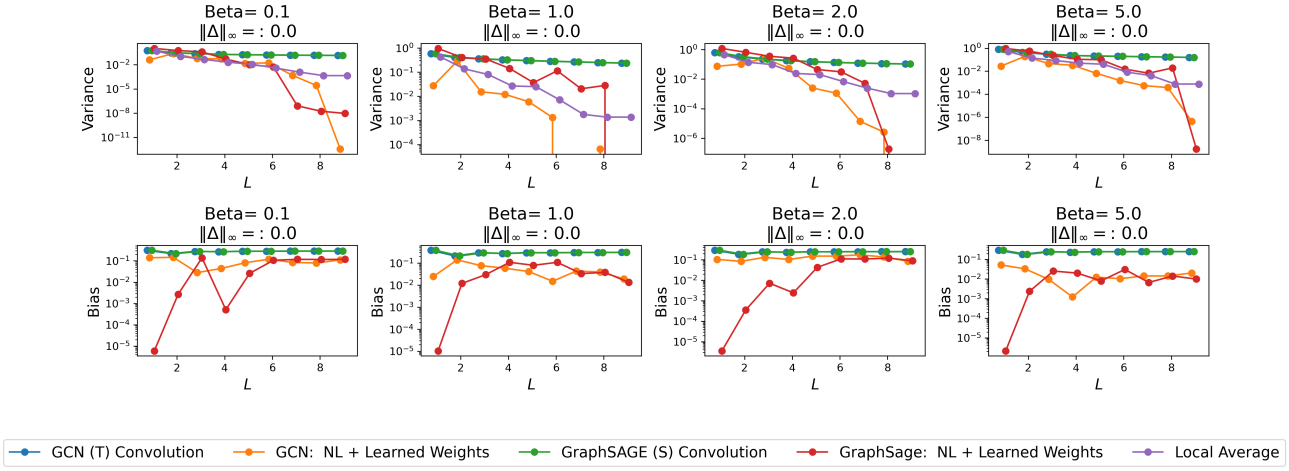


Figure 16: Bias-Variance as a function of L for the tree (with branching number equal to 4), $\sigma^2 = 1$.

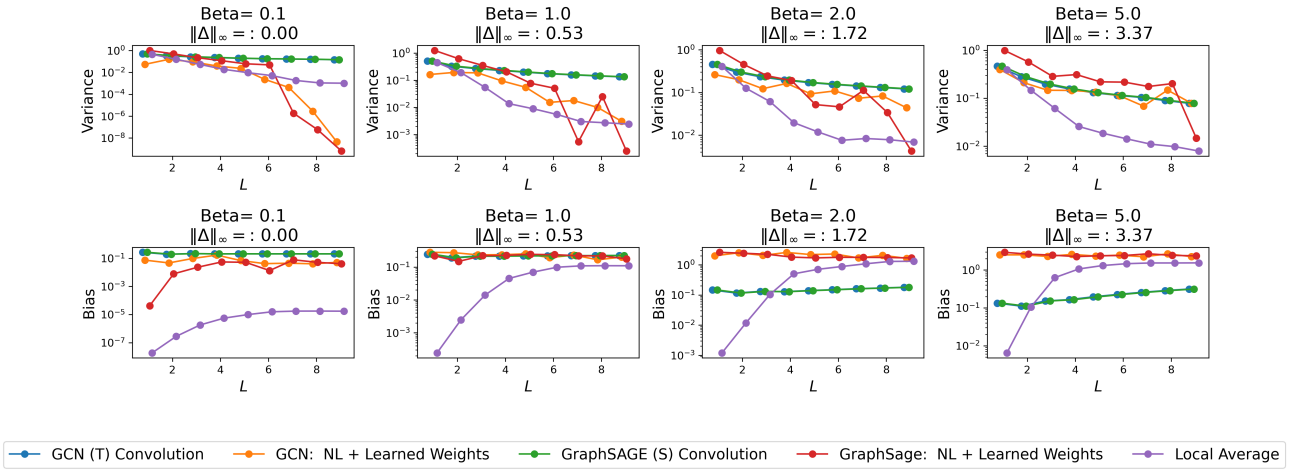


Figure 17: Bias-Variance as a function of L for the power law cluster graph ($m = 1, p = 0.1$), $\sigma^2 = 1$.

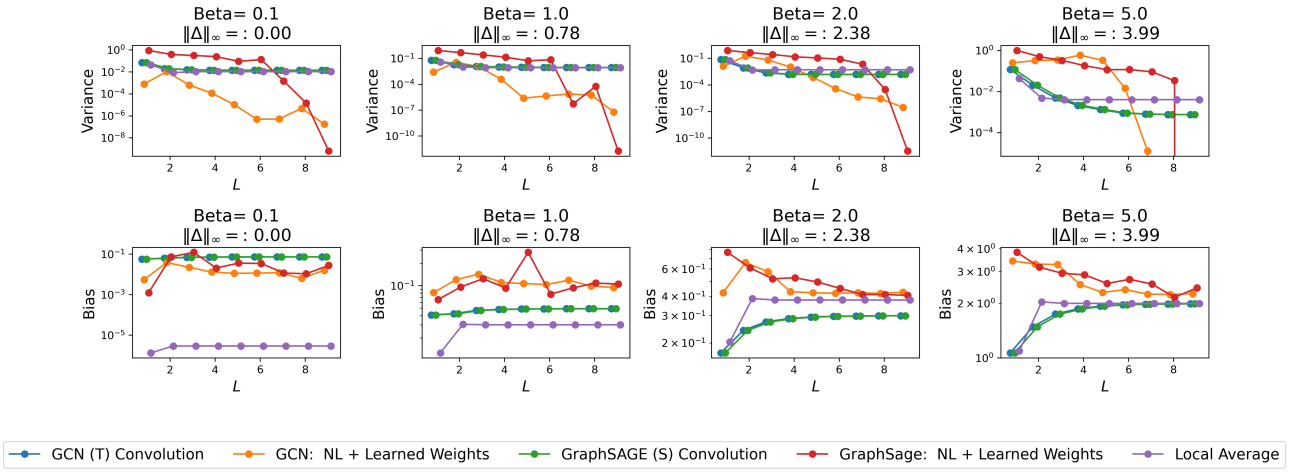


Figure 18: Bias-Variance as a function of L for the latent variable graph, $\sigma^2 = 1$.

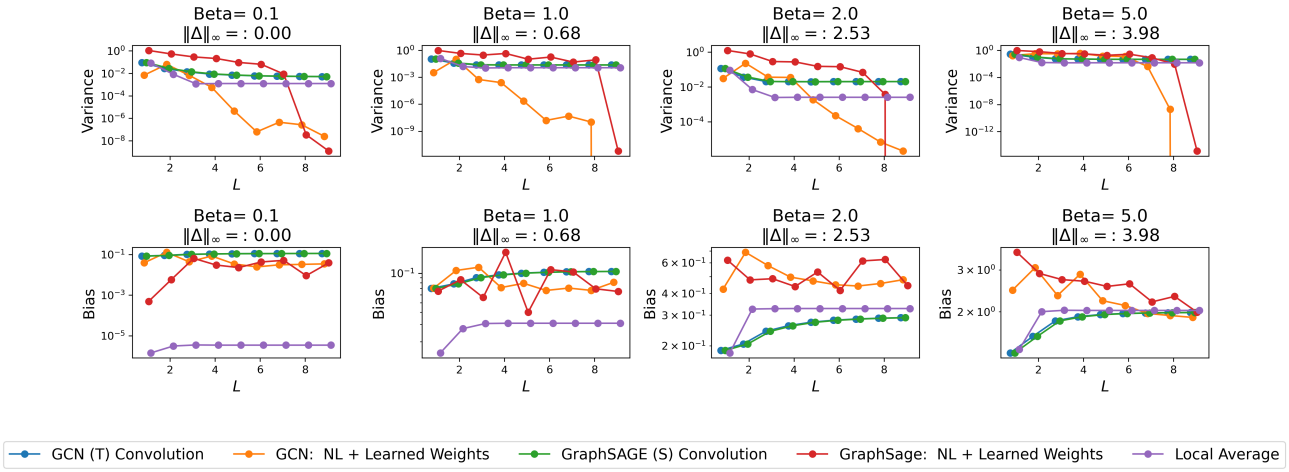


Figure 19: Bias-Variance as a function of L for the sparsified latent variable graph ($p = 0.5$), $\sigma^2 = 1$.

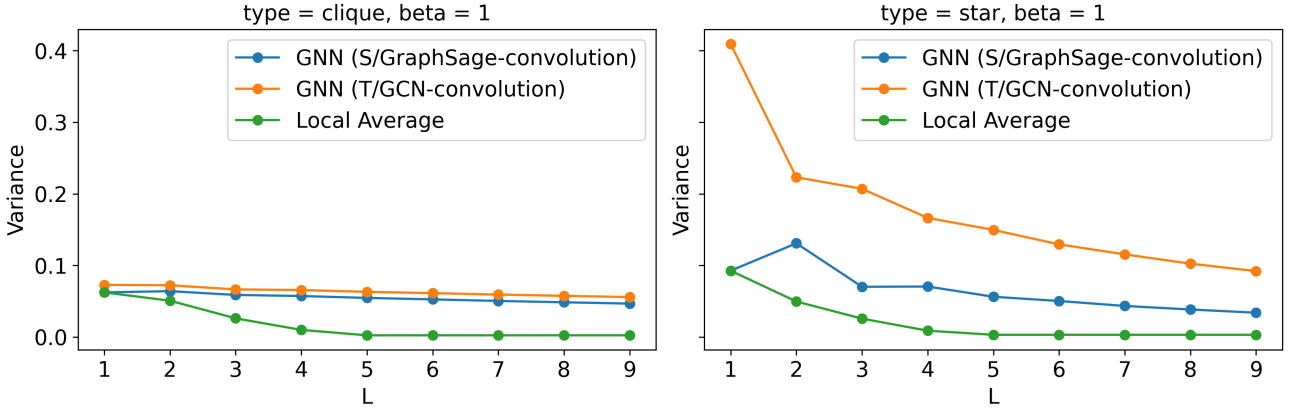


Figure 20: Variance decay of different estimators on a tree with a branching factor of 4 over four levels, with a clique of 10 nodes added (left) or a star of 10 nodes added (right).