Convolutional Neural Networks on Manifolds: From Graphs and Back

Anonymous Author(s)
Affiliation
Address
email

Abstract

In recent years, geometric deep learning has gained attraction due to both the need for machine learning on structured data (e.g., graphs) and the increasing availability of this type of data. Extensions of deep convolutional architectures to non-Euclidean domains in particular are a powerful technique in sensor network applications — which can be seen as graphs — and 3D model analysis — which can be seen as manifolds. While recent works have provided a better theoretical understanding of why convolutional neural network architectures work well on graphs of moderate size, in the large-scale regime that is the setting of most problems of interest, their behavior is not as well understood. In this paper, we bridge this gap by modeling large graphs as samples from manifolds and studying manifold neural networks (MNNs). Our main contribution is to define a manifold convolution operation which, when "discretized" in both the space and time domains, is consistent with the practical implementation of a graph convolution. We then show that graph neural networks (GNNs) can be particularized from MNNs, which in turn are the limits of these GNNs. We conclude with numerical experiments showcasing an application of the MNN to point-cloud classification.

1 Introduction

Convolutional neural networks (CNNs) have achieved impressive success in a wide range of applications, including but not limited to natural language processing [33], image denoising [42] and video analysis [40]. Due to their remarkable performance, CNNs are recognized as the de facto deep learning model to process data lying in Euclidean domains. However, many modern problems of practical interest — e.g., detection and recommendation in social networks [1, 9], resource allocation over wireless networks [35], and point cloud analysis for shape segmentation [39, 34] — require processing data that is non-Euclidean. In light of this, recent works on geometric deep learning [11, 5, 22, 20] have focused on extending convolutional models to such domains.

Among these models, graphs are commonly used to construct a discrete model that captures the underlying geometric structure. Convolutions can be readily extended to graph convolutions, which allows defining graph neural networks (GNNs) [13, 11, 43, 30, 7]. In graphs of moderate size, GNNs work well empirically, which is backed by recent theoretical findings on their expressive power [15, 41] and stability [14, 31, 29, 16]. However, in the large-scale regime which is the setting of most problems of practical interest, their behavior is not as well understood. Even if CNNs on manifolds, from which graphs can be seen as samples [3, 5], have also been proposed [37, 20, 22], the relationship between CNNs on graphs and CNNs on manifolds remains elusive. In this paper, we bridge this gap by defining a manifold convolution which can be particularized on sampled graphs from the manifold. This allows seeing GNNs as samples from manifold neural networks (MNNs).

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

Because of the convergence of graphs to manifolds [3, 6, 8], we can further show that GNNs built from these convolutions converge to MNNs.

Our definition of manifold convolution is based on the heat diffusion process. By cascading layers consisting of manifold convolutions and pointwise nonlinearities, we then define manifold neural networks (MNNs). Since we do not have access to the whole manifold — only a discrete set of its points — we show how the MNN particularizes a GNN on the graph connecting the sampled points from the manifold, i.e., how to go from manifolds to graphs. We also show how this GNN converges to the underlying MNN when the input data is bandlimited, i.e., how to go from graphs to manifolds. For practical implementation purposes, we further discretize the time axis of the heat diffusion process, recovering the usual (learnable) parametrization of the spatial graph convolution [13] when the time horizon is finite. Finally, we verify the performance of our proposed architecture with numerical experiments on a point cloud dataset.

Related works include neural networks built on graphons [27, 28], which are limits of sequences of dense graphs. Different from manifolds, graphons only represent the limits of graphs with unbounded degrees [19, 4]. The convergence of GNNs on random graphs is proposed in [16] while the limit is a constructed continuous GNN. The results in [18] show the convergence and transferability of GNNs if the graphs are sampled with a specific sampling operator from a continuous topological space. However, the sampling operator requires the knowledge of the spectrum of the underlying topological space, which in practice is often inaccessible. CNNs on manifolds have been constructed in [20] with geodesic polar coordinates and in [22] with mixtures of Gaussian kernels. Both works only focus on spatial approaches. Stability of MNNs has also been studied, by considering the perturbations to the Laplace-Beltrami operator [37, 36] in a spectral approach. All these works on MNNs lack the explicit relationship with GNNs, which are commonly implemented for realizations on manifolds [5, 22]. A general framework for algebraic neural networks has been proposed for architectures with commutative algebras [25].

The rest of the paper is organized as follows. We start with some preliminary concepts and define the manifold convolutions as well as the manifold neural networks in Section 2. In Section 3, we implement the discretization in space and time domains to make the MNNs realizable which also bring back GNNs and we also prove the convergence of GNNs to MNNs. Our proposed MNN is verified in a model classification problem in Section 4. The conclusions are presented in Section 5.

2 Manifold Convolutions and Manifold Neural Networks

2.1 Manifold Convolution

We consider a compact, smooth, and differentiable d-dimensional submanifold $\mathcal{M}$ embedded in $\mathbb{R}^N$. The embedding induces a Riemannian structure [10] on $\mathcal{M}$ which endows a measure μ over the manifold. *Manifold data* supported on $\mathcal{M}$ are smooth scalar functions $f : \mathcal{M} \to \mathbb{R}$, which we refer to as *manifold signals* in the following. We consider f belongs to $L^2(\mathcal{M})$ in which we define the inner product as

$$\langle f, g \rangle_{L^2(\mathcal{M})} = \int_{\mathcal{M}} f(x) g(x) \mathrm{d}\mu(x) \tag{1}$$

with the norm defined as $\|f\|^2_{L^2(\mathcal{M})} = \langle f, f \rangle_{L^2(\mathcal{M})}$, representing the energy of manifold signal f. We consider manifold signals f with finite energy.

The manifold is locally Euclidean and we denote the local Euclidean space around $x \in \mathcal{M}$ as tangent space $T_x\mathcal{M}$. We use $T\mathcal{M}$ to represent the disjoint union of all tangent spaces on $\mathcal{M}$. The *intrinsic gradient* for differentiation is a local operator [5] and can thus be written as an operator $\nabla : L^2(\mathcal{M}) \to L^2(T\mathcal{M})$ mapping scalar functions to tangent vector functions on $\mathcal{M}$. The adjoint operator of intrinsic gradient is the *intrinsic divergence* defined as $\mathrm{div} : L^2(T\mathcal{M}) \to L^2(\mathcal{M})$. Based on these two differentiation operators, the Laplace-Beltrami (LB) operator $\mathcal{L} : L^2(\mathcal{M}) \to L^2(\mathcal{M})$ can be defined as the intrinsic divergence of the intrinsic gradient [26], formally as

$$\mathcal{L} f = -\mathrm{div} \circ \nabla f = -\nabla \cdot \nabla f. \tag{2}$$

Similar to the Laplacian operator in Euclidean domains or the Laplace matrix in graphs [23], the LB operator evaluates how much the function value at point x differs from the average function value of its neighborhood [5]. Since $\mathcal{L}$, like ∇, is a local operator depending on the tangent space $T_x\mathcal{M}$

of each point $x \in \mathcal{M}$, in the following we make this dependence explicit by writing $\mathcal{L} = \mathcal{L}_x$ and $\nabla = \nabla_x$.

The LB operator is essential to capture the heat diffusion over manifolds by the *heat equation*

$$\frac{\partial u(x,t)}{\partial t} + \mathcal{L}u(x,t) = 0, \tag{3}$$

where $u(x,t) \in L^2(\mathcal{M})$ measures the temperature at $x \in \mathcal{M}$ at time $t \in \mathbb{R}^+$. This can be interpreted to mean that, at point x, the rate at which the manifold "cools down" is proportional to the difference between the temperature of x and the local average of the temperature of the points in its neighborhood. With initial condition given by $u(x,0) = f(x)$, the solution can be expressed as $u(x,t) = e^{-t\mathcal{L}}f(x)$. We define the manifold convolution with a filter impulse response function $\tilde{h}$ and input manifold signal f as follows.

Definition 1 (Manifold convolutional filter) *Let $\tilde{h} : \mathbb{R}^+ \to \mathbb{R}$ and let $f \in L^2(\mathcal{M})$ be the data supported on $\mathcal{M}$. The manifold filter with impulse response $\tilde{h}$, denoted as $\mathbf{h}$, is given by*

$$g(x) = (\mathbf{h}f)(x) := \int_0^\infty \tilde{h}(t)u(x,t)dt = \int_0^\infty \tilde{h}(t)e^{-t\mathcal{L}}f(x)dt = \mathbf{h}(\mathcal{L})f(x), \tag{4}$$

where $u(x,t)$ is the solution of the heat equation (3) *with $u(x,0) = f(x)$.*

Manifold filters are local spatial operators operating directly on points on the manifold based on the LB operator. The exponential term $e^{-t\mathcal{L}}$ can be interpreted as a shift operator like the time delay in a Linear-Time Invariant (LTI) filter [24] and the graph shift in a Linear-Shift Invariant (LSI) graph filter [11]. In fact, manifold filters can recover graph filters by discretization, which we discuss thoroughly in Section 3.

The LB operator $\mathcal{L}$ is self-adjoint and positive-semidefinite. Considering that manifold $\mathcal{M}$ is compact without boundary, the operator $\mathcal{L}$ possesses a real discrete spectrum $\{\lambda_i\}_{i=1}^\infty$ with the eigenvalues λ_i and the corresponding eigenfunctions ϕ_i satisfying $\mathcal{L}\boldsymbol{\phi}_i = \lambda_i\boldsymbol{\phi}_i$. Eigenvalue λ_i can be interpreted as the canonical frequency and the eigenfunction ϕ_i as the canonical oscillation mode. By projecting a manifold signal f onto the eigenfunction, we can write the *frequency representation* $\hat{f}$ as

$$[\hat{f}]_i = \langle f, \boldsymbol{\phi}_i \rangle_{L^2(\mathcal{M})} = \int_{\mathcal{M}} f(x)\boldsymbol{\phi}_i(x)\mathrm{d}\mu(x). \tag{5}$$

Based on this concept, we can define the bandlimited manifold signal as follows.

Definition 2 (Bandlimited manifold signal) *A manifold signal $f \in L^2(\mathcal{M})$ is defined as λ_M-bandlimited if $[\hat{f}]_i = 0$ for all i such that $\lambda_i > \lambda_M$.*

The spectrum and eigenbasis of the LB operator help to understand not only the frequency behavior of manifold signal but the manifold filter $\mathbf{h}(\mathcal{L})$. The frequency representation of manifold filter output g can be similarly written as

$$[\hat{g}]_i = \int_{\mathcal{M}} \int_0^\infty \tilde{h}(t)e^{-t\mathcal{L}}f(x)\mathrm{d}t\boldsymbol{\phi}_i(x)\mathrm{d}\mu(x). \tag{6}$$

By substituting $e^{-t\mathcal{L}}\phi_i = e^{-t\lambda_i}\phi_i$, we can get

$$[\hat{g}]_i = \int_0^\infty \tilde{h}(t)e^{-t\lambda_i}\mathrm{d}t[\hat{f}]_i. \tag{7}$$

The function solely dependent on λ_i is defined as the *frequency response* of the filter $\mathbf{h}(\mathcal{L})$, which can be stated formally as follows.

Definition 3 (Frequency response of manifold filter) *The frequency response of the filter $\mathbf{h}(\mathcal{L})$ is given by*

$$\hat{h}(\lambda) = \int_0^\infty \tilde{h}(t)e^{-t\lambda}dt, \tag{8}$$

which leads (7) *to $[\hat{g}]_i = \hat{h}(\lambda_i)[\hat{f}]_i$.*

Definition 3 indicates that the frequency response of manifold filter is point-wise in frequency domain. Combining the frequency representation of $\hat{g}$ over the whole spectrum, we can obtain the frequency representation of manifold filter $\mathbf{h}$ as

$$g = \mathbf{h}(\mathcal{L})f = \sum_{i=1}^{\infty} \hat{h}(\lambda_i)\langle f, \phi_i \rangle_{L^2(\mathcal{M})}\phi_i. \tag{9}$$

If we consider the input manifold signal f as λ_M-bandlimited (Definition 2), the output signal g can be written as

$$g = \mathbf{h}(\mathcal{L})f = \sum_{i=1}^{M} \hat{h}(\lambda_i)\langle f, \phi_i \rangle_{L^2(\mathcal{M})}\phi_i, \tag{10}$$

where $M = \#\{\lambda_i \leq \lambda_M\}_i$ counts the number of eigenvalues within the bandwidth.

2.2 Manifold Neural Networks

Manifold neural networks (MNNs) augment manifold filters as we have defined in Definition 1 with a point-wise nonlinear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ which is an independent application on each point of the manifold. We further make an assumption on its continuity as follows.

Assumption 1 (Normalized Lipschitz activation functions) *The activation function σ is normalized Lipschitz continuous, i.e., $|\sigma(a) - \sigma(b)| \leq |a - b|$, with $\sigma(0) = 0$.*

Note that this assumption is rather reasonable, since most common activation functions (e.g., the ReLU, the modulus and the sigmoid) are normalized Lipschitz by design.

In a single-layer MNN, the manifold signal f is passed through a manifold filter followed by a point-wise nonlinearity as

$$f_1(x) = \sigma\Bigg(\mathbf{h}(\mathcal{L})f(x)\Bigg), \tag{11}$$

which can be seen as a basic nonlinear processing of the input manifold signal. By stacking this procedure in layers, a multi-layer MNN can be constructed which can be formally written as a function composition. The output manifold signal of a layer becomes the input signal of the next layer. Let $l = 1, 2, \cdots, L$ stand for the index for the layer and $\mathbf{h}_l(\mathcal{L})$ as the manifold filter on each layer. For a specific layer l, filter $\mathbf{h}_l(\mathcal{L})$ takes the output $f_{l-1}(x)$ as the input produces the output of layer l as

$$f_l(x) = \sigma\Bigg(\mathbf{h}_l(\mathcal{L})f_{l-1}(x)\Bigg), \tag{12}$$

where $f_0(x) = f(x)$ is the given input manifold signal. After a recursive applications through L layers, we can get the output of the MNN as $f_L(x)$.

When considering multiple features in each layer to increase the representation power of MNN, the manifold filters map the input F_{l-1} features from layer $l-1$ to F_l intermediate features in layer l with a bank of manifold filters, i.e.,

$$y_l^p(x) = \sum_{q=1}^{F_{l-1}} \mathbf{h}_l^{pq}(\mathcal{L})f_{l-1}^q(x), \tag{13}$$

where $\mathbf{h}_l^{pq}(\mathcal{L})$ is the filter mapping the q-th feature from layer $l-1$ to the p-th feature of layer l, for $1 \leq q \leq F_{l-1}$ and $1 \leq p \leq F_l$. The intermediate features are then processed by the nonlinearity σ as

$$f_l^p(x) = \sigma\Bigg(y_l^p(x)\Bigg). \tag{14}$$

The output of layer L is the output of MNN with F_L features. To represent the MNN more precisely, we gather the impulse responses of all the manifold filters $\mathbf{h}_l^{pq}$ as a function set $\mathbf{H}$ and define the MNN as a map $\mathbf{\Phi}(\mathbf{H}, \mathcal{L}, f)$. This map emphasizes that the neural network is parameterized by both the filter functions $\mathbf{H}$ and the LB operator $\mathcal{L}$.

3 Discretization in the Space and Time Domains

MNNs are built from manifold convolutional filters (Definition 1) operating on a continuous manifold and over an infinite continuous time horizon. This makes it impractical to implement directly the neural network architecture described by (14) in applications. In this section, we discuss the practical application of MNNs and the connections with GNNs (14) over a set of discrete samples from the manifold in a finite and discrete time frame.

3.1 Discretization in the Space Domain

In practice, to operate on a continuous topological space, it is natural to access the underlying geometric structure by sampling points over the continuous domain. Therefore we sample uniformly on the manifold and connect the sampled points as a graph to approximate the underlying manifold [8, 3]. Specifically, we model the set of n sampled points as $X = \{x_1, x_2, \ldots, x_n\}$ which are sampled i.i.d. from measure μ of manifold $\mathcal{M} \subset \mathbb{R}^N$. A complete weighted symmetric graph $\mathbf{G}_n$ can be constructed by seeing the sampled points as the vertices of the graph. The edge weight w_{ij} connecting point x_i and x_j is defined as a graph signal

$$w_{ij} = \frac{1}{n} \frac{1}{t_n (4\pi t_n)^{k/2}} \exp\left(-\frac{\|x_i - x_j\|^2}{4t_n}\right), \tag{15}$$

with $\|x_i - x_j\|$ representing the Euclidean distance between x_i and x_j. Parameter t_n controls the chosen Gaussian kernel [3]. The adjacency matrix of $\mathbf{G}_n$ is thus defined as $[\mathbf{A}_n]_{ij} = w_{ij}$ for $1 \leq i, j \leq n$ with $\mathbf{A}_n \in \mathbb{R}^{n \times n}$. The correspondent graph Laplacian matrix $\mathbf{L}_n$ [21] thus can be calculated as $\mathbf{L}_n = \text{diag}(\mathbf{A}_n \mathbf{1}) - \mathbf{A}_n$.

We define a uniform sampling operator $\mathbf{P}_n : L^2(\mathcal{M}) \to L^2(\mathbf{G}_n)$ to discretize the signal f on manifold (Definition 2) as the data supported on the graph $\mathbf{G}_n$, which is denoted as

$$\mathbf{x}_n = \mathbf{P}_n f \text{ with } [\mathbf{x}_n]_i = f(x_i), \quad x_i \in X, \tag{16}$$

where the i-th entry of the graph signal $\mathbf{x}_n$ is the manifold signal f evaluated at the sampled point x_i.

Considering that the discrete points $\{x_1, x_2, \ldots, x_n\}$ are uniformly sampled from manifold $\mathcal{M}$ with measure μ, the empirical measure associated with $\mathrm{d}\mu$ can be denoted as $p_n = \frac{1}{n} \sum_{i=1}^n \delta_{x_i}$, where δ_{x_i} is the Dirac measure supported on x_i. Similar to the inner product defined in the $L^2(\mathcal{M})$ space (1), the inner product on $L^2(\mathbf{G}_n)$ is denoted as

$$\langle u, v \rangle_{L^2(\mathbf{G}_n)} = \int u(x) v(x) \mathrm{d}p_n = \frac{1}{n} \sum_{i=1}^n u(x_i) v(x_i). \tag{17}$$

The norm in $L^2(\mathbf{G}_n)$ is therefore $\|u\|^2_{L^2(\mathbf{G}_n)} = \langle u, u \rangle_{L^2(\mathbf{G}_n)}$, with $u, v \in L^2(\mathcal{M})$. For signals $\mathbf{u}, \mathbf{v} \in L^2(\mathbf{G}_n)$, the inner product is therefore $\langle \mathbf{u}, \mathbf{v} \rangle_{L^2(\mathbf{G}_n)} = \frac{1}{n} \sum_{i=1}^n [\mathbf{u}]_i [\mathbf{v}]_i$.

The parametrization of manifold filter (Definition 1) enables us to replace the LB operator with the graph Laplacian operator $\mathbf{L}_n$ and turns $\mathbf{h}$ to a graph convolutional filter, i.e.,

$$\mathbf{z}_n = \int_0^\infty \tilde{h}(t) e^{-t\mathbf{L}_n} \mathrm{d}t \mathbf{x}_n = \mathbf{h}(\mathbf{L}_n) \mathbf{x}_n, \ \mathbf{x}_n, \mathbf{z}_n \in \mathbb{R}^n, \tag{18}$$

where $\mathbf{z}_n$ is the output graph signal. By cascading the layers containing the graph convolutional filters as defined in (18) and point-wise nonlinearities, we can define a neural network on this constructed graph $\mathbf{G}_n$ as

$$\mathbf{x}^p_{n,l} = \sigma\left(\sum_{q=1}^{F_{l-1}} \mathbf{h}^{pq}_l(\mathbf{L}_n) \mathbf{x}^q_{n,l-1}\right). \tag{19}$$

Similarly, we can represent this neural network as a map $\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_n, \mathbf{x}_n)$ parametrized by the filter functions $\mathbf{H}$ and the graph Laplacian $\mathbf{L}_n$. As the number of sampling points n goes to infinity, the discrete graph signal $\mathbf{x}_n$ converges to the manifold signal f while the discrete graph Laplacian matrix $\mathbf{L}_n$ has also been proved to converge to the LB operator $\mathcal{L}$ of the underlying manifold [3, 6, 8]. By introducing the definition of Lipschitz continuous manifold filters in Definition 4, we can combine the convergence of signals ($\mathbf{x}_n \xrightarrow{p} f$) and Laplace operators ($\mathbf{L}_n \xrightarrow{p} \mathcal{L}$) to derive the convergence of $\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_n, \mathbf{x}_n) \xrightarrow{p} \mathbf{\Phi}(\mathbf{H}, \mathcal{L}, f)$.

Definition 4 (Manifold filter with Lipschitz continuity) *A manifold filter is C-Lispchitz if its frequency response is Lipschitz continuous with constant C, i.e,*

$$|\hat{h}(a) - \hat{h}(b)| \leq C|a - b| \text{ for all } a, b \in (0, \infty). \tag{20}$$

We conclude that the outputs of the neural network on the graph converge to the outputs of the continuous MNN as Theorem 1 shows under the mild assumption on the amplitude of the Lipschitz continuous manifold filter frequency response (Assumption 2).

Assumption 2 (Non-amplifying filters) *A manifold filter is non-amplifying if for all $\lambda \in (0, \infty)$, its frequency response $\hat{h}$ satisfies $|\hat{h}(\lambda)| \leq 1$.*

Note that this assumption is reasonable, because the filter function $\hat{h}(\lambda)$ can always be normalized.

Theorem 1 *Let $X = \{x_1, x_2, \ldots, x_n\}$ be a set of points sampled i.i.d. from a d-dimensional manifold $\mathcal{M} \subset \mathbb{R}^N$. Let $\mathbf{G}_n$ be a graph constructed from X with weight values set as* (15) *with $t_n = n^{-1/(d+2+\alpha)}$ and $\alpha > 0$. Let $\mathbf{x}_n$ be the data supported on $\mathbf{G}_n$ sampled by operator $\mathbf{P}_n$ as* (16) *from a bandlimited manifold signal f. Let $\mathbf{\Phi}(\mathbf{H}, \cdot, \cdot)$ be the neural network parameterized by the LB operator $\mathcal{L}$ of manifold $\mathcal{M}$* (14) *or by the graph Laplacian $\mathbf{L}_n$ of $\mathbf{G}_n$. Under the assumption that the filters in $\mathbf{H}$ are Lipschitz continuous and non-amplifying, it holds that*

$$\lim_{n\to\infty} \|\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_n, \mathbf{P}_n f) - \mathbf{P}_n \mathbf{\Phi}(\mathbf{H}, \mathcal{L}, f)\|_{L^2(\mathbf{G}_n)} = 0, \tag{21}$$

where the limits are taken in probability.

Proof. See Supplementary material. ■

Theorem 1 states that, for bandlimited input signals, neural networks on graphs generated from a manifold actually converge to MNNs under certain assumptions as the graph size grows to infinity. Neural networks supported on large graphs have been proved to be stable and transferable empirically in many applications [35, 12]. Our result provides a critical theoretical support for understanding these properties of large graph neural networks regardless of graph sparsity by considering the MNN as their limit. We remark that the definition of graph convolutional filter in (18) is a continuous form based on the integration over the whole time domain.

3.2 Discretization in the Time Domain

Generally, we operate on the neural networks in practice in digital systems with digital filters. This indicates that the time horizon on which the convolutional filters are usually defined is not continuous, but discrete. Therefore we discretize the impulse response function $\tilde{h}(t)$ of the manifold filter with a fixed sampling interval $T_s = 1$ and replace the filter response function with a series of coefficients $h_k = \tilde{h}(k)$, $k = 0, 1, 2 \ldots$. In order to make the digital filters learnable and realizable, we further fix a finite time horizon with K samples and rewrite the manifold filter as (22)

$$\mathbf{h}(\mathcal{L})f(x) = \sum_{k=0}^{K-1} h_k e^{-k\mathcal{L}} f(x), \tag{22}$$

which corresponds to the form of a finite impulse response (FIR) filter with a shift operator $e^{-\mathcal{L}}$.

The manifold convolution in the discrete time domain can therefore be written as a summation over K time steps instead of the integration form. If we discretize the time domain of the graph convolutional filter defined in (18), this leads to a completely practical manifold filter supported on a constructed graph model and discrete finite time steps, i.e.,

$$\mathbf{z}_n = \mathbf{h}(\mathbf{L}_n)\mathbf{x}_n = \sum_{k=0}^{K-1} h_k e^{-k\mathbf{L}_n} \mathbf{x}_n, \quad \mathbf{x}_n, \mathbf{z}_n \in \mathbb{R}^n. \tag{23}$$

We observe that (23) recovers the form of graph convolution [11, 31] with $e^{-\mathbf{L}_n}$ seen as the graph shift operator. By replacing the filter $\mathbf{h}_l^{pq}(\mathbf{L}_n)$ in (19) with the form of (23), we further recovers

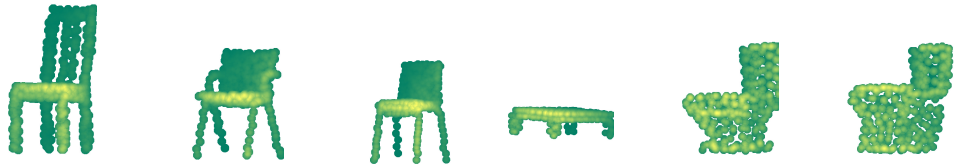

Figure 1: Point cloud models with $n = 300$ sampling points in each model. Our goal is to identify chair models from other models such as toilet and table.

GNNs with MNNs in this discrete finite time frame. Up until now, we have completed our process of building MNNs from graphs and back, i.e., we relate GNNs with MNNs in a two-way connection. We have shown that neural networks on the graphs sampled from the manifold converge to neural networks built on the manifold when the input manifold signal is bandlimited. This provides a theoretical support for analyzing the stability and transferability of large graph neural networks supported either on dense or sparse graphs. The discretization over the space and time domains enables the particularization as GNNs constructed on the sampled points over the manifold and the practical implementation of MNNs. We remark that Theorem 1 also indicates that our proposed MNN can be well approximated by a large enough GNN sampled from the MNN, which supports our simulation setup in the following section theoretically.

4 Numerical Experiments

We evaluate the performance of our proposed MNN structure with approximated GNNs. We carry out a classification problem with ModelNet10 dataset [38]. The dataset contains 3,991 meshed CAD models from 10 categories for training and 908 models for testing.

Architecture	error rates
GNN1Ly	$8.04\% \pm 0.88\%$
GNN2Ly	$4.30\% \pm 2.64\%$
GF1Ly	$13.77\% \pm 6.87\%$
GF2Ly	$12.22\% \pm 7.89\%$

Table 1: Classification error rates averaged over 5 data realizations.

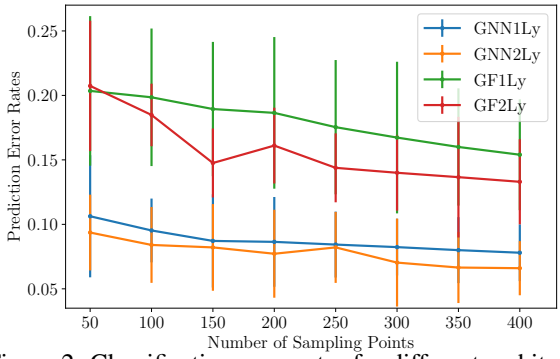

Figure 2: Classification error rates for different architectures on graphs with $n = 100, 200, 300, 400$ over 5 realizations.

We sample n points from each meshed model uniformly and construct a dense graph. Explicitly, we see each point as a node and the edge weights are calculated with (15). The point cloud models are approximated by the constructed graphs. The edge weights are calculated according to (15) with t_n set as $n^{-1/5}$. The Laplacian matrix is calculated for each point cloud model. Our goal is to identify the models for chair from other models as illustrated in Figure 1. We implement graph filters with 1 (GF1Ly) and 2 layers (GF2Ly) along with graph neural networks with 1 (GNN1Ly) and 2 layers (GNN2Ly) respectively. The single layer architectures of graph filters and graph neural networks contain $F_0 = 3$ input features which are the coordinates of every point in 3d space. The output features are set as $F_1 = 64$. The architectures with 2 layers include another layer with $F_2 = 32$ features. The number of filter taps in each layer are set as $K = 5$. The nonlinearity is a ReLu function. All the architectures are concluded with a linear readout layer mapping the output features to a binary scalar to estimate the classifications. We train all the architectures with an ADAM optimizer [17]

with the learning rate as 0.005 and decaying factor as 0.9, 0.999 to minimize the entropy loss. The training model set is divided into batches with 10 models over 40 epochs. We repeat 5 sampling realizations for all the architectures and calculate the average classification error rates as well as the standard deviation.

The classification error rates for $n = 300$ are shown in Table 1. We observe that GNNs perform better than the GFs while architectures with more layers learn more accurate models with more parameters learned. The averaged error rates for different architectures are shown in Figure 2 for increasing value of number of sampling points n. We observe that, for all architectures the error rates decrease with n. This indicates that GNN constructed on $\mathbf{L}_n$ converges as the graphs $\mathbf{G}_n$ grow, as expected from Theorem 1.

5 Conclusions

We have defined a manifold convolution operation with the heat diffusion controlled by the Laplace-Beltrami operator. We have further constructed a manifold neural network architecture by conscading the manifold filters and nonlinearities. To realize the MNN in practice, we have carried out discretization in both space and time domains which recovers the convolution and neural networks on graphs. We have proved the convergence of neural networks on sampled graphs to the MNN when the input signal is bandlimited. We have connected GNNs with MNNs by showing that the sequence of GNNs sampled from the manifold converge to MNN and the discretization of MNN brings back the particularization on GNNs. We finally verified the performance of MNN with a model classification problem.

References

[1] Manasvi Aggarwal and M Narasimha Murty. *Machine Learning in Social Networks: Embedding Nodes, Edges, Communities, and Graphs*. Springer Nature, 2020.

[2] Mikhail Belkin and Partha Niyogi. Convergence of laplacian eigenmaps. *Advances in neural information processing systems*, 19, 2006.

[3] Mikhail Belkin and Partha Niyogi. Towards a theoretical foundation for laplacian-based manifold methods. *Journal of Computer and System Sciences*, 74(8):1289–1308, 2008.

[4] Christian Borgs and Jennifer Chayes. Graphons: A nonparametric method to model, estimate, and design algorithms for massive networks. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, pages 665–672, 2017.

[5] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[6] Jeff Calder and Nicolas Garcia Trillos. Improved spectral convergence rates for graph laplacians on epsilon-graphs and k-nn graphs. *arXiv preprint arXiv:1910.13476*, 2019.

[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *arXiv preprint arXiv:1606.09375*, 2016.

[8] David B Dunson, Hau-Tieng Wu, and Nan Wu. Spectral convergence of graph laplacian and heat kernel reconstruction in l^∞ from random samples. *Applied and Computational Harmonic Analysis*, 55:282–336, 2021.

[9] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, pages 417–426, 2019.

[10] Jean Gallier and Jocelyn Quaintance. *Differential geometry and Lie groups: a computational perspective*, volume 12. Springer Nature, 2020.

[11] Fernando Gama, Elvin Isufi, Geert Leus, and Alejandro Ribeiro. Graphs, convolutions, and neural networks: From graph filters to graph neural networks. *IEEE Signal Processing Magazine*, 37(6):128–138, 2020.

[12] Fernando Gama, Qingbiao Li, Ekaterina Tolstaya, Amanda Prorok, and Alejandro Ribeiro. Decentralized control with graph neural networks. *arXiv preprint arXiv:2012.14906*, 2020.

[13] 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.

[14] Fernando Gama, Alejandro Ribeiro, and Joan Bruna. Stability of graph scattering transforms. *Advances in Neural Information Processing Systems*, 32:8038–8048, 2019.

[15] Vikas Garg, Stefanie Jegelka, and Tommi Jaakkola. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning*, pages 3419–3430. PMLR, 2020.

[16] Nicolas Keriven, Alberto Bietti, and Samuel Vaiter. Convergence and stability of graph convolutional networks on large random graphs. *Advances in Neural Information Processing Systems*, 33:21512–21523, 2020.

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Ron Levie, Wei Huang, Lorenzo Bucci, Michael Bronstein, and Gitta Kutyniok. Transferability of spectral graph convolutional neural networks. *Journal of Machine Learning Research*, 22(272):1–59, 2021.

[19] László Lovász. *Large networks and graph limits*, volume 60. American Mathematical Soc., 2012.

[20] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 37–45, 2015.

[21] Russell Merris. A survey of graph laplacians. *Linear and Multilinear Algebra*, 39(1-2):19–31, 1995.

[22] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5115–5124, 2017.

[23] Parry Moon and Domina E Spencer. *Field theory handbook: including coordinate systems, differential equations and their solutions*. Springer, 2012.

[24] Alan V Oppenheim, Alan S Willsky, Syed Hamid Nawab, Gloria Mata Hernández, et al. *Signals & systems*. Pearson Educación, 1997.

[25] Alejandro Parada-Mayorga and Alejandro Ribeiro. Algebraic neural networks: Stability to deformations. *IEEE Transactions on Signal Processing*, 69:3351–3366, 2021.

[26] Steven Rosenberg and Rosenberg Steven. *The Laplacian on a Riemannian manifold: an introduction to analysis on manifolds*. Number 31. Cambridge University Press, 1997.

[27] Luana Ruiz, Luiz Chamon, and Alejandro Ribeiro. Graphon neural networks and the transferability of graph neural networks. *Advances in Neural Information Processing Systems*, 33:1702–1712, 2020.

[28] Luana Ruiz, Luiz FO Chamon, and Alejandro Ribeiro. Graphon signal processing. *IEEE Transactions on Signal Processing*, 69:4961–4976, 2021.

[29] Luana Ruiz, Zhiyang Wang, and Alejandro Ribeiro. Graphon and graph neural network stability. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5255–5259. IEEE, 2021.

[30] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[31] Saurabh Verma and Zhi-Li Zhang. Stability and generalization of graph convolutional neural networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1539–1548, 2019.

[32] Ulrike Von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, pages 555–586, 2008.

[33] Wei Wang and Jianxun Gang. Application of convolutional neural network in natural language processing. In *2018 International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pages 64–70. IEEE, 2018.

[34] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[35] Zhiyang Wang, Mark Eisen, and Alejandro Ribeiro. Learning decentralized wireless resource allocations with graph neural networks. *IEEE Transactions on Signal Processing*, 70:1850–1863, 2022.

[36] Zhiyang Wang, Luana Ruiz, and Alejandro Ribeiro. Stability of neural networks on manifolds to relative perturbations. *arXiv preprint arXiv:2110.04702*, 2021.

[37] Zhiyang Wang, Luana Ruiz, and Alejandro Ribeiro. Stability of neural networks on riemannian manifolds. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 1845–1849. IEEE, 2021.

[38] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[39] Yuxing Xie, Jiaojiao Tian, and Xiao Xiang Zhu. Linking points with labels in 3d: A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*, 8(4):38–59, 2020.

[40] Kai Xu, Longyin Wen, Guorong Li, Liefeng Bo, and Qingming Huang. Spatiotemporal cnn for video object segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1379–1388, 2019.

[41] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

[42] Kai Zhang, Wangmeng Zuo, and Lei Zhang. Ffdnet: Toward a fast and flexible solution for cnn-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, 2018.

[43] Dongmian Zou and Gilad Lerman. Graph convolutional neural networks via scattering. *Applied and Computational Harmonic Analysis*, 49(3):1046–1074, 2020.

A Supplementary material

A.1 Proof of Theorem 1

For ease of presentation, we denote the norm $\|\cdot\|_{L^2(\mathbf{G}_n)}$ as $\|\cdot\|$ for short.

We first import the existing results from [2] which indicate the spectral convergence of the constructed graph Laplacian operator to the LB operator of the underlying manifold.

Theorem 2 (Theorem 2.1 [2]) *Let $X = \{x_1, x_2, ...x_n\}$ be a set of n points sampled i.i.d. from a d-dimensional manifold $\mathcal{M} \subset \mathbb{R}^N$. Let $\mathbf{G}_n$ be a graph approximation of $\mathcal{M}$ constructed from X with weight values set as* (15) *with $t_n = n^{-1/(d+2+\alpha)}$ and $\alpha > 0$. Let $\mathbf{L}_n$ be the graph Laplacian of $\mathbf{G}_n$ and $\mathcal{L}$ be the Laplace-Beltrami operator of $\mathcal{M}$. Let λ_i^n be the i-th eigenvalue of $\mathbf{L}_n$ and ϕ_i^n be the corresponding normalized eigenfunction. Let λ_i and ϕ_i be the corresponding eigenvalue and eigenfunction of $\mathcal{L}$ respectively. Then, it holds that*

$$\lim_{n\to\infty} \lambda_i^n = \lambda_i, \quad \lim_{n\to\infty} |\phi_i^n(x_j) - \phi_i(x_j)| = 0, j = 1, 2 \ldots, n \tag{24}$$

where the limits are taken in probability.

From the definitions of neural networks on the constructed graph $\mathbf{G}_n$ and manifold $\mathcal{M}$ respectively, the output difference can be written as

$$\|\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_n, \mathbf{P}_n f) - \mathbf{P}_n \mathbf{\Phi}(\mathbf{H}, \mathcal{L}, f))\| = \left\| \sum_{q=1}^{F_L} \mathbf{x}_{n,L}^q - \sum_{q=1}^{F_L} \mathbf{P}_n f_L^q \right\| \leq \sum_{q=1}^{F_L} \left\| \mathbf{x}_{n,L}^q - \mathbf{P}_n f_L^q \right\|. \tag{25}$$

By inserting the filter definition, we have

$$\left\| \mathbf{x}_{n,l}^p - \mathbf{P}_n f_l^p \right\| = \left\| \sigma \left(\sum_{q=1}^{F_{l-1}} \mathbf{h}_l^{pq}(\mathbf{L}_n) \mathbf{x}_{n,l-1}^q \right) - \mathbf{P}_n \sigma \left(\sum_{q=1}^{F_{l-1}} \mathbf{h}_l^{pq}(\mathcal{L}) f_{l-1}^q \right) \right\| \tag{26}$$

with $\mathbf{x}_{n,0} = \mathbf{P}_n f$ as the input of the first layer. Since the point-wise nonlinearity is normalized Lipschitz according to Assumption 1, we have

$$\|\mathbf{x}_{n,l}^p - \mathbf{P}_n f_l^p\| \leq \left\| \sum_{q=1}^{F_{l-1}} \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{x}_{n,l-1}^q - \mathbf{P}_n \sum_{q=1}^{F_{l-1}} \mathbf{h}_l^{pq}(\mathcal{L}) f_{l-1}^q \right\| \tag{27}$$

$$\leq \sum_{q=1}^{F_{l-1}} \left\| \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{x}_{n,l-1}^q - \mathbf{P}_n \mathbf{h}_l^{pq}(\mathcal{L}) f_{l-1}^q \right\|. \tag{28}$$

The difference can be further decomposed as

$$\|\mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{x}_{n,l-1}^q - \mathbf{P}_n \mathbf{h}_l^{pq}(\mathcal{L}) f_{l-1}^q\|$$

$$\leq \|\mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{x}_{n,l-1}^q - \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{P}_n f_{l-1}^q + \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{P}_n f_{l-1}^q - \mathbf{P}_n \mathbf{h}_l^{pq}(\mathcal{L}) f_{l-1}^q\| \tag{29}$$

$$\leq \left\| \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{x}_{n,l-1}^q - \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{P}_n f_{l-1}^q \right\| + \left\| \mathbf{h}_l^{pq}(\mathbf{L}_n)\mathbf{P}_n f_{l-1}^q - \mathbf{P}_n \mathbf{h}_l^{pq}(\mathcal{L}) f_{l-1}^q \right\| \tag{30}$$

The first term can be bounded as $\|\mathbf{x}_{n,l-1}^q - \mathbf{P}_n f_{l-1}^q\|$ with the initial condition $\|\mathbf{x}_{n,0} - \mathbf{P}_n f_0\| = 0$. Let us denote the second term as D_{l-1}^n. By induction, we have

$$\|\mathbf{\Phi}(\mathbf{H}, \mathbf{L}_n, \mathbf{P}_n f) - \mathbf{P}_n \mathbf{\Phi}(\mathbf{H}, \mathcal{L}, f)\| \leq \sum_{l=0}^{L} \prod_{l'=l}^{L} F_{l'} D_l^n.$$

Therefore, we can focus on the difference term D_l^n, so we omit the feature and layer index to work on a general form. Considering that the input manifold signal f is λ_M-bandlimited, we can write the convolution operation as follows.

$$\|\mathbf{h}(\mathbf{L}_n)\mathbf{P}_n f - \mathbf{P}_n \mathbf{h}(\mathcal{L}) f\| \leq \left\| \sum_{i=1}^{M} \hat{h}(\lambda_i^n) \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n - \sum_{i=1}^{M} \hat{h}(\lambda_i) \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}} \mathbf{P}_n \boldsymbol{\phi}_i \right\| \tag{31}$$

$$\leq \left\| \sum_{i=1}^{M} \hat{h}(\lambda_i^n) \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n - \sum_{i=1}^{M} \hat{h}(\lambda_i) \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n \right\|$$

$$+ \left\| \sum_{i=1}^{M} \hat{h}(\lambda_i) \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n - \sum_{i=1}^{M} \hat{h}(\lambda_i) \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}} \mathbf{P}_n \boldsymbol{\phi}_i \right\|. \tag{32}$$

The first term in (32) can be bounded by leveraging the C-Lipschitz continuity of the frequency response. From the convergence in probability stated in (24), we can claim that for each eigenvalue $\lambda_i \leq \lambda_M$, for all $\epsilon_i > 0$ and all $\delta_i > 0$, there exists some N_i such that for all $n > N_i$, we have

$$\mathbb{P}(|\lambda_i^n - \lambda_i| \leq \epsilon_i) \geq 1 - \delta_i, \tag{33}$$

Letting $\epsilon_i < \epsilon$ with $\epsilon > 0$, with probability at least $\prod_{i=1}^{M}(1 - \delta_i) := 1 - \delta$, the first term is bounded as

$$\left\| \sum_{i=1}^{M} (\hat{h}(\lambda_i^n) - \hat{h}(\lambda_i)) \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n \right\| \leq \sum_{i=1}^{M} |\hat{h}(\lambda_i^n) - \hat{h}(\lambda_i)| |\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n}| \|\boldsymbol{\phi}_i^n\| \tag{34}$$

$$\leq \sum_{i=1}^{M} C |\lambda_i^n - \lambda_i| \|\mathbf{P}_n f\| \|\boldsymbol{\phi}_i^n\|^2 \leq MC\epsilon, \tag{35}$$

for all $n > \max_i N_i := N$.

The second term in (32) can be bounded combined with the convergence of eigenfunctions in (37) as

$$\left\| \sum_{i=1}^{M} \hat{h}(\lambda_i) \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n - \sum_{i=1}^{M} \hat{h}(\lambda_i) \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}} \mathbf{P}_n \boldsymbol{\phi}_i \right\|$$

$$\leq \left\| \sum_{i=1}^{M} \hat{h}(\lambda_i) \left(\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n - \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \mathbf{P}_n \boldsymbol{\phi}_i \right) \right\|$$

$$+ \left\| \sum_{i=1}^{M} \hat{h}(\lambda_i) \left(\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \mathbf{P}_n \boldsymbol{\phi}_i - \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}} \mathbf{P}_n \boldsymbol{\phi}_i \right) \right\| \tag{36}$$

From the convergence in probability stated in (24), we can claim that for some fixed eigenfunction $\boldsymbol{\phi}_i$, for all $\epsilon_i > 0$ and all $\delta_i > 0$, there exists some N_i such that for all $n > N_i$, we have

$$\mathbb{P}(|\boldsymbol{\phi}_i^n(x_j) - \boldsymbol{\phi}_i(x_j)| \leq \epsilon_i) \geq 1 - \delta_i, \quad \forall x_j \in X. \tag{37}$$

Therefore, letting $\epsilon_i < \epsilon$ with $\epsilon > 0$, with probability at least $\prod_{i=1}^{M}(1-\delta_i) := 1 - \delta$, for all $n > \max_i N_i := N$, the first term in (36) can be bounded as

$$\left\| \sum_{i=1}^{M} \hat{h}(\lambda_i) \left(\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \boldsymbol{\phi}_i^n - \langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathcal{M}} \mathbf{P}_n \boldsymbol{\phi}_i \right) \right\| \leq \sum_{i=1}^{M} \|\mathbf{P}_n f\| \|\boldsymbol{\phi}_i^n - \mathbf{P}_n \boldsymbol{\phi}_i\| \leq M\epsilon, \tag{38}$$

considering that frequency response function is non-amplifying stated in Assumption 2. The last equation comes from the definition of norm in $L^2(\mathbf{G}_n)$. The second term in (36) can be written as

$$\begin{aligned} &\left\| \sum_{i=1}^{M} \hat{h}(\lambda_i^n)(\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} \mathbf{P}_n \boldsymbol{\phi}_i - \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}} \mathbf{P}_n \boldsymbol{\phi}_i) \right\| \\ &\qquad \leq \sum_{i=1}^{M} |\hat{h}(\lambda_i^n)| \, |\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} - \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}}| \, \|\mathbf{P}_n \boldsymbol{\phi}_i\|. \end{aligned} \tag{39}$$

Because $\{x_1, x_2, \cdots, x_n\}$ is a set of uniform sampled points from $\mathcal{M}$, based on Theorem 19 in [32] we can claim that

$$\lim_{n\to\infty} \mathbb{P}\left(|\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} - \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}}| \leq \epsilon\right) \geq 1 - \delta, \tag{40}$$

for all $\epsilon > 0$ and $\delta > 0$. Taking into consider the boundedness of frequency response $|\hat{h}(\lambda)| \leq 1$ and the bounded energy $\|\mathbf{P}_n \boldsymbol{\phi}_i\|$. Therefore, we have for all $\epsilon > 0$ and $\delta > 0$,

$$\lim_{n\to\infty} \mathbb{P}\left(\left\| \sum_{i=1}^{M} \hat{h}(\lambda_i^n) \left(\langle \mathbf{P}_n f, \boldsymbol{\phi}_i^n \rangle_{\mathbf{G}_n} - \langle f, \boldsymbol{\phi}_i \rangle_{\mathcal{M}} \right) \mathbf{P}_n \boldsymbol{\phi}_i \right\| \leq M\epsilon \right) \geq 1 - \delta. \tag{41}$$

Combining all these results, we can claim that for all $\epsilon' > 0$ and $\delta > 0$, there exists some N, such that for all $n > N$, we have

$$\mathbb{P}(\|\mathbf{h}(\mathbf{L}_n)\mathbf{P}_n f - \mathbf{P}_n \mathbf{h}(\mathcal{L}) f\| \leq \epsilon') \geq 1 - \delta. \tag{42}$$

With $\lim_{n\to\infty} D_l^n = 0$ in high probability, this concludes the proof.