How Universal Polynomial Bases Enhance Spectral Graph Neural Networks: Heterophily, Over-smoothing, and Over-squashing

Keke Huang [1] **Yu Guang Wang** [2 3 4] **Ming Li** [5 6] **Pietro Liò** [7]

Abstract

Spectral Graph Neural Networks (GNNs), alternatively known as *graph filters*, have gained increasing prevalence for heterophily graphs. Optimal graph filters rely on Laplacian eigendecomposition for Fourier transform. In an attempt to avert prohibitive computations, numerous polynomial filters have been proposed. However, polynomials in the majority of these filters are *predefined* and remain *fixed* across different graphs, failing to accommodate the varying degrees of heterophily. Addressing this gap, we demystify the intrinsic correlation between the spectral property of desired polynomial bases and the heterophily degrees via thorough theoretical analyses. Subsequently, we develop a novel adaptive heterophily basis wherein the basis vectors mutually form angles reflecting the heterophily degree of the graph. We integrate this heterophily basis with the homophily basis to construct a universal polynomial basis *UniBasis*, which devises a polynomial filter-based graph neural network – *UniFilter*. It optimizes the convolution and propagation in GNN, thus effectively limiting over-smoothing and alleviating over-squashing. Our extensive experiments, conducted on datasets with a diverse range of heterophily, support the superiority of UniBasis in the universality but also its proficiency in graph explanation.

[1]School of Computing, National University of Singapore, Singapore [2]Institute of Natural Sciences, School of Mathematical Sciences, Zhangjiang Institute for Advanced Study, Shanghai Jiao Tong University, Shanghai, China [3]Shanghai AI Laboratory, Shanghai, China [4]School of Mathematics and Statistics, University of New South Wales, Sydney, Australia [5]Zhejiang Institute of Optoelectronics, Jinhua, China [6]Zhejiang Key Laboratory of Intelligent Education Technology and Application, Zhejiang Normal University, Jinhua, China [7]Department of Computer Science and Technology, Cambridge University, Cambridge, UK. Correspondence to: Keke Huang <kkhuang@nus.edu.sg>, Ming Li <mingli@zjnu.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Spectral Graph Neural Networks (GNNs) (Kipf & Welling, 2017), also referred to as *graph filters*, have been extensively investigated in recent years due to their superior performance in handling heterophily graphs (Bo et al., 2023b). Optimal graph filters conduct Laplacian eigendecomposition for Fourier transform. To bypass computation complexity, existing graph filters leverage various polynomials to approximate the desired filters for graphs with varying heterophily degrees. For example, ChebNet (Defferrard et al., 2016) employs truncated Chebyshev polynomials (Mason & Handscomb, 2002; Hammond et al., 2011) and accomplishes localized spectral filtering. BernNet (He et al., 2021) utilizes Bernstein polynomials (Farouki, 2012) to acquire better controllability and interpretability. Later, Wang & Zhang (2022) propose JacobiConv by exploiting Jacobi polynomial bases (Askey, 1974) with improved generality.

However, existing polynomial filters overlook the diverse heterophily degrees of underlying graphs when implementing polynomial bases. This oversight of the graph homophily characteristic in the development of polynomial bases leads to suboptimal performance on real-world graphs, as demonstrated in our experiments (Sections 6.1 and 6.3). Meanwhile, Theorem 3.1 proves that frequencies of signals filtered by optimal graph filters are proportional to the heterophily degrees. Therefore, ideal polynomial bases are obligated to provide adaptability to the wide range of heterophily degrees. Consequently, a natural question arises: **how can we devise a universal polynomial basis that encapsulates the diverse graph heterophily degrees?**

To address the issue, we initially determine the relationship between the degree of heterophily and the frequency of optimally filtered signals (Theorem 3.1). Following this, we investigate how the distribution of polynomial bases within the Euclidean space affects the basis spectrum on regular graphs, as outlined in Theorem 4.2. These pivotal insights pave the way for creating an innovative adaptive heterophily basis, specifically designed to match the heterophily degrees in graphs. Ultimately, we combine the heterophily and homophily bases to establish a comprehensive universal basis, termed *UniBasis*. Utilizing UniBasis, we develop a graph neural network using a general polynomial filter, named

UniFilter, which is adaptable to various graph structures.

Many existing spectral and spatial GNNs usually suffer from over-smoothing (Li et al., 2018; Nguyen et al., 2023; Rusch et al., 2023) and over-squashing (Alon & Yahav, 2021; Topping et al., 2022; Giovanni et al., 2023). These scenarios occur when node features go consensus and exponential information is squeezed into fixed-sized vectors in long-range propagation. However, our approach, UniFilter, effectively tackles these two pivotal challenges by employing a convolutional matrix performing both information propagation and rotation concurrently, thereby effectively rewiring the graph structures. For a comprehensive evaluation, we compare UniFilter with 20 baselines across both real-world datasets and synthetic datasets with a wide range of heterophily degrees. The consistently outstanding performance of UniFilter across these datasets validates its efficacy and wide applicability. Moreover, we showcase the spectrum distribution of graph signals as captured by UniBasis across various datasets in Section 6.2. These findings clearly underscore UniBasis's potent potential as a groundbreaking tool for graph explanation, thereby deepening our comprehension of graph structures.

In summary, 1) we reveal that polynomials of desired polynomial filters are meant to align with degrees of graph heterophily; 2) we design a universal polynomial basis UniBasis tailored for different graph heterophily degrees and devise a general graph filter UniFilter; 3) we prove in theory that UniFilter effectively averts over-smoothing while alleviating over-squashing; 4) we evaluate UniFilter on both real-world and synthetic datasets against 20 baselines. The exceptional performance of UniFilter not only attests to the effectiveness and broad applicability of UniBasis but also highlights its significant potential in enhancing graph explanation. The code of UniFilter is accessed at `https://github.com/kkhuang81/UniFilter`.

2. Preliminaries

Notations and Definitions. We represent matrices, vectors, and sets with bold uppercase letters (e.g., $\mathbf{A}$), bold lowercase letters (e.g., $\mathbf{x}$), and calligraphic fonts (e.g., $\mathcal{N}$), respectively. The i-th row (resp. column) of matrix $\mathbf{A}$ is represented by $\mathbf{A}[i,\cdot]$ (resp. $\mathbf{A}[\cdot,i]$). We denote $[n]=\{1,2,\cdots,n\}$.

Let $\mathbf{G}=(\mathcal{V},\mathcal{E})$ be an undirected and connected graph with node set $|\mathcal{V}|=n$ and edge set $|\mathcal{E}|=m$. Let $\mathbf{X}\in\mathbb{R}^{n\times d}$ be the d-dimension feature matrix. For ease of exposition, we use node notation $u\in\mathcal{V}$ to denote its index, i.e., $\mathbf{X}_u=\mathbf{X}[u,\cdot]$. Let $\mathbf{Y}\in\mathbb{N}^{n\times|\mathcal{C}|}$ be the one-hot label matrix, i.e., $\mathbf{Y}[u,i]=1$ if node u belongs to class $\mathcal{C}_i$ for $i\in[|\mathcal{C}|]$, where $\mathcal{C}$ is the set of node labels. The neighbor set of node $u\in\mathcal{V}$ is denoted as $\mathcal{N}_u$ with degree $d_u=|\mathcal{N}_u|$. The adjacency matrix of $\mathbf{G}$ is denoted as $\mathbf{A}\in\mathbb{R}^{n\times n}$ that $\mathbf{A}[u,v]=1$ if edge $\langle u,v\rangle\in\mathcal{E}$; otherwise $\mathbf{A}[u,v]=0$. $\mathbf{D}\in\mathbb{R}^{n\times n}$ is the diagonal degree matrix of $\mathbf{G}$ with $\mathbf{D}[u,u]=d_u$. Let $\mathcal{L}$ and $\hat{\mathcal{L}}$ be the normalized Laplacian matrix of graph $\mathbf{G}$ without and with self-loops respectively, defined as $\mathcal{L}=\mathbf{I}-\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$ and $\hat{\mathcal{L}}=\mathbf{I}-\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ where $\mathbf{I}$ is the identity matrix and $\tilde{\mathbf{D}}=\mathbf{D}+\mathbf{I}$ and $\tilde{\mathbf{A}}=\mathbf{A}+\mathbf{I}$. Frequently used notations are summarized in Table 4 in Appendix A.

Spectral graph filters. In general, the eigendecomposition of the Laplacian matrix is denoted as $\mathcal{L}=\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, where $\mathbf{U}$ is the eigenvector matrix and $\mathbf{\Lambda}=\mathrm{diag}[\lambda_1,\cdots,\lambda_n]$ is the diagonal matrix of eigenvalues. Eigenvalues λ_i for $i\in[n]$ mark the *frequency* and the eigenvalue set $\{\lambda_1,\cdots,\lambda_n\}$ is the *graph spectrum*. Without loss of generality, we assume $0=\lambda_1\le\lambda_2\le\cdots\le\lambda_n\le 2$. When applying a spectral graph filter on graph signal $\mathbf{x}\in\mathbb{R}^n$, the process involves the following steps. First, the graph Fourier operator $\mathcal{F}(\mathbf{x})=\mathbf{U}^\top\mathbf{x}$ projects the graph signal $\mathbf{x}$ into the spectral domain. Next, a spectral filtering function $\mathrm{g}_{\mathbf{w}}(\cdot)$ parameterized by $\mathbf{w}\in\mathbb{R}^n$ is applied on the derived spectrum. Last, the filtered signal is transformed back via the inverse graph Fourier transform operator $\mathcal{F}^{-1}(\mathbf{x})=\mathbf{U}\mathbf{x}$. The process is

$$\begin{aligned}\mathcal{F}^{-1}(\mathcal{F}(\mathrm{g}_{\mathbf{w}})\odot\mathcal{F}(\mathbf{x}))&=\mathbf{U}\mathrm{g}_{\mathbf{w}}(\mathbf{\Lambda})\mathbf{U}^\top\mathbf{x}\\&=\mathbf{U}\,\mathrm{diag}(\mathrm{g}_{\mathbf{w}}(\lambda_1),\cdots,\mathrm{g}_{\mathbf{w}}(\lambda_n))\mathbf{U}^\top\mathbf{x},\end{aligned}\tag{1}$$

where $\odot$ is the Hadamard product.

Spectral graph filters enhance signals in specific spectra and suppress the rest parts according to objective functions. For node classification, homophily graphs are prone to contain low-frequency signals whilst heterophily graphs likely own high-frequency signals. To quantify the heterophily degrees of graphs, numerous homophily metrics have been introduced, e.g., *edge homophily* (Zhu et al., 2020), *node homophily* (Pei et al., 2020), *class homophily* (Lim et al., 2021), and a recent *adjusted homophily* (Platonov et al., 2022). By following the literature on spectral graph filters (Zhu et al., 2020; Lei et al., 2022), we adopt edge homophily in this work.

Definition 2.1 (Homophily Ratio h)**.** Given a graph $\mathbf{G}=(\mathcal{V},\mathcal{E})$ and its label matrix $\mathbf{Y}$, the homophily ratio h of $\mathbf{G}$ is the fraction of edges with two end nodes from the same class, i.e., $h=\frac{|\{\langle u,v\rangle\in\mathcal{E}:\,\mathbf{y}_u=\mathbf{y}_v\}|}{|\mathcal{E}|}$.

Discussion on homophily metrics. Lim et al. (2021); Platonov et al. (2022) point out that homophily metrics quantify the tendency of nodes to connect other nodes from the same classes and measure the deviation of the label distribution from a null model, denoted as *constant baseline* property. In this regard, it is supposed to eliminate the homophily degrees stemming from random edge connections. Yet in our paper, what matters in terms of message-passing GNNs is whether edges connect nodes from the same classes. Generally, the more two endpoints of edges

are from the same classes in graphs, the more beneficial the message passing for information aggregation. Therefore, we adopt edge homophily as the homophily metric.

In addition to the homophily metrics for *categorical* node labels, the similarity of *numerical* node signals can also be measured via *Dirichlet Enenrgy* (Zhou et al., 2021; Karhadkar et al., 2023) (Detailed discussion in Section 5). In terms of spectral perspective, we propose *spectral signal frequency*, a metric customized for node signals $\mathbf{x} \in \mathbb{R}^n$.

Definition 2.2 (Spectral Signal Frequency f)**.** Consider a graph $\mathbf{G} = (\mathcal{V}, \mathcal{E})$ with n nodes and Laplacian matrix $\mathcal{L}$. Given a normalized feature signal $\mathbf{x} \in \mathbb{R}^n$, the spectral signal frequency $f(\mathbf{x})$ on $\mathbf{G}$ is defined as $f(\mathbf{x}) = \frac{\mathbf{x}^\top \mathcal{L}\mathbf{x}}{2}$.

By nature of the Laplacian matrix, spectral signal frequency $f(x)$ quantifies the discrepancy of signal $\mathbf{x}$ on graph $\mathbf{G}$. Formally, the spectral signal frequency $f(x)$ holds that

Proposition 2.3. *For any normalized feature signal* $\mathbf{x} \in \mathbb{R}^n$ *on graph* $\mathbf{G}$*,* $f(\mathbf{x}) \in [0, 1]$.

3. Polynomial Graph Filters

Optimal graph filters perform eigendecomposition on the Laplacian matrix with a computation cost of $O(n^3)$. To bypass the significant computation overhead, substantial polynomial graph filters (Chien et al., 2021; Wang & Zhang, 2022; He et al., 2022; Guo & Wei, 2023; Huang et al., 2024) have been proposed to approximate optimal graph filters by leveraging different polynomials (see Table 5 in Appendix A). By identifying an appropriate propagation matrix $\mathbf{P}$, those polynomial filters on graph signal $\mathbf{x} \in \mathbb{R}^n$ is equally expressed as

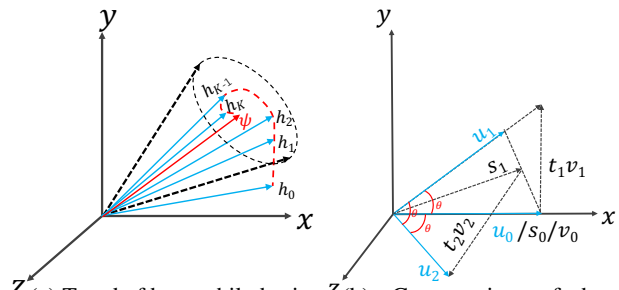
$$\mathbf{z} = \sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}^k \cdot \mathbf{x}, \tag{2}$$

where K is the propagation hops, $\mathbf{w} \in \mathbb{R}^{K+1}$ is the learnable weight vector, and $\mathbf{z} \in \mathbb{R}^n$ is the final representation. For example, BernNet (He et al., 2021) utilizes Bernstein polynomial as $\mathbf{z} = \sum_{k=0}^{K} \frac{\mathbf{w}'_k}{2^K}\binom{K}{k}(2\mathbf{I} - \mathcal{L})^{K-k}\mathcal{L}^k\mathbf{x}$. By setting $\mathbf{P} = \mathbf{I} - \frac{\mathcal{L}}{2}$ and reorganizing the equation, we derive an equivalent formulation as $\mathbf{z} = \sum_{k=0}^{K} \mathbf{w}_k \left(\mathbf{I} - \frac{\mathcal{L}}{2}\right)^k \mathbf{x}$ where

$$\mathbf{w}_k = \sum_{i=0}^{k} \mathbf{w}'_{k-i} \binom{K}{K-i}\binom{K-i}{k-i}(-1)^{k-i}$$

works as the new learnable parameter.

In Equation (2), the vectors $\mathbf{P}^k\mathbf{x}$ for $k \in \{0, 1, \cdots, K\}$ collectively constitute a signal basis $\{\mathbf{P}^0\mathbf{x}, \mathbf{P}^1\mathbf{x}, \cdots, \mathbf{P}^K\mathbf{x}\}$. Spectral graph filters aim to generate node representations within graphs with various heterophily degrees by learning a weighted combination of the signal basis. From the spectral perspective, they essentially derive signals with the desired frequency from the spectrum $\{f(\mathbf{P}^0\mathbf{x}), f(\mathbf{P}^1\mathbf{x}), \cdots, f(\mathbf{P}^K\mathbf{x})\}$, ensuring alignment with label signals $\mathbf{Y}$. According to Definition 2.1 and 2.2, the homophily ratio h and the frequency of label signals attempt to epitomize the inconsistency of label information distributed on graphs. This fact manifests the inherent correlation between the filtered signal $\sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}^k \mathbf{x}$ and h. Therefore, we propose a theorem to formally depict the correlation.



(a) Trend of homophily basis (b) Construction of heterophily basis

Figure 1: Left: Converging trend of homophily bases. Right: Construction procedure of heterophily bases.

Theorem 3.1. *Given a connected graph* $\mathbf{G} = (\mathcal{V}, \mathcal{E})$ *with homophily ratio* h*, consider an optimal polynomial filter* $\mathrm{F}(\mathbf{w}) = \sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}^k$ *with propagation matrix* $\mathbf{P}$ *and weights* $\mathbf{w} \in \mathbb{R}^{K+1}$ *for node classification. Given a feature signal* $\mathbf{x} \in \mathbb{R}^n$*, the spectral frequency* $f(\sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}^k \mathbf{x})$ *is proportional to* $1-h$.

Theorem 3.1 clarifies that ideal signal bases are obligated to incorporate the heterophily properties of graphs. This implies that high-frequency signals can be more desirable on heterophily graphs. However, the majority of existing polynomial filters ignore the graph homophily ratios while relying on predefined polynomials instead, thereby yielding suboptimal performance.

4. Universal Polynomial Basis for Graph Filters

In this section, we develop a novel universal polynomial basis, denoted as *UniBasis*, and subsequently we present a general polynomial filter, referred to as *UniFilter*, built upon UniBasis. Specifically, the construction of UniBasis involves the integration of two foundational components: the traditional homophily basis $\{\mathbf{x}, \mathbf{P}\mathbf{x}, \cdots, \mathbf{P}^K\mathbf{x}\}$ (Section 4.1), and an innovative adaptive heterophily basis $\{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_K\}$ (Section 4.2), governed by a hyperparameter $\tau \in [0, 1]$. This integration, formulated as $\tau\mathbf{P}^k\mathbf{x} + (1-\tau)\mathbf{u}_k$, enables UniBasis the adaptability for a variety of heterophily graphs. Consequently, the polynomial

filter UniFilter is crafted as

$$\mathbf{z} = \sum_{k=0}^{K} \mathbf{w}_k(\tau \mathbf{P}^k \mathbf{x} + (1-\tau)\mathbf{u}_k), \tag{3}$$

where vector $\mathbf{w} \in \mathbb{R}^{K+1}$ is the learnable weight. Subsequently, $\mathbf{z}$ is fed into Multilayer Perceptron (MLP) network for weight training, i.e.,

$$\mathbf{y} = \mathrm{Softmax}(\mathrm{MLP}(\sum_{k=0}^{K} \mathbf{w}_k(\tau \mathbf{P}^k \mathbf{x} + (1-\tau)\mathbf{u}_k))).$$

In the following sections, we first establish the property of homophily basis and then elaborate on the heterophily basis design. Ultimately, we introduce the graph explanation capability of UniBasis.

4.1. Theoretical Analysis of Homophily Basis

Conventional GNN models (Hamilton et al., 2017; Klicpera et al., 2019a; Yang et al., 2023; Huang et al., 2023) employ homophily as a strong inductive bias (Lim et al., 2021). In general, graph signal $\mathbf{x}$ is propagated to K-hop neighbors via propagation matrix $\mathbf{P} = \mathbf{I} - \mathcal{L}$ for information aggregation, yielding a $(K+1)$-length *homophily basis* $\{\mathbf{x}, \mathbf{P}\mathbf{x}, \cdots, \mathbf{P}^K\mathbf{x}\}$. We formulate a theorem to explore the property of the homophily basis, shedding light on its potential to accommodate homophily graphs.

Theorem 4.1. *Given a propagation matrix* $\mathbf{P}$ *and graph signal* $\mathbf{x}$*, consider an infinite homophily basis* $\{\mathbf{x}, \mathbf{P}\mathbf{x}, \cdots, \mathbf{P}^k\mathbf{x}, \mathbf{P}^{k+1}\mathbf{x}, \cdots\}$*. There exists an integer* $\eta \in \mathbb{N}$ *such that when the exponent* $k \geq \eta$ *and increases, the angle* $\arccos\left(\frac{\mathbf{P}^k\mathbf{x} \cdot \mathbf{P}^{k+1}\mathbf{x}}{\|\mathbf{P}^k\mathbf{x}\|\|\mathbf{P}^{k+1}\mathbf{x}\|}\right)$ *is progressively smaller and asymptotically approaches* 0*.*

The homophily basis exhibits *growing similarity* and *asymptotic convergence* to capture homophily signals. This phenomenon, however, leads to the *over-smoothing issue*. For better visualization, Figure 1a illustrates the converging trend of a homophily basis in Euclidean space.

4.2. Adaptive Heterophily Basis

As proved in Theorem 3.1, the desired signal bases are expected to adapt to homophily ratios. This raises a pertinent question: *how can we incorporate homophily ratios in a sensible manner when designing signal bases?* To answer this question, we initially explore the correlation between the basis distribution and its frequency on regular graphs for simplicity, deriving insights for general cases.

Theorem 4.2. *Consider a regular graph* $\mathbf{G}$*, a random basis signal* $\mathbf{x} \in \mathbb{R}^n$*, and a normalized all-ones vector* $\phi \in \mathbb{R}^n$ *with* $f(\phi) = 0$*. Suppose* $\theta := \arccos(\phi \cdot \mathbf{x})$ *denotes the angle formed by* $\mathbf{x}$ *and* ϕ*. It holds that the expectation of spectral signal frequency* $\mathbb{E}_{\mathbf{G}\sim\mathcal{G}}[f(\mathbf{x})]$ *over the randomness of* $\mathbf{G}$ *is monotonically increasing with* θ *for* $\theta \in [0, \frac{\pi}{2})$*.*

Heterophily basis construction. Heterophily bases aim to align the spectrum property to the homophily ratios of underlying graphs. Theorem 4.2 establishes that signal vectors exhibiting a greater angular separation from the 0-frequency vector ϕ possess a higher expected frequency on regular graphs. When attempting to extend to general graphs without involving the 0-frequency vector, we may consider the angular offsets (relative position) among the basis vectors themselves for the desired spectrum property.

Theorem 4.1 discloses the growing similarity and asymptotic convergence phenomenon within the homophily basis, i.e., angles among homophily basis approaching zero. To mitigate this over-smoothing issue, we can intuitively scatter and fix an angle of $\theta \in [0, \frac{\pi}{2}]$ among all pairs of basis vectors, as inspired by the insight from Theorem 4.2. Then, a natural question arises: *how do we determine an appropriate value for* θ*?*

Specifically, Theorem 3.1 proves the spectral frequency of ideal signals proportional to $1-h$. Meanwhile, Theorem 4.2 hints at the monotonic relation between the spectral frequency and angular offsets among basis vectors. By integrating these insights of linear proportion and monotonic relation, we empirically set the $\theta := \frac{\pi}{2}(1-h)$. Consequently, we derive a signal basis adept at encapsulating the varying heterophily degrees of graphs, which is formally denoted as the *heterophily basis*.

The procedure of constructing a $(K+1)$-length of heterophily basis is outlined in Algorithm 1 and illustrated in Figure 1b. To start with, we normalize the input signal $\mathbf{x}$ as the initial signal $\mathbf{u}_0$ and set $\theta := \frac{(1-\hat{h})\pi}{2}$ where $\hat{h}$ is the estimation of h. To manipulate the formed angles between signal vectors, we forge an orthonormal basis, denoted as $\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_K\}$ where $\mathbf{v}_0$ is initialized as $\mathbf{u}_0$. In particular, at the k-th iteration for $k \in [1, K]$, we set $\mathbf{v}_k := \mathbf{P}\mathbf{v}_{k-1}$. Subsequently, $\mathbf{v}_k$ is calculated as $\mathbf{v}_k := \mathbf{v}_k - (\mathbf{v}_k^\top \mathbf{v}_{k-1})\mathbf{v}_{k-1} - (\mathbf{v}_k^\top \mathbf{v}_{k-2})\mathbf{v}_{k-2}$ as per the *three-term recurrence theorem* (Gautschi, 2004; Liesen & Strakos, 2013; Guo & Wei, 2023). Signal vector $\mathbf{u}_k$ is set as $\mathbf{u}_k := \frac{\mathbf{s}_{k-1}}{k}$ where $\mathbf{s}_{k-1} := \sum_{i=0}^{k-1} \mathbf{u}_i$. Subsequently, $\mathbf{u}_k$ is updated as $\mathbf{u}_k := \frac{\mathbf{u}_k + t_k \mathbf{v}_k}{\|\mathbf{u}_k + t_k \mathbf{v}_k\|}$ where t_k is

$$t_k = \sqrt{\left(\frac{\mathbf{s}_{k-1}^\top \mathbf{u}_{k-1}}{k\cos\theta}\right)^2 - \frac{(k-1)\cos\theta + 1}{k}}. \tag{4}$$

The final vector set $\{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_K\}$ is returned as the heterophily basis. The desired property of the heterophily basis is proved in the following theorem. Detailed proofs are presented in Appendix B.

Theorem 4.3. *Consider a heterophily basis* $\{\mathbf{u}_0, \mathbf{u}_1, \cdots \mathbf{u}_K\}$ *constructed from Algorithm 1 for graphs*

Algorithm 1 Heterophily Basis

Input: Graph $\mathbf{G}$, propagation matrix $\mathbf{P}$, input feature signal $\mathbf{x}$, hop K, estimated homophily ratio $\hat{h}$

Output: Heterophily basis $\{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_K\}$

$\mathbf{u}_0 \leftarrow \frac{\mathbf{x}}{\|\mathbf{x}\|}$, $\mathbf{v}_0 \leftarrow \mathbf{u}_0$, $\mathbf{v}_{-1} \leftarrow \mathbf{0}$, $\mathbf{s}_0 \leftarrow \mathbf{u}_0$, $\theta \leftarrow \frac{(1-\hat{h})\pi}{2}$

for $k \leftarrow 1$ **to** K **do**

  $\mathbf{v}_k \leftarrow \mathbf{P}\mathbf{v}_{k-1}$

  $\mathbf{v}_k \leftarrow \mathbf{v}_k - (\mathbf{v}_k^\top \mathbf{v}_{k-1})\mathbf{v}_{k-1} - (\mathbf{v}_k^\top \mathbf{v}_{k-2})\mathbf{v}_{k-2}$

  $\mathbf{v}_k \leftarrow \frac{\mathbf{v}_k}{\|\mathbf{v}_k\|}$, $\mathbf{u}_k \leftarrow \frac{\mathbf{s}_{k-1}}{k}$

  t_k is calculated as in Equation (4)

  $\mathbf{u}_k \leftarrow \frac{\mathbf{u}_k + t_k \mathbf{v}_k}{\|\mathbf{u}_k + t_k \mathbf{v}_k\|}$, $\mathbf{s}_k \leftarrow \mathbf{s}_{k-1} + \mathbf{u}_k$

return $\{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_K\}$

with homophily ratio h. It holds that $\forall i, j \in \{0, 1, \cdots, K\}$,

$$\mathbf{u}_i \cdot \mathbf{u}_j = \begin{cases} \cos(\frac{(1-h)\pi}{2}) & \text{if } i \neq j, \\ 1 & \text{if } i = j. \end{cases}$$

As shown in Equation (3), UniBasis is constructed as $\tau\mathbf{P}^k\mathbf{x} + (1-\tau)\mathbf{u}_k$ by integrating homophily basis and heterophily basis with a parameter $\tau \in [0, 1]$. This integration yields enhanced adaptability of UniBasis to various graphs. Consequently, UniBasis is better equipped to learn optimal weight parameters during model training compared with fixed polynomials.

Estimation of homophily ratio. The accurate calculation of the homophily ratio h relies on the label set of the entire graphs, which remains inaccessible. To circumvent this issue, we estimate h through labels of training data, denoted as $\hat{h}$, as the input for UniFilter. We validate the estimation accuracy in the ablation study (Section 6.3) and robustness in Appendix D. As shown, $\mathbf{h}$ can be efficiently estimated via a small proportion of labeled training nodes without compromising the performance of UniFilter. This fact signifies that the estimated homophily ratio $\hat{h}$ can serve as an ideal substitute for h.

Time complexity. Algorithm 1 consists of K iterations. In the k-th iteration, it takes $O(m+n)$ to calculate the orthonormal basis and $O(n)$ to update $\mathbf{u}_k$. Therefore, the total time complexity of Algorithm 1 is $O(K(m+n))$, i.e., linear to propagation hops and input graph sizes.

4.3. UniBasis for Graph Explanation

Once UniFilter is well-trained, the derived UniBasis with the learned weights demystifies the spectral properties of graph signals. Specifically, given a graph $\mathbf{G}$ with node signal $\mathbf{x}$, UniBasis constitutes the spectrum $\{f(\mathbf{x}), f(\tau\mathbf{P}\mathbf{x} + (1-\tau)\mathbf{u}_1), \cdots, f(\tau\mathbf{P}^K\mathbf{x} + (1-\tau)\mathbf{u}_K)\}$. The k-th learned weight $\mathbf{w}_k$ from the weight vector $\mathbf{w} \in \mathbb{R}^{K+1}$ acts as the *amplification factor* of the signal in frequency $f(\tau\mathbf{P}^k\mathbf{x} + (1-\tau)\mathbf{u}_k)$. Consequently, the weight vector $\mathbf{w}$ discloses the significance of each frequency component in $\mathbf{G}$, thereby unfolding the distribution of graph signals and offering insights into the hidden spectral property.

As validated in Section 6.2, the experimental results (Figure 2) demonstrate the superior capability of UniBasis to capture the spectral characteristics of graphs across diverse heterophily degrees. Akin to the Fourier transform in signal processing, UniBasis manifests its potential as a promising approach for graph explanation.

5. Theoretical Analysis

Every pair of vectors $(\mathbf{u}_i, \mathbf{u}_j)$ from the heterophily basis form the angle of θ. Among them, $\mathbf{u}_0 \leftarrow \frac{\mathbf{x}}{\|\mathbf{x}\|}$ is the normalized input node signal $\mathbf{x}$. In this regard, any heterophily vector $\mathbf{u}_i$ for $i \in \{1, 2, \cdots, K\}$ can be obtained by rotating $\mathbf{x}$ with θ degree via a certain *rotation matrix* $\mathbf{P}_\theta$, i.e., $\mathbf{u}_i := \mathbf{P}_\theta\mathbf{x}$. Formally, the rotation matrix is defined as

Definition 5.1 (Rotation matrix)**.** Matrix $\mathbf{P}_\theta \in \mathbb{R}^{n\times n}$ is a rotation matrix if there exists a unitary matrix $\mathbf{U} \in \mathbb{C}^{n\times n}$ such that $\mathbf{R}(\theta) := \mathbf{U}^{-1}\mathbf{P}_\theta\mathbf{U}$ where

$$\mathbf{R}(\theta) = \begin{pmatrix} \cos\theta & -\sin(\theta) & 0 & \cdots \\ \sin(\theta) & \cos\theta & 0 & \cdots \\ 0 & 0 & 1 & \cdots \\ & & \cdots & \cdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Therefore, *UniFilter* in Equation (3) can be reformulated as

$$\mathbf{z} = \textstyle\sum_{k=0}^{K} \mathbf{w}_k \big((\tau\mathbf{P}^k + (1-\tau)\mathbf{P}_{\theta,k})\mathbf{x}\big), \qquad (5)$$

where $\mathbf{P}_{\theta,k}$ is the k-th rotation matrix and $\tau\mathbf{P}^k + (1-\tau)\mathbf{P}_{\theta,k}$ works as the *convolutional matrix* of the k-th layer.

Over-smoothing analysis. Consider a node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ where d is the feature dimension. The similarity of node representations at the k-th layer on graph $\mathbf{G}$ is measured by *Dirichlet energy* as $E(\mathbf{G}, \mathbf{X}^k) = \frac{1}{n}\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{N}_v}\|\mathbf{X}_v^k - \mathbf{X}_u^k\|_2^2$ where $\mathbf{X}^k$ is the resultant feature matrix at the k-th layer. Over-smoothing occurs if $E(\mathbf{G}, \mathbf{X}^k) \to 0$ for a sufficiently large k. In terms of this, we substantiate that UniFilter prevents over-smoothing.

Theorem 5.2. *Given a node feature matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$ on graph $\mathbf{G}$, let $\mathbf{X}^k = (\tau\mathbf{P}^k + (1-\tau)\mathbf{P}_{\theta,k})\mathbf{X}$. It holds that* $\lim_{K\to\infty} E(\mathbf{G}, \mathbf{X}^k) = (1-\tau)E(\mathbf{G}, \mathbf{X})$.

Over-squashing analysis. Over-squashing (Alon & Yahav, 2021; Topping et al., 2022) describes the phenomenon that exponentially growing information from distant nodes is squeezed into fixed-size vectors during message passing on graphs. Over-squashing is usually measured by the *Jacobian*

of node representations (Topping et al., 2022; Giovanni et al., 2023). Specifically, let $\mathbf{z}^{(k)} = \mathbf{w}_k\big((\tau\mathbf{P}^k + (1-\tau)\mathbf{P}_{\theta,k})\mathbf{x}\big)$ be the node signal vector of the k-th step in UniFilter and k be the distance between nodes u and v. Jacobian $|\partial\mathbf{z}_u^{(k)}/\partial\mathbf{x}_v|$ measures the sensitivity of information received at node u to the signal of node v when propagating k distance. Over-squashing occurs if $|\partial\mathbf{z}_u^{(k)}/\partial\mathbf{x}_v| \leq c\cdot(\mathbf{P}_M^k)_{uv}$ where c is a constant and $\mathbf{P}_M$ is the message-passing matrix, i.e., the sensitivity (node dependence) decays exponentially to propagation distance. However, we prove that UniFilter is able to mitigate over-squashing.

Theorem 5.3. *Consider $\mathbf{z}^{(k)}$ as the k-th step node signal vector in UniFilter. Jacobian $|\partial\mathbf{z}_u^{(k)}/\partial\mathbf{x}_v|$ is independent of propagation step k.*

6. Experiments

Datasets. We evaluate the performance of UniFilter on 6 real-world datasets with varied homophily ratios. Specifically, the three citation networks (Sen et al., 2008), i.e., Cora, Citeseer, and Pubmed, are homophily graphs with homophily ratios 0.81, 0.73, and 0.80 respectively; the two Wikipedia graphs, i.e., Chameleon and Squirrel and the Actor co-occurrence graph from WebKB3 (Pei et al., 2020) are heterophily graphs with homophily ratios 0.22, 0.23, and 0.22 respectively. The dataset details are presented in Table 6 in Appendix C.

Baselines. We compare UniFilter with 20 baselines in two categories, i.e., *polynomial filters* and *model-optimized methods*. For polynomial filters, we include monomial SGC (Wu et al., 2019), SIGN (Frasca et al., 2020), ASGC (Chanpuriya & Musco, 2022), GPR-GNN (Chien et al., 2021), and EvenNet (Lei et al., 2022), Chebyshev polynomial ChebNet (Defferrard et al., 2016) and its improved version ChebNetII (He et al., 2022), Bernstein polynomial BernNet (He et al., 2021), Jacobi polynomial JacobiConv (Wang & Zhang, 2022), the orthogonal polynomial OptBasisGNN (Guo & Wei, 2023) and learnable basis Specformer (Bo et al., 2023a). For model-optimized methods, they optimize the model architecture for improvement. We consider GCN (Kipf & Welling, 2017), GCNII (Chen et al., 2020), GAT (Velickovic et al., 2018), MixHop (Abu-El-Haija et al., 2019), H_2GCN (Zhu et al., 2020), LINKX (Lim et al., 2021), WRGAT (Suresh et al., 2021), ACM-GCN (Luan et al., 2022), and GloGNN++ (Li et al., 2022).

Experiment Settings. There are two common data split settings, i.e., $60\%/20\%/20\%$ and $48\%/32\%/20\%$ for training/validation/testing in the literature. Specifically, the polynomial filters are mostly tested in the previous setting (Wang & Zhang, 2022; Guo & Wei, 2023; Bo et al., 2023a) while the model-optimized methods are normally evaluated in the latter[1] (Zhu et al., 2020; Li et al., 2022; Song et al., 2023).

6.1. Node Classification Performance

Table 1 and Table 2 report the accuracy scores associated with standard deviations for UniFilter, tested polynomial filters, and model-optimized methods for node classification. For ease of exposition, we highlight the *highest* accuracy score in bold and underline the *second highest* score for each dataset.

As shown, our method UniFilter consistently achieves the highest accuracy scores on both the homophily datasets and heterophily datasets, except in one case on Actor in Table 1. UniFilter exhibits explicit performance advantages over both SOTA polynomial filter Specformer and SOTA model-optimized method GloGNN++ for the majority of cases. In particular, the performance improvements are remarkably significant on the two heterophily datasets Chameleon and Squirrel. Specifically, the corresponding performance gains reach up to 1.03% and 2.76% as shown in Table 1 and 2.45% and 6.38% as shown in Table 2 respectively. It is worth mentioning that the computation time of UniBasis is linear to graph sizes and propagation hops. The superior performance of UniFilter significantly confirms the outstanding effectiveness and universality of UniBasis.

6.2. Signal Spectrum for Graph Explanation

The superior performance of UniFilter explicitly implies the outstanding capability of UniBasis to capture the spectral characteristics of graphs. For a better demonstration, we first calculate the *spectral signal frequency* of each basis vector for all d-dimensions, resulting in d spectrum of length $K+1$. We then average the spectrum and associate it with the learned weights $\mathbf{w}$ accordingly, where weights $\mathbf{w} \in \mathbb{R}^{K+1}$ of UniBasis are trained for each dataset. The spectrum distributions of the trained UniBasis for the 6 datasets are plotted in Figure 2.

Recall that signals in specific frequencies with weights in large absolute are enhanced while signals with small weights are suppressed. As displayed, the majority of signals of the three homophily datasets lie within the relatively low-frequency intervals as expected, e.g., $[0.3, 0.5]$. We also observe some minor high-frequency information which also provides insightful information for node classification (Klicpera et al., 2019b; Chen et al., 2019; Balcilar et al., 2020; Zheng et al., 2022) and node distinguishability (Luan et al., 2023). On the contrary, UniBasis of the three heterophily datasets aims to remove low-frequency signals with negative weights and preserve high-frequency

[1]Please note that those model-optimized methods reuse the public data splits from Pei et al. (2020) which are actually in the splits of $48\%/32\%/20\%$ in the implementation.

Table 1: Accuracy (%) compared with polynomial filters.

Methods	Cora	Citeseer	Pubmed	Actor	Chameleon	Squirrel
SGC	86.83 ± 1.28	79.65 ± 1.02	87.14 ± 0.90	34.46 ± 0.67	44.81 ± 1.20	25.75 ± 1.07
SIGN	87.70 ± 0.69	80.14 ± 0.87	89.09 ± 0.43	41.22 ± 0.96	60.92 ± 1.45	45.59 ± 1.40
ASGC	85.35 ± 0.98	76.52 ± 0.36	84.17 ± 0.24	33.41 ± 0.80	71.38 ± 1.06	57.91 ± 0.89
GPR-GNN	88.54 ± 0.67	80.13 ± 0.84	88.46 ± 0.31	39.91 ± 0.62	67.49 ± 1.38	50.43 ± 1.89
EvenNet	87.77 ± 0.67	78.51 ± 0.63	90.87 ± 0.34	40.36 ± 0.65	67.02 ± 1.77	52.71 ± 0.85
ChebNet	87.32 ± 0.92	79.33 ± 0.57	87.82 ± 0.24	37.42 ± 0.58	59.51 ± 1.25	40.81 ± 0.42
ChebNetII	88.71 ± 0.93	80.53 ± 0.79	88.93 ± 0.29	41.75 ± 1.07	71.37 ± 1.01	57.72 ± 0.59
BernNet	88.51 ± 0.92	80.08 ± 0.75	88.51 ± 0.39	41.71 ± 1.12	68.53 ± 1.68	51.39 ± 0.92
JacobiConv	88.98 ± 0.72	80.78 ± 0.79	89.62 ± 0.41	41.17 ± 0.64	74.20 ± 1.03	57.38 ± 1.25
OptBasisGNN	87.00 ± 1.55	80.58 ± 0.82	90.30 ± 0.19	**42.39 ± 0.52**	74.26 ± 0.74	63.62 ± 0.76
Specformer	88.57 ± 1.01	81.49 ± 0.94	87.73 ± 0.58	41.93 ± 1.04	74.72 ± 1.29	64.64 ± 0.81
UniFilter	**89.49 ± 1.35**	**81.39 ± 1.32**	**91.44 ± 0.50**	40.84 ± 1.21	**75.75 ± 1.65**	**67.40 ± 1.25**

Table 2: Accuracy (%) compared with model-optimized methods.

Methods	Cora	Citeseer	Pubmed	Actor	Chameleon	Squirrel
GCN	86.98 ± 1.27	76.50 ± 1.36	88.42 ± 0.50	27.32 ± 1.10	64.82 ± 2.24	53.43 ± 2.01
GCNII	88.37 ± 1.25	77.33 ± 1.48	90.15 ± 0.43	37.44 ± 1.30	63.86 ± 3.04	38.47 ± 1.58
GAT	87.30 ± 1.10	76.55 ± 1.23	86.33 ± 0.48	27.44 ± 0.89	60.26 ± 2.50	40.72 ± 1.55
MixHop	87.61 ± 0.85	76.26 ± 1.33	85.31 ± 0.61	32.22 ± 2.34	60.50 ± 2.53	43.80 ± 1.48
H_2GCN	87.87 ± 1.20	77.11 ± 1.57	89.49 ± 0.38	35.70 ± 1.00	60.11 ± 2.15	36.48 ± 1.86
LINKX	84.64 ± 1.13	73.19 ± 0.99	87.86 ± 0.77	36.10 ± 1.55	68.42 ± 1.38	61.81 ± 1.80
WRGAT	88.20 ± 2.26	76.81 ± 1.89	88.52 ± 0.92	36.53 ± 0.77	65.24 ± 0.87	48.85 ± 0.78
ACM-GCN	87.91 ± 0.95	77.32 ± 1.70	90.00 ± 0.52	36.28 ± 1.09	66.93 ± 1.85	54.40 ± 1.88
GloGNN++	88.33 ± 1.09	77.22 ± 1.78	89.24 ± 0.39	37.70 ± 1.40	71.21 ± 1.84	57.88 ± 1.76
UniFilter	**89.12 ± 0.87**	**80.28 ± 1.31**	**90.19 ± 0.41**	**37.79 ± 1.11**	**73.66 ± 2.44**	**64.26 ± 1.46**

information. The distinct spectrum distributions of UniBasis disclose the unique spectral characteristics of each dataset. These results highlight the capability of UniBasis as a new method to analyze graphs with varying heterophily degrees in the spectral domain with enriched understanding.

6.3. Ablation Studies

Universality of UniBasis. To verify the effectiveness of UniBasis, we devise 3 variants of UniFilter with different polynomial bases. To this end, we alter UniFilter by changing UniBasis into 1) a filter simply using the heterophily basis (setting $\tau = 0$) denoted as HetFilter, 2) a filter simply using the homophily basis (setting $\tau = 1$) denoted as HomFilter, and 3) a filter using the orthonormal basis ($\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_k\}$) denoted as OrtFilter. For easy control, we generate a synthetic dataset $\mathbf{G}_s$ by adopting the graph structure and label set of Cora. Without loss of generality, we generate a random one-hot feature vector in 100 dimensions for each node. To vary the homophily ratios of $\mathbf{G}_s$, we permute nodes in a random sequence and randomly reassign node labels progressively, resulting in homophily ratios in $\{0.13^2, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.81\}$.

The performance advantage gaps of UniFilter over the three variants are presented in Figure 3. We omit the results of HetFilter beyond $h \geq 0.3$ since the corresponding performance gaps become significantly larger, which is as expected since the heterophily basis is incapable of tackling homophily graphs. In particular, the performance advantage of UniFilter over HomFilter gradually decreases as h grows larger. In contrast, the performance gaps of OrtFilter from UniFilter peak at $h = 0.3$ with a notable shortfall and then erratically decrease, ending with an accuracy gap of 0.71% at $h = 0.81$. The fluctuation of OrtFilter states the inferiority of the orthonormal basis over UniBasis.

Sensitivity of τ. To explore the sensitivity of UniFilter to the hyperparameter τ, we vary τ in $\{0, 0.1, \cdots, 0.9, 1\}$ and test UniFilter on the strong homophily dataset Cora and the strong heterophily dataset Squirrel. Figure 4 plots the performance development with varying τ. As displayed, UniFilter prefers the homophily basis on Cora, and the performance peaks at $\tau = 0.9$. On the contrary, the performance of UniFilter slightly fluctuates when $\tau \leq 0.7$ and then moderately decreases along the increase of τ on Squirrel. When $\tau = 1.0$, the accuracy score drops sharply since only the homophily basis is utilized in this scenario.

Homophily Ratio Estimation. Notice that the homophily ratio is estimated on the training set from each random split. To demonstrate the accuracy of the estimation, we present

[2]Note that this is the smallest homophily ratio we can possibly acquire by random reassignments.

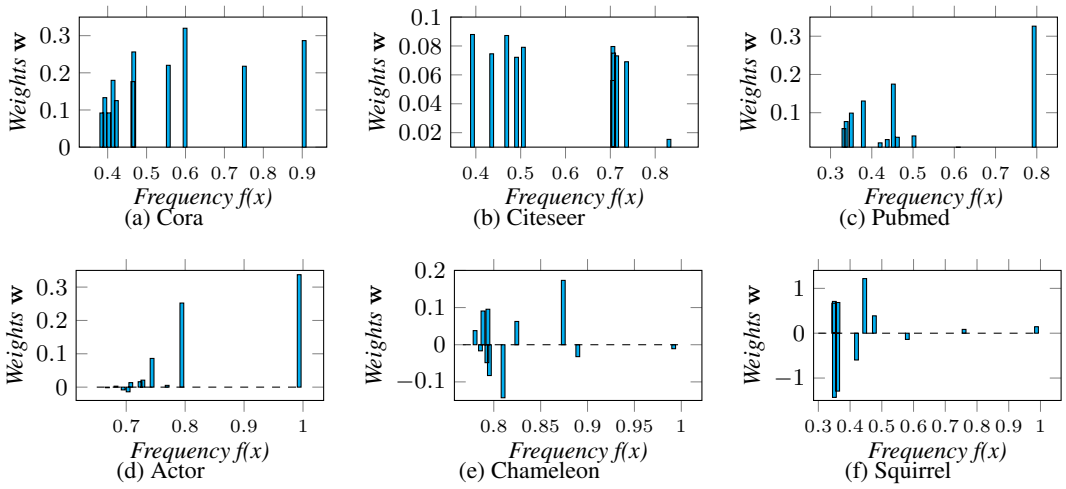


Figure 2: Spectrum distribution of trained UniBasis.

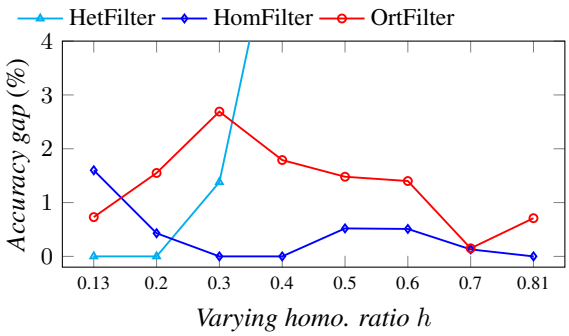


Figure 3: Accuracy gaps of the three variants from UniFilter on $\mathbf{G}_s$ across varying h.

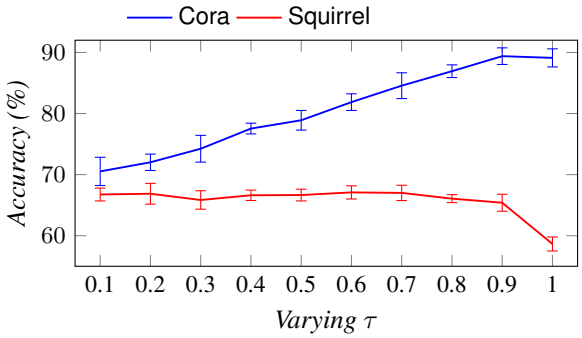


Figure 4: Accuracy (%) with varying τ.

Table 3: Estimated homophily ratios $\hat{h}$.

Dataset	Cora	Citeseer	Pubmed	Actor	Chameleon	Squirrel
True h	0.81	0.74	0.80	0.22	0.23	0.22
$\hat{h}_1$	0.82 ± 0.01	0.70 ± 0.01	0.79 ± 0.01	0.21 ± 0.004	0.24 ± 0.01	0.22 ± 0.01
$\hat{h}_2$	0.82 ± 0.01	0.69 ± 0.01	0.79 ± 0.01	0.21 ± 0.004	0.24 ± 0.01	0.22 ± 0.01

the estimated homophily ratio $\hat{h}$ with standard deviation averaged over the 10 estimations for all datasets in Table 3. In particular, $\hat{h}_1$ and $\hat{h}_2$ represent the averaged homophily ratio estimations for data splits settings of $60\%/20\%/20\%$ for polynomial filters and $48\%/32\%/20\%$ for model-optimized methods respectively. As shown in Table 3, the estimated values $\hat{h}_1$ and $\hat{h}_2$ closely align with the actual homophily ratio h. There is a difference within a 2% range across all datasets, except for Citeseer. As validated from the superior performance of UniFilter in Table 1 and Table 2, $\hat{h}$ is an ideal substitute for h as the input of UniFilter without compromising the performance.

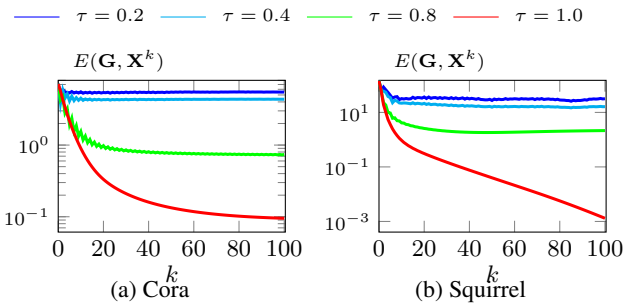


Figure 5: Dirichlet energy $E(\mathbf{G}, \mathbf{X}^k)$ with varying k.

Prevention of Over-smoothing. We calculate the Dirichlet energy $E(\mathbf{G}, \mathbf{X}^k)$ of node representations $\mathbf{X}^k$ on dataset Cora and Squirrel by varying $k \in [1, 100]$, demonstrating the capability of UniFilter to prevent the over-smoothing issue. Specifically, we set $\tau \in \{0.2, 0.4, 0.8, 1.0\}$ and plot the corresponding $E(\mathbf{G}, \mathbf{X}^k)$ in Figure 5. Note that UniFilter utilizes solely homophily basis when $\tau = 1$. Observe that the Dirichlet energy $E(\mathbf{G}, \mathbf{X}^k)$ approaches 0 on both datasets when $\tau = 1$ for sufficiently large k. This is attributed to the over-smoothing issue within the homophily basis, as elucidated in Theorem 4.1. In contrast, when $\tau > 0$, $E(\mathbf{G}, \mathbf{X}^k)$ remains a constant value larger than 0 as k increases. Furthermore, $E(\mathbf{G}, \mathbf{X}^k)$ increases with a decrease in τ, aligning with the findings in Theorem 5.2.

7. Related Work

Polynomial filters. As the seminal work, Bruna et al. (2014) proposed to generalize convolutional neural networks to graphs based on the spectrum of the graph Laplacian. Subsequently, ChebNet (Defferrard et al., 2016) utilized a K-order truncated Chebyshev polynomial and provided a K-hop localized filtering capability. GPR-GNN (Chien et al., 2021) simply adopted monomials instead and applied the generalized PageRank (Li et al., 2019) scores as the coefficients to measure node proximity. Meanwhile, Zheng et al. (2021) exploited framelet transform to enhance the performance. To enhance controllability and interpretability, BernNet (He et al., 2021) employed nonnegative Bernstein polynomials as the basis. Later, Wang & Zhang (2022) examined the expressive power of existing polynomials and proposed JacobiConv by leveraging Jacobi polynomials (Askey, 1974), achieving better adaptability to underlying graphs. Subsequently, He et al. (2022) revisited ChebNet and pinpointed the over-fitting issue in the Chebyshev approximation. To address the issue, they turned to Chebyshev interpolation and proposed ChebNetII. Recently, polynomial filter OptBasisGNN (Guo & Wei, 2023) orthogonalized the polynomial basis to maximize convergence speed. Instead of using fixed-order polynomials, Specformer (Bo et al., 2023a) resorted to Transformer (Vaswani et al., 2017) to derive learnable bases for each feature dimension. However, it required conducting eigendecomposition with the cost of $O(n^3)$, rendering it impractical for large graphs. Nonetheless, the above polynomial filters do not take the varying heterophily degrees of graphs into consideration when utilizing polynomials, which leads to suboptimal empirical performance, as verified in our experiments.

Model-optimized GNNs. One common technique in model design is to combine both low-pass and high-pass filters. GNN-LF/HF (Zhu et al., 2021) devised variants of the Laplacian matrix to construct low-pass and high-pass filters respectively. ACM-GCN (Luan et al., 2022) trained both low-pass and high-pass filters in each layer and then integrated their embeddings adaptively. Another strategy of model design is to extract homophily from both local and global graph structures. Specifically, H_2GCN (Zhu et al., 2020) utilized embeddings and intermediated representations of nodes at various distances, including central nodes, neighbors, and distant nodes. Similarly, GloGNN++ (Li et al., 2022) trained a coefficient matrix in each layer to measure the correlations between nodes to aggregate homophilous nodes globally. To explicitly capture the relations between distant nodes, WRGAT (Suresh et al., 2021) leveraged the graph rewiring (Topping et al., 2022; Karhadkar et al., 2023) technique by constructing new edges with weights to measure node proximity. Additionally, many GNNs handled heterophily graphs from other aspects. LINKX (Lim et al., 2021) learned embeddings from both node features and graph structure simultaneously. Ordered GNN (Song et al., 2023) constructed the hierarchy structure of neighbors and then constrained the neighbor nodes from specific hops into the same blocks, without mixing features across hops.

8. Conclusion

In this paper, we propose a universal polynomial basis UniBasis by incorporating the graph heterophily degrees in the premise of thorough theoretical analysis. Utilizing UniBasis, we devise a general graph filter UniFilter. Meanwhile, UniFilter is capable of effectively preventing over-smoothing and mitigating over-squashing by optimizing the convolution matrix. Comprehensive evaluation supports the superiority of UniFilter on a diverse range of both real-world and synthetic datasets, which validates the effectiveness of UniBasis for graphs with varying heterophily degrees. The UniBasis also serves as a graph explainer for the spectral distribution of graph signals.

Acknowledgments

This research is part of the programme DesCartes and is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Campus for Research Excellence and Technological Enterprise (CREATE) programme. Keke Huang acknowledged the support from the NUS Centre for Research in Privacy Technologies (No. A00080290000). Yu Guang Wang acknowledged the support of Shanghai City-level Key Special Science and Technology Project (No. 2021SHZDZX0102). Ming Li acknowledged the supports from the National Natural Science Foundation of China (No. U21A20473, No. 62172370), the Key R&D Program of Zhejiang Province (No. 2024C03262), and Jinhua Science and Technology Plan (No. 2023-3-003a). We thank the anonymous reviewers for their insightful comments.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abu-El-Haija, S., Perozzi, B., Kapoor, A., Alipourfard, N., Lerman, K., Harutyunyan, H., Steeg, G. V., and Galstyan, A. MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In *ICML*, pp. 21–29, 2019.

Alon, U. and Yahav, E. On the bottleneck of graph neural networks and its practical implications. In *ICLR*, 2021.

Askey, R. Positive jacobi polynomial sums, iii. In *Linear Operators and Approximation II*, pp. 305–312. 1974.

Balcilar, M., Renton, G., Héroux, P., Gauzere, B., Adam, S., and Honeine, P. Bridging the gap between spectral and spatial domains in graph neural networks. *arXiv preprint arXiv:2003.11702*, 2020.

Bo, D., Shi, C., Wang, L., and Liao, R. Specformer: Spectral graph neural networks meet transformers. In *ICLR*, 2023a.

Bo, D., Wang, X., Liu, Y., Fang, Y., Li, Y., and Shi, C. A survey on spectral graph neural networks. *CoRR*, abs/2302.05631, 2023b.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and deep locally connected networks on graphs. arxiv. *arXiv:1312.6203*, 2014.

Chanpuriya, S. and Musco, C. Simplified graph convolution with heterophily. In *NeurIPS*, 2022.

Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. Simple and deep graph convolutional networks. In *ICML*, pp. 1725–1735, 2020.

Chen, Y., Fan, H., Xu, B., Yan, Z., Kalantidis, Y., Rohrbach, M., Yan, S., and Feng, J. Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution. In *ICCV*, pp. 3434–3443, 2019.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. In *ICLR*, 2021.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, pp. 3837–3845, 2016.

Farouki, R. T. The bernstein polynomial basis: A centennial retrospective. *Computer Aided Geometric Design*, 29(6): 379–419, 2012.

Frasca, F., Rossi, E., Eynard, D., Chamberlain, B., Bronstein, M., and Monti, F. SIGN: Scalable inception graph neural networks. In *Graph Representation Learning and Beyond (GRL+) Workshop at ICML*, 2020.

Gautschi, W. *Orthogonal polynomials: computation and approximation*. OUP Oxford, 2004.

Giovanni, F. D., Giusti, L., Barbero, F., Luise, G., Lio, P., and Bronstein, M. M. On over-squashing in message passing neural networks: The impact of width, depth, and topology. In *ICML*, pp. 7865–7885, 2023.

Guo, Y. and Wei, Z. Graph neural networks with learnable and optimal polynomial bases. In *ICML*, pp. 12077–12097, 2023.

Hamilton, W. L., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. In *NIPS*, pp. 1024–1034, 2017.

Hammond, D. K., Vandergheynst, P., and Gribonval, R. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

He, M., Wei, Z., Xu, H., et al. BernNet: Learning arbitrary graph spectral filters via Bernstein approximation. In *NeurIPS*, 2021.

He, M., Wei, Z., and Wen, J. Convolutional neural networks on graphs with chebyshev approximation, revisited. In *NeurIPS*, 2022.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

Huang, K., Tang, J., Liu, J., Yang, R., and Xiao, X. Node-wise diffusion for scalable graph learning. In *WWW*, pp. 1723–1733, 2023.

Huang, K., Cao, W., Ta, H., Xiao, X., and Liò, P. Optimizing polynomial graph filters: A novel adaptive krylov subspace approach. In *WWW*, pp. 1057–1068, 2024.

Karhadkar, K., Banerjee, P. K., and Montúfar, G. Fosr: First-order spectral rewiring for addressing oversquashing in gnns. In *ICLR*, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

Klicpera, J., Bojchevski, A., and Günnemann, S. Predict then propagate: Graph neural networks meet personalized pagerank. In *ICLR*, 2019a.

Klicpera, J., Weißenberger, S., and Günnemann, S. Diffusion improves graph learning. In *NeurIPS*, pp. 13366–13378, 2019b.

Lei, R., Wang, Z., Li, Y., Ding, B., and Wei, Z. EvenNet: Ignoring odd-hop neighbors improves robustness of graph neural networks. In *NeurIPS*, 2022.

Li, P., Chien, I. E., and Milenkovic, O. Optimizing generalized pagerank methods for seed-expansion community detection. In *NeurIPS*, pp. 11705–11716, 2019.

Li, Q., Han, Z., and Wu, X. Deeper insights into graph convolutional networks for semi-supervised learning. In *AAAI*, pp. 3538–3545, 2018.

Li, X., Zhu, R., Cheng, Y., Shan, C., Luo, S., Li, D., and Qian, W. Finding global homophily in graph neural networks when meeting heterophily. In *ICML*, pp. 13242–13256, 2022.

Liesen, J. and Strakos, Z. *Krylov subspace methods: principles and analysis*. Oxford University Press, 2013.

Lim, D., Hohne, F., Li, X., Huang, S. L., Gupta, V., Bhalerao, O., and Lim, S. N. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *NeurIPS*, pp. 20887–20902, 2021.

Luan, S., Hua, C., Lu, Q., Zhu, J., Zhao, M., Zhang, S., Chang, X., and Precup, D. Revisiting heterophily for graph neural networks. In *NeurIPS*, 2022.

Luan, S., Hua, C., Xu, M., Lu, Q., Zhu, J., Chang, X., Fu, J., Leskovec, J., and Precup, D. When do graph neural networks help with node classification? investigating the homophily principle on node distinguishability. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S. (eds.), *NeurIPS*, 2023.

Mason, J. C. and Handscomb, D. C. *Chebyshev polynomials*. Chapman and Hall/CRC, 2002.

Nguyen, K., Hieu, N. M., Nguyen, V. D., Ho, N., Osher, S. J., and Nguyen, T. M. Revisiting over-smoothing and over-squashing using ollivier-ricci curvature. In *ICML*, pp. 25956–25979, 2023.

Pei, H., Wei, B., Chang, K. C., Lei, Y., and Yang, B. Geom-GCN: Geometric graph convolutional networks. In *ICLR*, 2020.

Platonov, O., Kuznedelev, D., Babenko, A., and Prokhorenkova, L. Characterizing graph datasets for node classification: Beyond homophily-heterophily dichotomy. *arXiv preprint arXiv:2209.06177*, 2022.

Rusch, T. K., Bronstein, M. M., and Mishra, S. A survey on oversmoothing in graph neural networks. *CoRR*, abs/2303.10993, 2023.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Galligher, B., and Eliassi-Rad, T. Collective classification in network data. *AI Magazine*, 29(3):93–93, 2008.

Song, Y., Zhou, C., Wang, X., and Lin, Z. Ordered GNN: ordering message passing to deal with heterophily and over-smoothing. In *ICLR*, 2023.

Suresh, S., Budde, V., Neville, J., Li, P., and Ma, J. Breaking the limit of graph neural networks by improving the assortativity of graphs with local mixing patterns. In *KDD*, pp. 1541–1551, 2021.

Tang, J., Sun, J., Wang, C., and Yang, Z. Social influence analysis in large-scale networks. In *KDD*, pp. 807–816, 2009.

Topping, J., Giovanni, F. D., Chamberlain, B. P., Dong, X., and Bronstein, M. M. Understanding over-squashing and bottlenecks on graphs via curvature. In *ICLR*, 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *NeurIPS*, pp. 5998–6008, 2017.

Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., and Bengio, Y. Graph attention networks. In *ICLR*, 2018.

Wang, X. and Zhang, M. How powerful are spectral graph neural networks. In *ICML*, pp. 23341–23362, 2022.

Wu, F., Jr., A. H. S., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Q. Simplifying graph convolutional networks. In *ICML*, pp. 6861–6871, 2019.

Yang, L., Shi, R., Zhang, Q., Niu, B., Wang, Z., Cao, X., and Wang, C. Self-supervised graph neural networks via low-rank decomposition. In *NeurIPS*, 2023.

Zheng, X., Zhou, B., Gao, J., Wang, Y., Lió, P., Li, M., and Montúfar, G. How framelets enhance graph neural networks. In *ICML*, pp. 12761–12771, 2021.

Zheng, X., Wang, Y., Liu, Y., Li, M., Zhang, M., Jin, D., Yu, P. S., and Pan, S. Graph neural networks for graphs with heterophily: A survey. *arXiv preprint arXiv:2202.07082*, 2022.

Zhou, K., Huang, X., Zha, D., Chen, R., Li, L., Choi, S., and Hu, X. Dirichlet energy constrained learning for deep graph neural networks. In *NeurIPS*, pp. 21834–21846, 2021.

Zhu, J., Yan, Y., Zhao, L., Heimann, M., Akoglu, L., and Koutra, D. Beyond homophily in graph neural networks: Current limitations and effective designs. In *NeurIPS*, 2020.

Zhu, M., Wang, X., Shi, C., Ji, H., and Cui, P. Interpreting and unifying graph neural networks with an optimization framework. In *WWW*, pp. 1215–1226, 2021.

A. Notations and Tables

Table 4: Frequently used notations.

Notation	Description
$\mathbf{G} = (\mathcal{V}, \mathcal{E})$	an undirected and connected graph with node set $\mathcal{V}$ and edge set $\mathcal{E}$
n, m	the numbers of nodes and edges in $\mathbf{G}$ respectively
$[n]$	the integer set $\{1, 2, \cdots, n\}$
$\mathcal{N}_u, d_u$	the neighbor set of node u and its degree
$\mathbf{X} \in \mathbb{R}^{n \times d}$	the feature matrix / node signals in d-dimension
$\mathbf{Y} \in \mathbb{N}^{n \times \lvert\mathcal{C}\rvert}$	the one-hot label matrix with the set of node labels $\mathcal{C}$
$\mathbf{A}, \tilde{\mathbf{A}}$	the adjacency matrix of $\mathbf{G}$ without and with self-loops respectively
$\mathbf{D}, \tilde{\mathbf{D}}$	the diagonal degree matrix of $\mathbf{G}$ without and with self-loops respectively
$\mathcal{L}, \hat{\mathcal{L}}$	the normalized Laplacian matrix of $\mathbf{G}$ without and with self-loops respectively
$\mathbf{P}, \mathbf{P}_{\theta,k}$	the propagation matrix and the k-th rotation matrix
λ_i	the i-th eigenvalue of Laplacian matrix $\mathcal{L}$
K	the number of propagation hop
$\mathbf{w} \in \mathbb{R}^{K+1}$	the learnable of weight vector of polynomial basis
$\mathbf{U}, \mathbf{\Lambda}$	the eigenvector matrix and diagonal eigenvalue matrix of $\mathcal{L}$
$\mathbf{Z} \in \mathbb{R}^{n \times d}$	the filtered node representations / signals in d-dimension
$h, \hat{h}$	the homophily ratio of graphs defined in Definition 2.1 and its estimation
$f(\cdot)$	the spectral signal frequency function defined in Definition 2.2

Table 5: Polynomial Graph Filters

	Poly. Basis	Graph Filter $\mathrm{g}_{\mathbf{w}}(\lambda)$	Prop. Matrix $\mathbf{P}$
ChebNet (Defferrard et al., 2016)	Chebyshev	$\sum_{k=0}^{K} \mathbf{w}_k T_k(\hat{\lambda})$	$2\mathcal{L}/\lambda_{max} - \mathbf{I}$
GPR-GNN (Chien et al., 2021)	Monomial	$\sum_{k=0}^{K} \mathbf{w}_k (1-\tilde{\lambda})^k$	$\mathbf{I} - \hat{\mathcal{L}}$
BernNet (He et al., 2021)	Bernstein	$\sum_{k=0}^{K} \frac{\mathbf{w}_k}{2^K}\binom{K}{k}(2-\lambda)^{K-k}\lambda^k$	$\mathbf{I} - \frac{\mathcal{L}}{2}$
JacobiConv (Wang & Zhang, 2022)	Jacobi	$\sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}_k^{a,b}(1-\lambda)$	$\mathbf{I} - \mathcal{L}$
OptBasisGNN (Guo & Wei, 2023)	Orthonormal	—	$\mathbf{I} - \mathcal{L}$

B. Proofs

B.1. Proof of Proposition 2.3

Proof. Given a graph $\mathbf{G} = (\mathcal{V}, \mathcal{E})$ and a normalized signal $\mathbf{x} \in \mathbb{R}^n$ on $\mathbf{G}$, the signal frequency $f(x)$ of $\mathbf{x}$ is

$$f(\mathbf{x}) = \frac{\mathbf{x}^\top \mathcal{L}\mathbf{x}}{2} = \frac{\sum_{\langle u,v\rangle \in \mathcal{E}} (\mathbf{x}_u - \mathbf{x}_v)^2}{2\sum_{u \in \mathcal{V}} \mathbf{x}_u^2 d_u}$$

where $\mathcal{L}$ is the normalized Laplacian matrix defined on $\mathbf{G}$. Therefore, $f(\mathbf{x}) \geq 0$ holds. Meanwhile, it is known that $(\mathbf{x}_u - \mathbf{x}_v)^2 \leq 2(\mathbf{x}_u^2 + \mathbf{x}_v^2)$ holds for each edge $\langle u, v\rangle \in \mathcal{E}$ and the equality holds when $\mathbf{x}_u = -\mathbf{x}_v$. Therefore, we have

$$\sup_{\mathbf{x} \in \mathbb{R}^n} \frac{\sum_{\langle u,v\rangle \in \mathcal{E}} (\mathbf{x}_u - \mathbf{x}_v)^2}{\sum_{u \in \mathcal{V}} \mathbf{x}_u^2 d_u} \leq 2,$$

i.e., $f(\mathbf{x}) \leq 1$ for any $\mathbf{x} \in \mathbb{R}^n$, which completes the proof. □

B.2. Proof of Theorem 3.1

Proof. Given a connected graph $\mathbf{G} = (\mathcal{V}, \mathcal{E})$ with a feature signal $\mathbf{x}$ and homophily ratio h, consider an optimal polynomial filter $\mathrm{F}(\mathbf{w})$. Denote $\mathbf{z} = \mathrm{F}(\mathbf{w})\mathbf{x} = \sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}^k \mathbf{x}$. Thus we have

$$f(\mathrm{F}(\mathbf{w})\mathbf{x}) = f(\mathbf{z}) = \frac{\mathbf{z}^\top \mathcal{L} \mathbf{z}}{2} = \frac{\sum_{\langle u,v\rangle \in \mathcal{E}} (\mathbf{z}_u - \mathbf{z}_v)^2}{2\sum_{u\in\mathcal{V}} \mathbf{z}_u^2 d_u}.$$

As $\mathrm{F}(\mathbf{w})$ is the optimal filter for node classification, it is reasonable to assume that the filtered signals of nodes belonging to the same classes tend to cluster together, while signals of nodes from distinct classes exhibit a notable spatial separation.

W.l.o.g., for $\forall u, v \in \mathcal{V}$, we assume a constant δ such that $|\mathbf{z}_u - \mathbf{z}_v| \leq c\delta$ with $c \ll 1$ if $\mathbf{Y}_u = \mathbf{Y}_v$; otherwise $|\mathbf{z}_u - \mathbf{z}_v| = g(\mathbf{Y}_u, \mathbf{Y}_v)\delta$ with $g(\mathbf{Y}_u, \mathbf{Y}_v) \geq 1$ where $\mathbf{Y} \in \mathbb{N}^{n\times|\mathcal{C}|}$ is the one-hot label matrix and $g(\cdot)$ is a function parameterized by $\mathbf{Y}_u$ and $\mathbf{Y}_v$. Consequently, the spectral frequency $f(\mathbf{z}) = \frac{\sum_{\langle u,v\rangle \in \mathcal{E}} (\mathbf{z}_u - \mathbf{z}_v)^2}{2\sum_{u\in\mathcal{V}} \mathbf{z}_u^2 d_u}$ approaches to

$$\frac{c^2\delta^2 hm + \sum_{\langle u,v\rangle \in \mathcal{E}, \mathbf{Y}_u \neq \mathbf{Y}_v} g^2(\mathbf{Y}_u, \mathbf{Y}_v)\delta^2}{2\sum_{u\in\mathcal{V}} \mathbf{z}_u^2 d_u}.$$

Since $c \ll 1$ and $g(\mathbf{Y}_u, \mathbf{Y}_v) \geq 1$, therefore $g^2(\mathbf{Y}_u, \mathbf{Y}_v) \gg c^2$ holds. In this regard, frequency $f(\sum_{k=0}^{K} \mathbf{w}_k \mathbf{P}^k \mathbf{x})$ is dominated by

$$\frac{\sum_{\langle u,v\rangle \in \mathcal{E}, \mathbf{Y}_u \neq \mathbf{Y}_v} g^2(\mathbf{Y}_u, \mathbf{Y}_v)\delta^2}{2\sum_{u\in\mathcal{V}} \mathbf{z}_u^2 d_u},$$

i.e., the number of edges connecting nodes from distinct classes, which is proportional to $1 - h$. □

B.3. Proof of Theorem 4.1

Proof. Consider a propagation matrix $\mathbf{P}$ and graph signal $\mathbf{x}$. Let $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$[3] be the eigenvalues of $\mathbf{P}$ associated with eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n\}$. For a general (non-bipartite) connected graph $\mathbf{G}$, we have $-1 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n = 1$ and $\mathbf{v}_i^\top \mathbf{v}_j = 0$ for $i \neq j$ and $\mathbf{v}_i^\top \mathbf{v}_j = 1$ for $i = j$. In particular, we have $\lambda_n = 1$ and $\mathbf{v}_n = \frac{\mathbf{D}^{\frac{1}{2}}\mathbf{1}}{\sqrt{2m}}$ where $\mathbf{1} \in \mathbb{R}^n$ is the all-one vector. Hence, $\mathbf{P}^k\mathbf{x}$ can be formulated as $\mathbf{P}^k\mathbf{x} = \sum_{i=1}^{n} \lambda_i^k (\mathbf{v}_i^\top \mathbf{x})\mathbf{v}_i$. Specifically, we have

$$\|\mathbf{P}^k\mathbf{x}\| = \sqrt{\left(\sum_{i=1}^{n} \lambda_i^k (\mathbf{v}_i^\top \mathbf{x})\mathbf{v}_i\right) \cdot \left(\sum_{j=1}^{n} \lambda_j^k (\mathbf{v}_j^\top \mathbf{x})\mathbf{v}_j\right)} = \sqrt{\sum_{i=1}^{n} \lambda_i^{2k} (\mathbf{v}_i^\top \mathbf{x})^2}.$$

Similarly, we have

$$\mathbf{P}^k\mathbf{x} \cdot \mathbf{P}^{k+1}\mathbf{x} = \left(\sum_{i=1}^{n} \lambda_i^k (\mathbf{v}_i^\top \mathbf{x})\right) \cdot \left(\sum_{i=1}^{n} \lambda_i^{k+1} (\mathbf{v}_i^\top \mathbf{x})\right) = \sum_{i=1}^{n} \lambda_i^{2k+1} (\mathbf{v}_i^\top \mathbf{x})^2.$$

Therefore, we have

$$\frac{\mathbf{P}^k\mathbf{x} \cdot \mathbf{P}^{k+1}\mathbf{x}}{\|\mathbf{P}^k\mathbf{x}\|\|\mathbf{P}^{k+1}\mathbf{x}\|} = \frac{\sum_{i=1}^{n} \lambda_i^{2k+1} (\mathbf{v}_i^\top \mathbf{x})^2}{\sqrt{\sum_{i=1}^{n} \lambda_i^{2k} (\mathbf{v}_i^\top \mathbf{x})^2}\sqrt{\sum_{i=1}^{n} \lambda_i^{2k+2} (\mathbf{v}_i^\top \mathbf{x})^2}}.$$

For ease of exposition, we denote $c_k = \|\mathbf{P}^k\mathbf{x}\|$ and let t be the integer index such that $\lambda_t < 0 \leq \lambda_{t+1}$. As a result, we have

$$\sum_{i=1}^{n} \left(\frac{\lambda_i^k (\mathbf{v}_i^\top \mathbf{x})}{c_k}\right)^2 = \sum_{i=1}^{n} \left(\frac{\lambda_i^{k+1} (\mathbf{v}_i^\top \mathbf{x})}{c_{k+1}}\right)^2 = 1 \tag{6}$$

and

$$\frac{\mathbf{P}^k\mathbf{x} \cdot \mathbf{P}^{k+1}\mathbf{x}}{\|\mathbf{P}^k\mathbf{x}\|\|\mathbf{P}^{k+1}\mathbf{x}\|} = \sum_{i=1}^{t} \frac{\lambda_i^{2k+1} (\mathbf{v}_i^\top \mathbf{x})^2}{c_k c_{k+1}} + \sum_{i=t}^{n} \frac{\lambda_i^{2k+1} (\mathbf{v}_i^\top \mathbf{x})^2}{c_k c_{k+1}}.$$

[3] For notation simplicity, we here reuse the notation λ_i as the i-th eigenvalue of propagation matrix $\mathbf{P}$.

Note that the exponent $2k+1$ remains odd integer for any k values. W.l.o.g., we denote the negative part as function

$$f_N(t,k) = \sum_{i=1}^{t} \frac{\lambda_i^{2k+1}(\mathbf{v}_i^\top \mathbf{x})^2}{c_k c_{k+1}}$$

and the positive part as function

$$f_P(t,k) = \sum_{i=t}^{n} \frac{\lambda_i^{2k+1}(\mathbf{v}_i^\top \mathbf{x})^2}{c_k c_{k+1}}.$$

Notice that $\lambda_i \in (-1,0)$ for $i \in \{1,\cdots,t\}$, $\lambda_i \in [0,1]$ ($\lambda_n = 1$) for $i \in \{t+1,\cdots,n\}$, and $(\mathbf{v}_i^\top \mathbf{x})^2$ for $i \in \{1,\cdots,n\}$ are constants. When k increases, numerator $\sum_{i=1}^{t} \lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2$ of $\frac{\sum_{i=1}^{t} \lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2}{c_k^2}$ monotonically decreases and asymptotically approaches to 0. According to Equation (6),

$$\sum_{i=1}^{t} \frac{\lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2}{c_k^2} + \sum_{i=t}^{n} \frac{\lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2}{c_k^2} = 1$$

holds for all k. Therefore, when the case $\sum_{i=1}^{t} \lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2 > \sum_{i=t}^{n} \lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2$ occurs for certain k, there exists an integer $\eta \in \mathbb{N}$ such that

$$\sum_{i=1}^{t} \lambda_i^{2(\eta-1)}(\mathbf{v}_i^\top \mathbf{x})^2 > \sum_{i=t}^{n} \lambda_i^{2(\eta-1)}(\mathbf{v}_i^\top \mathbf{x})^2$$

and

$$\sum_{i=1}^{t} \lambda_i^{2\eta}(\mathbf{v}_i^\top \mathbf{x})^2 \leq \sum_{i=t}^{n} \lambda_i^{2\eta}(\mathbf{v}_i^\top \mathbf{x})^2.$$

Considering the monotonicity of both $\sum_{i=1}^{t} \lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2$ and $\sum_{i=t}^{n} \lambda_i^{2k}(\mathbf{v}_i^\top \mathbf{x})^2$, when $k \geq \eta$, $f_N(t,k)$ monotonically decreases while $f_P(t,k)$ monotonically increases with k. As a consequence, $\frac{\mathbf{P}^k\mathbf{x}\cdot\mathbf{P}^{k+1}\mathbf{x}}{\|\mathbf{P}^k\mathbf{x}\|\|\mathbf{P}^{k+1}\mathbf{x}\|}$ is monotonically increasing with k, and thus the angle angle $\arccos\left(\frac{\mathbf{P}^k\mathbf{x}\cdot\mathbf{P}^{k+1}\mathbf{x}}{\|\mathbf{P}^k\mathbf{x}\|\|\mathbf{P}^{k+1}\mathbf{x}\|}\right)$ is progressively smaller.

Meanwhile, when $K \to \infty$, we have

$$\lim_{K\to\infty} \mathbf{P}^K\mathbf{x} = \lambda_n^K(\mathbf{v}_n^\top \mathbf{x})\mathbf{v}_n = \frac{\mathbf{v}_n^\top \mathbf{x}}{\sqrt{2m}}\mathbf{D}^{\frac{1}{2}}\mathbf{1}$$

where $\mathbf{v}_n = \frac{\mathbf{D}^{\frac{1}{2}}\mathbf{1}}{\sqrt{2m}}$. Therefore, $\lim_{K\to\infty} \arccos\left(\frac{\mathbf{P}^K\mathbf{x}\cdot\mathbf{P}^{K+1}\mathbf{x}}{\|\mathbf{P}^K\mathbf{x}\|\|\mathbf{P}^{K+1}\mathbf{x}\|}\right) \to 0$ holds. □

B.4. Proof of Theorem 4.2

Proof. We consider a t-regular graph $\mathbf{G} = (\mathcal{V},\mathcal{E})$ with n nodes without self-loops. Given a random normalized signal $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n)^\top$, $\phi \cdot \mathbf{x} = \sum_{i=1}^{n} \frac{\mathbf{x}_i}{\sqrt{n}}$ where $\phi \in \mathbb{R}^n$ is the normalized all-ones vector with $f(\phi) = 0$. Meanwhile, recall that

$$f(\mathbf{x}) = \frac{\mathbf{x}^\top \mathcal{L}\mathbf{x}}{2} = \frac{\sum_{\langle u,v\rangle\in\mathcal{E}}(\mathbf{x}_u - \mathbf{x}_v)^2}{2\sum_{u\in\mathcal{V}} \mathbf{x}_u^2 d_u} = \frac{\sum_{\langle u,v\rangle\in\mathcal{E}}(\mathbf{x}_u - \mathbf{x}_v)^2}{2t}$$

since $\sum_{u\in\mathcal{V}} \mathbf{x}_u^2 = 1$ and $d_u = t\ \forall u \in \mathcal{V}$. Over the randomness of $\mathbf{G}$, any two nodes $u, v \in \mathcal{V}$ is connected with probability $\frac{t}{n-1}$. Thus, the expectation of the spectral signal frequency is

$$\begin{aligned}
\mathbb{E}_{\mathbf{G}\sim\mathcal{G}}[f(\mathbf{x})] &= \tfrac{1}{2t}\cdot\tfrac{t}{n-1}\sum_{u\in\mathcal{V}}\sum_{v\in\mathcal{V}\setminus\{u\}}\tfrac{(\mathbf{x}_u-\mathbf{x}_v)^2}{2} \\
&= \tfrac{1}{4(n-1)}\sum_{u\in\mathcal{V}}\sum_{v\in\mathcal{V}\setminus\{u\}}(\mathbf{x}_u-\mathbf{x}_v)^2 \\
&= \tfrac{1}{4(n-1)}\left(\sum_{u\in\mathcal{V}}(n-1)\mathbf{x}_u^2 - \sum_{u\in\mathcal{V}}2\mathbf{x}_u\big(\sum_{v\in\mathcal{V}\setminus\{u\}}\mathbf{x}_v\big)\right) \\
&= \tfrac{1}{4(n-1)}\left(n-1-\sum_{u\in\mathcal{V}}2\mathbf{x}_u\big(\sum_{i=1}^{n}\mathbf{x}_i-\mathbf{x}_u\big)\right) \\
&= \tfrac{1}{4(n-1)}\left(n-1-2\big(\sum_{i=1}^{n}\mathbf{x}_i\big)^2+2\right) \\
&= \tfrac{1}{4(n-1)}\left(n+1-2\big(\sum_{i=1}^{n}\mathbf{x}_i\big)^2\right) \\
&= \tfrac{n+1}{4(n-1)} - \tfrac{1}{2(n-1)}\big(\sum_{i=1}^{n}\tfrac{\mathbf{x}_i}{\sqrt{n}}\big)^2 \\
&= \tfrac{n+1-2(\phi\cdot\mathbf{x})^2}{4(n-1)}.
\end{aligned}$$

As a consequence, if the angle $\theta := \arccos(\phi\cdot\mathbf{x})$ increases, $\phi\cdot\mathbf{x}$ decreases, resulting the increment of $\mathbb{E}_{\mathbf{G}\sim\mathcal{G}}[f(\mathbf{x})]$, which completes the proof. □

B.5. Proof of Theorem 4.3

Before the proof of Theorem 4.3, we first introduce the following Lemma.

Lemma B.1 (Proposition 4.3 (Guo & Wei, 2023)). *Vector $\mathbf{v}_k$ in Algorithm 1 is only dependent with $\mathbf{v}_{k-1}$ and $\mathbf{v}_{k-2}$.*

Based on Lemma B.1, it is intuitive that $\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_K\}$ forms an orthonormal basis. Before proceeding, we establish the following lemma.

Lemma B.2. *$\mathbf{v}_{k+1}$ constructed in Algorithm 1 is orthogonal to basis set $\{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_k\}$ for $k \in \{0, 1, \cdots, K-1\}$.*

Proof of Lemma B.2. In particular, we have $\mathbf{u}_0 = \mathbf{v}_0 = \frac{\mathbf{x}}{\|\mathbf{x}\|}$ and

$$\mathbf{u}_k = \frac{\mathbf{s}_{k-1}/k + t_k\mathbf{v}_k}{\|\mathbf{s}_{k-1}/k + t_k\mathbf{v}_k\|} = \frac{\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k}{\|\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k\|}$$

for $k \in [K]$ according to Algorithm 1. By recursively replacing $\mathbf{u}_i$ with $\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_i\}$, we obtain that $\mathbf{u}_k$ can be constructed by $\mathbf{v}_i$. In this regard, let $\{\alpha_0, \alpha_1, \cdots, \alpha_k\} \in \mathbb{R}$ be constants such that $\mathbf{u}_k = \sum_{i=0}^{k}\alpha_i\mathbf{v}_i$ holds. Since $\{\mathbf{v}_0, \mathbf{v}_1, \cdots, \mathbf{v}_K\}$ is an orthonormal basis, therefore $\mathbf{v}_{k+1}$ is orthogonal to $\{\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_k\}$ as $\mathbf{v}_{k+1}\cdot\mathbf{u}_k = \mathbf{v}_{k+1}\cdot(\sum_{i=0}^{k}\alpha_i\mathbf{v}_i) = 0$. □

Proof of Theorem 4.3. Let $\theta := \frac{(1-\hat{h})\pi}{2}$ be the predefined angle. First, we prove that $\mathbf{u}_0\cdot\mathbf{u}_1 = \cos\theta$. In particular, we have $\mathbf{u}_1 = \frac{\mathbf{u}_0+t_1\mathbf{v}_1}{\|\mathbf{u}_0+t_1\mathbf{v}_1\|}$ and $t_1 = \sqrt{(\frac{\mathbf{s}_0^\top\mathbf{u}_0}{\cos\theta})^2 - 1} = \sqrt{\frac{1}{\cos^2\theta} - 1}$. Then

$$\|\mathbf{u}_0 + t_1\mathbf{v}_1\| = \sqrt{\mathbf{u}_0\cdot\mathbf{u}_0 + t_1^2\mathbf{v}_1\cdot\mathbf{v}_1 + 2t_1\mathbf{u}_0\cdot\mathbf{v}_1} = \sqrt{1+t_1^2} = \tfrac{1}{\cos\theta}$$

as $\mathbf{u}_0 \cdot \mathbf{v}_1 = 0$ according to Lemma B.2. Hence, $\mathbf{u}_0 \cdot \mathbf{u}_1 = \mathbf{u}_0^\top(\mathbf{u}_0 + t_1\mathbf{v}_1)\cos\theta = \cos\theta$.

Second, we assume that $\mathbf{u}_i \cdot \mathbf{u}_j = \cos\theta$ holds for $\forall i, j \in \{0, 1, \cdots, k-1\}$ and $i \neq j$. In what follows, we then prove that $\mathbf{u}_k \cdot \mathbf{u}_j = \cos\theta$ holds for $j \in \{0, 1, \cdots, k-1\}$. Specifically, we have

$$\mathbf{u}_k = \frac{\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k}{\|\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k\|}$$

where

$$t_k = \sqrt{\left(\frac{\mathbf{s}_{k-1}^\top\mathbf{u}_{k-1}}{k\cos\theta}\right)^2 - \frac{(k-1)\cos\theta+1}{k}}, \text{ and } \mathbf{s}_{k-1} = \sum_{i=0}^{k-1}\mathbf{u}_i.$$

In particular, for the denominator of $\mathbf{u}_k$, we have

$$\begin{aligned}
\left\|\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k\right\| &= \sqrt{\left(\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i^\top + t_k\mathbf{v}_k^\top\right)\left(\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k\right)} \\
&= \sqrt{\frac{\sum_{i=0}^{k-1}\mathbf{u}_i^\top\mathbf{u}_i + 2\sum_{i=0}^{k-2}\mathbf{u}_i^\top(\sum_{j=i+1}^{k-1}\mathbf{u}_j)}{k^2} + t_k^2} \\
&= \sqrt{\frac{k+k(k-1)\cos\theta}{k^2} + \left(\frac{\mathbf{s}_{k-1}^\top\mathbf{u}_{k-1}}{k\cos\theta}\right)^2 - \frac{(k-1)\cos\theta+1}{k}} \\
&= \frac{\sum_{i=0}^{k-1}\mathbf{u}_i^\top \cdot \mathbf{u}_{k-1}}{k\cos\theta} \\
&= \frac{1+(k-1)\cos\theta}{k\cos\theta}.
\end{aligned}$$

Meanwhile, we have

$$\mathbf{u}_k \cdot \mathbf{u}_j = \frac{\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i^\top\mathbf{u}_j + t_k\mathbf{v}_k^\top\mathbf{u}_j}{\|\frac{1}{k}\sum_{i=0}^{k-1}\mathbf{u}_i + t_k\mathbf{v}_k\|} = \frac{\frac{1}{k}(1+(k-1)\cos\theta)\cdot k\cos\theta}{1+(k-1)\cos\theta} = \cos\theta.$$

Eventually, it is easy to verify that the derivation holds for $\forall j \in \{0, 1, \cdots, k-1\}$, which completes the proof. □

B.6. Proofs of Over-smoothing and Over-Squashing

Proof of Theorem 5.2. Given a feature matrix $\mathbf{X}^k$ from the k-th layer, over-smoothing occurs if Dirichlet energy $E(\mathbf{G}, \mathbf{X}^k) \to 0$ for a sufficiently large k. In what follows, we prove that $E(\mathbf{G}, \mathbf{X}^k)$ of $\mathbf{X}^k$ obtained by UniFilter remains constant when $k \to \infty$. Specifically, let $\mathbf{X}^k = (\tau\mathbf{P}^k + (1-\tau)\mathbf{P}_{\theta,k})\mathbf{X}$ be the k-th layer propagated feature matrix where $\mathbf{P}$ is the propagation matrix and $\mathbf{P}_{\theta,k}$ is the k-th rotation matrix.

For $E(\mathbf{G}, \mathbf{X}^k)$ and $k \to \infty$, we have

$$\begin{aligned}
E(\mathbf{G}, \mathbf{X}^k) &= \frac{1}{n}\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{N}_v}\|\mathbf{X}_v^{(k)} - \mathbf{X}_u^{(k)}\|_2^2 \\
&= \frac{1}{n}\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{N}_v}\|\tau\mathbf{P}^k\mathbf{X}[v] - \tau\mathbf{P}^k\mathbf{X}[u] + (1-\tau)\mathbf{P}_{\theta,k}\mathbf{X}[v] - (1-\tau)\mathbf{P}_{\theta,k}\mathbf{X}[u]\|_2^2.
\end{aligned}$$

When $k \to \infty$, $\lim_{K\to\infty}\mathbf{P}^K\mathbf{x}$ converges to $\mathbf{v}_n$, as pointed out in the proof of Theorem 4.1. Therefore $\tau\mathbf{P}^k\mathbf{X}[v] = \tau\mathbf{P}^k\mathbf{X}[u]$. Thus we have

$$\begin{aligned}
\lim_{K\to\infty} E(\mathbf{G}, \mathbf{X}^k) &= \frac{1}{n}\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{N}_v}\|(1-\tau)\mathbf{P}_{\theta,k}\mathbf{X}[v] - (1-\tau)\mathbf{P}_{\theta,k}\mathbf{X}[u]\|_2^2 \\
&= \frac{1-\tau}{n}\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{N}_v}\|\mathbf{P}_{\theta,k}\mathbf{X}[v] - \mathbf{P}_{\theta,k}\mathbf{X}[u]\|_2^2
\end{aligned}$$

According to Definition 5.1, there exists a unitary matrix $\mathbf{U} \in \mathbf{C}^{n\times n}$ such that $\mathbf{R}(\theta) := \mathbf{U}^{-1}\mathbf{P}_{\theta,k}\mathbf{U}$. Therefore, we have

$$\|\mathbf{P}_{\theta,k}\mathbf{X}[v] - \mathbf{P}_{\theta,k}\mathbf{X}[u]\|_2^2 = \|\mathbf{U}\mathbf{R}(\theta)\mathbf{U}^{-1}\mathbf{X}[v] - \mathbf{U}\mathbf{R}(\theta)\mathbf{U}^{-1}\mathbf{X}[u]\|_2^2.$$

Notice that both $\mathbf{R}(\theta)$ and $\mathbf{U}^{-1}$ are unitary matrices. According to the isometry property of the unitary matrix, we have

$$\begin{aligned}
\lim_{K\to\infty} E(\mathbf{G}, \mathbf{X}^k) &= \tfrac{1-\tau}{n} \sum_{v\in\mathcal{V}} \sum_{u\in\mathcal{N}_v} \|\mathbf{U}\mathbf{R}(\theta)\mathbf{U}^{-1}\mathbf{X}[v] - \mathbf{U}\mathbf{R}(\theta)\mathbf{U}^{-1}\mathbf{X}[u]\|_2^2 \\
&= \tfrac{1-\tau}{n} \sum_{v\in\mathcal{V}} \sum_{u\in\mathcal{N}_v} \|\mathbf{R}(\theta)\mathbf{U}^{-1}\mathbf{X}[v] - \mathbf{R}(\theta)\mathbf{U}^{-1}\mathbf{X}[u]\|_2^2 \\
&= \tfrac{1-\tau}{n} \sum_{v\in\mathcal{V}} \sum_{u\in\mathcal{N}_v} \|\mathbf{U}^{-1}\mathbf{X}[v] - \mathbf{U}^{-1}\mathbf{X}[u]\|_2^2 \\
&= \tfrac{1-\tau}{n} \sum_{v\in\mathcal{V}} \sum_{u\in\mathcal{N}_v} \|\mathbf{X}_v - \mathbf{X}_u\|_2^2
\end{aligned}$$

As a consequence, $\lim_{K\to\infty} E(\mathbf{G}, \mathbf{X}^k) = (1-\tau)E(\mathbf{G}, \mathbf{X})$, which completes the proof. □

Proof of Theorem 5.3. As discussed, Jacobian $|\partial \mathbf{z}_u^{(k)}/\partial \mathbf{x}_v|$ measures the sensitivity of information received at node u to the signal of node v when propagating k distance. Over-squashing occurs if $|\partial \mathbf{z}_u^{(k)}/\partial \mathbf{x}_v| \leq c \cdot (\mathbf{P}_M^k)_{uv}$ where c is a constant and $\mathbf{P}_M$ is the message-passing matrix, i.e., the sensitivity (node dependence) decays exponentially to propagation distance. According to Equation (5), we have

$$\left|\frac{\partial \mathbf{z}_u^{(k)}}{\partial \mathbf{x}_v}\right| = \mathbf{w}_k\big(\tau(\mathbf{P}^k)_{uv} + (1-\tau)(\mathbf{P}_{\theta,k})_{uv}\big)$$

in UniFilter. Here, $\mathbf{w}_k$ as the learnable parameter can be seen as a constant. Note that the rotation matrix $\mathbf{P}_{\theta,k}$ is determined by rotation angle θ and rotation axis without utilizing the underlying graph topology. Therefore, the value of $(\mathbf{P}_{\theta,k})_{uv}$ is independent of propagation step k, which completes the proof. □

Table 6: Dataset details.

Dataset	Cora	Citeseer	Pubmed	Actor	Chameleon	Squirrel	Penn94	Genius	Ogbn-arxiv
#Nodes (n)	2,708	3,327	19,717	7,600	2,277	5,201	41,554	421,961	169,343
#Edges (m)	5,429	4,732	44,338	26,659	31,371	198,353	1,362,229	984,979	1,166,243
#Features (d)	1,433	3,703	500	932	2,325	2,089	5	12	128
#Classes	7	6	3	5	5	5	2	2	40
Homo. ratio (h)	0.81	0.74	0.80	0.22	0.23	0.22	0.47	0.62	0.65

Table 7: Selections of τ for the tested datasets.

Dataset	Cora	Citeseer	Pubmed	Actor	Chameleon	Squirrel	Penn94	Genius	Ogbn-arxiv
τ	1.0	0.9	0.8	0.1	0.7	0.7	0.9	0.0	0.5

C. Experimental Settings

Datasets. Table 6 presents the detailed statistics of the 9 real-world datasets tested in our experiments. The three homophily datasets, i.e., Cora, Citeseer, and Pubmed are citation networks with homophily ratios of 0.81, 0.74, and 0.80 respectively. Each graph node represents a research paper, and each edge denotes a citation relationship. Feature vectors of nodes are bag-of-words representations. The one-hot label assigned to each node stands for one research field of the paper. The rest three small datasets i.e., Actor, Chameleon, and Squirrel are heterophily datasets with homophily ratios of 0.22, 0.23, and 0.22 respectively. Specifically, Actor is a co-occurrence graph from the film-director-actor-writer network from WebKB3 (Tang et al., 2009; Pei et al., 2020). Squirrel and Chameleon are two datasets extracted from Wikipedia web pages, and nodes are categorized by the average amounts of monthly traffic (Li et al., 2022). The additional three large datasets, i.e., Penn94, Genius, and Ogbn-arxiv are with homophily ratios of 0.47, 0.62, and 0.65 respectively. Specifically, Penn94 and Genius are from the LINKX datasets (Lim et al., 2021), and Ogbn-arxiv is a citation network from (Hu et al., 2020).

Running environment. All our experiments are conducted on a Linux machine with an NVIDIA RTX A5000 (24GB memory), Intel Xeon(R) CPU (2.80GHz), and 500GB RAM.

Parameter settings. During the training process, learnable parameters are tuned with Adam (Kingma & Ba, 2015) optimizer. We set a patience of early stopping with 200 epochs. For hyperparameters, we fix propagation hop $K = 10$ for all tested datasets. The rest hyperparameters are selected in the following ranges.

1. Learning rate: $[0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2]$;

2. Hidden dimension: $[64, 128, 256]$;

3. MLP layers: $[2, 3, 4, 5, 6]$;

4. Weight decays: $[0, 1e-4, 5e-4, 0.001]$;

5. Drop rates: $[0, 0.1, 0.2, \cdots, 0.9]$.

Choice of τ. Ideally, the selection of τ for each dataset is highly related to its homophily ratio. In our experiments, the τ values are set as in Table 7.

Homophily ratio estimation. For each dataset, 10 random splits of training/validation/test are generated. The homophily ratio is estimated using the training set from each split independently as the input to UniFilter for the construction of polynomial bases. The reported accuracy score is average with a standard deviation over the results on the 10 splits of each tested dataset.

D. Additional Experiments

Comparison on large datasets. For a comprehensive evaluation, we add three large datasets, i.e., Penn94, Genius, and Ogbn-arxiv which are summarized in Table 6. For large graphs, we stick to tuning parameters by following the parameter

Table 8: Accuracy (%) on large datasets.

Methods	Penn94	Genius	Ogbn-arxiv
GCN	82.47 ± 0.27	87.42 ± 0.37	71.74 ± 0.29
SGC	81.44 ± 0.15	87.59 ± 0.10	71.21 ± 0.17
SIGN	82.13 ± 0.28	88.19 ± 0.02	71.95 ± 0.12
ChebNet	82.59 ± 0.31	89.36 ± 0.31	71.12 ± 0.22
GPR-GNN	83.54 ± 0.32	90.15 ± 0.30	71.78 ± 0.18
BernNet	83.26 ± 0.29	90.47 ± 0.33	71.96 ± 0.27
ChebNetII	$\mathbf{84.86 \pm 0.33}$	$\underline{90.85 \pm 0.32}$	$\underline{72.32 \pm 0.23}$
OptBasisGNN	$\underline{84.85 \pm 0.39}$	90.83 ± 0.11	72.27 ± 0.15
UniFilter	84.46 ± 0.33	$\mathbf{90.94 \pm 0.22}$	$\mathbf{73.70 \pm 0.70}$

[4]The highest score is in bold while the second-highest score is underlined for each dataset.

Table 9: Estimated homophily ratio $\hat{h}$ over varying training percentages on Cora and Squirrel.

Dataset	10%	20%	30%	40%	50%	60%
Cora	0.83 ± 0.05	0.83 ± 0.04	0.83 ± 0.03	0.83 ± 0.01	0.82 ± 0.08	0.82 ± 0.01
Squirrel	0.23 ± 0.014	0.22 ± 0.011	0.22 ± 0.010	0.22 ± 0.006	0.22 ± 0.005	0.22 ± 0.005

Table 10: Estimated homophily ratio $\hat{h}$ over varying training percentages on Cora and Squirrel.

Varying $\hat{h}$	0.78	0.79	0.80	0.81	0.82
Squirrel	91.03 ± 0.61	91.28 ± 0.64	91.34 ± 0.62	91.19 ± 0.67	91.17 ± 0.69
Varying $\hat{h}$	0.19	0.20	0.21	0.22	0.23
Pubmed	66.22 ± 1.43	66.96 ± 1.38	67.09 ± 1.08	67.01 ± 1.25	66.69 ± 1.26

settings in Appendix C. In particular, we compare UniFilter against existing polynomial filters. The results are presented in Table 8. As demonstrated, UniFilter achieves the highest scores on Genius and Obgn-arxiv and approaches the highest score on Penn94. Specifically, UniFilter advances the accuracy score up to 1.38% on Ogbn-arxiv. Those observations further confirm the universality and efficacy of UniBasis.

Robustness of homophily ratio estimation. To further verify the robustness of the estimation, we vary the percentages of the training set in $\{10\%, 20\%, 30\%, 40\%, 50\%, 60\%\}$ on Cora and Squirrel respectively. Next, we generate 10 random training sets in each of the above percentages. Subsequently, we average the estimated homophily ratios $\hat{h}$ over the 10 splits and report the results in Table 9. As shown, the estimated homophily ratio $\hat{h}$ is approaching the true homophily ratio h across varying percentages of training data. This observation verifies that a high-quality estimation of the homophily ratio is accessible effectively by a small proportion of training sets.

Sensitivity of UniFilter to $\hat{h}$. We conduct an ablation study on the sensitivity of our proposed UniFilter to the estimated homophily ratio. Since UniFilter only utilizes the homophily basis on Cora ($\tau = 1$ for Cora as shown in Table 7) and does not rely on the homophily ratio, we thus test UniFilter on Pubmed and Squirrel instead. To this end, we vary the estimated homophily ratio $\hat{h} \in \{0.78, 0.79, 0.80, 0.81, 0.82\}$ on Pubmed and $\hat{h} \in \{0.19, 0.20, 0.21, 0.22, 0.23\}$ on Squirrel according to the estimation variances in Table 9. The achieved accuracy scores are presented in Table 10.

In Table 10, UniFilter demonstrates consistent accuracy scores across different homophily ratios for both Pubmed and Squirrel. Notably, the accuracy variations remain minor, staying within a 1% range from the scores under the true homophily

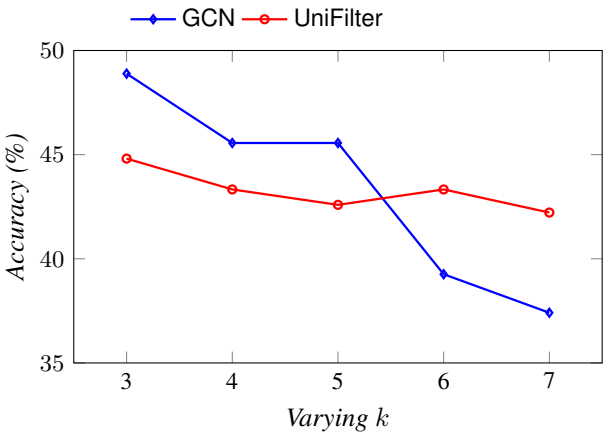


Figure 6: Accuracy (%) with varying k.

ratio, particularly on the Pubmed dataset.

Mitigation of Over-squashing. To validate that UniFilter is able to mitigate the over-squashing issue, we customize a synthetic graph for node classification. Specifically, we generate a binary tree of depth 7 empirically as the testing graph, denoted as $\mathbf{G}_B$. In particular, $\mathbf{G}_B$ consists of 127 nodes. Each node is assigned a random 100-dimension one-hot vector as a node feature and a random label from the label set $\{0, 1, 2\}$. We vary the propagation hop $k \in \{3, 4, 5, 6, 7\}$. As expected, the number of neighbors exponentially increases with the increment of k. In this ablation study, we compare UniFilter with GCN and tune their hyperparameters for the best possible performance on $\mathbf{G}_B$. The accuracy scores are plotted in Figure 6. As illustrated, the accuracy scores of GCN decrease along the increase of k, especially from $k = 5$ to $k = 6$. In contrast, UniFilter demonstrates consistency performance across the varying k. This observation aligns with the conclusion from Theorem 5.3 which posits that the Jacobian of the propagation matrix within UniFilter is independent of propagation step k.