G^{2}CN: Graph Gaussian Convolution Networks with Concentrated Graph Filters

Anonymous Authors[1]

Abstract

Recently, linear GCNs have shown competitive performance against non-linear ones with less computation cost, and the key lies in their propagation layers. Spectral analysis has been widely adopted in designing and analyzing existing graph propagations. Nevertheless, we notice that existing spectral analysis fails to explain why existing graph propagations with the same global tendency, such as low-pass or high-pass, still yield very different results. Motivated by this situation, we develop a new framework for spectral analysis in this paper called concentration analysis. In particular, we propose three attributes: concentration centre, maximum response, and bandwidth for our analysis. Through a dissection of the limitations of existing graph propagations via the above analysis, we propose a new kind of propagation layer, Graph Gaussian Convolution Networks (G^{2}CN), in which the three properties are decoupled and the whole structure becomes more flexible and applicable to different kinds of graphs. Extensive experiments show that we can obtain state-of-the-art performance on heterophily and homophily datasets with our proposed G^{2}CN.

1. Introduction

Graph Convolution Networks have achieved excellent performance on various graph-related tasks on social relationship (Li & Goldwasser, 2019; Qiu et al., 2018; Tong et al., 2019), traffic (Bogaerts et al., 2020; Cui et al., 2019; Li et al., 2018b), recommendation (Ying et al., 2018; Feng et al., 2022), medical researches and others (Zhao et al., 2019; Rathi et al., 2020; Jiang et al., 2021). GCNs mainly consist of the graph propagation part and feature transformation part. The graph propagation part first processes the graph signals to obtain the features with global or local graph properties. Then the signals are fed to a neural network with one or several layers. Several works (Wu et al., 2019; Zhu & Koniusz, 2020; Wang et al., 2021) have shown that even a linear transformation layer can achieve better performance on many benchmark datasets. Therefore, graph propagation is the key part of a GCN model and attracts a lot of attention these days. Among the researches on graph propagation, graph spectral analysis has been widely adopted in designing and analysing the existing graph propagation (Wu et al., 2019; Balcilar et al., 2021). It treats the graph propagations as graph filters on the spectrum of the graph Laplacian and categorize various graph propagation via the spectral response. For example, a graph filter that responds to amplify eigenvalues and diminish large eigenvalues will be categorized as a low-pass graph filter.

However, the former spectral analysis mainly focuses on the global tendency. These analyses cannot completely explain why GCNs with different graph propagation will show different results even if their corresponding graph filter share the same low-pass or high-pass tendency. Motivated by this, we analyze the filters' pass band since it determines the output of the graph propagation. Since the filters' responses are concentrated on the passband, we call our analysis as **spectral concentration analysis** to distinguish from the previous study. Specifically, we propose three attributes called the **concentration attributes** for spectral analysis on the graph propagation contrast to the filter's global tendency: maximum response $\mathcal{R}$, bandwidth $\mathcal{BW}$ and band concentration centre b.

We analyze existing graph propagation with these attributes and find that their different performance can be attributed to different concentration flexibility. Models with high concentration flexibility can apply different graph filters on various graphs with better performance. With these findings, we propose our Gaussian Graph Convolution (G^{2}C), whose graph filters can be assumed as the combination of Gaussian filters. Our G^{2}C decouples the above attributes and can be easily adapted to perform as different kinds of filters to suit the graph requirement. Then we implement our Gaussian graph filter in a linear graph model called Graph Gaussian Convolution Networks (G^{2}CN). The experiments show that our G^{2}CN can obtain state-of-art performance on homophily and heterophily datasets with efficient computation cost. We summarize the contributions of our work as below:

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. We extend the former spectral analysis to our concentration spectral analysis with our proposed concentration attributes. Compared with the former study, our proposed one does not only focuses on the graph filter's global tendency. And we can explain the different performances of various graph propagation with the same global trend.

2. We propose Gaussian Graph Convolution Network (G^2CN)and its graph propagation formulation from the above analysis. Our proposed filter enjoys sufficient concentration flexibility and can better apply to different graphs especially for the heterophilc ones.

3. We also conduct experiments to validate the analysis we proposed and to demonstrate the effectiveness of our proposed model.

2. Related Works

Graph Neural Networks can be generally categorized into spectral-based models (Bruna et al., 2014; Henaff et al., 2015; Li et al., 2018a), which use the eigendecomposition of the graph Laplacian, and spatial based model, which directly uses the graph Laplacian for graph propagation without the decomposition, which is widely used due to its efficiency. In this paper, we only discuss the latter propagation type. And we can divide the graph propagation for the current spatial-based GCNs from the spectral view into two kinds: GCNs with designed graph filters or learnable graph filters.

2.1. GCNs with Graph Propagation Constructed by Designed Graph Filters

As for the graph propagation part in Graph Convolution Networks, most models utilize the designed graph filter based on their priors on the graph information they need. For example, GCNs (Kipf & Welling, 2016) can be regarded as a low-pass filter (Balcilar et al., 2020; Wu et al., 2019; Xu et al., 2020). Furthermore, APPNP (Klicpera et al., 2019a) utilizes the Personalized PageRank and achieves a low-pass filter with different concentration properties compared to GCN as discussed in the following. Due to such limitations, these models show unstable performance on some real-world datasets. These models mainly focus on the local information on the graph and motivated many researchers to try to combine low-pass and high-pass filters (Zhu & Koniusz, 2020; Zhu et al., 2021) to balance the local and global information for graph data.

2.2. GCNs with the Graph Propagation Constructed by Learnable Graph Filters

Without considering the priors on the possible graphs filters, GCNs with learnable filters try to approximate any filter function for different graph datasets using the generalization ability of the neural network, such as GPR-GNN (Chien et al., 2020), ChebNet (Defferrard et al., 2016), BernNet (He et al., 2021) and others (Bianchi et al., 2021). These models use the combination of a series of polynomial bases they choose to approximate an arbitrary filter function, like Chebyshev polynomials and Bernstein polynomials. For this account, these models can obtain better results compared with traditional GCNs on many graph datasets, especially the heterophily ones.

3. On the Spectral Concentration Analysis of Graph Propagation

In this section, we first propose three concentration attributes: maximum response, concentration centre and bandwidth for spectral analysis on graph propagation. Our spectral analysis with these attributes discusses the weaknesses and advantages of different graph propagation in the following. Our analysis can explain why models like GCN, APPNP and Heat Diffusion's graph filters show the same spectral tendency will still show different performance on node classification tasks.

3.1. Notations and Preliminaries on Graph Propagation

For a graph defined as $\mathcal{G} = (\mathcal{V}, \mathbf{A})$, where $\mathcal{V} = \{v_1, ..., v_n\}$ denotes the vertex set of n nodes which contains a subset labeled nodes $\mathcal{V}_l \subset \mathbf{V}$ for training and unlabeled nodes $\mathcal{V}_u = \mathcal{V} \backslash \mathcal{V}_l$ to predict in the node classification task. $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the adjacent matrix where a_{ij} denotes the edge weight between node v_i and v_j. The degree matrix $\mathbf{D} = Diag(d_1, .., d_n)$ of $\mathbf{A}$ is a diagonal matrix with its $i-$th diagonal entry as $d_i = \sum_j a_{ij}$. Each node v_i is represented by a d-dimensional feature vector $\mathbf{x}_i \in \mathbb{R}^d$ and we use $\mathbf{X} \in \mathbb{R}^{n\times d} = [\mathbf{x}_1, .., \mathbf{x}_n]^\top$ to denote the feature matrix and use a one-hot vector $\mathbf{y}_i \in \{0, 1\}^C$ to denote the label of i-th node of C classes. In our paper, we only discuss the normalized graph Lapalacian matrix defined as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2} \in \mathcal{S}_+^n$ for convenience. The results can easily be extended to other types of Laplacian. Its eigendecomposition can be depicted as $\mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$, where $\mathbf{\Lambda}$ is the diagonal matrix consists of $\mathbf{L}$'s eigenvalues, and $\mathbf{U} \in \mathbb{R}^{n\times n}$ is made by the eigenvectors of the Laplacian matrix.

Then we define the graph Fourier transform of a graph singal $\mathbf{x}$ as $\hat{\mathbf{x}} = \mathbf{U}^\top\mathbf{x}$ and its inverse is $\mathbf{x} = \mathbf{U}\hat{\mathbf{x}}$. Thus the graph propagation for signal $\mathbf{x}$ with kernel $\mathbf{g}$ can be defined as:

$$\mathbf{H} = \mathbf{g} *_{\mathcal{G}} \mathbf{x} = \mathbf{U}\left((\mathbf{U}^\top\mathbf{g}) \odot \mathbf{U}\mathbf{x}\right) = \mathbf{U}\hat{\mathbf{G}}\mathbf{U}^\top\mathbf{X}, \quad (1)$$

where $\hat{\mathbf{G}} = diag(\hat{g}_1, ..., \hat{g}_2)$ denotes the spectral kernel coefficients. Exactly computing the graph convolution on a random kernel $\mathbf{g}$ is complicated due to the eigendecomposition for graph's Laplacian consumes unaffordable computation complexity. For this count, as we illustrated in the following, current works with spatial convolution usually use the polynomial functions $g(\mathbf{L})$ to approximate different kernels:

$$\mathbf{H} = g(\mathbf{L})\mathbf{x} = \mathbf{U}g(\mathbf{\Lambda})\mathbf{U}^{\top}\mathbf{X}, \tag{2}$$

where $g(\mathbf{\Lambda}) = Diag\left(g(\lambda_1), ..., g(\lambda_n)\right)$ and we call g the **graph filter** function for above graph propagation Eqn. (2). Then GCNs with such graph propagation can be considered as filtering the graph signals by the corresponding eigenvalues for Laplacian. Those g preserving small λ can be regarded as low-pass filters while those keeping large λ are high-pass filters. Apart from the graph filters global tendency, we also analyze the concentrated properties via the concentration attributes defined as follows.

3.2. Concentration Attributes on Graph Propagation

First, we would like to define the graph filter's maximum response and the concentration centre:

Definition 3.1. The maximum response $\mathcal{R}_g$ for graph filter g is denoted as:

$$\mathcal{R}_g = \max_{\lambda \in [0,2]} |g(\lambda)|, \tag{3}$$

since the eigenvalues of the graph Laplacian $\mathbf{L}$ lies in $[0, 2]$.

And the concentration centre can be defined as:

Definition 3.2. Then we define b as the **centre for graph filter** g if,

$$g(b) = \mathcal{R}_g, \tag{4}$$

Then we give the definition for the graph filter's bandwidth.

Definition 3.3. The bandwidth $\mathcal{BW}_g$ for a given graph filter $g(\lambda)$ can be defined as:

$$\mathcal{BW}_g = \int_0^2 \mathbb{I}\left(g(\lambda), \frac{\mathcal{R}_g}{\sqrt{2}}\right) d\lambda,$$

where $\mathbb{I}(g(\lambda), \frac{\mathcal{R}_g}{\sqrt{2}})$ is a indicator function:

$$\mathbb{I}(\gamma_1, \gamma_2) = \begin{cases} 1, & if\ \gamma_1 \geq \gamma_2 \\ 0, & if\ \gamma_1 \leq \gamma_2 \end{cases},$$

for $\gamma_1, \gamma_2 \in \mathbb{R}$.

The above attributes tell which spectrum the graph filter will amplify or diminish. And different graph filters enjoy unique concentration attributes and lead to different performances, as we have discussed. The bandwidth and concentration centre should be flexible to ensure the proper graph propagation for a different dataset, while the maximum response should be bounded for stable training. Then we try to dissect existing graph propagations via our spectral concentration analysis.

3.3. Dissecting Existing Deisgned Graph Propagations

Graph Convolutional Networks (GCN). Firstly, we dissect GCN (Kipf & Welling, 2016) via our concentration spectral analysis. The graph propagation for GCN can be formulated as:

$$\mathbf{H}_{gcn} = (\mathbf{2I} - \mathbf{L})^K \mathbf{X} \tag{5}$$

where $K \in \mathbb{N}^+$ is the propagation steps for GCN and its latter works (Wu et al., 2019; Sun et al., 2021; Xu et al., 2019b). Its graph filter can be formulated as $g_{gcn}(\lambda) = (2-\lambda)^K$, which is also a low-pass filter, whose concentration centre is 0 with $\mathcal{R}_{gcn} = 2^K$. And its bandwidth can be formulated as:

$$\mathcal{BW}_{gcn} = 2 - 2^{1-\frac{1}{2K}}. \tag{6}$$

One can see that the bandwidth of GCN is determined by its propagation steps. Therefore, the bandwidth for GCN's graph filter is limited and such defects restrict its performance since it cannot flexibly implement low-pass filters with different bandwidths as the graph desires.

Personalized PageRank (PPR). Then Klicpera et al. (2019a) proposed the personalized PageRank (PPR) graph propagation, trying to make the graph filter more flexible. Its graph propagation tries to approximate:

$$\mathbf{H}_{ppr} = \left[\mathbf{I} - (1-\alpha)(\mathbf{I} - \mathbf{L})\right]^{-1} \mathbf{X}, \tag{7}$$

with $\alpha \in (0, 1)$ to ensure the inverse exists. Its graph filter is $g_{ppr}(\lambda) = \frac{1}{1-(1-\alpha)(1-\lambda)}$ and its concentration centre is fixed at $\lambda = 0$ with $\mathcal{R}_{ppr} = \frac{1}{\alpha}$ thus also a low pass filter. The bandwidth for PPR can be formulated as:

$$\mathcal{BW}_{ppr} = \frac{(\sqrt{2}-1)\alpha}{1-\alpha}. \tag{8}$$

Since the bandwidth for PPR can continuously change with different α, models with PPR propagation are more flexible than original GCN and enjoy better performance. When $\mathbf{BW}_{ppr} \to 0$, $\alpha \to 0$, its maximum response will explode: $\mathcal{R}_{ppr} = \frac{1}{\alpha} \to \infty$. Therefore, we cannot choose a narrow PPR filter in practice, although the bandwidth for PPR can vary from 0 to ∞ in theory.

Heat Kernel (HK). Apart from the above models, another widespread designed graph propagation is the heat kernel propagation (Klicpera et al., 2019b; Wang et al., 2021) which is inspired by the heat diffusion. Such propagation tries to approximate the following graph propagation:

$$\mathbf{H}_{HK} = e^{-T\mathbf{L}}\mathbf{X}, \tag{9}$$

where $T \in \mathbb{R}^+$ is the diffusion factor and its graph filter is an exponential function $g_{HK}(\lambda) = e^{-T\lambda}$. Its maximum response is fixed to be 1 with fixed centre $\lambda = 0$. And the bandwidth for heat kernel is:

$$\mathcal{BW}_{HK} = \frac{log\sqrt{2}}{T}. \tag{10}$$

The above formula shows that the bandwidth of DGC is flexible and can be easily controlled by T, and the fixed maximum response can also stabilize the training procedure. Thereby, as illustrated in Klicpera et al. (2019b) and Wang et al. (2021), heat kernels can show better performance against the above graph propagations on most datasets. However, the unflexible concentration centre for the above models also hinders their performance on many datasets since these low-pass models will permanently cut off the graph information corresponding to large eigenvalues of the graph Laplacian.

3.4. Dissecting Graph Propagation with Learnable Graph Propagations

To approximate the graph filters with different concentration centres, many graph models combine a series of polynomial bases with learnable weights as their graph propagation. Although these models can theoretically approximate every filter with proper weight for each basis, different properties for the basis will influence the training difficulties of bases' balancing weights and lead to different performances. Thereby, this section discusses the concentration attributes for a single filter basis.

Chebshev Polynoimals. ChebNet (Defferrard et al., 2016) which utilizes the Chebyshev Polynomials to approximate various graph filters, whose i-th basis can be formulated as:

$$\mathbf{C}^{(i)} = 2\mathbf{C}^{(2)}\mathbf{C}^{(i-1)} - \mathbf{C}^{i-2}, \tag{11}$$

for $i > 2$ with $\mathbf{C}^{(1)} = \mathbf{I}$ and $\mathbf{C}^{(2)} = \frac{2\mathbf{L}}{\lambda_{\max}} - \mathbf{I}$ and its propagation for i-th basis is $\mathbf{H}^{i}_{Cheb} = \mathbf{C}^{(i)}\mathbf{X}$. Its concentration attributes are determined by its index i and the details of these attributes are calculated in Appendix E. Since i-th basis needs to multiply the graph Laplacian and input $\mathbf{X}$ for $i-1$ times, ChebNet's attributes is unflexible with fixed computation cost. Moreover, the bases for ChebNet have more than one concentration centre lies in $[0, 2]$ and relies on the complicated weights to approximate the usually used filters with few peaks.

Bernstein Polynoimals. BernNet (He et al., 2021) utilize the summation of Bernstein bases with balancing weights to approximate different graph filters, the graph propagation for i-th basis for an order-K BernNet graph propagation can be formulated as follows:

$$\mathbf{H}^{K,i}_{Bern} = \frac{1}{2^K}\binom{K}{k}(2\mathbf{I} - \mathbf{L})^{K-k}\mathbf{L}^k, \tag{12}$$

whose corresponding graph filter can be simplified as $(2-\lambda)^{K-i}\lambda^i$. Its concentration centre with order K is $\frac{2i}{K}$. And each Berstein filter only has one concentration centre, which makes the weights' training easier and leads to better performance compared with ChebNet.

The concentration centre and bandwidth (listed in Appendix C) for each Bernstein basis can only be changed by setting different K. Like ChebNet, using larger K will consumes a lot computation cost, since BernNet needs to compute the multiplication of graph Laplacian and input $\mathbf{X}$ for $O\left(K^2\right)$ times with order-K basis. For this account, BernNet still needs multi-layer transformation part to process each basis output since it cannot freely change its bandwidth and concentration center for different graphs without increasing computation complexity. Thereby, BernNet is not efficient and cannot easily adapt to different graphs. We summarize our analysis in this section as a Table in Appendix E.

4. The Proposed G^{2}CN Models.

4.1. Gaussian Graph Filters

From the above analysis, one can see that models with better flexibility like Heat Kernel propagation and BernNet perform better than the others since they can easily approximate different filters adapted to various graphs. However, as we state above, these models are not flexible enough as the concentration centre and bandwidth for Bernstein basis is unchangable and Heat Kernel based propagation always performs as a low-pass filter. Therefore, we would like to construct a series of bases with better concentration flexibility and combine them to approximate different filters for graphs.

A proper graph filter should have flexible concentration attributes, which means that its concentration centre and bandwidth can change smoothly and its maximum response is bounded, then it can easily adapt to various graphs. To build such a graph filter, we use the a series of Gaussian bases as our graph filter. We propose our Gaussian Graph Convolution inspired from the close form for graph heat diffusion. The graph propagation for a single Gaussian basis concentrated at b can be formulated as follows:

$$\mathbf{H}_G = e^{-T(\mathbf{L}-b\mathbf{I})^2}\mathbf{X}, \tag{13}$$

with $b \in [0, 2]$ and its maximum response 1. Then for its corresponding Gaussian filter, we can obtain is bandwidth as follows:

Proposition 4.1. *For the Gaussian filter denoted as:*

$$g_{G_{T,b}}(\lambda) = e^{-T(\lambda-b)^2},$$

with $b \in [0, 2]$ and T is the hyper-parameter related the bandwidth. Its bandwidth is:

$$\mathcal{BW}_{g_{T,b}} = \left\{ \begin{array}{cc} b, & b < \sqrt{\frac{log\sqrt{2}}{T}} \\ 2\sqrt{\frac{log\sqrt{2}}{T}}, & \sqrt{\frac{log\sqrt{2}}{T}} \le b \le 2 - \sqrt{\frac{log\sqrt{2}}{T}} \\ 2-b, & b > 2 - \sqrt{\frac{log\sqrt{2}}{T}} \end{array} \right\}$$

The proposition can be quickly proved by showing the $g_{G_{T,b}}(\lambda) \leq \frac{1}{\sqrt{2}}$ with $\lambda \in [0, 2]$. From the above results, we can demonstrate the advantages for the Gaussian filter. Like heat kernel filters, Gaussian filters decouple the three concentration attributes. For this account, we can flexibly choose different Gaussian filters and combine them to achieve our graph filter with the desired pass-band. We can choose T for different graph propagation to different approximated filters with various bandwidths and choose b to change the concentration centre smoothly. Empirical results show that such flexibility enables our G^{2}CN to simulate different graph filters and show satisfactory results on real-world datasets. As the following lemma shows, all square-integrable functions can approximate f with a series of Gaussian kernels.

Lemma 4.2. *(Calcaterra & Boldt, 2008) For any $f \in L^2(\mathbb{R})$ and any $\epsilon > 0$ there exists $T > 0$ and $b > 0$ and $N \in \mathbb{N}$ and $\theta_i \in \mathbb{R}$ such that:*

$$\left\| f(x) - \sum_{n=0}^{N} \theta_n e^{-T(x-nb)^2} \right\| \leq \epsilon.$$

We can obtain the above conclusion based on the findings proposed in (Calcaterra & Boldt, 2008). Thereby, we can approximate the most practical filters with proper series Gaussian filter. The above results demonstrates the expressive power of our proposed Gaussian Graph Convolution.

4.2. Formulation of the Gaussian Filter and Our G^{2}CN

To achieve the Gaussian filter in the graph propagation, we propose two formulations for the approximated graph propagation as listed below:

G^{2}CN-Taylor Firstly, we can obtain dissecting the Gaussian filter as graph propagation by K-order Taylor expansion:

$$\mathbf{H} = g_{G_{T,b}}(\mathbf{L})\mathbf{X} \approx \sum_{k=0}^{K} \frac{T^k}{k!}(\mathbf{L} - b\mathbf{I})^{2k}\mathbf{X}, \qquad (14)$$

where K here denotes the propagation times, which only influences the accuracy of the approximation.

G^{2}CN-Euler Since the equivalent graph filter $g_{G_{T,b}}$ can also be regarded as the close form solution for the following differential system at $t = 1$:

$$\frac{\partial \mathbf{X}(t)}{\partial t} = -T(\mathbf{L} - b\mathbf{I})^2\mathbf{X}(t), \qquad (15)$$

with $\mathbf{X}(0) = \mathbf{X}$, we can use the forward Euler method to solve the graph propagation with the step size equal to $1/K$.

$$\mathbf{X}\left(\frac{i}{K}\right) = \mathbf{X}\left(\frac{i-1}{K}\right) - \frac{T}{K}(\mathbf{L} - b\mathbf{I})^2\mathbf{X}\left(\frac{i-1}{K}\right),$$

with $i = 0, ..., K$ to obtain the final state $\mathbf{H} \approx \mathbf{X}(1)$. We only need to store the state i/K after the i-th iteration with such a scheme.

Using the above methods, we can obtain the equivalent graph propagation for the graph filter concentrated at b. Since one Gaussian filter can only respond to the eigenvalues around the basis, we can combine features obtained by Gaussian filters with different concentration centre as follows:

$$\mathbf{H} = \sum_{i=0}^{N} \theta_i g_{G_{T_i,b_i}}(\mathbf{L})\mathbf{X}, \qquad (16)$$

where N denotes the bases number we use, and θ_i are learnable weights balancing Gaussian filters with different concentration centres and bandwidth.

The Whole Structure of G^{2}CN. After combining our Gaussian Graph Convolution, we can obtain the graph information filtered by the graph spectral filter we desired. As shown in (He et al., 2021; Li et al., 2021), the node classification mainly needs graph information centered on the high-frequency or low-frequency part in practice, so we only adopt two Gaussian filters for our G^{2}CN with fixed centre $b = 0$ and $b = 2$ and different T for the following experiments. Since the high flexibility makes the weights training much easier, we only use a linear layer for classification after the propagation for computational efficiency. Then we finally get our G^{2}CN.

4.3. Numerical Analysis on G^{2}CN

In this section, we compare two type of G^{2}CN from the perspective of their approximation error from their closed-form solution, which can be defined as:

$$e_{\mathcal{M}}^{T,K,b} = \|\mathbf{H}_{\mathcal{M}} - e^{-T(b\mathbf{I}-\mathbf{L})^2}\mathbf{X}\|_2, \qquad (17)$$

with $\mathcal{M}$ can be chosen as "Eu" or "Ta" to denote G^{2}CN-Euler and G^{2}CN-Taylor and $\|\cdot\|$ denotes the operator norm. Then the numerical error for the Taylor method is:

Theorem 4.3. *For a single G^{2}CN-Taylor's graph filter with order K, width factor T and bias b, the approximation error $e_{Ta}^{T,K,b}$ compared with its closed-from $e^{-T(b\mathbf{I}-\mathbf{L})^2}\mathbf{X}$ is:*

$$e_{Ta}^{T,K,b} \leq \frac{(T\|b\mathbf{I} - \mathbf{L}\|_2^2)^{K+1}}{(K+1)!}\|\mathbf{X}\|_2. \qquad (18)$$

And the numerical error for the Euler method is:

Theorem 4.4. *For a single G^{2}CN-Euler's graph filter with order K, width factor T and bias b, if $2T < K$ the approximation error $e_{T}^{T,K,b}$ compared with its closed-from $e^{-T(b\mathbf{I}-\mathbf{L})^2}\mathbf{X}$ can be bounded by:*

$$e_{Eu}^{T,K,b} \leq \frac{T^2\|b\mathbf{I} - \mathbf{L}\|_2^4}{K}\|\mathbf{X}\|_2. \qquad (19)$$

Table 1. Datasets statistics

Dataset	Cora	Citeseer	Pubmed	Computers	Photo	Chameleon	Squirrel	Actor	Texas	Cornell	OGB-Arxiv
#Nodes	2708	3327	19717	13752	7650	2277	5201	7600	183	183	169343
#Edges	5278	4552	44324	245861	119081	36101	217073	26659	295	309	1166243
#Features	1433	3703	500	767	745	2325	2089	932	1703	1703	128
#Classes	7	6	3	10	8	5	5	5	5	5	40
$\mathcal{H}$	0.83	0.71	0.79	0.80	0.85	0.25	0.22	0.24	0.06	0.11	-

From the above results, one can see that the increase of K can reduce the numerical error of the Euler method. However, when $K < T\|b\mathbf{I} - \mathbf{L}\|_2^2$ the error of Taylor expansion will increase first as K increases and then decrease when K is larger enough and will return a smaller error than the Euler method when K is large enough. Therefore, we recommend G^{2}CN Euler when K is small due to the limited computation complexity. And as the results demonstrate, we only recommend the G^{2}CN-Taylor when K is significantly large compared with T, then G^{2}CN-Taylor will return a better approximation in this scenario.

5. Experiments

5.1. Homophily and Heterophily Datasets

The graph node classification datasets can be divided into homophily graphs and heterophily graphs with the homophily principle (McPherson et al., 2001). Homophily principle states that contact between similar nodes occurs at a higher rate than among dis-similar nodes. In recent work, Pei et al. (2020) designed an index denoted by $\mathcal{H}$ to measure the homophily in a graph:

$$\mathcal{H}(G) = \frac{1}{|V|}\sum_{v\in V}\frac{\text{Number of neighbors of } v \in V \text{ that have the same label as } v}{\text{Number of neighbors of } v}.$$

Lower $\mathcal{H}$ indicates a stronger heterophilic graph, where nodes with distinct labels are more likely to link together. As Wu et al. (2021) stated, the homophilic graphs need low-pass filters for a satisfactory performance while heterophilic graphs need other passbands in graph filters for generalization.

In our work, we conduct experiments on five widely used benchmark homophylic graphs as others (Wang et al., 2021) to demonstrate our Gaussian filter achieve the comparable state-of-the-art performance compared with other low-pass graph filters, including the standard citation graphs, Cora, CiteSeer and PubMed (Sen et al., 2008; Yang et al., 2016) and the Amazon co-purchase graphs, Computers and Photo (McAuley et al., 2015; Shchur et al., 2018).

Then we finish the experiments for five benchmark heterophilic graphs used in Pei et al. (2020), including Wikipedia graphs, Chameleon and Squirrel (Rozemberczki et al., 2021), the Actor co-occurrence graph and the WebKB graphs, Texas and Cornell (Pei et al., 2020). Experiments on heteophilc graphs demonstrate the superiority of our G^{2}CN models compared with other linear or non-linear graph models due to the flexibility of the Gaussian Graph Filters we use. More rigorously, we also conduct the comparison on a large scale node classification dataset, OGB-Arxiv (Hu et al., 2021). An overview of characteristics of the datasets is given in Table 1.

5.2. Experimental Settings

Firstly, we experiment on the semi-supervised node classification task for standard citation networks with the standard data split as in other works (Wang et al., 2021; Kipf & Welling, 2016; Veličković et al., 2017; Xu et al., 2020; 2019a; Poli et al., 2019) with the same setting as DGC. Furthermore, we also evaluate the performance of the homophilic Amazon co-purchase graph, Computers and Photo, for fully supervised node classification tasks with the same setting as BernNet, which randomly splits the node sets into train/validation/test set with ratio $60\%/20\%/20\%$.

Then for the heterophilic graphs, we use the same setting as BernNet and finish the experiments on the fully-supervised node classification tasks for these graphs. We compare G^{2}CN with three baseline models: GCN (Kipf & Welling, 2016) , GAT (Veličković et al., 2017), APPNP (Klicpera et al., 2019a), BernNet (He et al., 2021), SGC (Wu et al., 2019), DGC (Wang et al., 2021). We tune the hyper-parameters for non-linear models follows the same steps as He et al. (2021) stated and the channel numbers for hidden layers in non-linear models are 64. As for linear GCNs, we perform a hyper-parameter search for linear models' hyper-parameter (T and weight decay) on the validation set as used in DGC for 1000 trials. We choose $K = 200$ for DGC and $K = 100$ for our G^{2}CN to ensure the same propagation steps with graph Laplacian for fairness.

We use Adam optimizer (Kingma & Ba, 2014) for training with 100 epochs for semi-supervised tasks and 1000 epochs for fully-supervised tasks as BernNet for fair with learning rate 0.5 for our model and report results averaged over ten random runs. And we use G^{2}CN-Euler with $2T < K$ in the

Table 2. Test accuracy (%) comparison on heterophily datasets. Reported results are averaged over 10 runs. The former for models are designed with low-pass filters.

Dataset	GCN	GAT	APPNP	DGC	BernNet	G^{2}CN
Chameleon	42.98 ± 1.41	47.62 ± 0.83	43.07 ± 1.55	67.08 ± 1.58	54.11 ± 1.73	$\mathbf{73.61 \pm 0.89}$
Squirrel	28.41 ± 0.57	27.33 ± 0.81	31.71 ± 0.47	50.51 ± 0.81	41.35 ± 0.81	$\mathbf{66.91 \pm 1.13}$
Actor	33.23 ± 1.16	33.93 ± 2.47	39.66 ± 0.55	40.35 ± 0.73	$\mathbf{41.79 \pm 1.01}$	41.44 ± 0.76
Texas	77.38 ± 3.28	80.82 ± 2.13	90.98 ± 1.64	94.74 ± 0.33	96.22 ± 0.79	$\mathbf{96.72 \pm 0.73}$
Cornell	65.90 ± 4.43	78.21 ± 2.95	91.81 ± 1.96	92.45 ± 1.21	92.29 ± 2.74	$\mathbf{94.11 \pm 1.81}$

following since its performance is more stable when T is large.

5.3. Experimental Results on Homophily Datasets

The test results for homophily datasets are shown in Table 3. The results show that our G^{2}CN can show comparable performance on the semi-supervised citation graphs. Furthermore, one can see that our G^{2}CN enjoys an evident advantage on the fully-supervised graph tasks for large graphs. The empirical results demonstrate that our G^{2}CN can return better graph signals output by our graph filters on homophilic graphs, which mainly depend on the graph's low-frequent composition.

Table 3. Test accuracy (%) comparison on common used citation graphs with semi-supervised scheme (above) and Amazon co-purchase graphs with fully-supervised scheme (blow). The results are averaged over 10 runs.

Type	**Dataset**	Cora	CiteSeer	Pubmed
Nonlinear	GCN	81.5	70.3	79.0
	GAT	83.0	72.5	79.0
	APPNP	83.3	71.8	80.1
Linear	SGC	81.1	71.9	78.9
	DGC	**83.3**	73.3	80.3
	G^{2}CN	82.7	**73.8**	**80.4**

Type	**Dataset**	Computers	Photo
Nonlinear	GCN	83.32 ± 0.33	88.26 ± 0.73
	GAT	83.32 ± 0.39	90.94 ± 0.68
	APPNP	85.32 ± 0.32	88.51 ± 0.31
	BernNet	87.64 ± 0.44	94.50 ± 0.40
Linear	SGC	84.74 ± 0.27	86.48 ± 0.41
	DGC	89.54 ± 0.64	94.89 ± 0.50
	G^{2}CN	$\mathbf{91.46 \pm 0.35}$	$\mathbf{95.29 \pm 0.48}$

We notice that BernNet performs worse than our G^{2}CN on the homophilic graphs. The proper passband for these graphs shown in Table 4 can explain such a phenomenon.

The table shows that the bandwidth for each Bernstein basis in BernNet is not flexible. For example, the pass bands for last three filters of BernNet ($K = 10$) are $[1.3, 1.92]$, $[1.61, 1.92]$, $[1.93, 2.0]$. For this account, BernNet needs to rely on complicated balancing weights for each basis to approximate the narrow high-pass filter stated in G^{2}CN and it will make the learning task much harder. Thereby, it cannot utilize proper high-frequency graph information as ours which leads to worse results than ours.

Table 4. Bandwidth for DGC and G^{2}CN (Bandwidth for two Gaussian filters with $b = 0, 2$) on Computers and Photo.

Datasets	DGC	G^{2}CN
Computers	0.21	(0.35, 0.15)
Photo	0.23	(0.37, 0.12)

5.4. Experimental Results on Heterophily Datasets

Apart from the homophilic graphs, we also conduct experiments on the heterophilic graphs. The results are shown in Table 2. Compared with the homophilic graphs, the advantages for our G^{2}CN over the low-pass models are more evident since the heterophilic graphs need the graph signals with different frequency for their tasks. Compared with BernNet, our G^{2}CN can still show better performance on these models except Actor. The above results can also demonstrate the superiority of our proposed model.

Understandings via Our Concentration Analysis From the experimental results, one can see that our G^{2}CN shows significant performance on Chameleon and Squirrel, while on Cornell, Actor and Texas, the difference is not significant compared with low-pass graph models. To explain such a phenomenon, we calculate the bandwidth of DGC and G^{2}CN in Table 5.

Table 5. Bandwidth for DGC and G^{2}CN (Bandwidth for two Gaussian filters with $b = 0, 2$) for heterophilic graphs.

Datasets	DGC	G^{2}CN
Texas	2	(1.13, 0.16)
Actor	2	(2.0, 0.27)
Cornell	2	(2.0, 0.08)
Squirrel	0.09	(0.12, 0.37)
Chameleon	0.07	(0.11, 0.27)

The table shows that the Texas, Actor and Cornell mainly need to apply an all-pass filter. For this account, APPNP and DGC can also perform comparable results with our G^{2}CN. GCN and GAT cannot perform well on these dataset because their bandwidth are fixed while APPNP and DGC's can easily change. As for Chameleon and Squirrel datasets, the best bandwidth adapted for DGC and G^{2}CN demonstrate that they need a comb graph filter with narrow pass-bands. Therefore, graph models with low-pass filters cannot perform well since they lack high-frequent graph signals. As for BernNet, the proper passband is not suitable for its basis. Thus approximating the proper passband like ours is much harder and leads to worse results. Empirical results demonstrate the effectiveness of our G^{2}CN and the rationality of our concentration spectral analysis since we can use it to understand different models' performance on various datasets.

5.5. Performance on Large Scale Datasets

Apart from the above datasets, we also finish experiments for our G^{2}CN on OGB-Arxiv. Since the OGB-Arxiv dataset is large and hard, we replace the linear layer with 2-layer MLP after the output of our Gaussian graph filters. The results are listed in Table 6. The results on the large graph OGB-Arxiv demonstrate the effectiveness of our G^{2}CN.

Table 6. Test accuracy (%) comparison on a large scale dataset OGB-Arxiv. OOM: out of memory.

Method	Accuracy
GCN	OOM
BernNet	OOM
SGC	68.9
DGC+MLP	71.2
G^{2}CN+MLP	**71.6**

5.6. Computation Time for Different Models

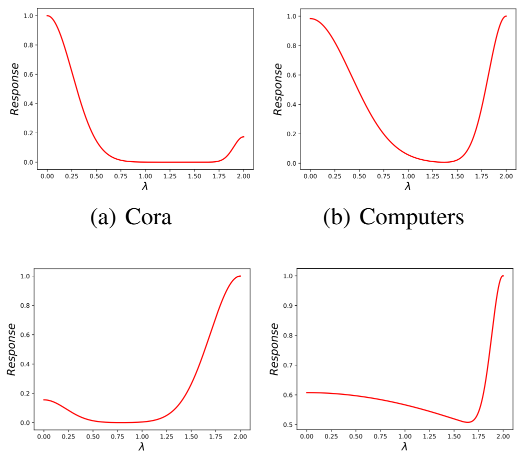
We list the computation time for different models trained on large graphs for 1000 epochs in Table 7. The models are evaluated on a single RTX-3070 and Intel I5-10400 (2.90GHz). As linear GCNs, DGC and G^{2}CN are much faster than non-linear GCNs like GCN and BernNet. Our G^{2}CN is only slightly slower compared with DGC. This is because our G^{2}CN needs an additional weighted summation for the outputs of two filters.

5.7. Ablation Studies on Spectral Response Curve For Selected Dataset

In this section, we draw the approximation spectral response curve of our G^{2}CN's Gaussian graph filter for selected datasets in Figure 1. The curve depicts whether our Gaussian graph filter amplifies or diminishes the signals corresponding to eigenvalue λ. From the above figures, one can see Cora needs a low-pass filter while Computers and Squirrel need comb filters with narrow bands. Moreover, Squirrel mostly relies on its graph signals with high frequency for classification which is the same as Chameleon listed in the Appendix. Since our G^{2}CN can easily approximate such a filter, it outperforms other methods with significant advantages. The proper filter for Cornell is more likely to be an all-pass filter. Other curves are shown in Appendix F.

Table 7. Comparison of total training time for different models trained for 1000 epochs. The number in the brackets are preprocessing times and those outside the brackets are total times.

Model	Pubmed	Computers	Photo
BernNet	59.33s(-)	107.1s(-)	53.33s(-)
DGC	1.04s(19ms)	2.01s(20ms)	1.40s(19ms)
G^{2}CN	1.24s(18ms)	2.27s(21ms)	1.56s(19ms)



(a) Cora (b) Computers

(c) Squirrel (d) Cornell

Figure 1. G^{2}CN's equivalent graph filters for selected real-world datasets.

6. Conclusions

In this work, we first propose three attributes called concentration attributes to analyze the spectral properties for different graph models. So our theory is called the concentration spectral analysis. Our analysis can explain why various graph models will show different results even if their global tendency is the same. Furthermore, we analyze the graph filters' weaknesses and advantages for different models. We notice that these models' concentration attributes are not flexible, making them hard to apply proper graph filters for different graphs. We then propose our Gaussian graph filters and our G^{2}CN using this filter. The concentration attributes for our G^{2}CN is much more flexible and can easily apply proper filters for different graphs. Empirical results also demonstrate the superiority of our model.

References

Balcilar, M., Renton, G., Héroux, P., Gaüzère, B., Adam, S., and Honeine, P. Analyzing the expressive power of graph neural networks in a spectral perspective. In *International Conference on Learning Representations*, 2020.

Balcilar, M., Renton, G., Héroux, P., Gaüzère, B., Adam, S., and Honeine, P. Analyzing the expressive power of graph neural networks in a spectral perspective. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=-qh0M9XWxnv.

Bianchi, F. M., Grattarola, D., Livi, L., and Alippi, C. Graph neural networks with convolutional arma filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

Bogaerts, T., Masegosa, A. D., Angarita-Zapata, J. S., Onieva, E., and Hellinckx, P. A graph cnn-lstm neural network for short and long-term traffic forecasting based on trajectory data. *Transportation Research Part C: Emerging Technologies*, 112:62–77, 2020.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. Spectral networks and locally connected networks on graphs, 2014.

Calcaterra, C. and Boldt, A. Approximating with gaussians, 2008.

Chien, E., Peng, J., Li, P., and Milenkovic, O. Adaptive universal generalized pagerank graph neural network. *arXiv preprint arXiv:2006.07988*, 2020.

Cui, Z., Henrickson, K., Ke, R., Pu, Z., and Wang, Y. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting, 2019.

Defferrard, M., Bresson, X., and Vandergheynst, P. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29:3844–3852, 2016.

Feng, L., Cai, Y., Wei, E., and Li, J. Graph neural networks with global noise filtering for session-based recommendation. *Neurocomputing*, 472:113–123, 2022. doi: 10.1016/j.neucom.2021.11.068. URL https://doi.org/10.1016/j.neucom.2021.11.068.

He, M., Wei, Z., Huang, Z., and Xu, H. Bernnet: Learning arbitrary graph spectral filters via bernstein approximation. *arXiv preprint arXiv:2106.10994*, 2021.

Henaff, M., Bruna, J., and LeCun, Y. Deep convolutional networks on graph-structured data, 2015.

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, 2021.

Jiang, D., Wu, Z., Hsieh, C.-Y., Chen, G., Liao, B., Wang, Z., Shen, C., Cao, D., Wu, J., and Hou, T. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13 (1):1–23, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

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

Klicpera, J., Weißenberger, S., and Günnemann, S. Diffusion improves graph learning. *Advances in Neural Information Processing Systems*, 32:13354–13366, 2019b.

Li, C. and Goldwasser, D. Encoding social information with graph convolutional networks forpolitical perspective detection in news media. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2594–2604, 2019.

Li, R., Wang, S., Zhu, F., and Huang, J. Adaptive graph convolutional neural networks, 2018a.

Li, S., Kim, D., and Wang, Q. Beyond low-pass filters: Adaptive feature propagation on graphs. *Lecture Notes in Computer Science*, pp. 450–465, 2021. ISSN 1611-3349. doi: 10.1007/978-3-030-86520-7_28. URL http://dx.doi.org/10.1007/978-3-030-86520-7_28.

Li, Y., Yu, R., Shahabi, C., and Liu, Y. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, 2018b.

McAuley, J., Targett, C., Shi, Q., and Van Den Hengel, A. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pp. 43–52, 2015.

McPherson, M., Smith-Lovin, L., and Cook, J. M. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1):415–444, 2001.

Pei, H., Wei, B., Chang, K. C.-C., Lei, Y., and Yang, B. Geom-gcn: Geometric graph convolutional networks. *arXiv preprint arXiv:2002.05287*, 2020.

Poli, M., Massaroli, S., Park, J., Yamashita, A., Asama, H., and Park, J. Graph neural ordinary differential equations. *arXiv preprint arXiv:1911.07532*, 2019.

Qiu, J., Tang, J., Ma, H., Dong, Y., Wang, K., and Tang, J. Deepinf: Social influence prediction with deep learning. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2110–2119, 2018.

Rathi, P. C., Ludlow, R. F., and Verdonk, M. L. Practical high-quality electrostatic potential surfaces for drug discovery using a graph-convolutional deep neural network. *Journal of Medicinal Chemistry*, 63 (16):8778–8790, 2020. doi: 10.1021/acs.jmedchem.9b01129. URL https://doi.org/10.1021/acs.jmedchem.9b01129. PMID: 31553186.

Rozemberczki, B., Allen, C., and Sarkar, R. Multi-scale attributed node embedding. *Journal of Complex Networks*, 9(2):cnab014, 2021.

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.

Shchur, O., Mumme, M., Bojchevski, A., and Günnemann, S. Pitfalls of graph neural network evaluation. *arXiv preprint arXiv:1811.05868*, 2018.

Sun, K., Zhu, Z., and Lin, Z. Adagcn: Adaboosting graph convolutional networks into deep models, 2021.

Tong, P., Zhang, Q., and Yao, J. Leveraging domain context for question answering over knowledge graph. *Data Science and Engineering*, 4(4):323–335, 2019.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Wang, Y., Wang, Y., Yang, J., and Lin, Z. Dissecting the diffusion process in linear graph convolutional networks, 2021.

Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., and Weinberger, K. Simplifying graph convolutional networks. In *International conference on machine learning*, pp. 6861–6871. PMLR, 2019.

Wu, Z., Pan, S., Long, G., Jiang, J., and Zhang, C. Beyond low-pass filtering: Graph convolutional networks with automatic filtering. *CoRR*, abs/2107.04755, 2021. URL https://arxiv.org/abs/2107.04755.

Xu, B., Shen, H., Cao, Q., Qiu, Y., and Cheng, X. Graph wavelet neural network. *arXiv preprint arXiv:1904.07785*, 2019a.

Xu, B., Shen, H., Cao, Q., Cen, K., and Cheng, X. Graph convolutional networks using heat kernel for semi-supervised learning. *arXiv preprint arXiv:2007.16002*, 2020.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. How powerful are graph neural networks?, 2019b.

Yang, Z., Cohen, W., and Salakhudinov, R. Revisiting semi-supervised learning with graph embeddings. In *International conference on machine learning*, pp. 40–48. PMLR, 2016.

Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., and Leskovec, J. Graph convolutional neural networks for web-scale recommender systems. *CoRR*, abs/1806.01973, 2018. URL http://arxiv.org/abs/1806.01973.

Zhao, L., Peng, X., Tian, Y., Kapadia, M., and Metaxas, D. N. Semantic graph convolutional networks for 3d human pose regression. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 3425–3435. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00354. URL http://access.thecvf.com/content_CVPR_2019/html/Zhao_Semantic_Graph_Convolutional_Networks_for_3D_Human_Pose_Regression_CVPR_2019_paper.html.

Zhu, H. and Koniusz, P. Simple spectral graph convolution. In *International Conference on Learning Representations*, 2020.

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

A. Proof for Theorem 4.3

The proof for Theorem 4.3 is as follows:

Proof. From the definition, the approximation error for G^{2}CN-Taylor is:

$$\begin{aligned}
e_{Ta}^{T,K,b} &= \left\| \mathbf{H}_{Ta} - e^{T(b\mathbf{I}-\mathbf{L})^2}\mathbf{X} \right\|_2 && (20)\\
&= \left\| \sum_{i=0}^{K} \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^i}{i!}\mathbf{X} - \sum_{i=0}^{\infty} \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^i}{i!}\mathbf{X} \right\|_2 && (21)\\
&= \left\| \sum_{i=K+1}^{\infty} \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^i}{i!}\mathbf{X} \right\|_2 && (22)\\
&= \left\| (-T(b\mathbf{I}-\mathbf{L})^2)^{K+1} \sum_{i=0}^{\infty} \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^i}{(i+K+1)!}\mathbf{X} \right\|_2 && (23)\\
&= \left\| (-T(b\mathbf{I}-\mathbf{L})^2)^{K+1} \right\|_2 \left\| \sum_{i=0}^{\infty} \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^i}{(i+K+1)!} \right\|_2 \|\mathbf{X}\|_2 && (24)\\
&\leq \left\| \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^{K+1}}{(K+1)!} \right\|_2 \left\| \sum_{i=0}^{\infty} \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^i}{i!} \right\|_2 \|\mathbf{X}\|_2 && (25)\\
&= \left\| \frac{(-T(b\mathbf{I}-\mathbf{L})^2)^{K+1}}{(K+1)!} \right\|_2 \left\| e^{-T(b\mathbf{I}-\mathbf{L})^2} \right\|_2 \|\mathbf{X}\|_2 && (26)\\
&\leq \frac{(T\|(b\mathbf{I}-\mathbf{L})\|^2)^{K+1}}{(K+1)!} \|\mathbf{X}\|_2 && (27)\\
& && (28)
\end{aligned}$$

Then Theorem 4.3 is proved. □

B. Proof for Theorem 4.4

The proof for Theorem 4.4 is as follows:

Proof. The Euler Scheme is to solve the following differential equation at $t = 1$:

$$\begin{cases} \frac{d\mathbf{X}_t}{dt} = & -T(b\mathbf{I}-\mathbf{L})^2\mathbf{X}_t, \\ \mathbf{X}_0 = & \mathbf{X}, \end{cases} \tag{29}$$

with step size $1/K$ for K steps. Then consider a general Euler forward scheme for our problem:

$$\hat{\mathbf{X}}(k+1) = \hat{\mathbf{X}^{(k)}} - h\mathbf{L}_{b,T}^2\hat{\mathbf{X}}^{(k)},\ k = 0, ..., K-1,\ \mathbf{X}^{(0)} = \mathbf{X}, \tag{30}$$

with $\hat{\mathbf{X}}^{(k)}$ denotes approximated $\mathbf{X}(k/K)$ by the forward Euler, $h = \frac{1}{K}$ and $\mathbf{L}_{b,T} = \sqrt{T}(b\mathbf{I}-\mathbf{L})$. Then we denote the error at step k as:

$$\mathbf{e}_k = \mathbf{X}^{(k)} - \hat{\mathbf{X}}^{(k)}, \tag{31}$$

then the approximation error is $e_{Eu}^{T,K,b} = \mathbf{e}_K$, and the truncation error of the Euler forward finite difference (Eqn (30)) at step k as:

$$\mathbf{T}^{(k)} = \frac{\mathbf{X}^{(k+1)} - \mathbf{X}^{(k)}}{h} + \mathbf{L}_{b,T}^2\mathbf{X}^{(k)}, \tag{32}$$

which can be reformulated as follows:

$$\mathbf{X}^{(k+1)} = \mathbf{X}^{(k)} + h(\mathbf{T}^{(k)} - \mathbf{L}_{b,T}\mathbf{X}^{(k)}). \tag{33}$$

Then subtract Eqn 33 by Eqn 30, we can obtain:

$$\mathbf{e}^{(k+1)} = (\mathbf{I} - h\mathbf{L}_{b,T}^2)\mathbf{e}^{(k)} + h\mathbf{T}^{(k)}, \tag{34}$$

whose norm can be upper bounded as:

$$\left\|\mathbf{e}^{(k+1)}\right\|_2 \leq \left\|\mathbf{I} - h\mathbf{L}_{b,T}^2\right\|_2 \left\|\mathbf{e}^{(k)}\right\|_2 + h\left\|\mathbf{T}^{(k)}\right\|. \tag{35}$$

Since $\mathbf{L}_{b,T} = (b\mathbf{I} - \mathbf{L})$ with $b \in [0,2]$ and the eigenvalue of the symmetric matrix $\mathbf{L}$ also lies in $[0,2]$, Then the eigenvalue of $\mathbf{L}_{b,T}^2$ lies in $[0,4T]$ and eigenvalue of $h\mathbf{L}_{b,T}^2$ lies in $[0,2]$ since $h = T/K < 0.5$. Thereby, the spectral norm of $\mathbf{I} - h\mathbf{L}_{b,T}^2$ is upper bounded by 1. Then with $M = \max_{0\leq k\leq K-1}\|\mathbf{T}^{(k)}\|_2$ be the upper bound on the truncation error, we have:

$$\left\|\mathbf{e}^{k+1}\right\|_2 \leq \|\mathbf{e}^{(k)}\|_2^{(k)} + hM. \tag{36}$$

Since $\mathbf{e}^{(0)} = \mathbf{X}^{(0)} - \hat{\mathbf{X}}^{(0)} = 0$, we have the following formula by induction:

$$\left\|\mathbf{e}^{(K)}\right\|_2 \leq KhM = M. \tag{37}$$

Then since $\frac{d\mathbf{X}^{(k)}}{dt} = -L\mathbf{X}^{(k)}$ and applying Taylor's theorem, there exists $\delta \in [kh, (k+1)h]$ such that the truncation error $\mathbf{T}^{(k)}$ in Eqn 32 follows:

$$\mathbf{T}^{(k)} = \frac{h}{2}\mathbf{L}_{b,T}^4\mathbf{X}_\delta. \tag{38}$$

Furthermore, since

$$\|\mathbf{X}_\delta\|_2 = \left\|e^{-T\delta\mathbf{L}_b^2}\mathbf{X}_0\right\| \leq \|\mathbf{X}\|_2. \tag{39}$$

Thereby, we can bound the truncated error:

$$\|\mathbf{T}^{(k)}\|_2 = \frac{h}{2}\|\mathbf{L}_b\|^4\|\mathbf{X}_\delta\|_2 \leq \frac{h}{2}\|\mathbf{L}_b\|^4\|\mathbf{X}\|_2, \tag{40}$$

Finally, we have

$$\left\|\mathbf{e}_{Eu}^{T,K,b}\right\|_2 = \left\|\mathbf{e}^{(K)}\right\|_2 \leq M \leq \frac{T^2}{2K}\|b\mathbf{I} - \mathbf{L}\|^4\|\mathbf{X}\|_2. \tag{41}$$

Then we complete the proof. □

C. Bandwidth and Concentration Centers for Graph filters approximated by Polynomial Bases.

We list the bandwidth of each Bernstein basis for BernNet with $K = 10$ listed in Table C.

Table 8. Concentration Attributes for Bernstein Basis.

Index	0	1	2	3	4	5	6	7	8	9	10
Concentration Center	0.0	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0
PassBand	$[0.0, 0.069]$	$[0.079, 0.390]$	$[0.219, 0.635]$	$[0.381, 0.855]$	$[0.555, 1.062]$	$[0.741, 1.258]$	$[0.938, 1.445]$	$[1.145, 1.619]$	$[1.365, 1.781]$	$[1.610, 1.921]$	$[1.931, 2.0]$
Maximum Response	1	0.387	0.301	0.267	0.251	0.246	0.251	0.267	0.301	0.387	1

D. Ablation Study on the Propagation Stpes K.

We also finish the experiments on Photo with different propagation times K for our G^{2}CN Euler to explore K's influence. The results are shown in Figure 2. Same to our analysis stated in Theorem 4.3, as K increases, the approximation error will decrease and the performance will increase and finally converge.

E. Summary for our concentration analysis on different graph propagation.

We summarize the concentration attributes for different graph propagation as follows:

F. Response Curve for Other Graphs

We list the response curve for the rest Graphs as follows:

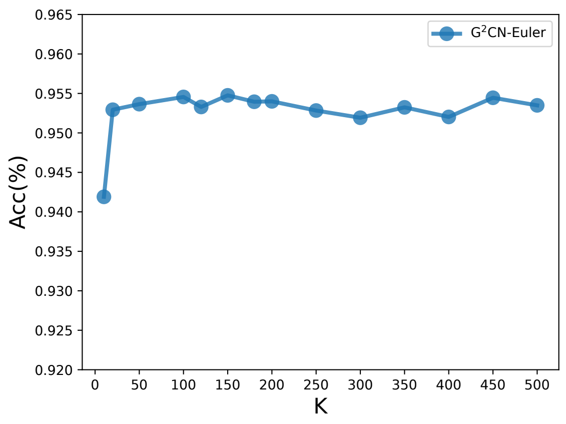
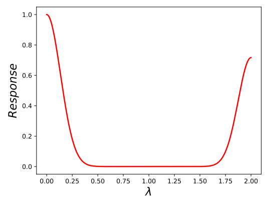


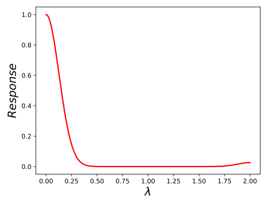
Figure 2. Test Accuracy with respect to different propagation times for our G^{2}CN on Photo.

Table 9. Concentration attributes for different graph propagation.

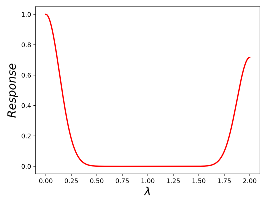
	Graph Propagation Kernel (Or Basis)	$\mathcal{R}$	b	$\mathcal{BW}$
GCN	$(2\mathbf{I}-\mathbf{L})^K$	2^K	0	$2-2^{1-\frac{1}{2K}}$
PPR	$(\mathbf{I}-(1-\alpha)(1-\mathbf{L}))^{-1}$	$\frac{1}{\alpha}$	0	$\frac{(\sqrt{2}-1)\alpha}{1-\alpha}$
Heat Kernel	$e^{-T\mathbf{L}}$	1	0	$\frac{log(\sqrt{2})}{T}$
ChebNet	$\mathbf{C}^{(1)}=\mathbf{I}$ $\mathbf{C}^{(2)}=2\mathbf{L}/\lambda_{\max}-\mathbf{I}$ $\mathbf{C}^{(k)}=2\mathbf{C}^{(2)}\mathbf{C}^{(s-1)}-\mathbf{C}^{(s-2)}$	1	$[0,2]$ $1,-1$	2 $2-\sqrt{2}$ ....
BernNet	$\frac{1}{2^K}\binom{K}{k}(2\mathbf{I}-\mathbf{L})^{K-k}\mathbf{L}^k$	Appendix C	$\frac{2}{K}$	Appendix C
Graph Gaussian	$e^{-T(\mathbf{b}-\mathbf{L})^2}$	1	b	Proposition 4.1



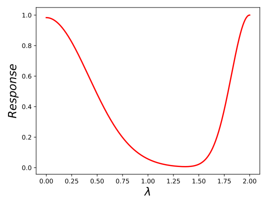
(a) Pubmed



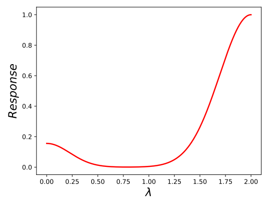
(b) Citeseer



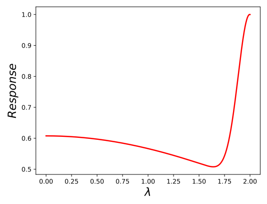
(c) Pubmed



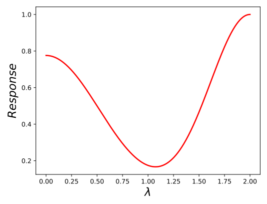
(d) Computers



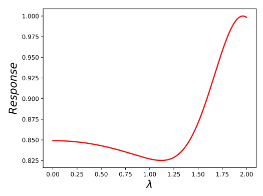
(e) Squirrel



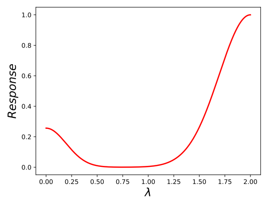
(f) Cornell



(g) Photo



(h) Actor



(i) Chameleon

Figure 3. G^{2}CN's equivalent graph filters for real-world datasets.