Graph Linear Canonical Transform: Definition, Vertex-Frequency Analysis and Filter Design

Jian Yi Chen and Bing Zhao Li

***Abstract*—This paper proposes a graph linear canonical transform (GLCT) by decomposing the linear canonical parameter matrix into fractional Fourier transform, scale transform, and chirp modulation for graph signal processing. The GLCT enables adjustable smoothing modes, enhancing alignment with graph signals. Leveraging traditional fractional domain time-frequency analysis, we investigate vertex-frequency analysis in the graph linear canonical domain, aiming to overcome limitations in capturing local information. Filter design methods, including optimal design and learning with stochastic gradient descent, are analyzed and applied to image classification tasks. The proposed GLCT and vertex-frequency analysis present innovative approaches to signal processing challenges, with potential applications in various fields.**

***Index Terms*—Graph signal processing, graph linear canonical transform, vertex-frequency analysis, filter design.**

I. Introduction

Graph signal processing has a broad spectrum of applications in computer vision, image processing, and social network analysis. Advancements in these technologies will catalyze technological innovation and societal progress, offering abundant developmental opportunities and benefits for humanity [**?**], [**?**], [**?**]. The essence of graph signal processing lies in the construction of dictionary atoms, followed by the representation of graph signals as linear combinations of these atoms [**?**]. This representation technique finds diverse applications, including data visualization analysis, statistical analysis, compression, and regularization in machine learning, and ill-posed inverse problems such as repair, denoising, and classification tasks.

When the atom is the eigenvector (Fourier basis) obtained from the Laplacian matrix decomposition of the graph, different Fourier bases correspond to distinct levels of smoothness. Their associated eigenvalues provide a gradient characterization of the smoothness of the graph signal, effectively functioning as frequencies [**?**], [**?**]. Lower eigenvalues indicate slower changes in the corresponding Fourier basis, resulting in consistent signal values across similar nodes. Conversely, higher eigenvalues signify more rapid changes in the Fourier basis, leading to inconsistent signal values across similar nodes. Thus, the coefficients of the graph Fourier transform (GFT) represent the amplitude of the graph signal on the corresponding frequency component, reflecting the strength of the signal on that frequency component.

Based on the GFT, various graph analysis methods have emerged, with the core focus being on designing different dictionary atoms to represent graph signals. The effectiveness of signal analysis refers to the ability to capture important information in a signal with fewer mathematical descriptions [**?**], which essentially requires atoms to be able to sparsely represent the signal. Therefore, the effectiveness of vertex-frequency analysis of graph signals requires atoms to have the following characteristics in the spatial domain:

- Interpretability: The methods should enable the explanation of the underlying graph structure.
- Multiresolution: They should be able to continuously approximate signals from low-frequency resolution to high-frequency resolution.
- Locality: Emphasis should be placed on the local structure of the graph.
- Multiscale: They should encompass vertex-frequency distributions of various scales.
- Smooth diversity: The ability to select different smoothing modes should be included.

Traditional vertex-frequency analysis methods can effectively fulfill the first four characteristics but are inadequate in addressing the last one [**?**], [**?**], [**?**]. The fixed smoothing mode, based on GFT, solely relies on its Laplacian matrix. Consequently, traditional vertex-frequency analysis methods are confined to characterizing signals from the horizontal and vertical directions of the vertex-frequency plane. This limitation stems from the mathematical structure of their atoms, typically comprising time shift, frequency shift, or scaling operations. As a result, these methods can only impart changes to atoms in the horizontal and vertical directions of the vertex-frequency plane [**?**], [**?**]. Consequently, traditional vertex-frequency analysis methods fail to fully exploit the inherent geometric characteristics of the signal itself and encounter bottlenecks in enhancing signal processing performance further. An effective solution entails the introduction of a new vertex-frequency analysis method.

The Linear canonical transform (LCT) is a more generalized integral transform, with the Fourier transform, fractional Fourier transform (FrFT), and Fresnel transform all being regarded as special forms of LCT [**?**], [**?**], [**?**]. It demonstrates strong flexibility due to its additional parameters. Building upon the advantages of this transform method, it has been further integrated into graph signal processing technology, leading to significant research achievements. These include the

J. Y. Chen is with the School of Mathematics and Statistics, Beijing Institute of Technology , Beijing 100081, China. (e-mail: 15030218436@163.com)

B. Z. Li is with the Beijing Key Laboratory on MCAACI, Beijing Institute of Technology, Beijing 100081, China. (e-mail: li_bingzhao@bit.edu.cn)

0000–0000/00$00.00 © 2021 IEEE

development of graph fractional Fourier transform [**?**], graph linear canonical transform (GLCT) [**?**] based on adjacency matrix , and related vertex-frequency analysis methods [**?**], [**?**]. The linear canonical parameter matrix can be decomposed into a linear combination of FrFT, scale transform, and chirp modulation (CM). Leveraging this pivotal property, this paper proposes a GLCT based on the graph Laplacian matrix, which significantly diverges from previous work [**?**]. The GLCT facilitates the adjustment of the smoothing mode to better align with the graph signal. Additionally, building upon the research foundation of traditional fractional domain time-frequency analysis methods [**?**], [**?**], [**?**], [**?**], this paper explores vertex-frequency analysis methods in the graph linear canonical domain. This research endeavors to address the inherent limitations of graph signal processing techniques in characterizing local information.

The vertex-frequency analysis method based on GLCT enables atoms to select different smoothing modes, facilitating optimal matching of intrinsic local features of the signal. This approach offers new ideas and methods for addressing problems in signal processing. This paper first presents a systematic framework for vertex-frequency analysis based on GLCT. It then discusses three definitions of filters: (1) The filter is constructed based on the convolution and translation operators of GLCT, leading to the definition of windowed GLCT. (2) By utilizing the spectral shift form of the kernel of the bandpass function in the linear canonical domain of the graph, we can define GLCT-based wavelet transform and S-transform. (3) A filter constructed based on the vertex neighborhood can define local GLCT. In practical applications, selecting the appropriate filter is crucial. For instance, various spectral neural networks primarily rely on different filter learning methods [**?**], [**?**], [**?**], [**?**]. Therefore, we analyze methods for filter design, discussing optimal filter design as well as filter learning based on stochastic gradient descent methods. Subsequently, we apply the designed filters to image classification tasks. The filter design proposed in this paper can be utilized as convolutional layers in spectral graph neural networks [**?**], [**?**], [**?**], [**?**], showcasing significant potential for various applications.

This paper is structured through the following steps. Section **??** presents the basic theory of graph signal processing and defines time-frequency analysis methods based on LCT. In Section **??**, the GLCT is proposed through FrFT, scaling, and CM. In Section **??**, the definition of the vertex-frequency analysis method based on GLCT is discussed and summarized. In Section **??**, we discuss optimal filter design and filter learning, and we apply the designed filters to image classification tasks. Finally, Section **??** concludes this paper.

II. Preliminaries

A. The graph signal

Consider a weighted, undirected, and connected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathrm{W}\}$, where $\mathcal{V} = \{v_0, v_1, \cdots, v_{N-1}\}$ with $|\mathcal{V}| = N$ represents a finite set of vertices, $\mathcal{E}$ denotes the set of edges, and W represents the weighted adjacency matrix. In this graph, if there exists an edge $e = (i, j)$ connecting vertices i and j, the weight of the edge is denoted by $w_{i,j}$; otherwise, $w_{i,j} = 0$ [**?**].

The non-normalized graph Laplacian is a symmetric difference operator $\mathcal{L} = \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ represents a diagonal degree matrix of the graph $\mathcal{G}$. The i-th diagonal element d_i of $\mathbf{D}$ is equal to the sum of the weights of all the edges incident to vertex i. Since the matrix $\mathcal{L}$ is real and symmetric, it possesses a complete set of orthonormal eigenvectors denoted as $\{\mathbf{u}_k\}_{k=0,1,\cdots,N-1}$. These eigenvectors are sorted in ascending order according to their corresponding eigenvalues $0 = \widetilde{\lambda}_0 < \widetilde{\lambda}_1 \leq \widetilde{\lambda}_2 \leq \cdots \leq \widetilde{\lambda}_{N-1}$. Thus, using the unitary column matrix $\mathbf{U} = [\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_{N-1}]$ and the diagonal matrix $\mathbf{\Lambda} = \mathrm{diag}\left(\left[\widetilde{\lambda}_0, \widetilde{\lambda}_1, \cdots, \widetilde{\lambda}_{N-1}\right]\right)$, we can express

$$\mathcal{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^H \tag{1}$$

where the superscript H represents the Hermitian transpose operation. While we use the non-normalized graph Laplacian $\mathcal{L}$ for explanation in this article, we can still use the normalized graph Laplacian $\mathcal{L}_{\text{norm}} = \mathbf{E}_N - \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$, where $\mathbf{E}_N$ represents an N-dimensional identity matrix [**?**].

A signal defined on the graph $\mathcal{G}$ can be represented as a mapping [**?**]

$$\begin{aligned} f : \mathcal{V} &\to \mathbb{R} \\ v_i &\mapsto f(i). \end{aligned} \tag{2}$$

Alternatively, we can express $\mathbf{f}$ as a real-valued vector $\mathbf{f} = [f(0), f(1), \cdots, f(N-1)]^T \in \mathbb{R}^N$, where $f(i)$ represents the value associated with the i-th vertex. The Laplacian matrix is an operator that captures the local smoothness of a graph signal. When it acts on the graph signal f, it is defined as

$$(\mathcal{L}\mathbf{f})(i) = \sum_{j \in \mathcal{N}i} w_{i,j}[f(i) - f(j)] \tag{3}$$

where the neighborhood $\mathcal{N}_i$ represents the set of vertices connected to vertex i by an edge. This operator quantifies the difference in signal between the central node and its neighboring nodes.

Example 1. *We simulate two distinct graphs: a 128-node sensor network graph (Fig.* **??***. (a)) and a 256-node Swiss roll graph (Fig.* **??***. (b)). We consider the random signal for both cases, as shown in Fig.* **??** *(c) and (d), respectively.*

B. The graph fractional Fourier transform

1) Graph Fourier transform: For any function $\mathbf{f} \in \mathbb{R}^N$ defined on the vertices of $\mathcal{G}$, its Graph Fourier Transform (GFT) is defined as [**?**]

$$\hat{f}(k) = \langle f, u_k \rangle = \sum_{n=0}^{N-1} f(n) u_k^*(n), \quad k = 0, 1, \cdots, N-1. \tag{4}$$

The inverse GFT is given by

$$f(n) = \sum_{k=0}^{N-1} \hat{f}(k) u_k(n), \quad n = 0, 1, \cdots, N-1. \tag{5}$$

Therefore, we can observe that the GFT can be represented using the matrix $\mathbf{U}$, such that $\hat{\mathbf{f}} = \mathcal{F}\mathbf{f} = \mathbf{U}^H\mathbf{f}$ [**?**]. In the case where the GFT matrix $\mathcal{F}$ is ortho-diagonalized, it can be expressed as $\mathcal{F} = \mathbf{U}^H = \mathbf{Q}\mathbf{R}\mathbf{Q}^H$, where the matrices $\mathbf{Q} = [\mathbf{q}_0, \mathbf{q}_1, \cdots, \mathbf{q}_{N-1}]$ and $\mathbf{R} = \mathrm{diag}([r_0, r_1, \cdots, r_{N-1}])$ are obtained through the spectral decomposition of the GFT matrix.

2) Graph fractional Fourier transform: By denoting $\mathbf{J} = \mathrm{diag}([\zeta_0, \zeta_1, \cdots, \zeta_{N-1}]) = R^\alpha$, where $\zeta_l = r_l^\alpha$, the graph fractional Fourier transform (GFrFT) is defined as $\hat{\mathbf{f}}_\alpha = \mathcal{F}_\alpha\mathbf{f} = \mathbf{Q}\mathbf{J}\mathbf{Q}^H\mathbf{f}$ [**?**]. There is [**?**]

$$\hat{f}_\alpha(k) = \sum_{n=0}^{N-1}\sum_{l=0}^{N-1} f(n)q_l(k)\zeta_l q_l^*(n) \tag{6}$$

where the papameter α is the order of the fractional Fourier transform, and $k = 0, 1, \cdots, N-1$. Similar to (**??**), we define the graph fractional Laplacian operator $\mathcal{L}_\alpha$ as follows

$$\mathcal{L}_\alpha = \mathcal{F}_\alpha^H\mathbf{\Lambda}\mathcal{F}_\alpha. \tag{7}$$

The inverse GFrFT is given by

$$f(n) = \sum_{k=0}^{N-1}\hat{f}_\alpha(k)q_l^*(k)\zeta_l q_l(n), \quad n = 0, 1, \cdots, N-1. \tag{8}$$

C. Linear canonical transform

The linear canonical transform (LCT) of the signal $f(t) \in L^2(\mathbb{R})$ with the parameters $A = (a, b, c, d)$, where $ad - bc = 1$, is defined as follows [**?**], [**?**], [**?**]

$$L^A\{f\}(u) = \begin{cases} \int_{-\infty}^{\infty} f(t)K_A(t,u)\mathrm{d}t & b \neq 0 \\ \sqrt{d}e^{\mathbf{i}(cd/2)u^2}f(dt) & b = 0 \end{cases} \tag{9}$$

$$K_A(t,u) = \sqrt{\frac{1}{\mathbf{i}2\pi b}}e^{\mathbf{i}\left(\frac{d}{b}u^2 - \frac{2}{b}ut + \frac{a}{b}t^2\right)} \tag{10}$$

where $\mathbf{i}$ reprents an imaginary unit. When the parameter $\mathrm{A} = (\cos\alpha, \sin\alpha, -\sin\alpha, \cos\alpha)$ is used, the LCT becomes the α-th order fractional Fourier transform (FrFT)

$$L^{(\cos\alpha,\sin\alpha,-\sin\alpha,\cos\alpha)}\{f\}(u) = \int_{-\infty}^{\infty} K_\alpha(t,u)f(t)\mathrm{d}t$$

$$K_\alpha(t,u) = \begin{cases} A_\alpha \exp\left\{\mathbf{i}\frac{u^2\cot\alpha}{2} - \mathbf{i}tu\csc\alpha + \mathbf{i}\frac{t^2\cot\alpha}{2}\right\} \\ \alpha \neq n\pi \\ \delta(u-t), \alpha = 2n\pi \\ \delta(u+t), \alpha = 2n\pi - \pi \end{cases}$$

$$A_\alpha = \sqrt{\frac{1 - \mathbf{i}\cot(\alpha)}{2\pi}}, 0 < |\alpha| < \pi. \tag{11}$$

The LCT with parameter $A = (a, b, c, d)$ can be decomposed into a combination of various specialized transforms, achieved through the decomposition of the parameter matrix. This article will utilize the following decomposition

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \xi & 1 \end{bmatrix}\begin{bmatrix} \sigma & 0 \\ 0 & \sigma^{-1} \end{bmatrix}\begin{bmatrix} \cos\beta & \sin\beta \\ -\sin\beta & \cos\beta \end{bmatrix} \tag{12}$$

where A is decomposed into FrFT, scaling, and chirp modulation (CM), and ξ is the CM parameter, σ denotes the scaling parameter, β is the FrFT parameter. The parameter relations between (a, b, c, d) and (ξ, σ, β) are $\xi = \frac{ac+bd}{a^2+b^2}, \sigma = \sqrt{a^2+b^2}, \beta = \cos^{-1}\left(\frac{a}{\sigma}\right) = \sin^{-1}\left(\frac{b}{\sigma}\right)$. In Section **??**, we will present the definition of graph linear canonical transform (GLCT) based on this parameter matrix factorization.

III. Graph linear canonical transform

Building upon the centrality and scalability of the LCT eigendecomposition approach, Zhang proposes the definition of the GLCT based on the adjacency matrix by integrating the graph chirp-Fourier transform, graph scale transform, and graph fractional Fourier transform [**?**]. Similarly, this article proposes a GLCT based on the Laplacian matrix through decomposition of the LCT parameter matrix in (**??**). However, there is a significant difference between the definition in this article and the definition in [**?**].

Since any LCT matrix is non-commutative, it should be implemented in the order of FrFT, scaling, CM. To derive the definition of GLCT based on the Laplacian matrix, we accomplish it through the following three stages.

Step1 GFrFT stage

For a graph signal, initiate the GFrFT as delineated in Section **??**, expressed as

$$\hat{\mathbf{f}}_\beta = \mathcal{F}_\beta\mathbf{f} = \mathbf{Q}\mathcal{K}\mathbf{Q}^H\mathbf{f} \tag{13}$$

where $\mathcal{K} = \mathrm{diag}([\kappa_0, \kappa_1, \cdots, \kappa_{N-1}]) = R^\beta$, $\kappa_l = r_l^\beta, l = 0, \cdots, N-1$, and $\beta = \cos^{-1}\left(\frac{a}{\sigma}\right) = \sin^{-1}\left(\frac{b}{\sigma}\right)$.

Step2 Graph scaling stage

In graph signal processing, defining scale transform differs from traditional signal processing methods. Hammond introduced the concept of scale on the graph by utilizing the scale property of the Fourier transform [**?**], defining the scale operators at scale s as $S_f^s = f(s\mathcal{L})$. Therefore, this article considers the scale Laplacian matrix $\mathbf{S} = \sigma\mathcal{L}$.

Pei noted that the scaled signal can be derived by decomposing the signal into orthogonal Hermite eigenspaces and subsequently reconstructing the signal using scale Hermite eigenvectors with identical inner product coefficients [**?**]. We employ this concept to define the graph scale transform.

The graph spectrum decomposition of $\mathbf{S}$ is represented as $\mathbf{S} = \mathbf{Y}\mathbf{X}\mathbf{Y}^H$, and $\mathbf{Y}^H = \mathbf{P}\Xi\mathbf{P}^H$, where $\mathbf{P} = [\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{N-1}]$ and $\Xi = \mathrm{diag}([\iota_0, \iota_1, \cdots, \iota_{N-1}])$. The graph scale transform is defined as

$$ST_\sigma\mathbf{f} = \mathbf{P}\mathbf{Q}^H\mathbf{f} \tag{14}$$

where $\sigma = \sqrt{a^2+b^2}$.

Step3 Graph CM stage

In classical signal processing, the modulation operator is defined by $M_\xi f = e^{\mathbf{i}\pi\xi t}f(t)$. Shuman defined the modulation operator on the graph as $M_k f = f(n)u_k^2(n)$, where $k \in \{0, \cdots, N-1\}$, recognizing that the essence of the modulation operator lies in multiplying by the Laplacian eigenvectors.

However, the scenario addressed in this article pertains to CM, where $CM_\xi f = e^{\mathbf{i}\pi\xi t^2} f(t)$. The concept of CM on the graph has not yet been clearly defined, and defining it from a practical perspective is quite challenging. Hence, we define it based on the chirp signal spectrum perspective.

In traditional signal processing, the Fourier spectrum of a chirp signal $s(t) = \text{rect}\left(\frac{t}{T}\right) e^{j\pi\xi t^2}$ can be approximated as $F\{s\}(u) \approx \text{rect}\left(\frac{u}{|\xi|T}\right) e^{-j\pi\frac{u^2}{\xi}}$ [**?**]. Similarly, we define a signal in a graph spectral domain as

$$\hat{s}_\xi(k) = \begin{cases} e^{j\frac{\tilde{\lambda}_k^2}{\xi}}, & \xi \neq 0 \\ \mathbf{1}_N, & \xi = 0 \end{cases} \tag{15}$$

where $\mathbf{1}_N$ represents an N-dimensional column vector with all elements being 1. Next, the inverse GFT is performed to obtain its vertex domain form as $s(n) = \mathbf{U}\hat{s}_\xi(k)$. We utilize this signal as the CM term for the graph signal. Let

$$\Upsilon = \text{diag}\left(\mathbf{U}\hat{s}_\xi(k)\right) \tag{16}$$

thus, defining the CM operator on the graph as

$$CM_\xi \mathbf{f} = \Upsilon \mathbf{f} \tag{17}$$

where $\xi = \frac{ac+bd}{a^2+b^2}$.

Based on the aforementioned three stages, we define the GLCT by performing FrFT in (**??**), scaling in (**??**), and CM in (**??**) on a graph signal $\mathbf{f}$.

Definition 1. *For the parameter $A = (a, b, c, d)$ of LCT, denote $\sigma = \sqrt{a^2 + b^2}$, $\xi = \frac{ac+bd}{\sigma^2}$, $\beta = \cos^{-1}\left(\frac{a}{\sigma}\right) = \sin^{-1}\left(\frac{b}{\sigma}\right)$. The graph linear canonical transform (GLCT) is defined by*

$$\hat{\mathbf{f}}_A = \mathcal{F}_A\mathbf{f} = \Upsilon(\mathbf{P}\mathbf{Q}^H)(\mathbf{Q}\mathcal{K}\mathbf{Q}^H)\mathbf{f} = \Upsilon\mathbf{P}\mathcal{K}\mathbf{Q}^H\mathbf{f}. \tag{18}$$

Its algebraic form is

$$\hat{f}_A(k) = \sum_{n=0}^{N-1}\sum_{l=0}^{N-1} f(n)\epsilon_k p_l(k)\kappa_l q_l^*(n). \tag{19}$$

The corresponding eigenvalues are $\mathbf{\Lambda}_A = \text{diag}\left([\lambda_0, \lambda_1, \cdots, \lambda_{N-1}]\right)$, where $\lambda_l = \sigma\tilde{\lambda}_l^\beta, l = 0, \cdots, N-1$.

Likewise, when $\mathrm{A} = (\cos\alpha, \sin\alpha, -\sin\alpha, \cos\alpha)$ is used, the equation $\hat{f}_{A=(\cos\alpha,\sin\alpha,-\sin\alpha,\cos\alpha)}(k) = \hat{f}_\alpha(k)$ holds true. We can also define graph linear canonical Laplacian operator as

$$\mathcal{L}_A = \mathcal{F}_A^H \mathbf{\Lambda}_A \mathcal{F}_A. \tag{20}$$

The inverse GLCT (IGLCT) is defined as $\mathbf{f} = \mathcal{F}_A^H\hat{\mathbf{f}}_A = \mathbf{Q}\mathcal{K}^H\mathbf{P}^H\Upsilon^H\hat{\mathbf{f}}_A$, which is equal to

$$f(n) = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} \hat{f}_A(k) q_l(n)\kappa_l^* p_l^*(k)\epsilon_k^*. \tag{21}$$

Example 2. *Perform the GLCT with parameters $\sigma = 1/3$, $\xi = 0$, and $\beta = 0.8$ on the graph and graph signal in Example* **??**. *The results are depicted in Fig.* **??** *(e) and (f), respectively.*

Example 3. *To further elucidate the role of GLCT, we visualize its feature vectors. Consider a K-nearest neighbors (K-NN) graph with 100 vertices, where each vertex connects to 4 edges. Figures* **??** *(a-c) depict the 9th, 49th, and 89th eigenvectors of GFT. Figures* **??** *(d-f) illustrate the 9th, 49th, and 89th eigenvectors of GLCT under parameters $\sigma = 1$, $\xi = 0.8$, and $\beta = 0.95$. Figures* **??** *(g-i) illustrate the 9th, 49th, and 89th eigenvectors of GLCT under parameters $\sigma = 0.4$, $\xi = 0$, and $\beta = 0.9$. It can be observed that as the eigenvalues increase, the eigenvectors become less smooth. The GLCT can modify the smoothing mode by adjusting parameters, which enables a better fit for various graph signals.*

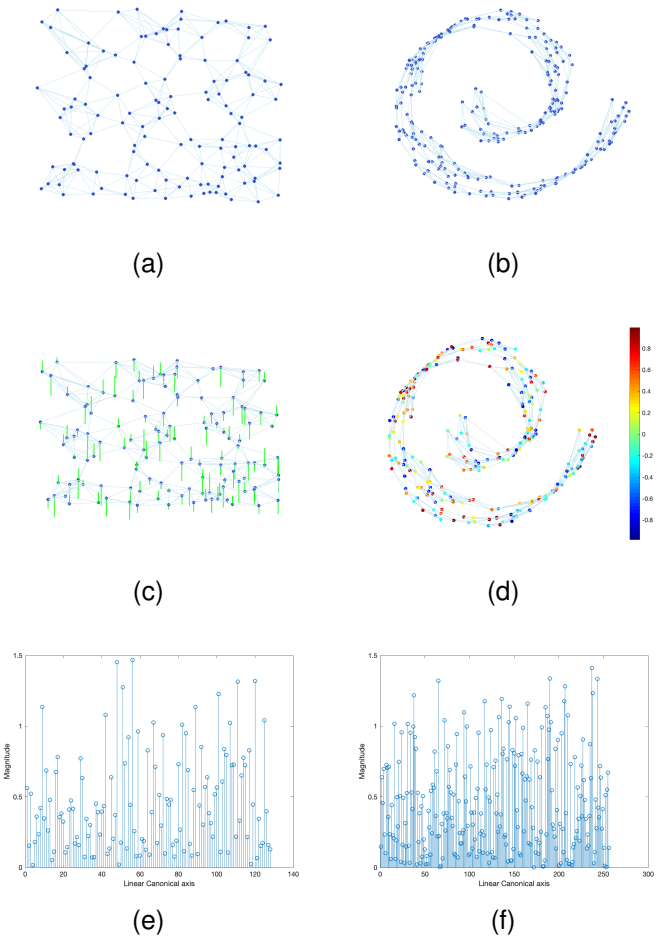

Fig. 1. (a) Sensor network graph. (b) Swiss roll graph. (c) A random signal on (a). (d) A random signal on (b). (e) The GLCT with $\sigma = 1/3$, $\xi = 0$, $\beta = 0.8$ of (c). (f) The GLCT with $\sigma = 1/3$, $\xi = 0$, $\beta = 0.8$ of (d).

IV. Vertex-Frequency Analysis Method Associated with GLCT

Like classical time-frequency analysis, the spectral transformation of signals localized around a specific vertex n, forms the fundamental formula for vertex-frequency analysis. This spectral transformation is typically achieved using a localization window. Hence, a generalized method for vertex-frequency analysis, denoted as VGLCT, can be derived by combining the GLCT with local window functions [**?**], [**?**]. The VGLCT is obtained by computing the GLCT of a signal

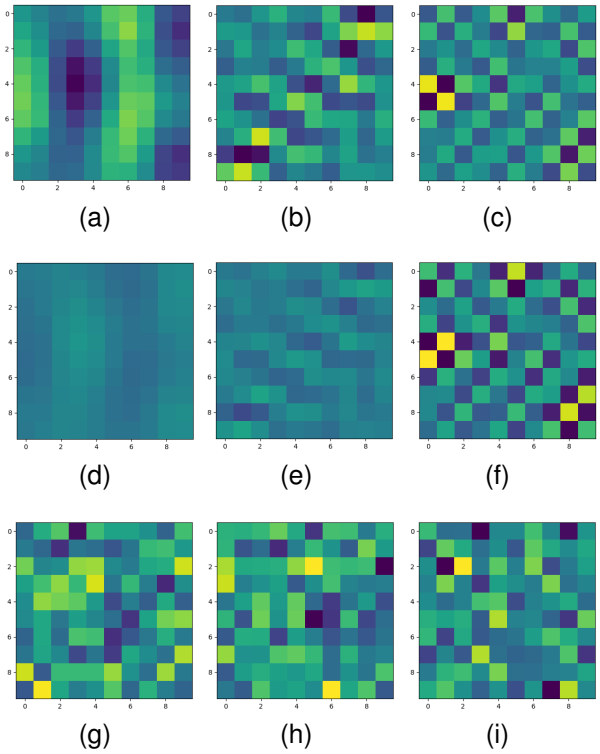

Fig. 2. Reshape the eigenvector to a 10×10 matrix. (a-c) The 9th, 49th, and 89th eigenvectors of the GFT. (d-f) The 9th, 49th, and 89th eigenvectors of GLCT with $\sigma = 1$, $\xi = 0.8$, $\beta = 0.95$. (g-i) The 9th, 49th, and 89th eigenvectors of GLCT with $\sigma = 0.4$, $\xi = 0$, $\beta = 0.95$.

$f(n)$ and multiplying it by a suitable vertex localization window function, $h_i(n)$, resulting in the following expression

$$\mathcal{V}_h^A f(i,k) = \sum_{n=0}^{N-1}\sum_{l=0}^{N-1} f(n)h_i(n)\epsilon_k p_l(k)\kappa_l q_l^*(n). \tag{22}$$

In general, the VGLCT initially localizes the signal around vertex i using the graph window function and then evaluates its GLCT performance. Specifically, the graph window function should have a value close to 1 at vertex i, indicating that the vertex is in close proximity to vertex i or its nearest neighbors. Conversely, for vertices that are distant from vertex i, the graph window function should have a value close to 0, indicating negligible influence on the analysis.

We can categorize the methods for constructing the graph window into three categories:

- The window function is constructed based on the convolution and translation operators of GLCT, leading to the definition of windowed GLCT.
- By utilizing the spectral shift form of the kernel of the bandpass function in the linear regular domain of the graph, we can define GLCT-based wavelet transform and S-transform.
- A window function constructed based on the vertex neighborhood can define local GLCT.

A. Windowed graph linear canonical transform

Unlike the traditional Fourier transform, the windowed Fourier transform (or short-time Fourier transform) divides the signal into a series of short-time windows and applies the Fourier transform to the signal within each window to obtain the spectral information of each window. In graph signal processing, the windowed graph Fourier transform (WGFT) has also been proposed [**?**], [**?**]. The basic idea is to define the WGFT as the inner product of the signal f and a shifted and modulated window function h. The WGFT is defined by incorporating convolution, translation, and modulation operations into the graph signal [**?**]. In this section, we begin by defining the graph linear canonical convolution operator and the graph linear canonical translation operator. Subsequently, we define the windowed graph linear canonical convolution transform (WGLCT) by multiplying the signal f with the translated window function prior to applying the GLCT.

Graph linear canonical convolution operator. Consider two signals, $f(n)$ and $h(n)$, defined on a graph, with their corresponding GLCTs represented as $\hat{f}_A$ and $\hat{h}_A$. The graph linear canonical convolution $*_A$ of these signals is defined as follows

$$\begin{aligned}(f *_A h)(n) &= IGLCT\{\hat{f}_A(k)\hat{h}_A(k)\} \\ &= \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} \hat{f}_A(k)\hat{h}_A(k)q_l(n)\kappa_l^* p_l^*(k)\epsilon_k^*. \end{aligned}\tag{23}$$

The concept of 'shift' on a graph cannot be directly and uniquely extended to graph signals using an analogy with classical signal shifts. In the realm of vertex frequency analysis, there has been an intriguing effort to generalize convolution and define corresponding shift operators on graphs. In this context, we aim to further generalize the calculation of shifts based on graph linear regularization transforms.

Graph linear canonical translation operator. For any signal $h(n)$ defined on a graph, and the delta function located at a vertex $i \in \{0, 1, \cdots, N-1\}$, given by $\delta_i(n) = \delta(n-i)$. The GLCT of delta function $\delta_i(n)$, is given by

$$\begin{aligned}\Delta_i(k) = GLCT\{\delta_i(n)\} &= \sum_{n=0}^{N-1}\sum_{l=0}^{N-1} \delta_i(n)\epsilon_k p_l(k)\kappa_l q_l^*(n) \\ &= \sum_{l=0}^{N-1} \epsilon_k p_l(k)\kappa_l q_l^*(i). \end{aligned}\tag{24}$$

The graph linear canonical translation T_i^A is defined as

$$\begin{aligned}T_i^A h(n) = (h *_A \delta_i)(n) &= \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} \hat{h}_A(k) \\ &\times \left(\sum_{s=0}^{N-1} \epsilon_k p_s(k)\kappa_s q_s^*(i)\right) q_l(n)\kappa_l^* p_l^*(k)\epsilon_k^*. \end{aligned}\tag{25}$$

Therefore, the WGLCT can be written as

$$\mathcal{V}_{h,1}^A f(i,k) = \sum_{n=0}^{N-1}\sum_{l=0}^{N-1} f(n)T_i^A h^*(n)\epsilon_k p_l(k)\kappa_l q_l^*(n) \tag{26}$$

where $T_i^A h(n)$ is defined in (**??**). By applying the graph window function $T_i^A h(n)$ to the signal $f(n)$ on each vertex, and then summing over all vertices i, a constant overlap-add

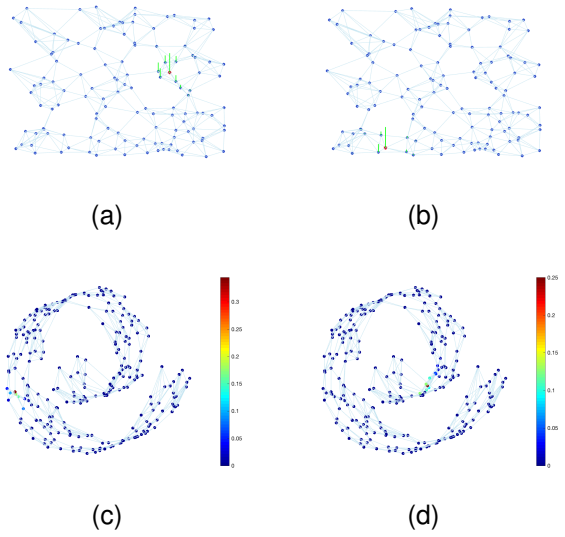

Fig. 3. Graph linear canonical translation. The red dot in (a) (b) represents the selected vertex i, while the green line represents the magnitude of the function value. (a) $T_{30}^{A}h$ on sensor network graph. (b) $T_{90}^{A}h$ on sensor network graph. (c) $T_{30}^{A}h$ on swiss roll graph. (d) $T_{200}^{A}h$ on swiss roll graph.

relation is achieved. Then the inverse WGLCT is

$$f(n)=\frac{1}{\sum_{i=0}^{N-1}T_i^A h(n)}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\mathcal{V}^A_{T_i^A h}f(i,k)q_l(n)\kappa_l^* p_l^*(k)\epsilon_k^*. \tag{27}$$

The basic form of this window, $h(n)$, can be conveniently defined in the spectral domain, for example as $H(k)=ce^{-\lambda_k\tau}$, where c denotes the window amplitude and $\tau>0$ is a constant which determines the window width.

Example 4. *Let $H(k)=e^{-\lambda_k/2}$ be the spectral domain-defined window, and $\sigma=1/3$, $\xi=0$, $\beta=0.8$. For the sensor network graph depicted in Fig.* **??** *(a) in Example* **??**, *we performed local point frequency analysis on vertex 30 and vertex 90, respectively. The results are displayed in Fig.* **??** *(a) and (b). For the Swiss roll graph shown in Fig.* **??** *(b), we conducted local point frequency analysis on vertex 30 and vertex 200, respectively. The corresponding results are presented in Fig.* **??** *(c) and (d).*

Example 5. *Consider the real part of graph signal as $f_1(n)=\Re(\varsigma_{30}(n))+\Re(\varsigma_{80}(n))$ for the sensor network graph on Fig.* **??** *(a), where $\varsigma_k(n)=\sum_{l=0}^{N-1}\epsilon_k p_l(k)\kappa_l q_l^*(n)$ is derived from the graph linear canonical Laplacian matrix of the sensor network graph. Let $H(k)=e^{-400\lambda_k}$, $\sigma=1/3$, $\xi=0$, and $\beta=0.8$, the WGLCT of f_1 is depicted in Figure* **??** *(a).*

Example 6. *Consider a signal on the Swiss roll graph shown in Fig.* **??** *(b), $f_2(n)=\Re(\varsigma_{20}(n))+\Re(\varsigma_{50}(n))+\Re(\varsigma_{70}(n))$, where $\varsigma_k(n)=\sum_{l=0}^{N-1}\epsilon_k p_l(k)\kappa_l q_l^*(n)$. Here, $\varsigma_k(n)$ is derived from the graph linear canonical Laplacian matrix of the Swiss roll graph. Let $H(k)=e^{-3500\lambda_k}$, $\sigma=1/3$, $\xi=0$, and $\beta=0.8$, the WGLCT of f_2 is depicted in Figure* **??** *(b).*

Below, we present a rapid computation approach for WGLCT. In the traditional scenario of one-dimensional signals, the windowed Fourier transform can be computed by taking the inverse FT from the so-called α-domain [**?**], [**?**], [**?**]. The α-domain is defined as the FT of the windowed Fourier domain. We define α-domain in linear canonical domain by

$$\alpha(u,u')=\int_{-\infty}^{\infty}V_h^A f(l,u)K_A(l,u')\mathrm{d}l \tag{28}$$

where $V_h^A f(l,u)$ represents the windowed LCT of f [**?**], that is, (**??**) applies LCT to $V_h^A f(l,u)$. The fast WLCT can be obtained by applying inverse LCT to α on the variable u'

$$FV_h^A f(l,u)=\int_{-\infty}^{\infty}\alpha(u,u')K_{A^{-1}}(l,u')\mathrm{d}u'. \tag{29}$$

Similarly, we define the α-domain as the GLCT of WGLCT in graph linear canonical domain as follows

$$\alpha(k,k')=\sum_{n=0}^{N-1}\sum_{l=0}^{N-1}\mathcal{V}^A_{T_i^A h}f(n,k')\epsilon_k p_l(k)\kappa_l q_l^*(n). \tag{30}$$

Then, the discretization of α is

$$\begin{aligned}\alpha(k,k')&=\sum_{n=0}^{N-1}\sum_{l=0}^{N-1}\sum_{m=0}^{N-1}\sum_{r=0}^{N-1}f(m)\sum_{d=0}^{N-1}\sum_{y=0}^{N-1}\sum_{s=0}^{N-1}\hat{h}_A^*(d)\\&\quad\times q_s(n)\kappa_s^* p_s^*(d)\epsilon_d^*\epsilon_d p_y(d)\kappa_y q_y^*(m)\\&\quad\times\epsilon_{k'}p_r(k')\kappa_r q_r^*(m)\epsilon_k p_l(k)\kappa_l q_l^*(n)\\&=\sum_{m=0}^{N-1}\sum_{r=0}^{N-1}\sum_{d=0}^{N-1}\sum_{y=0}^{N-1}\sum_{s=0}^{N-1}f(m)\hat{h}_A(d)\epsilon_{k'}p_r(k')\kappa_r q_r^*(m)\\&\quad\times\epsilon_d p_y(d)\kappa_y q_y^*(m)\delta_{dk}\\&=\sum_{m=0}^{N-1}\sum_{r=0}^{N-1}\sum_{y=0}^{N-1}f(m)\hat{h}_A(k)\epsilon_{k'}p_r(k')\kappa_r q_r^*(m)\\&\quad\times\epsilon_k p_y(k)\kappa_y q_y^*(m).\end{aligned} \tag{31}$$

The fast WGLCT is achieved by performing the inverse GLCT on the variable k with respect to α, that is

$$\begin{aligned}F\mathcal{V}^A_{h,1}f(i,k')&=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\alpha(k,k')q_l(n)\kappa_l^* p_l^*(k)\epsilon_k^*\\&=\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\tilde{f}(k,k')\hat{h}_A(k)q_l(n)\kappa_l^* p_l^*(k)\epsilon_k^*\end{aligned} \tag{32}$$

where

$$\begin{aligned}\tilde{f}(k,k')&=\sum_{m=0}^{N-1}\sum_{r=0}^{N-1}\sum_{y=0}^{N-1}f(m)\epsilon_{k'}p_r(k')\kappa_r q_r^*(m)\\&\quad\times\epsilon_k p_y(k)\kappa_y q_y^*(m).\end{aligned} \tag{33}$$

Example 7. *To compare the results of directly calculating the WGLCT with those obtained through the fast calculation method, we utilize the signals f_1 and f_2 from Example* **??** *and Example* **??**, *respectively. As depicted in Figure* **??**, *it is evident that both the fast calculation and direct calculation methods yield the same results, indicating that they have an equivalent effect.*

Algorithm 1 Fast windowed graph linear canonical transform

Input:

Graph signal f.

An $N \times N$ matrix $\mathbf{F}$ (each row of $\mathbf{F}$ is the signal f).

Output:

The fast WGLCT matrix $\mathbf{FV}_{h,2}^{A}$.

1: Compute $\mathcal{F}_A = \Upsilon \mathbf{P} \mathcal{K} \mathbf{Q}^H$.

2: Calculate $\tilde{f}(k,k')$ using the expression $\tilde{\mathbf{F}} = [\tilde{f}(k,k')] = (\mathbf{F} \circ \mathcal{F}_A^H) \cdot \mathcal{F}_A$, where $\circ$ represents element-wise multiplication (Hadamard multiplication), and $\cdot$ denotes standard matrix multiplication.

3: Construct a matrix where each row corresponds to the GLCT of the window function

$$\mathbf{H}_A = \begin{pmatrix} \hat{h}_A(r_0) & \hat{h}_A(k_1) & \dots & \hat{h}_A(k_{N-1}) \\ \hat{h}_A(k_0) & \hat{h}_A(k_1) & \dots & \hat{h}_A(k_{N-1}) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{h}_A(k_0) & \hat{h}_A(k_1) & \dots & \hat{h}_A(k_{N-1}) \end{pmatrix}.$$

4: Compute the α-domain representation by multiplying $\tilde{\mathbf{F}}$ and $\mathbf{H}_A^*$, i.e., $\alpha = \mathbf{H}_A^* \cdot \tilde{\mathbf{F}}$.

5: The fast WGLCT at each vertex is obtained by applying inverse GLCT to σ, i.e., $\mathbf{FV}_{h,2}^{A} = \mathcal{F}_A \cdot \alpha$.

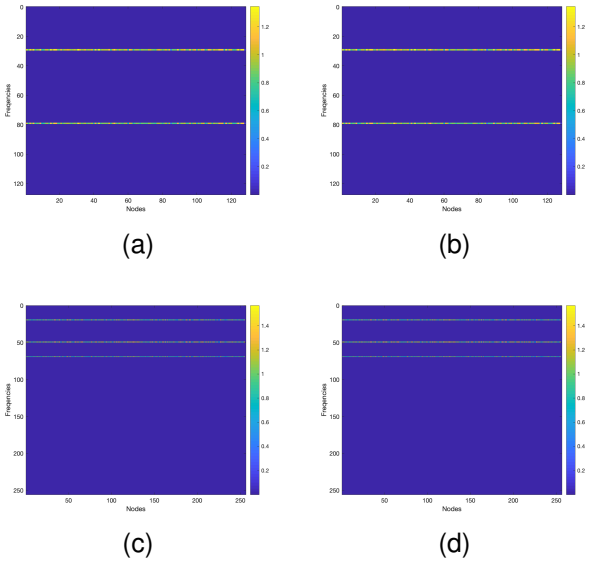

Fig. 4. (a)The WGLCT of f_1 in Example **??**. (b)The fast WGLCT of f_1 in Example **??**. (c)The WGLCT of f_2 in Example **??**. (d)The fast WGLCT of f_2 in Example **??**.

B. Graph linear canonical domain localization

The classical time-frequency analysis method employs windows for frequency localization in the spectral domain, enabling the derivation of the dual form of the time-frequency analysis method using the DFT of the original signal and the spectral window. Similarly, for graph signals, we can apply this method to localize them in the graph linear canonical domain and obtain VGLCT. For graph signals, we can also employ this approach to achieve localization in the spectral domain. In this case, the VGLCT is obtained as the IGLCT of $\hat{f}(z)$, which is localized using a spectral domain window $H(k-z)$

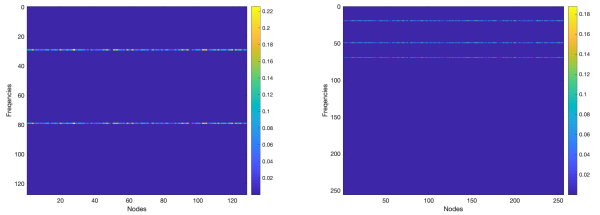

Fig. 5. The WGLCT of graph signals. (a) The WGLCT of a signal on sensor network graph. (b) The WGLCT of a signal on swiss roll graph.

centered around the spectral index k [**?**]. Thus, the VGLCT can be expressed as follows

$$\mathcal{V}_h^A f(i,k) = \sum_{p=0}^{N-1} \sum_{l=0}^{N-1} \hat{f}_A(p) H(k-z) q_l(n) \kappa_l^* p_l^*(z) \epsilon_z^*. \quad (34)$$

It can be imlemented using band-pass transfer functions, $H(k-z) = H_k(\lambda_z)$, as

$$\mathcal{V}_{h,2}^A f(i,k) = \sum_{p=0}^{N-1} \sum_{l=0}^{N-1} \hat{f}_A(z) H_k(\lambda_z) q_l(n) \kappa_l^* p_l^*(z) \epsilon_z^*. \quad (35)$$

The spectral domain localization window for a band-pass graph system can be expressed as a transfer function using eigenvalues λ_z as the center and its surroundings [**?**], [**?**]. It is defined in a polynomial form, represented by the following equation

$$H_k(\lambda_z) = h_{0,k} + h_{1,k}\lambda_z + \cdots + h_{C-1,k}\lambda_z^{C-1} \quad (36)$$

where C reprents the polynomial order, and $k = 0, 1, \cdots, K-1$ with K being the number of bands. Then the elements $\mathcal{V}_{h,2}^A f(i,k), i = 0, 1, \cdots, N-1$ of the VGLCT matrix $\mathbf{V}_{h,2}^A$ can be written in a matrix form

$$\begin{aligned} \mathbf{v}_{h,2}^{k,A} &= IGLCT\{\hat{f}_A(z) H_k(\lambda_z)\} = \mathbf{Q}\mathcal{K}^{-1}\mathbf{P}^H\Upsilon^{-1}\hat{\mathbf{f}}_A H_k(\mathbf{\Lambda}_A) \\ &= \mathbf{Q}\mathcal{K}^{-1}\mathbf{P}^H\Upsilon^{-1} H_k(\mathbf{\Lambda}_A)\Upsilon\mathbf{P}\mathcal{K}\mathbf{Q}^H\mathbf{f} \\ &= H_k(\mathcal{L}_A)\mathbf{f} = \sum_{m=0}^{C-1} h_{m,k}\mathcal{L}_A^m\mathbf{f} \end{aligned} \quad (37)$$

where $\mathbf{v}_{h,2}^{k,A}, k = 1, \cdots, K-1$ is the column vector with elements $\mathcal{V}_{h,2}^A f(i,k)$, $i = 0, 1, \cdots, N-1$, and $H_k(\mathbf{\Lambda}_A)$ is a diagonal matrix with elements $H_k(\lambda_z), z = 1, 2, ..., N$. In this scenario, the number of shifted windows, K, is not related to the total number of indices N.

Based on this formulation of the VGLCT, we can further define two specific vertex-frequency analysis methods: the graph linear canonical wavelet transform (GLWT) and the graph linear canonical stockwell transform (GLST).

1) Graph linear canonical wavelet trasform: In classical signal processing theory, time-frequency analysis and wavelet transform share common objectives. However, these two domains are typically treated independently. Classic time-frequency analysis aims to explore the frequency and time characteristics of non-stationary signals. This approach finds widespread application in signal processing, communication,

and audio analysis for tasks like signal analysis, parameter estimation, detection, and denoising. On the other hand, wavelet analysis offers a multi-resolution approach that captures both local and global signal features. Through detailed and approximate decomposition of the signal, wavelet analysis enables tasks such as signal compression, denoising, and feature extraction.

In graph signal processing, it is often assumed that these two signal processing methods are equivalent. Therefore, only the GLWT, which is directly linked to frequency variation (VGLCT), is considered. It can be seen as a specific instance of vertex-frequency analysis, rather than a transform for graph signal compression and its wavelet-like multi-resolution analysis.

Linear canonical wavelet operators. Assuming a band-pass filter behavior, the wavelet definition in the spectral domain can be represented by $H(\lambda_z)$ as the basic form. We define the linear canonical wavelet operator at scale s_j as

$$\widehat{T_H^{s_j} f}(z) = H(s_j \lambda_z) \hat{f}(z) \tag{38}$$

where $j = 1, \cdots, J$. This operator is achieved by applying each linear canonical mode to a given function f.

Employing the IGLCT on (**??**) yields

$$T_H^{s_j} f(z) = \sum_{z=0}^{N-1} \sum_{l=0}^{N-1} H(s_j \lambda_z) \hat{f}(z) q_l(n) \kappa_l^* p_l^*(z) \epsilon_z^*. \tag{39}$$

The linear canonical wavelets are then realized through localizing it by applying it to the impulse on a single vertex

$$\psi_{s_j,i}(n) = T_H^{s_j} \delta_i(n) = \sum_{z=0}^{N-1} \sum_{l=0}^{N-1} H(s_j \lambda_z) \times \left(\sum_{r=0}^{N-1} \epsilon_k p_r(k) \kappa_r q_r^*(i) \right) q_l(n) \kappa_l^* p_l^*(z) \epsilon_z^*. \tag{40}$$

The GLWT is defined as

$$\begin{aligned} \mathcal{W}_\psi^A(i, s_j) &= \sum_{n=0}^{N-1} f(n) \psi_{s_j,i}(n) \\ &= \sum_{n=0}^{N-1} \sum_{p=0}^{N-1} \sum_{l=0}^{N-1} f(n) H(s_j \lambda_z) \left(\sum_{r=0}^{N-1} \epsilon_k p_r(k) \kappa_r q_r^*(i) \right) \\ &\quad \times q_l(n) \kappa_l^* p_l^*(z) \epsilon_z^* \\ &= \sum_{p=0}^{N-1} \sum_{l=0}^{N-1} H(s_j \lambda_z) \hat{f}_A(z) q_l(n) \kappa_l^* p_l^*(z) \epsilon_z^*. \end{aligned} \tag{41}$$

Example 8. *We use wavelet generation kernels similar to Meyer wavelets [**?**], defined as*

$$H(\lambda) = \begin{cases} \sin\left(\frac{\pi}{2} \nu\left(\frac{1}{\lambda_1}|\lambda| - 1\right)\right) & \text{if } \lambda_1 \le \lambda \le \lambda_2 \\ \cos\left(\frac{\pi}{2} \nu\left(\frac{1}{\lambda_2}|\lambda| - 1\right)\right) & \text{if } \lambda_2 \le \lambda \le \lambda_3 \end{cases} \tag{42}$$

*where $v(x) = x^4(35 - 84x + 70x^2 - 20x^3)$, and $\lambda_1 = 2/3$, $\lambda_2 = 2\lambda_1$, $\lambda_3 = 4\lambda_1$. The J wavelet scales are defined as $s_j = 2^j \lambda_{max}^{-1}$ for $j = 1, \cdots, J$, where λ_{max} is obtained by the linear caonical Laplacian matrix (**??**). To handle the low-pass spectral components (the interval for λ closest to $\lambda = 0$), the low-pass type scale function, $G(\lambda)$ is added in the form*

$$G(\lambda) = \begin{cases} 1 & \text{if } \lambda \le \lambda_1 \\ \cos\left(\frac{\pi}{2} \nu\left(\frac{1}{\lambda_1}|\lambda| - 1\right)\right) & \text{if } \lambda_1 \le \lambda \le \lambda_2 \end{cases}. \tag{43}$$

*Let $J = 4$, the generated linear regular wavelet kernel is shown in Fig. **??**. (a). Assuming values of $\sigma = 1$, $\xi = 0$ and $\beta = 0.98$, we can derive four scales: $s_1 = 1.693, s_2 = 0.846, s_3 = 0.423$, and $s_4 = 0.212$. By centering the wavelet on vertex 90, we can compute it and visualize it in Fig. **??** (b)-(e).*

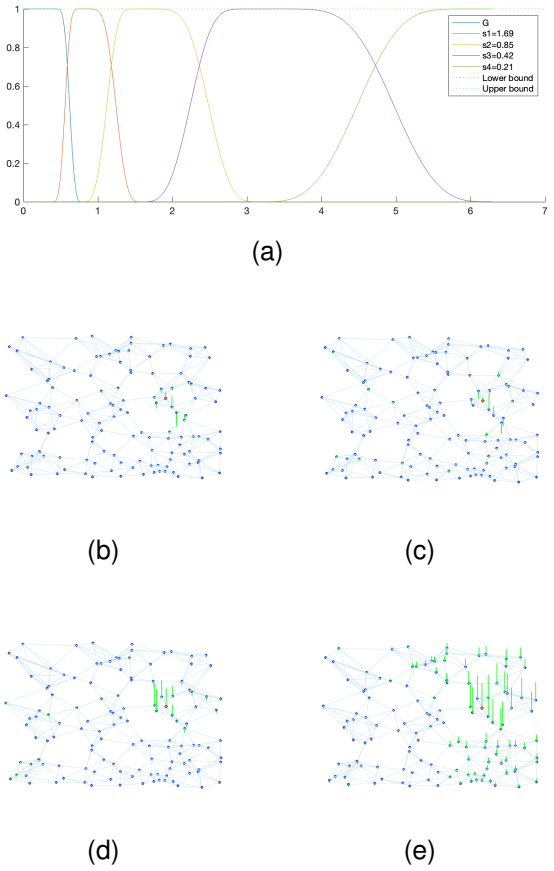

Fig. 6. Graph linear canonical wavelet on Fig. **??**. The red dot represents the selected vertex $i = 90$, while the green line represents the magnitude of the function value. (a) Wavelet kernels $H(s_j \lambda_z)$. (b) $s_1 = 1.693, i = 90$. (c) $s_2 = 0.846, i = 90$. (d) $s_3 = 0.423, i = 90$. (e) $s_4 = 0.212, i = 90$.

2) Graph linear canonical Stockwell trasform: The Stockwell transform is a time-frequency analysis technique that shares similarities with both the wavelet transform and the windowed Fourier transform. The S-transform can be considered as an extension of the WT and WFT, offering a variable time-frequency resolution by using a variable-sized Gaussian window that adapts to the local characteristics of the signal.

Graph linear canonical modulation operator. For any signal $f(n)$ defined on the graph $\mathcal{G}$ and any $k = 0, 1, \cdots, N-1$, define the graph linear canonical modulation M_k^A by

$$M_k^A f(n) = \sum_{l=0}^{N-1} f(n) q_l(n) \kappa_l^* p_l^*(k) \epsilon_k^*. \tag{44}$$

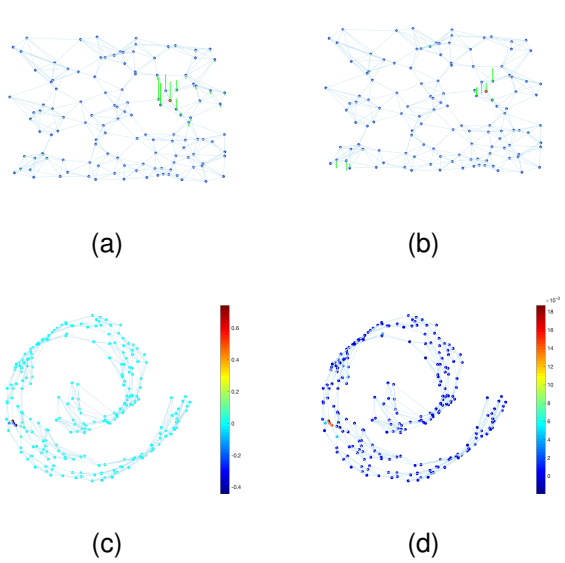

Fig. 7. (a)The GLWT kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (a). (b)The GLST kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (a). (c)The GLWT kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (b). (d)The GLST kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (b).

The graph linear canonical Stockwell trasform can be defined as modulation of the GLWT [**?**]. For $s = 1$, denote its corresponding GLWT as $\mathcal{W}_{\psi}^{A}(i, s_j)$, the GLST is defined as

$$\begin{aligned}\mathcal{S}_h^A(i,s_j) &= \sum_{l=0}^{N-1} \mathcal{W}_h^A(i,s_j) q_l(i)\kappa_l^* p_l^*(k)\epsilon_k^* \\ &= \sum_{l=0}^{N-1}\sum_{z=0}^{N-1}\sum_{r=0}^{N-1} H(s_j\lambda_z)\hat{f}_A(p) q_r(n)\kappa_r^* p_r^*(z)\epsilon_z^* \\ &\quad \times q_l(i)\kappa_l^* p_l^*(k)\epsilon_k^*. \end{aligned} \tag{45}$$

Example 9. *Briefly compare the differences in kernel functions between GLWT and GLST. Utilizing the basis function provided in Example* **??**, *set the linear canonical parameter uniformly to* $\sigma = 1$, $\xi = 0$, $\beta = 0.9$ *and select vertex 30. The resulting figure illustrates the outcome (Fig.* **??**).

Example 10. *Considering a path graph with trivial edge weights, we examine the signal* $f_3 = \sin(60\pi n/N)$, *where* $1 \le n \le 256$. *This signal represents a sampled sinusoidal waveform with a single frequency. Continuing with the utilization of the wavelet from Example* **??**, *we set the parameters as* $J = 4$, $\sigma = 0.5$, $\xi = 0$, *and* $\beta = 1.04$. *As a result, we obtain four scales:* $s_1 = 1.819$, $s_2 = 0.909$, $s_3 = 0.455$, *and* $s_4 = 0.227$. *Figure 6 depicts the GLCT and GLST of the signal* f_3 *at scale* $s_3 = 0.455$.

3) Chebyshev polynomial approximation: When employing a bandpass function in the windowing process and directly calculating it, conducting spectral analysis on the complete graph becomes imperative for deriving the basis function. To facilitate efficient computation, this section presents a polynomial approximation method grounded in the bandpass function $H_k(\lambda)$. This method enables vertex-frequency analysis using solely local signals and graph connections, thereby enhancing computational speed.

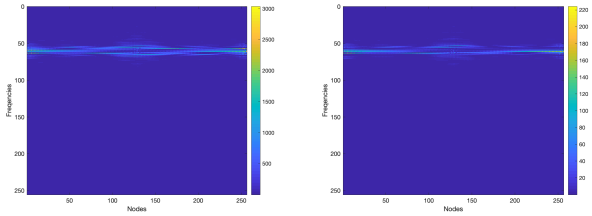

Fig. 8. (a) The GLWT of signal f_3. (b) The GLST of signal f_4.

The discrete points $\lambda = \lambda_z$ are employed for the bandpass function $H_k(\lambda)$. However, since the polynomial approximation is continuous for $0 \le \lambda \le \lambda_{max}$, the "min-max" Chebyshev polynomial is naturally selected as it exhibits the least deviation from the desired bandpass function. This choice ensures the accuracy of the approximation. When the weight function is $\rho(x) = \frac{1}{\sqrt{1-x^2}}$, the orthogonal polynomial obtained by orthogonalizing the sequence $\{1, x, \cdots, x^m, \cdots\}$ is known as the Chebyshev polynomial, which can be represented as $T_m(x) = \cos(m \arccos x)$, where $|x| \le 1$. The Chebyshev polynomial has the following important recursive relationships

$$\begin{aligned} &T_{m+1}(x) = 2xT_m(x) - T_{m-1}(x),\ m = 1, 2, \cdots \\ &T_0(x) = 1,\ T_1(x) = x. \end{aligned} \tag{46}$$

By employing finite $(M-1)$-order Chebyshev polynomials to approximate the spectral domain localization window in (**??**), i.e., $H_k(\mathcal{L}_A) = \tilde{P}_{k,M-1}(\mathcal{L}_A), k = 0, \cdots, N-1$. The approximation has the form

$$\tilde{P}_{k,M-1}(\lambda) = \frac{c_{k,0}}{2} + \sum_{m=1}^{M-1} c_{k,m}\tilde{T}_m(\lambda) \tag{47}$$

where $\tilde{T}_m(\lambda) = \frac{2\lambda}{\lambda_{max}-1}T_m$ is utilized to map the argument from the interval $0 \le \lambda \le \lambda_{max}$ to the interval $[-1, 1]$. The Chebyshev coefficients are obtained by the Chebyshev polynomial inversion property, that is

$$c_{k,m} = \frac{\pi}{2}\int_{-1}^{1} \frac{\tilde{P}_{k,M-1}(\lambda)\tilde{T}_m(\lambda)}{\sqrt{1-\lambda^2}} \mathrm{d}\lambda. \tag{48}$$

This implementation is particularly noteworthy when the graph represents large-scale data.

Example 11. *Considering the path graph and utilizing the signal* f_3 *from Example* **??** *along with the wavelet mentioned in Example* **??**, *we proceed with the following parameters:* $J = 4$, $\sigma = 1$, $\xi = 0$, *and* $\beta = 0.98$. *By applying these parameters, we obtain six scales:* $s_1 = 8.611$, $s_2 = 4.306$, $s_3 = 2.153$, $s_4 = 1.0764$, $s_5 = 0.538$, *and* $s_6 = 0.269$. *We obtained six filter kernels at different scales, as depicted in Fig.* **??**. *(a). Subsequently, we approximated these wavelet kernels using a 30th order Chebyshev polynomial. The resulting approximation is presented in Fig.* **??**. *(b). Fig.* **??**. *(c) illustrates the vertex-frequency representation results obtained using the original filter at scale* $s_4 = 1.0764$, *while Fig.* **??**. *(d) displays*

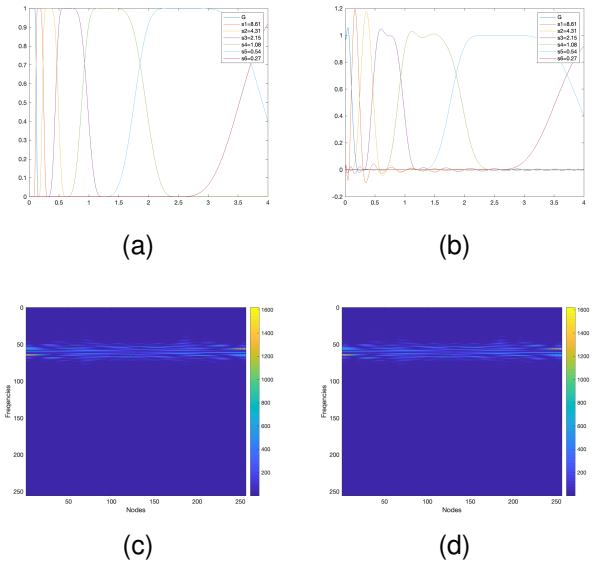

Fig. 9. (a)The GLWT kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (a). (b)The GLST kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (a). (c)The GLWT kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (b). (d)The GLST kernel with $s_3 = 0.423, i = 90$ on Fig. **??**. (b).

the corresponding results obtained using the approximation filter.

C. Localized graph linear canonical transform

The methods mentioned above all operate in the linear canonical domain of the graph. In this section, we discuss a vertex domain localization method denoted as localized graph linear canonical transform (LGLCT). Let $d_{n,i}$ represent the length of the shortest path from vertex i to vertex n. We then define the localized window function as $h(d_{n,i})$, where $h(\cdot)$ represents any basic window function commonly used in classical signal processing. For instance, the Blackman window can be employed [**?**], which is defined as follows

$$h(d_{n,i}) = 0.42 - 0.5cos\left(\frac{2\pi d_{n,i}}{\eta}\right) + 0.08cos\left(\frac{4\pi d_{n,i}}{\eta}\right) \tag{49}$$

where η is the assumed window width.

The distance $d_{n,i}$ can be calculated using the adjacency matrix $\mathbf{A}$. In an unweighted graph, a vertex belonging to a neighborhood of vertex i is represented by a unit value element in row i of the adjacency matrix $\mathbf{A}$. In the case of a weighted graph, the adjacency matrix $\mathbf{A}$ can be derived from the weighted matrix $\mathbf{W}$ by taking the sign of each element, i.e., $\mathbf{A} = \text{sign}(\mathbf{W})$. Then we define the following matrix form

$$\mathbf{A}_{d_{n,i}} = \begin{cases} \mathbf{A} & d_{n,i} = 1 \\ (\mathbf{A} \odot \mathbf{A}_{d_{n,i}-1}) \circ (\mathbf{1} - \mathbf{A}_{d_{n,i}-1}) \circ (\mathbf{1} - \mathbf{I}) & d_{n,i} \geq 2 \end{cases} \tag{50}$$

where $\odot$ represents the logical (Boolean) matrix product, $\circ$ denotes the Hadamard (element-by-element) product, and $\mathbf{1}$ is a matrix with all elements equal to 1. The i-th row of the matrix $(\mathbf{A} \odot \mathbf{A}d_{n,i} - 1)$ provides information about all vertices

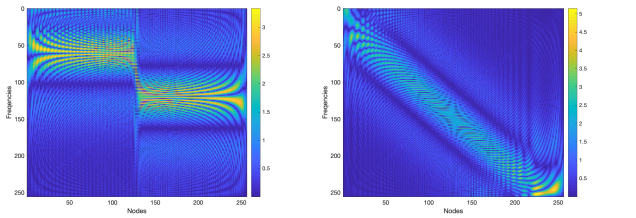

Fig. 10. The LGLCT of graph signals. (a) The LGLCT of signal f_4. (b) The LGLCT of signal f_5.

connected to vertex i through walks of length $K = d_{n,i}$ or lower. It is important to note that the element-wise multiplication of $(\mathbf{A} \odot \mathbf{A}d_{n,i} - 1)$ by the matrix $(\mathbf{1} - \mathbf{A}d_{n,i} - 1)$ removes the vertices connected by walks of length $d_{n,i} - 1$, while the multiplication by $(\mathbf{1} - \mathbf{I})$ eliminates the diagonal elements [**?**].

For a window function $\mathbf{h} = [h(0), \cdots, h(N-1)]^T \in \mathbb{R}^N$, the localized window function (denoted as $\mathbf{T_A h} = [T_{\mathbf{A}} h(n,i)]_{N-1 \times N-1}$) with assumed graph window width η can be given by

$$\mathbf{T_A h} = h(0)\mathbf{I} + h(1)\mathbf{A}_1 + \cdots + h(\eta-1)\mathbf{A}_{\eta-1}. \tag{51}$$

The LGLCT of a graph signal $f(n)$, then becomes

$$\mathcal{V}^A_{h,3} f(i,k) = \sum_{n=0}^{N-1} \sum_{l=0}^{N-1} f(n) T_{\mathbf{A}} h(n,i) \epsilon_k p_l(k) \kappa_l q_l^*(n). \tag{52}$$

Example 12. *Given a path graph with trivial edge weights, we consider the following two time series:* $f_4 = \begin{cases} \sin(60\pi n/N), & 1 \leq n \leq 128 \\ \sin(90\pi n/N), & 129 < n \leq 256 \end{cases}$ *is a sampled sinusoidal signal with two different frequencies,* $f_5(n) = \sin\left(\left(20n + 0.5n^2\right)\pi/N\right), 1 \leq n \leq 256$ *is a chirp signal. The LGLCT of signals* f_4 *and* f_5 *is depicted in Fig.* **??**.

V. Filter Design

Within a basic veryex-frequency analysis framework, different filter selections can correspond to various transformations. The selection of an appropriate filter is crucial in practical applications. For instance, different spectral graph neural networks mainly rely on distinct filter learning approaches [**?**], [**?**], [**?**]. Therefore, in this section, we will initially explore optimization learning methods for filters and subsequently apply them to image classification tasks.

A. Optimal Filter

In Section **??**, the graph linear canonical convolution operator was presented as $\mathbf{f} *_A \mathbf{h} = \mathcal{F}_A^H (\mathcal{F}_A \mathbf{f} \odot \mathcal{F}_A \mathbf{h})$, as shown in (**??**), where $\odot$ reprents a point-wise product. In general, designing filters on a graph involves designing a diagonal matrix $\mathbf{H}(\mathbf{\Lambda}_A) = \text{diag}(\hat{\mathbf{h}})$, where $\hat{\mathbf{h}} = [h_1, h_2, \cdots, h_N]^\top$.

Then, the graph linear canonical convolution operator can be rewritten as

$$\mathbf{f} *_A \mathbf{h} = \mathcal{F}_A^H \mathbf{H}(\mathbf{\Lambda}_A)\mathcal{F}_A\mathbf{f} = \mathbf{H}(\mathcal{L}_A)\mathbf{f}. \tag{53}$$

It follows that the signal $\mathbf{f}$ is filtered by $\mathbf{H}(\mathcal{L}_A)$.

Remark 1. *Extending convolution to inputs $\mathbf{f}$ with multiple input channels is a straightforward process. If $\mathbf{f}$ is a signal with M input channels and N locations, we apply the GLCT $\mathcal{F}_A$ to each channel independently. Then, to perform the convolution, we utilize multipliers $\hat{\mathbf{h}} = (h_{i,j}; i \leq N, j \leq M)$.*

Consider an input signal $\mathbf{f}$, where the actual output signal is represented as

$$\mathbf{g} = \mathbf{G}\mathbf{f} + \mathbf{n}, \tag{54}$$

where $\mathbf{G}$ is a known matrix, $\mathbf{n}$ is the additive noise term, and $\mathbf{f}$ is a stochastic graph signal [**?**].

Our next objective is to learn an optimized filter that minimizes the loss function while predicting the output signal $\widetilde{\mathbf{g}}$. We define the squared loss function as $R(\mathbf{H}) = \frac{1}{N}\|\mathbf{H}(\mathcal{L}_A)\mathbf{f} - \mathbf{g}\|_F^2$. Therefore, our objective is to design the matrix $\mathbf{H}(\mathcal{L}_A)$ to solve the following optimization problem

$$\min_{\mathbf{H}} \quad \frac{1}{N}\|\mathbf{H}(\mathcal{L}_A)\mathbf{f} - \mathbf{g}\|_F^2 \tag{55}$$

for any A. Subsequently, we will explore different possible values of A to identify A^* that minimizes the loss.

Theorem 1. *The problem defined in (**??**) will yield identical loss values for any A if there are no constraints on $\mathbf{H}(\mathcal{L}_A)$.*

Proof. Consider any parameters A_1 and A_2 such that $A_1 \neq A_2$, and define

$$\mathbf{H}(\mathbf{\Lambda}_{A_j}) = \arg\min_{\mathbf{H}} \left\{ \left\| \mathbf{H}(\mathcal{L}_{A_j})\mathbf{f} - \mathbf{g} \right\|_F^2 \right\} \tag{56}$$

where $j = 1, 2$. We note that for any choice of $\mathbf{H}(\mathbf{\Lambda}_{A_j})$, according to $j = 1$ in (**??**), we have

$$\left\| \mathcal{F}_{A_1}^H \mathbf{H}(\mathbf{\Lambda}_{A_1})\mathcal{F}_{A_1}\mathbf{f} - \mathbf{g} \right\|_F^2 \leq \left\| \mathcal{F}_{A_1}^H \mathbf{H}(\mathbf{\Lambda})\mathcal{F}_{A_1}\mathbf{f} - \mathbf{g} \right\|_F^2.$$

Then, by choosing $\mathbf{H}(\mathbf{\Lambda}) = \mathcal{F}_{A_1}\mathbf{H}(\mathcal{L}_{A_2})\mathcal{F}_{A_1}^H$, we can obtain from $j = 2$ in (**??**) that

$$\left\| \mathcal{F}_{A_1}^H \mathbf{H}(\mathbf{\Lambda}_{A_1})\mathcal{F}_{A_1}\mathbf{f} - \mathbf{g} \right\|_F^2 \leq \left\| \mathcal{F}_{A_2}^H \mathbf{H}(\mathbf{\Lambda}_{A_2})\mathcal{F}_{A_2}\mathbf{f} - \mathbf{g} \right\|_F^2.$$

Similarly, it can be easily shown that

$$\left\| \mathcal{F}_{A_2}^H \mathbf{H}(\mathbf{\Lambda}_{A_2})\mathcal{F}_{A_2}\mathbf{f} - \mathbf{g} \right\|_F^2 \leq \left\| \mathcal{F}_{A_1}^H \mathbf{H}(\mathbf{\Lambda}_{A_1})\mathcal{F}_{A_1}\mathbf{f} - \mathbf{g} \right\|_F^2.$$

Therefore, we can conclude that the same minimum value is achieved for any A_1 and A_2. □

By deriving the objective function in (**??**), we can identify the optimal solution. The subsequent theorem delineates the solution to the optimal filtering problem stated in (**??**).

Theorem 2. *For the problem in (**??**), the optimal filter coefficients $\hat{\mathbf{h}}^{opt} = [h_1^{opt}, \cdots, h_N^{opt}]^\top$ are obtained from the following matrix equation*

$$(\mathcal{F}_A\mathbf{f} \odot \mathcal{F}_A\mathbf{f})\mathbf{1}_N \odot \hat{\mathbf{h}}^{opt} = (\mathcal{F}_A\mathbf{f} \odot \mathcal{F}_A\mathbf{g})\mathbf{1}_N. \tag{57}$$

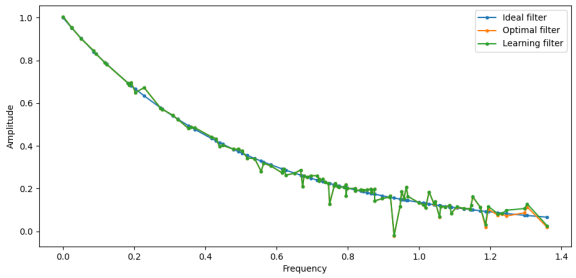

Fig. 11. Ideal Filter, optimal filter and learning filter.

Proof. Consider a fixed parameter A, the squared loss function is expressed as $R = \frac{1}{N}\|\mathcal{F}_A^H\mathbf{H}(\mathbf{\Lambda})\mathcal{F}_A\mathbf{f} - \mathbf{g}\|_F^2$. We rewrite it in the spectral domain as

$$\hat{R} = \frac{1}{N}\|\mathbf{H}(\mathbf{\Lambda})\mathcal{F}_A\mathbf{f} - \mathcal{F}_A\mathbf{g}\|_F^2. \tag{58}$$

Taking the derivative of $\hat{R}$ with respect to $\hat{\mathbf{h}}$, we obtain

$$\nabla_{\hat{\mathbf{h}}}\hat{R} = \frac{2}{N}\left(\mathcal{F}_A\mathbf{f} \odot (\hat{\mathbf{h}} \odot \mathcal{F}_A\mathbf{f} - \mathcal{F}_A\mathbf{g})\right)\mathbf{1}_N. \tag{59}$$

Let $\nabla_{\hat{\mathbf{h}}}\hat{R} = 0$ and $\hat{\mathbf{h}}^{\text{opt}} = [h_1^{\text{opt}}, \cdots, h_N^{\text{opt}}]^\top$, we obtain

$$(\mathcal{F}_A\mathbf{f} \odot \mathcal{F}_A\mathbf{f})\mathbf{1}_N \odot \hat{\mathbf{h}}^{\text{opt}} = (\mathcal{F}_A\mathbf{f} \odot \mathcal{F}_A\mathbf{g})\mathbf{1}_N. \tag{60}$$

□

Example 13. *Consider the graph structure depicted in Example **??**, where the spectral content of the graph signal length on each vertex follows a normal distribution, i.e., $\hat{\mathbf{f}}_6(k) \sim \mathcal{N}(e^{-1.5*\lambda_k}, (e^{-1.5*\lambda_k}/2)^2)$. Thus, the graph signal $\mathbf{f}_6$ is a matrix of size 100 × 200, where $\mathbf{f}_6(n) = \mathcal{F}_A^H\hat{\mathbf{f}}_6(k)$ denotes the signal on the nth vertex, a vector with a length of 200. Let the parameters of GLCT be set to $\sigma = 1.5$, $\xi = 0.5$, $\beta = 0.95$. The ideal filter is defined as $h_k = e^{-2\lambda_k}$, where $k = 0, \cdots, N-1$, and the noise term $\mathbf{n}$ follows a normal distribution $\mathbf{n} \sim \mathcal{N}(0, 0.1^2)$. Consequently, the actual output signal is given by $\mathbf{g} = \mathcal{F}_A^H\mathbf{H}(\mathbf{\Lambda})\mathcal{F}_A\mathbf{f}_6 + \mathbf{n}$. According to Theorem **??**, the optimal filter design result can be derived. Figure **??** illustrates the obtained optimal filter result depicted by the orange curve, while the blue curve represents the ideal filter.*

B. Learning Filter

In the aforementioned optimal filter design method, the gradient of the loss function constitutes a non-convex function, rendering it challenging to guarantee the discovery of the global optimal solution. To accommodate diverse datasets and problem scenarios more flexibly, optimization algorithms like gradient descent can be employed for filter learning. This is particularly pertinent when dealing with more intricate models or elevated performance criteria. Furthermore, alongside the fundamental gradient descent method, numerous enhanced optimization algorithms exist, including stochastic gradient descent (SGD), the momentum method, Adam, Adagrad, and others. These algorithms offer various advantages tailored to

different problem domains, aiding in expediting convergence and enhancing algorithmic performance.

Here, we introduce a method for learning filters via SGD, which serves as the groundwork for future advancements in spectrogram neural networks, as shown in Algorithm **??**.

Algorithm 2 SGD-based filter learning algorithm

Input:

Input graph signal: $\mathbf{f}$, Target signal: $\mathbf{g}$.
Initial value of filter: $\mathbf{H}^0$.
The parameters of GLCT: $A = (\xi, \sigma, \beta)$.
Learning rate: ϵ, Size of the mini-batch: N_{mini}, Stopping criterion, Maximum number of iterations.

Output:

The learned filter: $\mathbf{H}^{\text{sgd}} = \text{diag}(\hat{\mathbf{h}}^{\text{sgd}})$.
Loss function value: $\hat{R}(\mathbf{H}^{\text{sgd}})$.

1: Compute $\mathcal{F}_A = \Upsilon \mathbf{P} \mathcal{K} \mathbf{Q}^H$.
2: Compute the loss function in (**??**) of the initial filter: $\hat{R}(\mathbf{H}^0)$.
3: Let $\mathbf{H}^{\text{sgd}} = \mathbf{H}^0$.
4: **while** the stopping criteria are not met **do** :
Retrieve a mini-batch of samples with a length of N_{mini} from the training set.
Calculate the gradient of the loss function: $\nabla_{\hat{\mathbf{h}}^{\text{sgd}}} \hat{R} = \frac{2}{N} \left(\mathcal{F}_A \mathbf{f} \odot (\hat{\mathbf{h}}^{\text{sgd}} \odot \mathcal{F}_A \mathbf{f} - \mathcal{F}_A \mathbf{g}) \right) \mathbf{1}_N$.
Filter update: $\hat{\mathbf{h}}^{\text{sgd}} = \hat{\mathbf{h}}^{\text{sgd}} - \epsilon \nabla_{\hat{\mathbf{h}}^{\text{sgd}}} \hat{R}$.
end while
5: Compute the loss function in (**??**) of the learned filter: $\hat{R}(\mathbf{H}^{\text{sgd}})$.

Example 14. *Continuing with the consideration of the graph signal* $\mathbf{f}$ *and target signal* $\mathbf{g}$ *in Example* **??**, *we set the initial filter to a random sequence following a standard distribution, i.e.,* $\hat{\mathbf{h}}^0 \sim \mathcal{N}(0, 1)$. *The parameters of GLCT are set to* $\sigma = 1.5$, $\xi = 0.5$, *and* $\beta = 0.95$, *with a learning rate of* $\epsilon = 0.6$, *a mini-batch size of* $N_{mini} = 100$, *a stopping criterion of* 0.001, *and a maximum number of iterations of* 100. *The final learned filter is depicted in Figure 10, where the green curve represents the result.*

C. Image Classification based on designed filter

In addition to the direct utilization of advanced filters [**?**], [**?**], many researchers also employ polynomial form filters, which are extended in polynomial form (e.g., Chebyshev polynomial expansion as discussed in Section **??**). Polynomial filters offer the advantages of achieving locality and reducing learning complexity. Several notable examples include CheyNet [**?**], GPRGNN [**?**], JacobiConv [**?**], among others.

In this section, we employ the filter representation of Chebyshev polynomials. Subsequently, leveraging this filter formulation, we construct a fundamental linear learner for image classification described as $\mathbf{H}(\mathbf{\Lambda}) = \sum_{m=0}^{M-1} c_{k,m} \tilde{T}_m(\mathbf{\Lambda})$, where $c_{k,m}$ represents the polynomial coefficient, as elucidated in (**??**). This coefficient ensemble forms the essential parameter set for training the learner layer. To augment the fitting capability of the learner, a parameterized weight matrix Θ is introduced to perform radial transformations on the input graph signal matrix, yielding $\widetilde{\mathbf{g}} = \mathcal{F}_A^H \mathbf{H}(\mathbf{\Lambda}) \mathcal{F}_A \mathbf{f} \Theta$. The squared loss function in spectral domain is expressed as $\hat{R} = \frac{1}{N} \|\mathbf{H}(\mathbf{\Lambda}) \mathcal{F}_A \mathbf{f} \Theta - \mathcal{F}_A \mathbf{g}\|_F^2$. Subsequently, the filter undergoes training using Algorithm **??** to address image classification tasks.

To assess the validity of our model, we applied it to the Euclidean scenario using the benchmark MNIST classification problem [**?**], which comprises a dataset of 70,000 digits depicted on a 2D grid measuring 28 × 28. We specifically focused on discriminating between the challenging digits 4 and 9, each comprising 6,824 samples, totaling 13,648 instances. For our graph-based model, we constructed a 4-NN graph from the 2D grid, resulting in a graph with $N = \|\mathcal{V}\| = 784$ nodes. The weights of the K-NN similarity graph were computed using $w_{i,j} = \exp\left(-\frac{\|z_i - z_j\|_2^2}{\epsilon^2}\right)$, where z_i denotes the 2D coordinate of pixel i. For the dataset comprising 13,648 samples, 70% were randomly selected for training, while the remaining 30% were allocated to the test set.

Example 15. *Let the parameters of GLCT be* $\sigma = 0.5$, $\xi = 0$, *and* $\beta = 1.1$, *with a learning rate of* $\epsilon = 0.001$. *The degree of the Chebyshev polynomial is* $M = 10$. *The accuracy achieved on the training set is 95.15%, while the accuracy on the test set is 93.58%.*

Example 16. *We conducted tests on the varying outcomes of the linear canonical parameter amidst minor fluctuations, while keeping all other parameters constant (change* σ *and* β, *, keeping* $\xi = 0$ *fixed). As depicted in Table* **??**, *the highest accuracy was attained when the parameters were* $\sigma = 0.9$, $\xi = 0$, *and* $\beta = 0.5$. *This suggests that fine-tuning the linear canonical parameter can enhance model performance.*

VI. Conclusion

This paper proposes a new graph linear canonical transform (GLCT). Furthermore, we discuss the vertex-frequency analysis framework based on this GLCT, delving into the specific delineations of different vertex-frequency analysis methods defined by various types of filters. Lastly, we examine filter design, encompassing optimal design and filter learning based on stochastic gradient descent, and apply them in image classification. The GLCT proposed in this paper demonstrates promising application prospects, with the filter design section akin to the convolutional layers of spectral graph neural networks (SGNNs), which can potentially be employed in large-scale SGNNs in the future. Subsequently, we will delve deeper into exploring the impact of linear canonical parameters to ensure their widespread applicability in practice.

Acknowledgments

This work was funded by the National Natural Science Foundation of China [No. 62171041], and the Natural Science Foundation of Beijing Municipality [No. 4242011].

TABLE I
PREDICTION RESULTS OF TEST SET WITH DIFFERENT PARAMETERS (%)

	$\sigma = 0.5$	$\sigma = 0.6$	$\sigma = 0.7$	$\sigma = 0.8$	$\sigma = 0.9$	$\sigma = 1$	$\sigma = 1.1$	$\sigma = 1.2$	$\sigma = 1.3$	$\sigma = 1.4$
$\beta = 0.5$	93.16	92.9	93.78	93.32	**94.09**	93.38	93.54	93.62	93.23	93.67
$\beta = 0.6$	93.19	93.14	93.78	93.58	93.87	93.56	93.25	93.43	93.19	93.19
$\beta = 0.7$	93.32	92.97	93.73	93.47	93.91	93.38	93.52	93.69	92.86	93.65
$\beta = 0.8$	93.27	93.73	93.56	93.32	93.5	92.64	92.94	93.41	93.32	93.73
$\beta = 0.9$	93.36	93.62	93.34	93.43	93.45	91.78	93.16	93.89	93.47	93.36
$\beta = 1$	93.69	93.56	93.76	93.41	93.45	90.83	92.86	93.58	93.45	93.38
$\beta = 1.1$	93.58	93.52	93.23	93.25	93.34	91.65	92.75	93.73	93.25	93.19
$\beta = 1.2$	93.56	93.45	93.23	93.38	93.43	92.75	93.08	94.08	93.41	93.21
$\beta = 1.3$	93.67	93.16	93.25	93.32	93.47	92.5	92.81	94.06	92.57	93.23
$\beta = 1.4$	93.36	93.23	93.43	93.95	93.71	93.34	93.34	93.8	92.64	93.47

REFERENCES

[1] A. Sandryhaila and J. M. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE signal processing magazine*, vol. 31, no. 5, pp. 80–90, 2014.

[2] ——, "Discrete signal processing on graphs," *IEEE transactions on signal processing*, vol. 61, no. 7, pp. 1644–1656, 2013.

[3] A. Ortega, P. Frossard, J. Kovačević, J. M. Moura, and P. Vandergheynst, "Graph signal processing: Overview, challenges, and applications," *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[4] D. I. Shuman, "Localized spectral graph filter frames: A unifying framework, survey of design considerations, and numerical comparison," *IEEE Signal Processing Magazine*, vol. 37, no. 6, pp. 43–63, 2020.

[5] D. Shuman, S. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emmerging field of signal processing on graphs," *IEEE Signal Proc. Magazine*, 2013.

[6] A. Sandryhaila and J. M. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Transactions on signal processing*, vol. 62, no. 12, pp. 3042–3054, 2014.

[7] R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.

[8] L. Stanković, M. Daković, and E. Sejdić, "Vertex-frequency analysis: A way to localize graph spectral components [lecture notes]," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 176–182, 2017.

[9] L. Stanković, D. Mandic, M. Daković, B. Scalzo, M. Brajović, E. Sejdić, and A. G. Constantinides, "Vertex-frequency graph signal processing: A comprehensive review," *Digital signal processing*, vol. 107, p. 102802, 2020.

[10] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "A windowed graph fourier transform," in *2012 IEEE Statistical Signal Processing Workshop (SSP)*. Ieee, 2012, pp. 133–136.

[11] ——, "Vertex-frequency analysis on graphs," *Applied and Computational Harmonic Analysis*, vol. 40, no. 2, pp. 260–291, 2016.

[12] L. Stanković and E. Sejdić, *Vertex-frequency analysis of graph signals*. Springer, 2019.

[13] S. C. Pei and J. J. Ding, "Eigenfunctions of linear canonical transform," *IEEE Transactions on Signal Processing*, vol. 50, no. 1, pp. 11–26, 2002.

[14] J. J. Healy, M. A. Kutay, H. M. Ozaktas, and J. T. Sheridan, *Linear canonical transforms: Theory and applications*. Springer, 2016, vol. 198.

[15] S. C. Pei and J. J. Ding, "Relations between fractional operations and time-frequency distributions, and their applications," *IEEE Transactions on Signal Processing*, vol. 49, no. 8, pp. 1638–1655, 2001.

[16] Y. Q. Wang, B. Z. Li, and Q. Y. Cheng, "The fractional fourier transform on graphs," in *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2017, pp. 105–110.

[17] Y. Zhang and B. Z. Li, "Discrete linear canonical transform on graphs," *Digital Signal Processing*, vol. 135, p. 103934, 2023.

[18] J. Wu, F. Wu, Q. Yang, Y. Zhang, X. Liu, Y. Kong, L. Senhadji, and H. Shu, "Fractional spectral graph wavelets and their applications," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–18, 2020.

[19] F. J. Yan and B. Z. Li, "Windowed fractional fourier transform on graphs: Properties and fast algorithm," *Digital Signal Processing*, vol. 118, p. 103210, 2021.

[20] R. A. Brown, M. L. Lauzon, and R. Frayne, "A general description of linear time-frequency transforms and formulation of a fast, invertible transform that samples the continuous s-transform spectrum nonredundantly," *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 281–290, 2009.

[21] D. Y. Wei, Y. J. Zhang, and Y. M. Li, "Linear canonical stockwell transform: theory and applications," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1333–1347, 2022.

[22] D. Y. Wei and Y. M. Li, "Generalized wavelet transform based on the convolution operator in the linear canonical transform domain," *Optik*, vol. 125, no. 16, pp. 4491–4496, 2014.

[23] K. I. Kou and R. H. Xu, "Windowed linear canonical transform and its applications," *Signal Processing*, vol. 92, no. 1, pp. 179–188, 2012.

[24] X. Chen, "Understanding spectral graph neural network," *arXiv preprint arXiv:2012.06660*, 2020.

[25] D. Bo, X. Wang, Y. Liu, Y. Fang, Y. Li, and C. Shi, "A survey on spectral graph neural networks," *arXiv preprint arXiv:2302.05631*, 2023.

[26] X. Wang and M. Zhang, "How powerful are spectral graph neural networks," in *International Conference on Machine Learning*. PMLR, 2022, pp. 23 341–23 362.

[27] C. Ozturk, H. M. Ozaktas, S. Gezici, and A. Koç, "Optimal fractional fourier filtering for graph signals," *IEEE Transactions on Signal Processing*, vol. 69, pp. 2902–2912, 2021.

[28] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," *Advances in neural information processing systems*, vol. 29, 2016.

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

[30] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *arXiv preprint arXiv:1312.6203*, 2013.

[31] R. Liao, Z. Zhao, R. Urtasun, and R. S. Zemel, "Lanczosnet: Multi-scale deep graph convolutional networks," *arXiv preprint arXiv:1901.01484*, 2019.

[32] R. G. Stockwell, L. Mansinha, and R. Lowe, "Localization of the complex spectrum: the s transform," *IEEE transactions on signal processing*, vol. 44, no. 4, pp. 998–1001, 1996.

[33] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.

[34] S. C. Pei and Y. C. Lai, "Discrete linear canonical transforms based on dilated hermite functions," *JOSA A*, vol. 28, no. 8, pp. 1695–1708, 2011.

[35] I. G. Cumming and F. H. Wong, "Digital processing of synthetic aperture radar data," *Artech house*, vol. 1, no. 3, pp. 108–110, 2005.

[36] I. Jestrović, J. L. Coyle, and E. Sejdić, "A fast algorithm for vertex-frequency representations of signals on graphs," *Signal processing*, vol. 131, pp. 483–491, 2017.

[37] N. Leonardi and D. Van De Ville, "Wavelet frames on graphs defined by fmri functional connectivity," in *2011 IEEE International Symposium*

on Biomedical Imaging: From Nano to Macro. IEEE, 2011, pp. 2136–2139.

[38] P. Podder, T. Z. Khan, M. H. Khan, and M. M. Rahman, "Comparative performance analysis of hamming, hanning and blackman window," *International Journal of Computer Applications*, vol. 96, no. 18, pp. 1–7, 2014.