A fast algorithm for vertex-frequency representations of signals on graphs

Iva Jestrović[a], James L. Coyle[b], Ervin Sejdić[a,*]

[a]*Department of Electrical and Computer Engineering, Swanson School of Engineering, University of Pittsburgh, Pittsburgh, PA, USA*

[b]*Department of Communication Science and Disorders, School of Health and Rehabilitation Sciences, University of Pittsburgh, Pittsburgh, PA, USA*

Abstract

The windowed Fourier transform (short time Fourier transform) and the S-transform are widely used signal processing tools for extracting frequency information from non-stationary signals. Previously, the windowed Fourier transform had been adopted for signals on graphs and has been shown to be very useful for extracting vertex-frequency information from graphs. However, high computational complexity makes these algorithms impractical. We sought to develop a fast windowed graph Fourier transform and a fast graph S-transform requiring significantly shorter computation time. The proposed schemes have been tested with synthetic test graph signals and real graph signals derived from electroencephalography recordings made during swallowing. The results showed that the proposed schemes provide significantly lower computation time in comparison with the standard windowed graph Fourier transform and the fast graph S-transform. Also, the results showed that noise has no effect on the results of the algorithm for the fast windowed graph Fourier transform or on the graph S-transform. Finally, we showed that graphs can be reconstructed from the vertex-frequency representations obtained with the proposed algorithms.

Keywords: Graph signal processing, vertex-frequency analysis, windowed graph Fourier transform, graph S-transform.

1. Introduction

Graph theory is a mathematical approach for analyzing the proprieties of data sets that can be represented as graphs [1]. In the graph representation, the vertices (i.e., nodes) refer to objects of interest, and edges describe relationships between the vertices [2]. When applied to data sets, graph theory provides information about the geometric structure of the data and the relationship between data points [3, 4, 5]. For example, relationships between the objects from a given data set can be represented as a weighted undirected graph, where the vertices will describe the position of objects and the weights of the edges will describe similarities between the vertices. Furthermore, a graph representing represents data sets can be modeled such that each

*Corresponding author

Email address: `esejdic@ieee.org` (Ervin Sejdić)

vertex contains samples describing the observation of the object. These samples on vertices represent a signal on a graph [6, 7]. From signals on graphs, a new field emerged for the analysis of these data representations called signal processing on graphs [8, 9, 10, 11, 12, 13].

Signal processing on graphs is an extension of classical signal processing that enables spectral graph analysis, providing deeper insight into graph structures. Spectral graph analysis based on the Laplacian matrix along with its eigenvalues and eigenvectors, is very similar to classical signal processing notation [14, 15]. The advantages of the Laplacian matrix enabled the implementation of basic signal processing operations in graph settings (i.e., the Fourier transform on graphs, signal filtering on graphs, and signal shifting [14, 15]). Signal processing operations on graphs provide advanced graph analysis, such as finding a specific sub-graph in a larger graph [16], detecting very dense or frequently occurring sub-graphs [17, 18], or detecting certain behavioral patterns [19].

In spectral graph analysis, vertex-frequency distributions of the signal on the graph can give important information about structural properties of the graph. In classical signal processing, the windowed Fourier transform is one of the most commonly used tools for extracting time-frequency information from signals [20]. However, the windowed Fourier transform has the limitation of fixed window size that can result in poor time-frequency resolution. The S-transform originates from the windowed Fourier transform with a window size that is dependent on frequency, which overcomes limitations of the fixed window of the windowed Fourier transform method. In one of the previous studies, following the notation from classical signal processing and adopting it for signals on graphs, Shuman et al. [21] developed the windowed graph Fourier transform ($WGFT$) for the extraction of the vertex-frequency content from signals on graphs. As $WGFT$ is also limited by its fixed window size, it would be straightforward to implement the graph S-transform (GST) using the algorithm for $WGFT$ and change the size of the windowed function for the different frequencies, as we have shown in the next section. However, algorithms for the $WGFT$ and the GST have the drawback of high computation complexity, which results in long computation times for larger graphs. This means that potential applications that would use the windowed graph Fourier transform or the graph S-transform would be impractical.

In order to overcome computational burdens, the $WGFT$ and GST could be computed by operating on the Fourier spectrum of the signal and the Fourier spectrum of the window function. This approach has been used previously in classical signal processing for more efficiently calculating the windowed Fourier transform and S-transform of a one-dimensional signal [22]. In classical signal processing, this fast algorithm resulted in a significantly lower computational time than the original approach. Therefore, taking advantage of operating with the signal and window spectra, the fast $WGFT$ and GST could significantly decrease computation complexity and overall computation time.

This paper presents fast algorithms for calculating $WGFT$ and GST; therefore these new techniques are referred to as the fast windowed graph Fourier transform ($FWGFT$) and the fast graph S-transform ($FGST$), respectively. The proposed schemes were tested using synthetic test signals on graphs. Performances are compared with the $FWGFT$ and $WGFT$ algorithms, as well as with the $FGST$ and GST methods. Results showed significant computational improvements for the fast algorithms. These proposed algorithms could prove useful in many applications that use the signal processing on graph analysis.

2. Windowed Fourier transform and S-transform

2.1. Classical signal processing case

The windowed Fourier transform in classical signal processing is a commonly used technique for extracting frequency information from one dimensional signals. Mathematically, the windowed Fourier transform of a function $x \in L^2(\mathbb{R})$ can be written as the inner product of the original signal and the windowed Fourier atom [23]:

$$WFT(\tau, f) := \langle x(t), w_{\tau,f}(t)\rangle = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-j2\pi ft}dt, \tag{1}$$

where the classical windowed Fourier atoms are defined using notation for translation $((T_\tau f)(t) := f(t-\tau))$ and modulation $((M_f x)(t) := xe^{-j2\pi ft})$ as:

$$w_{\tau,f}(t) := (M_f T_\tau w)(t) = w(t-\tau)e^{-j2\pi ft}, \tag{2}$$

where $w(t)$ is the fixed window function. The fixed window in the windowed Fourier transform is the limitation that can influence the resolution of the time-frequency signal representation.

The S-transform overcome the limitations of the fixed window by changing window size for the different frequency points. S-transform is extension of the continuous wavelet transform ($CWT(t, f)$), and can be represented as modulation of the $CWT(t, f)$ [24]:

$$ST(\tau, f) := CWT(t, f)e^{-j2\pi ft} = \int_{-\infty}^{\infty} x(t)g(t-\tau, f)e^{-j2\pi f\tau}dt, \tag{3}$$

where $CWT(t, f)$ is defined as:

$$CWT(t, f) = \int_{-\infty}^{\infty} x(t)g(t-\tau, f)dt. \tag{4}$$

Thus, from the equation (4), the S-atom are defined as:

$$s_{\tau,f}(t) := (M_f CWT(t, f))(t) = g(t-\tau, f)e^{-j2\pi f\tau}, \tag{5}$$

where $g(t, f)$ is a wavelet defined as the frequency dependent function. This means that for higher frequencies the window size will be more narrow. Such a defined window should adjust the time-frequency resolution to provide better frequency resolution for the lower frequencies and better time resolution for the higher frequencies.

2.2. Signals on graphs

The windowed Fourier transform and S-transform can be adopted for signals on graphs using the Laplacian matrix. The Laplacian matrix enables the adjustment of basic classical signal processing operations to graph signals [14, 15]. The (unnormalized) graph Laplacian is defined as $\mathcal{L} := \mathbf{D} - \mathbf{W}$, where $\mathbf{D}$ is a diagonal matrix that contains the degree values of each node from the graph, and $\mathbf{W}$ is the connectivity matrix of the graph. The graph Laplacian $\mathcal{L}$ has a complete set of orthonormal eigenvectors $\{\mathcal{X}_l\}_{l=0,1,...,N-1}$, and real, nonnegative eigenvalues $\{\lambda_l\}_{l=0,1,...,N-1}$ that correspond to each eigenvector. We assume that nonnegative Laplacian eigenvalues are ordered as $0 = \lambda_0 \leq \lambda_1 \leq \lambda_2 ... \leq \lambda_{N-1} := \lambda_{max}$. Eigenvectors that correspond to the lower eigenvalues have smooth oscillations, and eigenvectors that correspond to the higher eigenvalues have more rapid oscillations. Therefore, we can tell that in graph signal processing notation, the eigenvalue will correspond to the frequency of the graph signal.

Since in the classical signal processing the Fourier transform can be represented as the expansion of a signal in terms of its eigenfunctions, we can say that the graph Fourier transform of a signal on the vertices of a graph $x \in \mathbb{R}^N$, can be defined as the expansion of x in terms of the eigenfunctions of the graph Laplacian [25]:

$$\hat{x}(\lambda_l) := \langle x, \mathcal{X}_l \rangle = \sum_{n=1}^{N} x(n)\mathcal{X}_l^*(n). \tag{6}$$

The inverse graph Fourier transform is then given by:

$$x(n) = \sum_{l=1}^{N} \hat{x}(\lambda_l)\mathcal{X}_l(n). \tag{7}$$

Following basic signal processing on graph notation, translation and modulation on graphs can be defined as (derivation provided in [21]):

$$(T_i x)(n) := \sqrt{N}(x * \delta_i)(n) = \sqrt{N} \sum_{l=0}^{N-1} \hat{x}(\lambda_l)\mathcal{X}_l^*(i)\mathcal{X}_l(n), \tag{8}$$

$$(M_k x)(n) := \sqrt{N} x(n)\mathcal{X}_k^*(n). \tag{9}$$

Equipped with the above defined generalized notions of translation and modulation of signals on graphs, we can now define the windowed graph Fourier atom as [21]:

$$W_{n,l}(n) := (M_l T_n w)(n) = N\mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{w}(\lambda_l)\mathcal{X}_l^*(n)\mathcal{X}_l(n), \tag{10}$$

where $\hat{w}$ is the Fourier transform of the window function. For calculating the windowed graph Fourier atoms, we will use a kernel function defined directly in the spectral domain as $\hat{w}(\lambda_l) = Ce^{-\tau\lambda_l}$, where C is chosen such that $||\hat{w}||_2 = 1$. In the case of the windowed graph Fourier transform, the window size is fixed, which means that the τ parameter in the kernel function is arbitrarily chosen such that the vertex frequency representation has the most optimal resolution. Finally, the windowed graph Fourier transform ($WGFT$) of a function $x \in \mathbb{R}$, is defined as the inner product of the signal and the windowed graph Fourier atoms:

$$\begin{aligned} & WGFT(n,l) = \\ & = \sum_{n=1}^{N} x(n) \Big(N \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{w}(\lambda_l) \mathcal{X}_l^*(n) \mathcal{X}_l(n) \Big) = \\ & = \langle x, W_{n,l} \rangle \qquad . \end{aligned} \tag{11}$$

The graph S-transform can be defined as modulation of the graph wavelet transform (GWT). The GWT is defined as [14]:

$$GWT(n,l) := \sqrt{N} \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \hat{f}(\lambda_l) \mathcal{X}_l(n). \tag{12}$$

Thus, doing modulation of the (12), the S-transform is defined as:

$$GST(n,l) := (M_l GWT(n,l))(n) = N \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \hat{f}(\lambda_l) \mathcal{X}_l(n) \tag{13}$$

The GST can be evaluated as:

$$\begin{aligned} GST(n,l) &= N \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \hat{f}(\lambda_l) \mathcal{X}_l(n) = \\ &= N \mathcal{X}_l^*(n) \sum_{n=1}^{N} x(n) \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \mathcal{X}_l(n) = \\ &= \sum_{n=1}^{N} x(n) \Big(N \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \mathcal{X}_l^*(n) \mathcal{X}_l(n) \Big) = \\ &= \langle x, S_{n,l} \rangle , \end{aligned} \tag{14}$$

where $S_{n,l}(n)$ are S atoms defined as:

$$S_{n,l}(n) := N \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{g}(\lambda_l) \mathcal{X}_l^*(n) \mathcal{X}_l(n), \tag{15}$$

Wavelet function is defined in the vertex domain as $g(n,l) = \frac{|\lambda_l|}{\sqrt{2\pi}} e^{-\frac{n^2 \lambda_l^2}{2}}$.

From the equations (11) and (14) one may gather the expression for both the $WGFT$ and GST looks very similar. The only difference between expression for the $WGFT$ and GST is the choice of the window function.

Moreover, the signal $f(n)$ can be recovered from the $WGFT$ or GST representation using the expression [21]:

$$x(n) = \frac{1}{N||T_n g||_2^2} \sum_{i=1}^{N} \sum_{k=0}^{N-1} SG(n,l) W_{n,l}, \tag{16}$$

where $SG(n,l)$ is either $WGFT$ or GST.

$WGFT$ or GST requires high computational complexity [21]. For the N number of nodes, the vertex-frequency representation will generate an $N \times N$ spectrum for Fourier graph atoms. Each of those $N \times N$ points will generate N additional iterations in order to calculate the translation of the signal. Those N^3 points in the next step will integrate with the N signal points. Therefore, the total computational complexity will have $O(N^4)$ operations. Such a high computational complexity requires too long of a computation time for larger graphs.

3. The proposed scheme

3.1. Classical signal processing case

In order to overcome computational burdens, we can calculate the windowed Fourier transform and S-transform by operating on the Fourier spectrum of the signal on the graph as well as the window function on the graph. This approach has been used previously in classical signal processing for calculating the windowed Fourier transform of a one-dimensional signal [22].

In the classical case of the one-dimensional signal, the windowed Fourier transform and the S-transform can be calculated by performing the inverse Fourier transform from the "α domain". The α domain is defined as the Fourier transform of the windowed Fourier domain or the S-transform defined in (1) and (4). This domain can be presented mathematically as:

$$\alpha(f', f) = \int_{-\infty}^{\infty} S(\tau, f) e^{-j2\pi f'\tau} d\tau, \tag{17}$$

where $S(\tau, f)$ is the time frequency representation obtained using either WFT or ST. The f' axis is produced by taking the Fourier transform along the τ axis of the time-frequency domain. The α domain could also be expressed as a representation of the Fourier transform of the original signal as shifted by one sample and multiplied by the Fourier transform of the windowed function:

$$\begin{aligned}
\alpha(f', f) &= \int_{-\infty}^{\infty} S(\tau, f) e^{-jf'\tau} d\tau \\
&= \int_{-\infty}^{\infty} \left\{ \int_{-\infty}^{\infty} x(t) w(t-\tau, \sigma) e^{-j2\pi ft} dt \right\} e^{-j2\pi f'\tau} d\tau \\
&= \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} W(f, \sigma) e^{-j2\pi f'\tau} dt \\
&= W(f, \sigma) \int_{-\infty}^{\infty} x(t) e^{-j2\pi(f+f')t} dt \\
&= X(f' + f) \cdot W(f', \sigma)
\end{aligned} \tag{18}$$

where σ will determine the size of the window function. If $\sigma = C$ is a constant $w(t-\tau, \sigma)$ is fixed window, and $S(\tau, f)$ will represent time frequency representation obtained using WFT. In the case $\sigma = f$, $w(t-\tau, \sigma) = w(t-\tau, f)$ is frequency varying window size, and $S(\tau, f)$ will represent time frequency representation obtained using ST. Finally, the windowed Fourier transform and the S-transform can be recovered by applying the inverse Fourier transform on each row of the matrix representing represents the α domain:

$$FWFT(\tau, f) = \int_{-\infty}^{\infty} \alpha(f', f) e^{j2\pi f' \tau} df', \tag{19}$$

In the case of the windowed Fourier Transform, σ will be fixed, while for the S-transform, σ will change over various frequencies.

3.2. Fast windowed graph Fourier transform and fast graph S-transform

Using the idea from Section 3.1 to calculate the windowed graph Fourier transform and the graph S-transform, we first define the α domain as the Fourier transform of the $WGFT/GST$. The parameter δ will determine the width of a window function. In the case of $\delta = constant$, the width of the window function $w(n, \delta)$ is fixed, and $w(n, \delta)$ will be used for calculating $FWGFT$. Hence, the Fourier transform of the window function is defined as a heat kernel $\hat{w}(\lambda_l, k) = Ce^{-k\lambda_l}$, where C is chosen such that $||\hat{w}||_2 = 1$. For calculating the $FGST$, the parameter $\delta = l'$ where $l' = 0, 1, ..., N-1$, the window size will be frequency dependent and the window function is defined in the vertex domain as $w(n, l) = e^{-\frac{n^2\lambda_l^2}{2}}$. In the case of the $FWGFT$, the width of the windowed function will be the same across all frequencies, while for the $FGST$, the window width will tend to be more narrow for higher frequencies. However, in the case of signals on graph, it is not a simple multiplication of the shifted graph signal spectrum and the spectrum of the window function in the α domain like in the case of classical signal processing:

$$\begin{aligned}
\alpha(\lambda_l, \lambda_{l'}) &= \sum_{n=1}^{N} \left\{ \sum_{n=1}^{N} x(n)[T_n w(n, \delta)] \mathcal{X}_l^*(n) \right\} X_{l'}^*(n) \\
&= \sum_{n=1}^{N} x(n) \mathcal{X}_{l'}^*(n) \sum_{n-1}^{N} [T_n w(n, \delta)] \mathcal{X}_l^*(n) \\
&= \sum_{n=1}^{N} x(n) \mathcal{X}_{l'}^*(n) \sum_{n-1}^{N} \mathcal{X}_l^*(n) \sum_{l=0}^{N-1} \hat{w}(\lambda_l, \delta) \mathcal{X}_l^*(n) \mathcal{X}_l(n) \\
&= \sum_{n=1}^{N} x(n) \mathcal{X}_l^*(n) \mathcal{X}_{l'}^*(n) \sum_{n-1}^{N} w(n, \delta) \mathcal{X}_l^*(n) \\
&= \hat{w}(\lambda_l, \delta) \sum_{n=1}^{N} x(n) \mathcal{X}_l^*(n) \mathcal{X}_{l'}^*(n) \\
&= \hat{w}(\lambda_l, \delta) \hat{x}(\lambda_l, \lambda_{l'}),
\end{aligned} \tag{20}$$

where $\hat{x}(\lambda_l, \lambda_{l'}) = \sum_{n=1}^{N} x(n) \mathcal{X}_l^*(n) \mathcal{X}_{l'}^*(n)$ which is computationally more complex than a simple shift of the graph signal spectrum. Finally, the vertex-frequency domain can be recovered by taking the inverse Fourier transform of each row from the matrix which represents the α domain of the signal on the graph:

$$FSG(n, \lambda_l) = \sum_{l'=0}^{N-1} \alpha(\lambda_l, \lambda_{l'}) \mathcal{X}_{l'}(n) \tag{21}$$

where n refers to the node, and l is the signal frequency which for the signal on the graph is the eigenvalue λ_l of the Laplacian.

Moreover, the signal $x(n)$ can be recovered from the fast windowed graph Fourier representation using the expression:

$$x(n) = \sum_{l=0}^{N-1} \mathcal{X}_l(n) \sum_{n=1}^{N} FSG(n, \lambda_l). \tag{22}$$

3.2.1. Algorithm for calculating $FWGFT$ and $FGST$

The algorithm for calculating $FWGFT$ and $FGST$ is defined through the following steps:

(1) Calculate an $N \times N$ matrix which represents:

$$\hat{x}(\lambda_l, \lambda_{l'}) = \sum_{n=1}^{N} x(n)\mathcal{X}_l^*(n)\mathcal{X}_{l'}^*(n). \tag{23}$$

This step has $O(N^3)$ computational complexity. An alternative approach for calculating $\hat{x}(\lambda_l, \lambda_{l'})$ is using matrix calculus as:

$$\hat{X}(\lambda_l, \lambda_{l'}) = \hat{x}(\lambda_l, \lambda_{l'}) = (X * \mathcal{X}'_l) \cdot \mathcal{X}_l, \tag{24}$$

where the " $*$ " sign represents element-by-element multiplication of the two matrices, and " $\cdot$ " sign represents standard matrix multiplication. The variable X is $N \times N$ matrix formed such that each row of the matrix is the signal $x(n)$, and $\mathcal{X}_l$ is also $N \times N$ that represents a given set of eigenvectors.

(2) Form a matrix where each row is the Fourier transform of the window function $\hat{w}$.

$$\hat{W}(\lambda_l, \delta) = \\ = \begin{vmatrix} \hat{w}(\lambda_0, \delta_0) & \hat{w}(\lambda_1, \delta_0) & \cdots & \hat{w}(\lambda_{l-1}, \delta_0) \\ \hat{w}(\lambda_0, \delta_1) & \hat{w}(\lambda_1, \delta_1) & \cdots & \hat{w}(\lambda_{l-1}, \delta_1) \\ \vdots & \vdots & \ddots & \vdots \\ \hat{w}(\lambda_0, \delta_{l'-1}) & \hat{w}(\lambda_1, \delta_{l'-1}) & \cdots & \hat{w}(\lambda_{l-1}, \delta_{l'-1}) \end{vmatrix} \tag{25}$$

In the case of $FWGFT$, parameter δ is constant, and $\hat{w}(\lambda_l, \delta)$ will be directly defined and each row in the $\hat{W}(\lambda_l, \delta)$ matrix will be the same. Therefore, this step will not influence computational complexity of the algorithm. However, if we calculate $FGST$, parameter δ is equal l', therefore, $\hat{w}(\lambda_l, \delta) = \hat{w}(\lambda_l, l')$ is defined as the Fourier transform of the window function that is dependent on frequency ($w(n, l') = \frac{|\lambda_{l'}|}{\sqrt{2\pi}} e^{-\frac{n^2 \lambda_{l'}^2}{2}}$). That means that each row of the $\hat{W}(\lambda_l, l')$ matrix will be calculated by obtaining the Fourier transform of the window function that corresponds to the particular frequency point. Thus, in the case of $FGST$, this step will have $O(N^3)$ computational complexity.

(3) Calculate the α domain representation by multiplying each row of the $\hat{X}(\lambda_{l'}, \lambda_l)$ matrix with the appropriate window function:

$$\alpha(\lambda_{l'}, \lambda_l) = \hat{X}(\lambda_{l'}, \lambda_l) * \hat{W}(\lambda_{l'}, \delta), \tag{26}$$

where the " $*$ " sign represents element-by-element multiplication of the two matrices. The space complexity of this operation is $O(N^2)$.

(4) Finally, the vertex-frequency content is calculated by doing the inverse Fourier transform using formula (7) for each row from the $\alpha(\lambda_{l'}, \lambda_l)$. According to the formula (7), the computational complexity of the inverse Fourier transform is $O(N^2)$. Thus, the computational complexity of this step will be $O(N^3)$.

In the algorithm described above, step (2) and step (4) do not affect the computational complexity. This means that the fast algorithm for calculating the vertex frequency representation will have a $O(2N^3)$ computational complexity in the case of $FWGFT$, and $O(3N^3)$ in the case of $FGST$. For the higher N, this computational complexity is a significant improvement in comparison with the original algorithm which has a $O(N^4)$ computational complexity.

4. Performance evaluation of vertex-frequency algorithms

In order to evaluate the performance of the fast windowed graph Fourier transform and the fast graph S-transform, we generated examples of the graphs where we know the expected frequency. For that purpose we used examples of the time series as path graphs, where each time point represents nodes on the graph, and time samples are signals on the graph. Weights of the edges between nodes are equal to one. We considered a sampled sinusoidal signal that contains one frequency for the entire duration of the signal (s_1), a signal that contains different frequencies during different time points (s_2), and a chirp signal (s_3) (see Figure 1). Signals s_1, s_2, and s_3 are defined as:

$$s_1(n) = \sin(60\pi n), \quad 0 \leq n \leq 200 \tag{27}$$

$$s_2(n) = \begin{cases} \sin(150\pi n) & 0 \leq n < 65 \\ \sin(50\pi n) & 65 \leq n < 135 \\ \sin(100\pi n) & 135 \leq n \leq 200 \end{cases} \tag{28}$$

$$s_3(n) = \sin((10n + kn^2)\pi), \quad 0 \leq n \leq 200 \tag{29}$$

Next, we calculated the vertex-frequency representation of the graph examples using the $FWGFT$, $FGST$, $WGFT$, and $FGST$ methods. Then, we normalized all representations to have the same energy and calculated the mean squared error (MSE) between $FWGFT$ and $WGFT$, and between $FGST$ and GST. The MSE between the vertex-frequency representations is calculated using the formula [26]:

$$MSE = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{l=1}^{N} (\hat{Y}_{i,\lambda_l} - Y_{i,\lambda_l})^2, \tag{30}$$

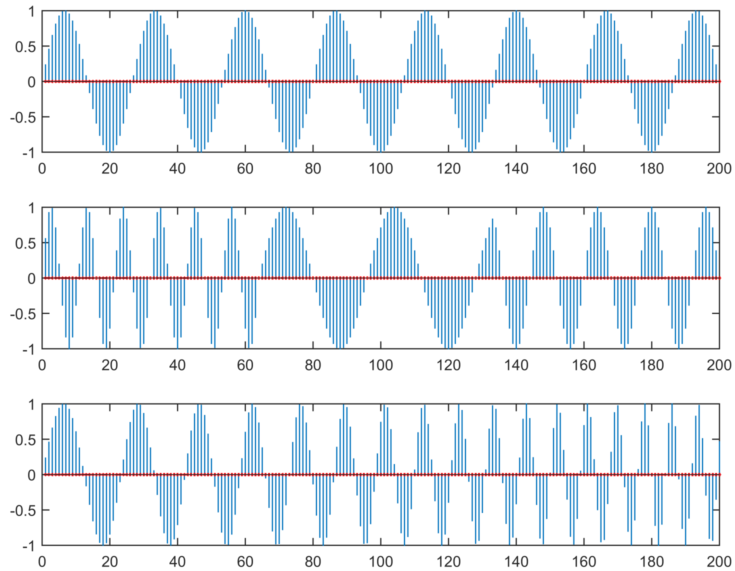

Figure 1: Signals used for testing the algorithm.

where $\hat{Y}_{i,k}$ is the vertex-frequency representation of the reconstructed signal, $Y_{i,k}$ is the vertex-frequency representation of the original signal, and N is the number of nodes of the graph. We also calculated the MSE between the reconstructed and the original signal, as well as the MSE between the reconstructed and the original signal contaminated with noise. The MSE between the reconstructed and the original signal is calculated as:

$$mse = \frac{1}{N}\sum_{n=1}^{N}(\hat{x}(n) - x(n))^2, \tag{31}$$

where $\hat{x}(n)$ is the reconstructed signal, and $x(n)$ is the original graph signal. Then we compared the required computation time between the fast algorithms and the slow algorithms. Finally, we checked the performance of the fast algorithm by applying it to the real brain network. The real brain network was formed using an electroencephalography (EEG) signal recorded during a swallowing activity.

Figure 2 shows the calculated vertex-frequency representation for each graph using $FWGFT$, $WGFT$, $FGST$, and GST. The heat kernel $\hat{w}(\lambda_l) = Ce^{-\tau\lambda_l}$ with the $\tau = 60$ is used as the spectral domain of the window function for calculating the $FWGFT$ and $WGFT$, where the constant C is chosen such that $||w||_2 = 1$. For calculating $FGST$ and GST, we used the window function $w(n,l) = \frac{|\lambda_l|}{\sqrt{2\pi}}e^{-\frac{n^2\lambda_l^2}{2}}$ defined in the vertex domain.

According to Figure 2, one may see that all methods produce the expected results. Results showed that the mean squared error (MSE) between the normalized representations of $FWGFT$ and the normalized representations of $WGFT$, as well as between the normalized representations of $FGST$ and the normalized

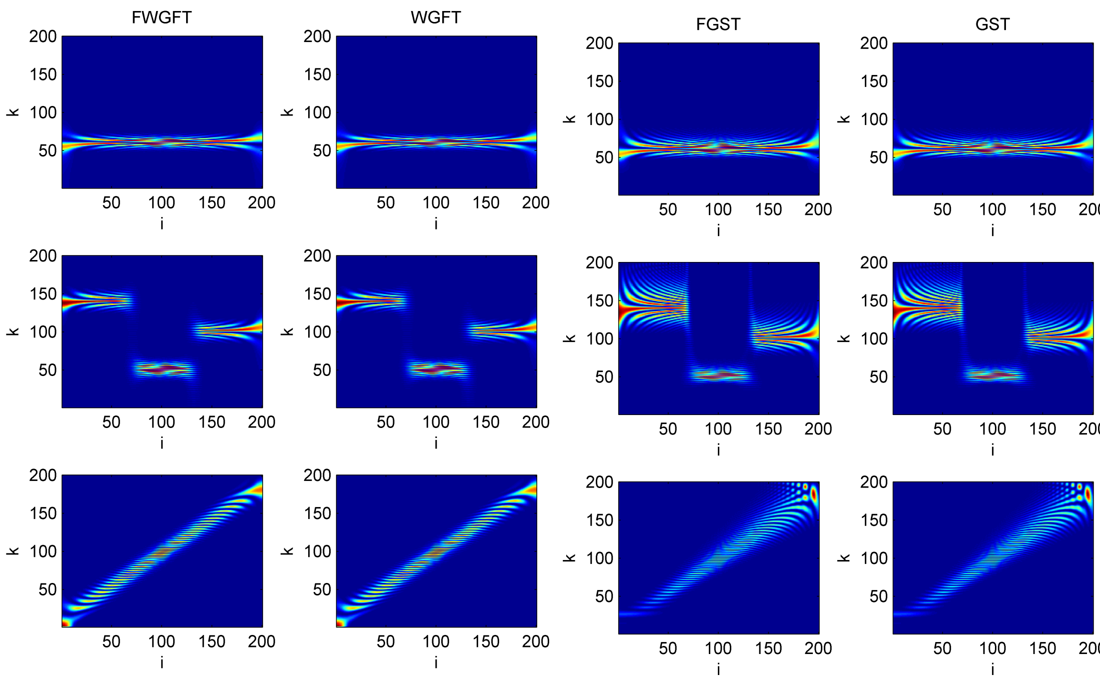

Figure 2: Vertex-frequency representation of the signals s_1, s_2, and s_3 using $FWGFT$, $WGFT$, $FGST$, and GST.

representations of GST are less than 10^{-31} for the all signals used in testing (s_1, s_2, and s_3).

The MSE between the original and the signal reconstructed using $FWGFT$ or $WGFT$ for all three test signals is significantly less than 10^{-29}, while for $FGST$ and GST, it is slightly higher, but still it is less than 10^{-6}. The MSE between the original signal contaminated with the noise and the reconstructed signal did not change across the different levels of noise using any of the four algorithms ($FWGFT$, $WGFT$, $FGST$, and GST). This means that noise will have no effect on the reconstruction error for each vertex-frequency technique used in this study.

We also considered Minnesota road and swiss roll graphs with the 300 nodes (3). As a signal on the Minnesota road graph, we used summation of the $\mathcal{X}_{500}(n)$, $\mathcal{X}_{510}(n)$, $\mathcal{X}_{520}(n)$, and $\mathcal{X}_{530}(n)$ eigenvectors that corresponds to the graph Laplacian from the Minnesota road graph. As a signal on the Swiss roll graph, we used $\mathcal{X}_{60}(n)$ eigenvector that corresponds to the graph Laplacian from the Swiss roll graph.

$$s_m(n) = \mathcal{X}^m_{500}(n) + \mathcal{X}^m_{510}(n) + \mathcal{X}^m_{520}(n) + \mathcal{X}^m_{530}(n) \tag{32}$$

$$s_s(n) = \mathcal{X}^s_{60}(n) \tag{33}$$

Figure 4 shows the evaluated vertex-frequency representations for both Minnesota road and swiss roll graphs using $FWGFT$, $WGFT$, $FGST$, and GST. For calculating $FWGFT$ and $WGFT$ of the Minnesota

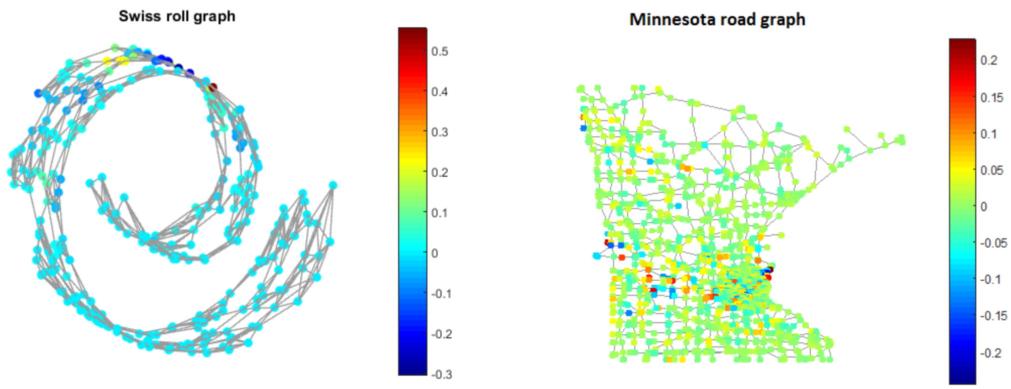

Figure 3: Swiss roll and Minnesota road graph.

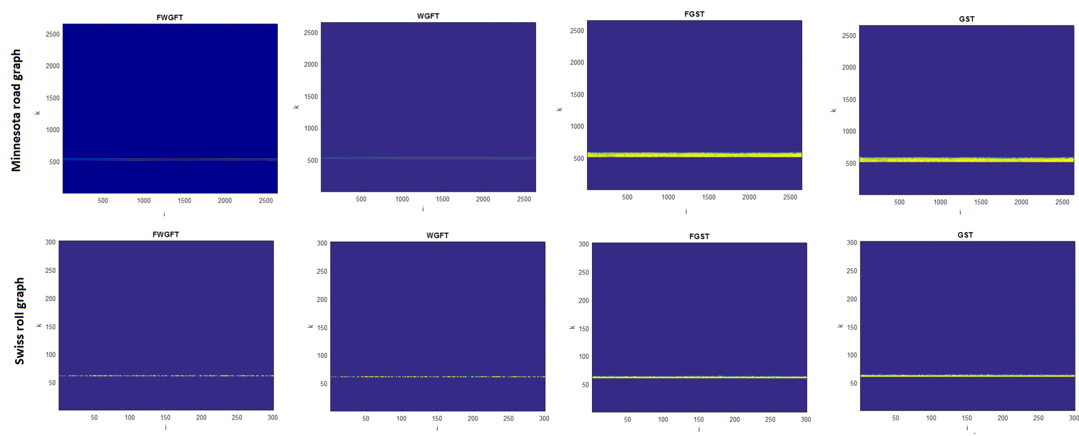

Figure 4: Vertex-frequency representation of the Swiss roll and Minnesota road graph using $FWGFT$, $WGFT$, $FGST$, and GST.

road graph using, the heat kernel $\hat{w}(\lambda_l) = Ce^{-\tau\lambda_l}$ with the $\tau = 300$ was used, while for calculating $FWGFT$ and $WGFT$ of the Swiss roll graph, the heat kernel $\hat{w}(\lambda_l) = Ce^{-\tau\lambda_l}$ with the $\tau = 150$ was used.

For considered cases, the MSEs between the normalized representations of $FWGFT$ and $WGFT$, as well as between the normalized representations of $FGST$ and GST are less than 10^{-31} for both Minnesota road and Swiss roll graphs.

To estimate computational complexities of these approaches, we formed random graphs with random node values. We considered graphs from 200 to 5000 nodes. For each formed graph, we measured computational times using the original and fast algorithms. Figure 5 shows the computation time for the $FWGFT$ versus $WGFT$ and the computation time for the $FGST$ versus GST. From these results, it is obvious that the computation time improved for the fast algorithms in comparison to the original algorithms. One may see from the graph containing 5000 nodes that the computation time for the $WGFT$ and GST is around 40

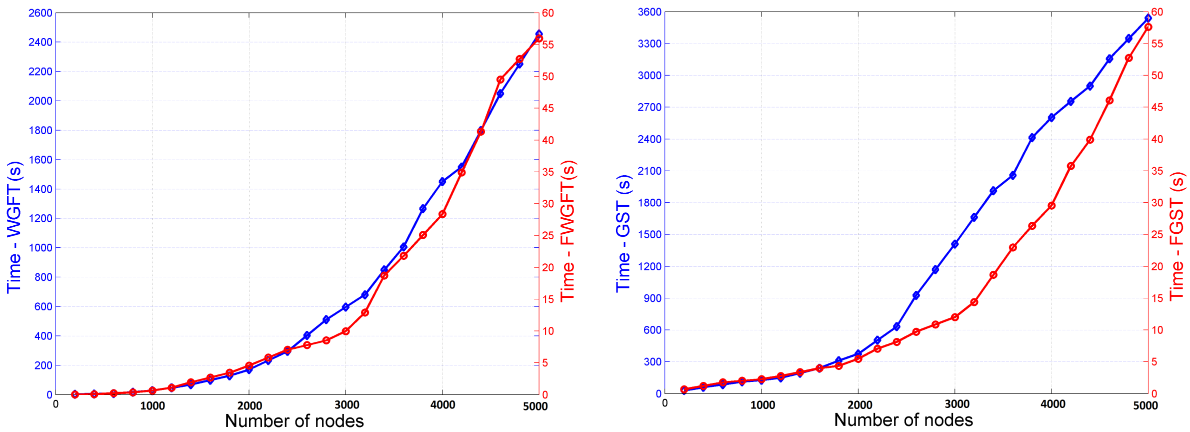

Figure 5: The required time for computing $WGFT$ versus $FWGFT$ and GST versus $FGST$ for the different graph sizes. Blue lines represent computation time for the original algorithms while red lines represent computation time for the fast algorithms.

minutes. Using this fast approach, computing vertex-frequency changes of the signal on the graph with 5000 nodes will take less than one minute.

4.1. An illustrative example of the performances of the proposed algorithm

It has been shown that brain activity during swallowing can be analyzed using EEG signals [27] and the graph theory approach [28] in order to provide distinctive information that could lead to the better diagnostic and rehabilitation techniques. The data collection process, swallowing segmentation, and pre-processing steps are described in our previous study [29]. Weighted connectivity networks are formed using the time-frequency based phase synchrony measure (i.e., reduced interference Rihaczek distribution) proposed by Aviyente et al [30]. From the constructed weighted connectivity networks, weak connections are removed by applying a threshold to the network. Then, in order to provide the signal on the graph, we formed a line graph [31] from the weighted connectivity network that corresponds to the synchronization between signals from the EEG electrodes during swallowing. With the new formed line graph, each vertex will represent one edge from the original graph. Vertices from the line graph will be connected if corresponding edges from original graph are connected to the same node.

Figure 6 shows the calculated vertex-frequency representation for each graph using $FWGFT$, $WGFT$, $FGST$, and GST. The $FWGFT$ and $WGFT$ are calculated from the spectral domain of the window function. A heat kernel represents the spectral domain of the window function, and is defined as $\hat{w}(\lambda_l) = Ce^{-\tau\lambda_l}$ with $\tau = 0.05$. When calculating the $FGST$ and GST, we used the window function as $w(n,l) = \frac{|\lambda_l|}{\sqrt{2\pi}}e^{-\frac{n^2\lambda_l^2}{2}}$. In both cases the constant C is chosen such that $||w||_2 = 1$. Computational time required for calculating vertex-frequency information from these swallowing brain networks is around 20 seconds using $WGFT$ and GST, while using $FWGFT$ and $FGST$ computational time is around 0.6 seconds.

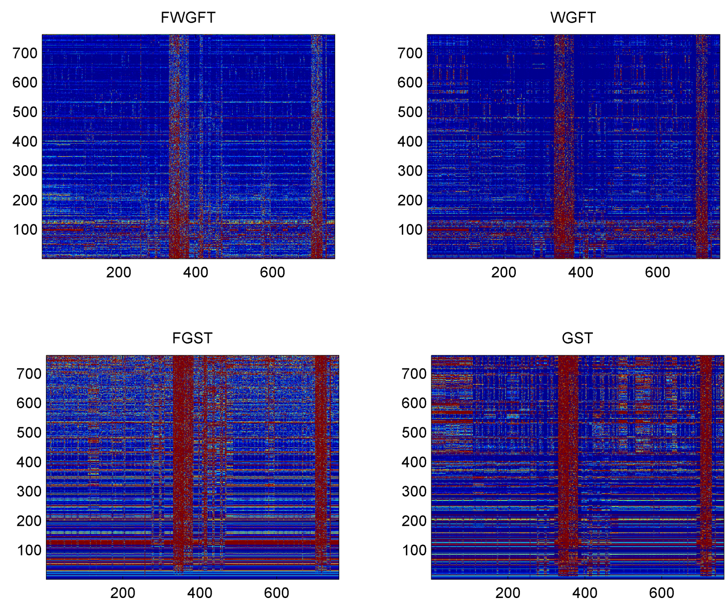

Figure 6: Vertex-frequency representations of the sample signals representing the brain network formed from EEG signals recorded during swallowing using $FWGFT$, $WGFT$, $FGST$, and GST.

5. Discussion

In this paper, we have described the basic signal processing operations on graphs: the graph Fourier transform, translation on graphs, modulation on graphs, the windowed graph Fourier transform, and the graph S-transform. We have also described algorithms for calculating the fast windowed graph Fourier transform and the fast graph S-transform. We showed that $FWGFT$ and $FGST$ have a significantly lower computational complexity than do $WGFT$ and GST [21].

Figure 2 summarizes the vertex-frequency representations for the signals s_1, s_2, and s_3 on the path graph, and Figure 6 summarizes the vertex-frequency representation for an example of the real signal on the swallowing brain network formed using EEG signals. From Figure 2 it can be seen that all methods used produce results that would match a spectrogram from classical signal processing. As in classical signal processing, the $FWGFT$ and $WGFT$ have the limitation of a fixed window size. $FGST$ and GST overcome the limitation of a fixed window by using a window size that is dependent on frequency. It is well known from classical signal processing that a wide window gives good frequency resolution but poor vertex resolution, while a narrow window trades improvements in frequency resolution for decreased quality of the vertex resolution. Such trends can be seen from Figure 6 in the signal on graph vertex-frequency representations using $FGST$ and GST, where we have poor vertex resolution at the very low frequencies and poor frequency resolution at the very high frequencies. Therefore, similarly as in classical signal processing [32], we can say that $FGST$ and GST provide a good concentration of the energy at lower frequencies, but poor concentration of the energy at higher frequencies. However, it needs to be mentioned that vertex-frequency representations suffer from the same drawbacks as classical time-frequency presentations (e.g., the uncertaininty principle). Hence, future investigations should consider adaptive vertex-frequency representations optimized for each signal or each signal class.

Results of the MSE between the original and the reconstructed signal showed that $FWGFT$ and $WGFT$ show an almost perfect reconstruction of the signal. However, the $FGST$ and GST exhibited higher MSE between the original and the reconstructed signal. This means that varying window size in the case of the $FGST$ and GST affects the reconstruction formula. Our results also showed that the MSE between the original and the reconstructed signal contaminated with noise is approximately constant for various levels of noise. This means that noise will have no effect on the reconstruction error when we use any of the methods for analysis of the vertex-frequency changes in the graph signal.

Finally, we showed significant computation time improvement in the algorithm for the $FWGFT$ and $FGST$ in comparison with the original $WGFT$ and GST (see Figure 5). Also, from Figure 5, it can be seen that the computation time for the $WGFT$ and GST is good for smaller graphs. However, as the number of nodes increases computation time progressively increases as well. Fast approaches could be applied in any application that is sensitive to time or memory, or in those applications that deal with large graphs. Further improvements can be obtained if the proposed algorithm is parallelizable using procedures similar to those obtained for traditional time-frequency representations (e.g., [33], [34]).

6. Conclusion

In this study, we developed fast algorithms for the windowed graph Fourier transform and the graph S-transform. The developed algorithms are referred to as the fast windowed graph Fourier transform ($FWGFT$) and the fast graph S-transform ($FGST$). Performance of the fast algorithms are evaluated and compared with the standard windowed graph Fourier transform and the graph S-transform algorithms using synthetic graph signals and a real graph signal. Results showed that the proposed approach significantly decreases the computation time for extracting vertex-frequency information from graph signals. We also showed that the inverse formula for the proposed algorithms provides almost perfect reconstruction. Additionally, we showed that the signal reconstruction is not affected by the noise.

7. Acknowledgments

Research reported in this publication was supported by the Eunice Kennedy Shriver National Institute Of Child Health & Human Development of the National Institutes of Health under Award Number

R01HD074819. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

[1] J. A. Bondy, U. S. R. Murty, Graph theory with applications, Vol. 290, Macmillan London, London, UK, 1976.

[2] Z. Mihalić, N. Trinajstić, A graph-theoretical approach to structure-property relationships, Journal of Chemical Education 69 (9) (1992) 701–712.

[3] O. Morris, M. Lee, A. Constantinides, Graph theory for image analysis: an approach based on the shortest spanning tree, IEEE Proceedings on Communications, Radar and Signal Processing 133 (2) (1986) 146–152.

[4] L. Dobrjanskyj, F. Freudenstein, Some applications of graph theory to the structural analysis of mechanisms, Journal of Manufacturing Science and Engineering 89 (1) (1967) 153–158.

[5] M. C. Etter, J. C. MacDonald, J. Bernstein, Graph-set analysis of hydrogen-bond patterns in organic crystals, Acta Crystallographica Section B: Structural Science 46 (2) (1990) 256–262.

[6] A. Sandryhaila, J. M. Moura, Discrete signal processing on graphs, IEEE Transactions on Signal Processing 61 (7) (2013) 1644–1656.

[7] A. Sandryhaila, J. M. Moura, Discrete signal processing on graphs: graph Fourier transform., in: ICASSP, 2013, pp. 6167–6170.

[8] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, P. Vandergheynst, The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains, IEEE Signal Processing Magazine 30 (3) (2013) 83–98.

[9] S. K. Narang, A. Ortega, Compact support biorthogonal wavelet filterbanks for arbitrary undirected graphs, IEEE Transactions on Signal Processing 61 (19) (2013) 4673–4685.

[10] N. Leonardi, D. Van De Ville, Tight wavelet frames on multislice graphs, IEEE Transactions on Signal Processing 61 (13) (2013) 3357–3367.

[11] S. Chen, R. Varma, A. Sandryhaila, J. Kovačević, Discrete signal processing on graphs: Sampling theory, IEEE Transactions on Signal Processing 63 (24) (2015) 6510–6523.

[12] A. G. Marques, S. Segarra, G. Leus, A. Ribeiro, Sampling of graph signals with successive local aggregations, IEEE Transactions on Signal Processing 64 (7) (2016) 1832–1843.

[13] A. Loukas, Distributed graph filters, Ph.D. thesis, TU Delft (2015).

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

[15] S. K. Narang, A. Ortega, Perfect reconstruction two-channel wavelet filter banks for graph structured data, IEEE Transactions on Signal Processing 60 (6) (2012) 2786–2799.

[16] B. Gelbord, Graphical techniques in intrusion detection systems, in: Proceedings on the 15th International Conference on Information Networking, IEEE, 2001, pp. 253–258.

[17] Y. Asahiro, R. Hassin, K. Iwama, Complexity of finding dense subgraphs, Discrete Applied Mathematics 121 (1) (2002) 15–26.

[18] M. Deshpande, M. Kuramochi, N. Wale, G. Karypis, Frequent substructure-based approaches for classifying chemical compounds, IEEE Transactions on Knowledge and Data Engineering 17 (8) (2005) 1036–1050.

[19] T. R. Coffman, S. E. Marcus, Pattern classification in social network analysis: a case study, in: Proceedings of the 2004 IEEE Aerospace Conference, Vol. 5, (Big Sky, MT, Mar. 2004), 2004.

[20] S. H. Nawab, T. F. Quatieri, Short-time Fourier transform, in: Advanced Topics in Signal Processing, Prentice-Hall, Inc., 1987, pp. 289–337.

[21] D. I. Shuman, B. Ricaud, P. Vandergheynst, Vertex-frequency analysis on graphs, Applied and Computational Harmonic Analysis 40 (2) (2016) 260–291.

[22] R. A. Brown, M. L. Lauzon, 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 58 (1) (2010) 281–290.

[23] K. Gröchenig, Foundations of time-frequency analysis, Springer Science & Business Media, New York City, NY, 2001.

[24] R. G. Stockwell, L. Mansinha, R. Lowe, Localization of the complex spectrum: the S-transform, IEEE Transactions on Signal Processing 44 (4) (1996) 998–1001.

[25] F. R. Chung, Spectral graph theory, American Mathematical Society, Providence, RI, 1997.

[26] A. V. Oppenheim, R. W. Schafer, J. R. Buck, Discrete-time signal processing, Vol. 2, Prentice-hall Englewood Cliffs, Upper Saddle River, NJ, 1989.

[27] I. Jestrović, J. L. Coyle, E. Sejdić, Decoding human swallowing via electroencephalography: a state-of-the-art review, Journal of Neural Engineering 12 (5) (2015) 1–15.

[28] I. Jestrović, J. L. Coyle, E. Sejdić, Characterizing functional connectivity patterns during saliva swallows in different head positions, Journal of Neuroengineering and Rehabilitation 12 (1) (2015) 1–11.

[29] I. Jestrović, J. Coyle, E. Sejdić, The effects of increased fluid viscosity on stationary characteristics of EEG signal in healthy adults, Brain Research 1589 (1) (2014) 45–53.

[30] S. Aviyente, E. M. Bernat, W. S. Evans, S. R. Sponheim, A phase synchrony measure for quantifying dynamic functional integration in the brain, Human Brain Mapping 32 (1) (2011) 80–93.

[31] D. B. West, Introduction to graph theory, Vol. 2, Prentice hall Upper Saddle River, Upper Saddle River, NJ, 2001.

[32] E. Sejdić, I. Djurović, J. Jiang, Time–frequency feature representation using energy concentration: An overview of recent advances, Digital Signal Processing 19 (1) (2009) 153–183.

[33] P. Waskito, S. Miwa, Y. Mitsukura, H. Nakajo, Parallelizing Hilbert-Huang transform on a GPU, in: Proc. of the First International Conference on Networking and Computing (ICNC), 2010, pp. 184–190.

[34] I. Orović, S. Stanković, B. Jokanović, A suitable hardware realization for the cohen class distributions, IEEE Transactions on Circuits and Systems II: Express Briefs 60 (9) (2013) 607–611.