SPECTRAL GRAPH BASED JOINT VERTEX-FREQUENCY WIENER FILTERING FOR IMAGE AND GRAPH SIGNAL DENOISING

Doctoral Dissertation

Ali Can YAĞAN

Eskişehir 2020

SPECTRAL GRAPH BASED JOINT VERTEX-FREQUENCY WIENER FILTERING FOR IMAGE AND GRAPH SIGNAL DENOISING

Ali Can YAĞAN

DOCTORAL DISSERTATION

Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Mehmet Tankut ÖZGEN

Eskişehir

Eskişehir Technical University

Institute of Graduate Programs

October 2020

ÖZET

GÖRÜNTÜ VE ÇİZGE SİNYALİ TEMİZLEME İÇİN SPEKTRAL ÇİZGE TABANLI ORTAK DÜĞÜM-SIKLIK WIENER FİLTRELEMESİ

Ali Can YAĞAN

Elektrik-Elektronik Mühendisliği Anabilim Dalı
Eskişehir Teknik Üniversitesi, Lisansüstü Eğitim Enstitüsü, Ekim 2020
Danışman: Prof. Dr. Mehmet Tankut ÖZGEN

Bu tezde, görüntü ve çizge sinyallerinin temizlenmesi için spektral çizge tabanlı düğüm-sıklık Wiener filtreleme işlemi önerilmiş ve geliştirilmiştir. İlk olarak, klasik sinyal işlemedeki Zadeh zaman-sıklık filtreleme kavramı çizge sinyallerine uyarlanmıştır ve bu filtrenin düğüm-sıklık transfer fonksiyonu, onun orijinal ve tahmin edilen çizge sinyali arasındaki ortalama karesel hatasını enküçülten, düğümle değişen dürtü cevabından elde edilmiştir. Türetilen Wiener filtresini kullanabilmek için bir delta sinyali ile genelleştirilmiş konvolüsyon olarak tanımlanan bir çizge öteleme operatörüne dayandırılan çizge Rihaczek düğüm-sıklık dağılımının (ÇRD) detaylı bir türetimi sunulmuştur. Ulaşılan Wiener filtresinin yapısı klasik sinyal işlemede yaygın olan zaman-sıklık Wiener filtrelerininkinden oldukça farklıdır. Ayrıca, kullanılan ÇRD'nın terslenebilirliği de incelenmiştir. Bir başka çalışma olarak, sadece düzensiz yapıdaki çizge sinyallerinin temizlenmesi için pencerelenmiş çizge Fourier dönüşümü alanında bir düğüm-sıklık çizge Wiener filtreleme yapısı önerilmiş ve geliştirilmiştir. Çizge sinyalinin deterministik olduğu varsayımı altında, bu önerilen düğüm-sıklık Wiener filtrelerini uygulamak amacıyla bazı algoritmalar önerilmiştir ve bazı görüntü ve düzensiz çizge sinyallerinin temizlenmesi için uygulanarak performansları güncel temizleme yöntemleriyle karşılaştırılmıştır.

Anahtar Sözcükler: Spektral çizge yöntemleri, Wiener filtresi, Zadeh zaman-sıklık filtresi, Çizge Rihaczek dağılımı, Görüntü ve çizge sinyali temizleme

ABSTRACT

SPECTRAL GRAPH BASED JOINT VERTEX-FREQUENCY WIENER FILTERING FOR IMAGE AND GRAPH SIGNAL DENOISING

Ali Can YAĞAN

Department of Electrical and Electronics Engineering
Eskişehir Technical University, Institute of Graduate Programs, October 2020
Supervisor: Prof. Dr. Mehmet Tankut ÖZGEN

In this thesis, spectral graph based vertex-frequency Wiener filtering scheme is proposed and developed for image and graph signal denoising. Zadeh time-frequency filter concept in classical signal processing is first extended to graph signals and vertex-frequency transfer function of this filter is obtained from its vertex varying impulse response minimizing the mean square error between original and estimated graph signal. A detailed derivation of graph Rihaczek vertex-frequency distribution (GRD) based on a graph shift operator defined by generalized convolution with a delta signal is presented to facilitate the derived Wiener filter. The form of the obtained Wiener filter is different than those of time-frequency Wiener filters prevalent in the classical signal processing. Moreover, the invertibility of the employed GRD is investigated. As another work, a vertex-frequency graph Wiener filter framework in the windowed graph Fourier transform domain is proposed and developed to denoise irregularly structured graph signals only. Under the assumption that the graph signal is deterministic, some algorithms are proposed to implement these proposed vertex-frequency Wiener filters, and they are applied for denoising of a set of images and irregular graph signals, and their performances are compared with recent denoising methods.

Keywords: Spectral graph methods, Wiener filter, Zadeh time-frequency filter, Graph Rihaczek distribution, Image and graph signal denoising

ACKNOWLEDGMENTS

I would like to express my special appreciation to and thank sincerely my supervisor Prof. Dr. Mehmet Tankut Özgen for his guidance and support during my study. He helped and encouraged me at every stage of this thesis.

I also would like to thank Assoc. Prof. Dr. Hakan Güray Şenel and Assist. Prof. Dr. Ediz Şaykol for serving in my Thesis Monitoring and PhD Dissertation Defence Committes, and also thank Prof. Dr. Ömer Nezih Gerek, Prof. Dr. Kemal Leblebicioğlu and Prof. Dr. Orhan Arıkan for serving in my PhD Dissertation Defence committee. Their precious and insightful comments provide me new perspectives.

I especially thank to my beloved wife Gizem Yağan for her endless trust and support during this thesis. I am grateful to her for suggestions and proof reading. I always feel so lucky that she is my wife.

Ali Can YAĞAN

08/10/2020

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with "scientific plagiarism detection program" used by Eskişehir Technical University, and that "it does not have any plagiarism" whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

....

Ali Can YAĞAN

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Page

GLOSSARY OF SYMBOLS AND ABBREVIATIONS

BF	: Bilateral Filter
BM3D	: Block-Matching and Three-Dimensional filtering
GFT	: Graph Fourier Transform
GHWT	: Generalized Haar-Walsh Transform
GRD	: Graph Rihaczek vertex-frequency energy Distribution
GRS	: Graph Rihaczek Spectrum
GSO	: Graph Shift Operator
GSP	: Graph Signal Processing
HGLET	: Hierarchical Graph Laplacian Eigen Transform algorithm
LPFW	: Lowpass Filter based vertex-frequency graph Wiener filtering algorithm
MSE	: Mean Square Error
NLGBT	: Non Local Graph Based Transform
NLM	: Non Local Means
OGLR	: Optimal Graph Laplacian Regularization
PLOW	: Patch-based Locally Optimal Wiener
PSNR	: Peak Signal-to-Noise Ratio
PWS	: Piecewise Smooth
SHTW	: Smoothing and Hard Thresholding based vertex-frequency graph Wiener filtering algorithm
SNR	: Signal-to-noise Ratio
SSIM	: Structural Similarity
STFT	: Short-Time Fourier Transform
TF	: Time-Frequency
WGFT	: Windowed Graph Fourier Transform
WS	: Smoothing and hard thresholding based vertex-frequency graph Wiener filtering algorithm in the WGFT domain, employing a window in the spectral domain

WV	: Smoothing and hard thresholding based vertex-frequency graph Wiener filtering algorithm in the WGFT domain, employing the Hann window in the spectral domain
Scalar	: Lowercase letter
Vector	: Bold lowercase letter
Matrix	: Bold uppercase letter
$(\cdot)^T$	: Transpose of a vector or a matrix
$(\cdot)^H$	: Conjugate transpose of a vector or a matrix
$(\cdot)^*$	: Complex conjugation
$\|\mathbf{x}\|$	: The Euclidean norm of the vector $\mathbf{x}$
$\circ$	: Hadamard (elementwise) product of matrices
$\mathbf{1}_{K\times K}$	: $K \times K$ matrix with all elements equal to unity
$\mathbf{I}$	: $L \times L$ identity matrix
$\mathbf{0}$	: A zero column-vector with dimension L
$\delta(n)$	: Kronecker delta, i.e., $\delta(n) = 1$ for $n = 0$, and 0 otherwise
$\mathbf{V}(n, :)$	: nth row of $\mathbf{V}$
$\mathbf{V}(:, \lambda_l)$	: lth column of $\mathbf{V}$ (lth eigenvector of $\boldsymbol{\mathcal{L}}$)
$\mathbf{V}^{-\circ} = [1/V(n, \lambda_l)]$	: A matrix formed by inverses of nonzero graph basis elements, assuming that they are nonzero
$\mathbf{V} = [V(n, \lambda_l)]$	: The eigenvector matrix of the graph Laplacian $\boldsymbol{\mathcal{L}}$ of size $L \times L$, nth element of the eigenvector of $\boldsymbol{\mathcal{L}}$ associated with its lth eigenvalue λ_l, for $l = 0, 1, \ldots, L-1$

1. INTRODUCTION

1.1. Background and Motivation

In the past decade, graph signal processing (GSP) based on the idea of defining a signal on a regular or irregular structure has been rapidly developed in many signal processing applications such as social, sensor or biological networks, etc. [1, 2]. Many classical signal processing approaches have been extended to graph signals for filtering, denoising and coding purposes [3, 4, 5], i.e., signals samples of which are defined on and indexed by vertices (nodes) of a graph with edges between vertices to represent sample relationships and possible edge weights to model pairwise similarity between those vertices or samples residing on them.

Spectral graph based methods based on the graph Fourier transform (GFT) have also been widely used in signal processing applications. Adjacency matrix of a signal defined over a graph is of high importance in these methods since the GFT is computed by the projection of corresponding signal onto the eigenbasis of it [4, 6, 7] or onto that of a graph Laplacian matrix derived from it [3, 8]. Alternative and more elaborate GFT definitions have also been proposed recently [9]–[13]. Spectral analysis of graph signals [14]–[19], filtering operations in the graph spectral domain [3, 4], [6]–[8], [20, 21], stationarity of a graph signal [15]–[19], [22] and adaptive graph signal processing [23]–[25] are some notions facilitated by the GFT, to analyze and process a signal, in spectral graph based methods.

Spectral graph based denoising methods have been developed for graph signals defined over irregular structures and images recently [26]. Viewing pixels of an image as vertices (nodes) of a graph consisting of a two-dimensional (2-D) regular grid of those vertices connected by edges representing image pixel similarities with their assigned weights enables some image processing techniques such as smoothing, denoising and filtering to be applied in GFT domain [20] and also leads to discrete regularization frameworks on the weighted graph obtained from image to be processed [27]. Such regularization frameworks are related with denoising techniques employed in GFT domain [3, 6, 7, 21], [28]–[31], i.e., l_0, l_1 and l_2 regularization yield hard thresholding [32]–[34], soft thresholding [6, 7] and lowpass filtering [35]–[37] in the graph Fourier domain. Total variation minimization methods applied to graph signals are also developed for denoising in GFT domain [6, 7, 27, 38, 39].

Fast spectral graph filter implementations are also available based on polynomial approximations of graph filter frequency responses that enable to apply polynomials of a sparse graph shift operator matrix to an input signal locally in the vertex domain [6, 8, 40, 41]. A related fast filtering approach is to project the input noisy signal onto a Krylov subspace of the graph Laplacian matrix [42]–[44].

Some of the more recent image denoising methods are kernel based image filtering techniques [45] that weight similar image patches or pixels locally or nonlocally [46]–[48], such as the bilateral filters (BF) [49, 50] and nonlocal means (NLM) [51]–[53] filters, and that also adapt to local structures of patches [54]. Such methods are closely related with spectral graph based denoising methods in that kernel similarity matrices are analogous to weighted adjacency matrices whose elements representing similarity between two image pixels serves as the weight of the graph edge connecting those two pixels (vertices). Kernel matrices can be adapted to local structures of images as well, for improved performance [54]. In such kernel based image filtering methods, row-normalized kernel matrices are applied to input vectors composed of observed, noisy image pixels [45, 54]. However, the matrix inversion involved in the normalization is shown to be relieved successfully to reduce the computational cost [55].

Two of the patch based image denoising methods employing Wiener filtering, i.e., patch-based locally optimal Wiener (PLOW) filtering and the block-matching and three-dimensional (BM3D) filtering, achieve effective denoising performances resulting them to be called as state-of-the-art methods. In particular, the PLOW filter performs Wiener filtering in spatial patch domain [56] while the BM3D filtering performs it in a fixed three-dimensional domain [57, 58]. Spatial domain Wiener filtering techniques based on estimating pixelwise image statistics locally [59]–[61] or non-locally [62] developed for various noise models; stationary or nonstationary, additive or multiplicative, Gaussian or Poisson [59]–[62] are also developed in image denoising literature.

In GSP context, stationarity notion for graph signals has also been defined and stationary graph Wiener filters in GFT domain have been developed for deconvolving or denoising of stationary graph signals [15]–[19], [21, 63]. Such filters are based on different concepts such as graph shift operator (GSO) matrices [18], correlation functions of stationary graph signals computed by an isometric GSO to obtain a graph version of the Wiener-Hopf equation [19], and power spectral densities [15, 18, 21, 22, 63] obtained from eigenvalues of such signal autocorrelation matrices whose eigenvectors are same with those of graph Laplacian [15, 21, 22, 63]. Image denoising and in-painting applications have also been performed by such filters [15, 18].

Images have been divided into different categories such as natural or piecewise smooth (PWS) images including depth, synthetic images, etc., with respect to their structural and textural properties in some of the more recent studies in image processing, to demonstrate the performance of proposed algorithms effectively. Natural images are generally regarded as 2-D stationary signals defined over regular structures [15, 64], however, images and graph signals of interest can be nonstationary or can even be modeled as deterministic. In particular, PWS images can be considered as 2-D deterministic signals on grid graphs. It has also been shown that there is still room for improvement in denoising performance over the state-of-the-art methods especially for less textured, simple structured and smoother images called as PWS images [65].

Wiener filtering has been developed in a fixed domain such as the cosine, wavelet or discrete Fourier domain [52, 66, 67], and in the eigenspace of a filter kernel matrix [45, 68], under the assumption that the signal of interest is deterministic. Such a filtering is proposed in [68], applying a kernel based denoising filtering iteratively to optimize estimated mean square error (MSE) from input noisy image, with respect to truncation of filter eigenmodes. A graph Wiener filtering scheme is proposed and developed in [69] in GFT domain as a post-processing step in order to improve image denoising performances obtained by various spectral graph based denoising methods. This method uses cleaned graph Fourier coefficients of a noisy image provided by such a method, and the estimated noise variance, to estimate graph Fourier coefficients of the original, noiseless image, first; and then estimates the Wiener filter frequency response using these estimated coefficients of the original image [69]. As such, it is realized directly in the GFT domain without requiring to compute graph power spectral densities or graph correlation functions first.

Wiener filtering approaches existing in related literature have been developed to perform filtering operations in a fixed transform domain, spatial domain or a GFT domain facilitated by a graph formed from the signal of interest. However, if a graph signal of irregular structure or of regular 2-D grid (for an image) is considered as deterministic, the Wiener filter becomes shift varying (node variant or vertex varying for graph signals). In that case, as clarified in detail in this thesis, transfer function of such filter should be formulated in the joint vertex-frequency domain as in the time-frequency (TF) domain for classical signal processing. Therefore, in this thesis, a vertex-frequency (or vertex varying) Wiener filtering scheme is first proposed and developed for denoising of images and graph signals [70]. As another work, a vertex-frequency graph Wiener filter class in the windowed graph Fourier transform (WGFT) domain [71]–[76] is proposed and derived for graph signal denoising as a generalization of the Wiener filter developed in [70] that includes it as a limiting case.

1.2. Related Work

In this thesis, a graph Zadeh vertex-frequency filter is proposed by extending the Rihaczek-Zadeh filtering scheme [77, 78] that is one of the commonly used linear TF filters in the classical signal processing [79, 80]. The proposed Zadeh filter is used for graph signal and image denoising, as a vertex-frequency graph Wiener filter while it can also be employed for seperation of graph signals.

More recent TF Wiener filters, as reviewed in [81], employ the Weyl TF transfer function formulated in terms of Wigner-Ville spectra of the original, noiseless signal and the noise process [81, 82]. However, interpolation required to extend this approach to graph signals makes this process difficult to achieve [3].

Evolutionary spectra [83] of the signal of interest is employed to formulate the transfer function of TF Wiener filters employing Zadeh's TF transfer function [84]–[86]. Nevertheless, this transfer function is also related to the Rihaczek distribution (RD) [87] of the input signal of the time varying filter [78]. Therefore, in this thesis, the Zadeh's TF transfer function [77] is first extended to graph signal filtering, then, a vertex-frequency Wiener filtering scheme is proposed for image and graph signal denoising, whose graph Zadeh's vertex-frequency transfer function is expressed in terms of the graph Rihaczek vertex-frequency energy distribution (GRD) [88, 89] of the original, noiseless signal of interest.

The short-time Fourier transform (STFT) and the wavelet transform domains of the noisy signal have been employed by Wiener filter-type transfer functions in other TF

Wiener filtering approaches. Transfer functions are expressed in terms of spectrograms (magnitude squared STFT) and scalograms (magnitude squared wavelet transform), in [90, 91] and [66], respectively, of the original, noiseless signal and the noise process. Such filtering approaches are also used for image denoising and restoration as in local adaptive filtering proposed in [67]. Windowed image portions are filtered in a fixed transform domain by weighting windowed spectral coefficients by an estimated Wiener filter frequency response specific to that image patch, in this work [67]. Hence, this work is actually a form of space-frequency filtering of the analysis-masking-synthesis type [80] mentioned above. The BM3D method [57, 58] also seems to be a 3-D extension of such a space-frequency filtering; since similar image patches at different locations on a noisy image are grouped together before applying a 3-D Wiener filter to them in a transform domain, making this method a space varying filtering operation. The third dimension is related to the similarity of the grouped patches.

The graph Wiener filtering scheme developed in [69] can be considered as a graph extention of TF Wiener filtering methods employing the STFT whose graph counterparts are the WGFT [71]–[76], or the graph S-transform [73], when it is applied patchwise in a locally adaptive manner. However, it can only be a very decimated and crude way of a vertex-frequency graph filtering operation. Firstly, choosing different image patches [69] amounts to the classical spatial shift operation, instead of a graph shift operation applicable to generic graph signals defined on possibly irregular graphs. Secondly, the GFT coefficients of selected patches are employed in this method, instead of using the WGFT or the S-transform. Therefore, this method proposed in [69] is not appropriate to be generalized for processing of generic graph signals.

Recently proposed vertex-frequency graph signal representations such as the WGFT or the S-transform, mentioned above, have only been employed to analyze vertex varying graph frequency contents of involved signal of interest, so far. They have not been employed yet for any vertex-frequency filtering operations. Nevertheless, such a filtering idea is hinted in [92] as masking the WGFT coefficients of an input, noisy signal and inverting them by inversion methods listed there to attain the filter output signal. This filtering is probably proposed for a denoising operation, since thresholding the WGFT of the input, noisy signal is proposed to be used for obtaining the filter transfer function. However, a filtering application is not obtained in that work. Therefore, as another work presented in this thesis, a vertex-frequency graph Wiener filter framework in the WGFT domain is proposed and developed to denoise graph signals of irregular structure, only. It is achieved by weighting the WGFT of the input, noisy signal by the filter transfer function and then obtaining the filter output signal by one of the WGFT inversion methods presented in [92]. Assuming that the original signal to be recovered is viewed as a deterministic signal, the Wiener vertex-frequency transfer function obtained by minimizing the MSE is analytically expressed in terms of the GRD of the original, noiseless signal of interest and the window signal in the vertex domain.

In some recent works, the idea of denoising the noisy transform coefficients by thresholding has been employed for images [93] and graph signals [94], as a vertex-frequency filtering, in the graph wavelet and multiscale transform domains. For details of such methods and a comprehensive review of multiscale transforms and graph wavelets please see [1, 3, 8, 94] and references therein.

Spatial domain Wiener filtering methods mentioned previously [56], [59]–[62] also involve space varying filtering, but they are not formulated or implemented in a joint space-frequency domain.

As an approach reminiscent to vertex-frequency filtering, convolutional neural networks are devised for graph structured data where WGFT coefficients of such data are implemented, and weighted by network weights acting as spectral multipliers, in network layers [95]–[97] for deformable shape correspondence and classification applications [97, 98]. Although such a localized graph spectral convolution operation is also a form of vertex-frequency filtering, it is not explicitly formulated and designed in a joint vertex-frequency domain.

The node-invariant (shift varying or vertex varying) graph filters introduced in detail in [99] are used in analog network coding to optimally approximate a predetermined linear transformation that is applied to graph signals [99], while they are employed for denoising purposes in this thesis. Nevertheless, the vertex-frequency transfer function is also given in Eq. (4.3). As emphasized below, this node-variant graph frequency response differs from the Zadeh vertex-frequency transfer function and the Zadeh vertex-frequency graph filtering concepts proposed in Chapter 4 of the thesis. Fig. 1.1 presents a flowchart of proposed image denoising methods employing a vertex-frequency transfer function of the proposed graph Zadeh filter.

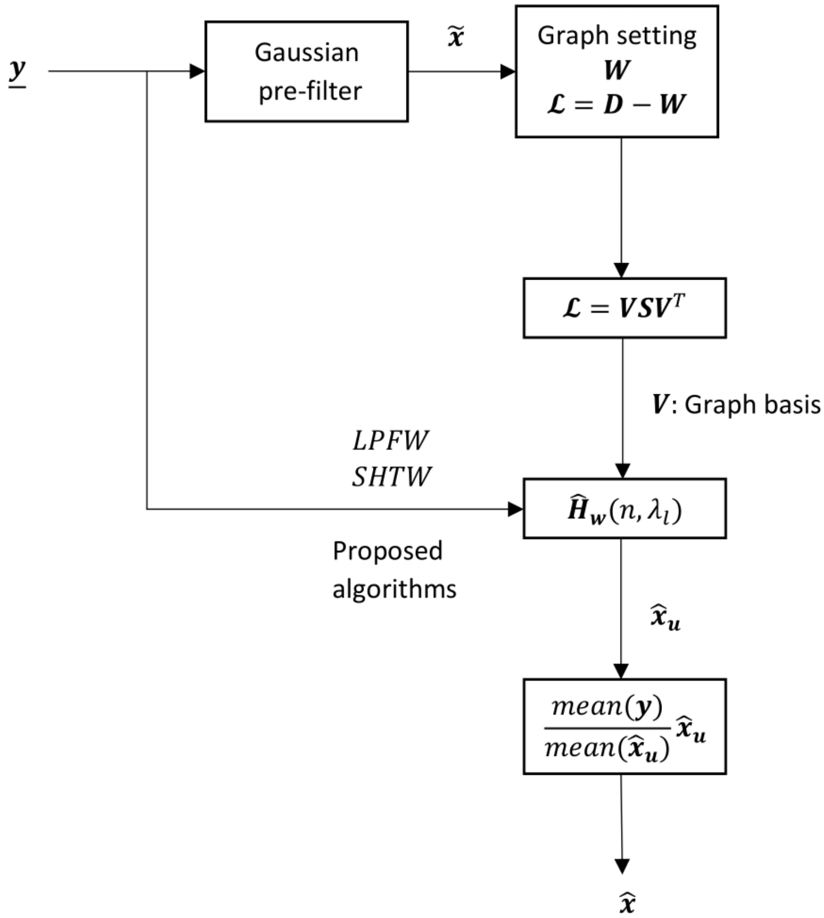

Figure 1.1. *A flowchart demostrating the proposed image denoising methods*

Node-invariant (shift-invariant) graph filters are usually described by impulse reponse matrices expressed as polynomials of GSO matrices [1, 3, 4, 18, 19, 99, 100]. Node-variant filters are obtained in [99] by generalizing scalar polynomial coefficients to diagonal coefficient matrices multiplying powers of the GSO matrix so that each graph node applies different coefficients to corresponding samples of shifted input graph signals. Then, for each graph node, a different graph frequency response is revealed as a polynomial of eigenvalues of the GSO matrix serving as graph frequencies, with polynomial coefficients specific to that node (Eq. (4.3) in [99]). In the proposed graph Zadeh filtering scheme, the entire graph filter impulse response matrix is considered, whether it is expressible as a polynomial of a GSO matrix or not, and a transformation to it is applied in a similar way to the one encountered in the classical Zadeh time varying filtering [77] to obtain a vertex-frequency transfer function of the proposed graph Zadeh filter. This derivation will be given in Chapter 4.

1.3. Thesis Contributions

Contributions of this thesis are listed as follows.

(1) A graph Zadeh vertex-frequency filtering scheme for graph signals is proposed.

(2) A vertex-frequency graph Wiener filter for denoising of such signals is proposed with a vertex-frequency transfer function obtained from transformation of the impulse response of the Wiener filter.

(3) The obtained Wiener transfer function is expressed in terms of the GRD [88, 89] of the original, noiseless signal to be estimated. The form of the attained vertex-frequency transfer function is notably different from than those of TF Wiener filters widely used in the classical signal processing, as reviewed in [81].

(4) The employed GRD originally proposed in [88, 89] has been shown to satisfy vertex and frequency marginal properties in those works; but, a more detailed development of it is presented in this work, as follows:

(4.1) The GRD based on a graph shift operator defined by generalized convolution with a delta signal [3, 8, 72] is derived, among different GSO's defined for graph signals. This derivation follows the original definition of the classical RD [87]: The graph signal of interest is first translated, conjugated and correlated with the unshifted signal, respectively; then, a GFT is finally computed from the graph shift or lag variable. In particular, it is emphasized that first the graph shift operation should be applied to the graph signal than the complex conjugation process, not the other way around, to obtain the GRD proposed in [88, 89] that satisfies marginal properties.

Furthermore, derivation presented in this thesis analogous to the classical one also enables to define a cross GRD of two graph signals on the same graph, which is not defined in [88, 89].

(4.2) It is shown that the vertex-frequency transfer function of the proposed graph Zadeh filter matches the GRD of the filter input signal, in a similar way as has been shown in [78] for the classical case.

(4.3) Mean and variance of the GRD of an observed, noisy graph signal are also derived, since they are employed in one of the Wiener filter based denoising algorithms proposed and implemented in this work.

(4.4) Graph signal of interest is recovered back from its GRD only within a fixed phase factor.

(4.5) Two alternative GRD definitions based on two isometric GSO's developed in [101] and [19] are also derived. It is shown that they satisfy marginal properties only for the standard complex exponential basis, for which case all three GRD definitions reduce to the classical RD. Furthermore, invertibilities of them turn out to be less favorable than that of the GRD proposed in [88, 89].

(5) It is shown that the proposed vertex-frequency graph Wiener filtering scheme is reduced to stationary graph Wiener filters [15, 18, 21, 22] developed for stationary graph signals.

(6) Assuming that the original, noiseless graph signal to be estimated can be considered as deterministic, two algorithms are developed to apply the proposed vertex-frequency graph Wiener filter to denoising. One of them is built on the output signal of a denoising lowpass filter in the GFT domain [3, 6, 7, 21], [28]–[31] to estimate the GRD of the original, noiseless signal that is used to facilitate the transfer function of the Wiener filter. The other algorithm estimates the GRD of the original signal by smoothing and hard thresholding GRD coefficients of the input, noisy graph signal.

(7) The proposed Wiener filter algorithms are applied to denoise various images of different texture and structure such as natural and PWS images including depth and synthetic images, and obtained averaged peak signal-to-noise ratio (PSNR) and structural similarity (SSIM) index values are compared with those yielded by the BM3D, NLM, BF and some recent graph based denoising methods developed especially for PWS images such as Optimal graph Laplacian regularization (OGLR) and Nonlocal graph based transform (NLGBT) methods. Real graph signals such as dendritic tree thickness, Toronto traffic volume and Minnesota road graph data are also denoised by the proposed algorithms, and averaged signal-to-noise (SNR) results are compared with those of graph wavelet shrinkage based denoising methods, such as the hierarchical graph Laplacian eigen transform (HGLET) and the generalized Haar-Walsh transform (GHWT), proposed for graph signal denoising in [94].

(8) As another contribution of this thesis, a vertex-frequency graph Wiener filter class in the WGFT domain for graph signal denoising is also proposed and derived as a generalization of the Wiener filter obtained in [70] that includes it as a limiting case. Under the deterministic signal assumption, a denoising algorithm based on hard thresh-

olding and smoothing that of the observed, noisy signal is proposed and applied to graph signals. The performance of the proposed Wiener filter algorithm in the WGFT domain is compared in averaged SNR results with those of graph wavelet shrinkage based denoising methods [94] mentioned above, in denoising real graph datasets.

1.4. Thesis Organization

Graph signal model and the WGFT are presented in Chapter 2. In Chapter 3, the vertex domain Wiener filter for denoising applications is presented. Graph Zadeh vertex-frequency filtering scheme is proposed in Chapter 4. Vertex-frequency graph Wiener filter developed for denoising of graph signals and images is introduced in Chapter 5. Appendix A presents two general forms of the proposed graph Wiener vertex-frequency transfer function expressed in terms of the graph Rihaczek spectrum (GRS) of the graph signal to be estimated, without the assumption that the graph signal is deterministic. For stationary graph signals, this vertex-frequency transfer function is reduced to the stationary graph Wiener transfer function of [15, 18, 21, 22, 63] developed for graph signals, which is also presented in Appendix A.

Chapter 6 presents derivational details of the GRD based on a graph shift operator defined by generalized convolution with a delta signal [3, 8, 72], in order to match the transfer function of the graph Zadeh filter proposed in Chapter 4. Mean and variance of the GRD of input, noisy graph signal to be used in one of the proposed denoising algorithms are also presented in this chapter and their derivations for the real and complex signals and graph Fourier bases are given in Appendices C and D, respectively, while Appendix B presents two inversion methods for the employed GRD. Two alternative GRD definitions with different isometric GSO's proposed in [101, 19] are separately derived in Appendix E where their invertibilities and whether they satisfy marginal properties or not, are also given. In Chapter 7, an elaborate derivation of the proposed vertex-frequency graph Wiener filtering class in the WGFT is presented with the assumption that the graph signal is deterministic.

Two algorithms are proposed with their one-step and two-step versions, to perform the proposed vertex-frequency graph Wiener filter for image and graph signal denoising in Chapter 8 where another two step algorithm implementing proposed vertex-frequency graph Wiener filtering scheme in WGFT domain is also proposed for graph signal denoising only, as another work. Performances of these algorithms in image and graph signal denoising are separately presented by comparing them with the state-of-the-art methods' denoising results in Chapter 9. Chapter 10 concludes the thesis.

2. BACKGROUND

2.1. Graph Signal Model

A weighted, connected and undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$ contains a set $\mathcal{V}$ of L vertices, a set $\mathcal{E}$ of edges between those vertices, and a weighted adjacency matrix $\mathbf{W}$. If pixels (vertices) m and n are connected via an edge $e = (m, n)$ directed from n to m, the element w_{mn} represents the positive weight of the edge as a measure of similarity between those pixels. If they are not connected, $w_{mn} = 0$. Undirected graphs for which $w_{mn} = w_{nm}$ are considered. Hence, $\mathbf{W}$ is a real, symmetric matrix.

A graph signal $x : \mathcal{V} \to \mathbb{C}$ defined on the vertices of the graph $\mathcal{G}$ can be viewed as a complex valued vector $\mathbf{x} = [x(1), x(2), \ldots, x(L)]^T \in \mathbb{C}^L$ where $x(n)$ represents the signal sample at vertex n of the graph.

An image X of size $M \times N$ can also be considered as a graph signal defined on a 2-D regular grid graph with vertices corresponding to image pixels. A column-stacked vectorized form of the image, $\mathbf{x} \in \mathbb{R}^L$ with dimension $L = MN$ can be regarded as a real valued graph signal defined on such a graph.

The unnormalized graph Laplacian $\boldsymbol{\mathcal{L}}$ is defined as $\boldsymbol{\mathcal{L}} = \mathbf{D} - \mathbf{W}$, where the degree matrix $\mathbf{D}$ is diagonal whose ith diagonal element given by the sum of elements of the ith row of $\mathbf{W}$. Since $\boldsymbol{\mathcal{L}}$ is also a real symmetric matrix, it has a complete set of orthonormal eigenvectors taken as columns of a unitary matrix $\mathbf{V}$ such that $\boldsymbol{\mathcal{L}} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^H$, with $\mathbf{V}\mathbf{V}^H = \mathbf{I}$ [3, 20]. Its eigenvalues are real and nonnegative eigenvalues of the graph Laplacian $\boldsymbol{\mathcal{L}}$, λ_l, for $l = 0, 1, \ldots, L-1$, sorted from the smallest to the largest, with $\lambda_0 = 0$ being the only zero eigenvalue for a connected graph [3]. Eigenvalues represent graph frequencies from the lower to the higher when sorted this way [3, 20, 101].

The GFT $\mathbf{x_f} = [x_f(\lambda_0), x_f(\lambda_1), \ldots, x_f(\lambda_{L-1})]^T$ of the graph signal $\mathbf{x}$ is defined as $\mathbf{x_f} = \mathbf{V}^H\mathbf{x}$ [3, 8], yielding GFT coefficients as

$$x_f(\lambda_l) = \sum_{n=1}^{L} V^*(n, \lambda_l)\, x(n) \tag{2.1}$$

for $l = 0, 1, \ldots, L-1$. The graph signal $\mathbf{x}$ is obtained back from its GFT by the inverse

GFT, $\mathbf{x} = \mathbf{V}\mathbf{x_f}$ [3, 8], revealed as

$$x(n) = \sum_{l=0}^{L-1} V(n, \lambda_l)\, x_f(\lambda_l) \tag{2.2}$$

for $n = 1, 2, \ldots, L$.

The image measurement model is given by

$$\mathbf{y} = \mathbf{x} + \mathbf{w} \tag{2.3}$$

where $\mathbf{y}$ and $\mathbf{x}$ denote column-stacked vectorized forms of the observed noisy and original noiseless images, respectively, of size L. $\mathbf{w}$ is a noise vector of additive, zero-mean, white Gaussian noise (AWGN) samples. For the real signal model, i.e., $\mathbf{x} \in \mathbb{R}^L$, it is assumed that $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ containing independent, identically distributed (i.i.d.), zero-mean Gaussian noise samples with variance σ^2.

For the complex signal model, i.e., $\mathbf{x} \in \mathbb{C}^L$, noise samples, $w(n) = w_R(n) + jw_I(n)$, $n = 1, 2, \ldots, L$, are assumed to be i.i.d., complex, zero-mean, circular Gaussian random variables with variance σ^2; $\mathbb{E}\{w(m)w^*(n)\} = \sigma^2\delta(m-n)$ [102] for which

$$\mathbb{E}\{w_R(m)w_R(n)\} = \mathbb{E}\{w_I(m)w_I(n)\} = (\sigma^2/2)\delta(m-n) \tag{2.4}$$

$$\mathbb{E}\{w_R(m)w_I(n)\} = 0 \tag{2.5}$$

for m, $n = 1, 2, \ldots, L$, where $w_R(n)$ and $w_I(n)$ are real and imaginary parts of $w(n)$, respectively. $\mathbb{E}(\cdot)$ denotes the statistical expectation operation above.

For both real and complex signal models, $\mathbf{x}$ and $\mathbf{w}$ are assumed to be uncorrelated. The denoising problem is to estimate $\mathbf{x}$ given its noisy version $\mathbf{y}$, in the minimum MSE sense.

2.2. Windowed Graph Fourier Transform

The WGFT of the graph signal $\mathbf{x}$ can be expressed in the vertex domain as the GFT of a windowed version of $\mathbf{x}$ [74, 92],

$$S_x(m, \lambda_k) = \sum_{n=1}^{L} x(n) h_m(n) V^*(n, \lambda_k) \tag{2.6}$$

for $m = 1, 2, \ldots, L$ and $k = 0, 1, \ldots, L-1$. $h_m(n)$ is a window function localized around the vertex m, and, $V(n, \lambda_k)$ denotes nth element of the kth eigenvector of the graph Laplacian associated with its eigenvalue λ_k, i.e., an element of the graph basis matrix $\mathbf{V}$.

Five inversion methods are presented for the WGFT in [92] to recover back the signal $\mathbf{x}$. The proposed filtering scheme in the WGFT domain is based on the following [92]:

$$x(n) = \sum_{m=1}^{L} \sum_{k=0}^{L-1} S_x(m, \lambda_k) V(n, \lambda_k) \tag{2.7}$$

for $n = 1, 2, \ldots, L$, when the window function is normalized such that

$$\sum_{m=1}^{L} h_m(n) = 1 \tag{2.8}$$

for each vertex n.

3. VERTEX DOMAIN WIENER FILTER

An estimate of the original, noiseless graph signal $\mathbf{x}$ is obtained as $\widehat{\mathbf{x}} = \mathbf{H}\mathbf{y}$, where $\mathbf{H} = [h(m,n)]_{L\times L}$ is a graph filter matrix [4, 18, 19, 21], in order to minimize

$$\text{MSE} = \frac{1}{L} \times \mathbb{E}\left\{\|\mathbf{x} - \widehat{\mathbf{x}}\|^2\right\} = \frac{1}{L} \times \mathbb{E}\left\{\|\mathbf{x} - \mathbf{H}\mathbf{y}\|^2\right\}.$$

By evaluating derivatives of the MSE with respect to complex conjugates of filter coefficients, $h^*(m,n)$'s, with the assumption that $h(m,n)$ and $h^*(m,n)$ are independent variables according to Wirtinger calculus [103], setting these derivatives to zero, using (2.4) and (2.5) for the complex, circular noise model in the obtained linear system of equations, and solving this equation system, The graph Wiener filter matrix in the vertex domain [15, 104] is obtained as

$$\mathbf{H}_W = \mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\}\left[\sigma^2\mathbf{I} + \mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\}\right]^{-1} = \left[\sigma^2\mathbf{I} + \mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\}\right]^{-1}\mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\}. \tag{3.1}$$

Since the above matrices commute, the second equality above follows from the first one. Under the assumption that the graph signal of interest $\mathbf{x}$ is deterministic; by removing the statistical expectation $\mathbb{E}(\cdot)$ and applying the matrix inversion lemma [102], (3.1) can be written as

$$\mathbf{H}_W = \frac{\mathbf{x}\mathbf{x}^H}{\sigma^2 + \|\mathbf{x}\|^2} \tag{3.2}$$

where $\|\mathbf{x}\|^2 = \mathbf{x}^H\mathbf{x}$ above. For the real signal model, $\mathbf{H}_W$ is obtained simply by replacing the Hermitian operators with the transpose operators, i.e., $\mathbf{x}^H$ terms with $\mathbf{x}^T$ in (3.1) and (3.2) via a similar derivation.

The real form of the ideal (true) vertex domain graph Wiener filter given by (3.2) is applied to denoise vectorized forms of 14 noisy gray scale images of size 256×256 with zero-mean AWGN with standard deviation of 15. PSNR values are calculated in dB using

$$\text{PSNR} = 10\log_{10}\left(\frac{255^2}{\text{MSE}}\right)$$

Table 3.1 presents PSNR values averaged over 20 independent trials. For normalized images, 255 is replaced by 1 in PSNR definition above. Remarkable results obtained not only for PWS synthetic and depth images but also for natural images as presented in Table

Table 3.1. *Ideal (true) Wiener filter performances in average PSNR are presented for noise level of 15*

Patch Size	***Image***													
	Box	Stripes	Teddy	Cones	C.man	Montage	Peppers	House	Lena	Boats	Barbara	Parrot	Baboon	Grass
64×64	60.85	61.09	60.82	60.61	60.57	60.31	60.21	60.89	60.97	60.75	60.91	60.89	60.66	60.78
128×128	66.14	66.96	66.67	66.58	67.09	67.42	66.18	66.92	67.01	67.15	66.78	67.48	66.46	67.26
256×256	77.59	79.88	77.93	77.85	78.51	79.61	78.42	79.57	78.97	76.74	77.29	75.67	74.99	73.36

3.1, hint that it is reasonable to consider even textured natural images, as noise-like as the grass image, as deterministic graph signals.

From (3.2), the vertex domain Wiener filter coefficients $h_W(m,n) \propto x(m)x^*(n)$. Therefore, it is obvious that at least for signals defined over regular grid graphs like images, the Wiener filter $\mathbf{H}_W$ becomes vertex varying (or shift varying), since it is not a Toeplitz matrix with elements $h_W(m,n)$ dependent on $m-n$, in general. Hence, Wiener filter transfer function should better be formulated in the joint vertex-frequency domain, as developed in Chapter 5.

4. GRAPH ZADEH VERTEX-FREQUENCY FILTERING

The Zadeh TF filtering concept is generalized to the graph signal setting and a graph Zadeh vertex-frequency filtering scheme is proposed in this chapter. Its derivation is similar to that of the classical Zadeh TF filter [77].

Let $y_f(\lambda_l)$, $l = 0, 1, \ldots, L-1$ denote GFT coefficients of an observed, noisy graph signal $\mathbf{y}$ computed according to the GFT definition given by (2.1). To obtain output signal samples $\widehat{x}(n)$, $n = 1, 2, \ldots, L$, a graph Zadeh filter should weight $V(n, \lambda_l)y_f(\lambda_l)$ terms in the inverse GFT definition (2.2) by coefficients of a vertex-frequency transfer function $\mathcal{H}(n, \lambda_l)$:

$$\widehat{x}(n) = \sum_{l=0}^{L-1} V(n, \lambda_l)\, \mathcal{H}(n, \lambda_l)\, y_f(\lambda_l) \tag{4.1}$$

for $n = 1, 2, \ldots, L$. Input GFT coefficients $y_f(\lambda_l)$ computed according to (2.1) is then substituted into (4.1);

$$\widehat{x}(n) = \sum_{l=0}^{L-1} V(n, \lambda_l)\, \mathcal{H}(n, \lambda_l) \left[\sum_{m=1}^{L} V^*(m, \lambda_l)\, y(m) \right], \tag{4.2}$$

$n = 1, 2, \ldots, L$. The output signal $\widehat{x}(n)$ is attained in terms of the input signal $y(m)$ by interchanging the order of summations in (4.2) as

$$\widehat{x}(n) = \sum_{m=1}^{L} h(n, m)\, y(m) \tag{4.3}$$

$n = 1, 2, \ldots, L$, where impulse response samples are expressed as

$$h(n, m) = \sum_{l=0}^{L-1} V^*(m, \lambda_l) V(n, \lambda_l) \mathcal{H}(n, \lambda_l) \tag{4.4}$$

n, $m = 1, 2, \ldots, L$, in terms of the vertex-frequency transfer function $\mathcal{H}(n, \lambda_l)$. Therefore, from (4.4), the impulse response matrix $\mathbf{H} = [h(n, m)]_{L\times L}$ of the graph Zadeh filter can be written in terms of its vertex-frequency transfer function matrix $\boldsymbol{\mathcal{H}} = [\mathcal{H}(n, \lambda_l)]_{L\times L}$ as

$$\mathbf{H} = (\mathbf{V} \circ \boldsymbol{\mathcal{H}})\, \mathbf{V}^H. \tag{4.5}$$

Since $\mathbf{V}^H\mathbf{V} = \mathbf{I}$, by using the impulse response matrix of the graph Zadeh filter, its vertex-frequency transfer function matrix is obtained as

$$\boldsymbol{\mathcal{H}} = \mathbf{V}^{-\circ} \circ (\mathbf{H}\mathbf{V})\, . \tag{4.6}$$

(4.5) and (4.6) relating the vertex-domain impulse response and the vertex-frequency transfer function to each other constitute the proposed graph Zadeh vertex-frequency filtering scheme. The output signal $\widehat{\mathbf{x}}$ can be computed from either the filter input, noisy signal $\mathbf{y}$ (or GFT coefficients of it $y_f(\lambda_l)$) directly in the joint vertex-frequency domain by (4.1) for given the vertex-frequency transfer function $\mathcal{H}(n, \lambda_l)$. Alternatively, impulse response of the filter $\mathbf{H}$ can be computed by (4.5) and the output signal of the filter can be computed subsequently in the vertex domain as $\widehat{\mathbf{x}} = \mathbf{H}\mathbf{y}$.

The proposed Zadeh filter can also be used for separation of graph signals. Firstly, the GRD [88, 89] or the WGFT [71, 72] of the input, noisy signal $\mathbf{y}$ can be computed to obtain its joint vertex-frequency display. Then, vertex-frequency region of the desired signal component in $\mathbf{y}$ can be identified and the Zadeh transfer function $\mathcal{H}(n, \lambda_l)$ can be set to isolate corresponding region. Finally, the designed graph Zadeh filter as such can be implemented to the input, noisy signal $\mathbf{y}$, to recover the desired component as the filter output $\widehat{\mathbf{x}}$.

However, in this work, this Zadeh vertex-frequency filtering scheme is employed to develop a vertex-frequency graph Wiener filtering framework for image and graph signal denoising, only. This is achieved by transforming the impulse response matrix of the Wiener filter $\mathbf{H}$, via (4.6), to obtain its vertex-frequency transfer function $\boldsymbol{\mathcal{H}}$, in Chapter 5.

5. VERTEX-FREQUENCY GRAPH WIENER FILTER

The vertex-frequency transfer function of the proposed graph Wiener filter is derived in this chapter, under the assumption that the original graph signal of interest is deterministic. For the random graph signal case, the graph Wiener vertex-frequency transfer function will be derived in Appendix A.

The Wiener impulse response matrix $\mathbf{H}_W$ given in (3.2) is substituted into (4.6). Then, by using $\mathbf{x_f} = \mathbf{V}^H\mathbf{x}$ expression as the GFT of $\mathbf{x}$, the vertex-frequency transfer function matrix of the Wiener filter is obtained as

$$\boldsymbol{\mathcal{H}}_W = \mathbf{V}^{-\circ} \circ \left(\frac{\mathbf{x}\mathbf{x_f}^H}{\sigma^2 + \|\mathbf{x}\|^2} \right). \tag{5.1}$$

The Wiener vertex-frequency transfer function coefficients, $\mathcal{H}_W(n, \lambda_l)$, are defined as

$$\mathcal{H}_W(n, \lambda_l) = \frac{x(n)\, x_f^*(\lambda_l)}{V(n, \lambda_l)\,(\sigma^2 + \|\mathbf{x}\|^2)}, \tag{5.2}$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$. The GRD of the original, noiseless graph signal $\mathbf{x}$, as introduced in [88, 89],

$$R_x(n, \lambda_l) = x(n)\, x_f^*(\lambda_l)\, V^*(n, \lambda_l), \tag{5.3}$$

the Wiener vertex-frequency transfer function in (5.2) is expressed in terms of (5.3) as

$$\mathcal{H}_W(n, \lambda_l) = \frac{R_x(n, \lambda_l)}{|V(n, \lambda_l)|^2\,(\sigma^2 + \|\mathbf{x}\|^2)} \tag{5.4}$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$.

(5.3), (5.4) and (4.1) constitute the proposed vertex-frequency graph Wiener filter for signal estimation in the minimum MSE sense, under the deterministic signal assumption. The Wiener filter vertex-frequency transfer function given in (5.4) is applied to the GFT coefficients of the observed, noisy image $\mathbf{y}$ modeled as (2.3)–(2.5), via (4.1), by substituting (5.4) into (4.1). The denoised signal $\widehat{\mathbf{x}}$ is obtained as the output of the Wiener filter.

For the real signal model, i.e., the graph signal $\mathbf{x}$ and the basis matrix $\mathbf{V}$ are real, the denominator term of (5.4), $|V(n, \lambda_l)|^2$, is replaced with $V^2(n, \lambda_l)$ and the GRD is defined as [88, 89]

$$R_x(n, \lambda_l) = x(n)\, x_f(\lambda_l)\, V(n, \lambda_l).$$

In a practical setting, the GRD coefficients of the original, noiseless signal, $\widehat{R}_x(n, \lambda_l)$, should be estimated from those of the observed, noisy, real signal, given by

$$R_y(n, \lambda_l) = y(n)\, y_f(\lambda_l)\, V(n, \lambda_l), \tag{5.5}$$

and an estimate of the Wiener vertex-frequency transfer function in (5.4) should be employed for the denoising graph filter, given as

$$\widehat{\mathcal{H}}_W(n, \lambda_l) = \max\left\{\min\left[\frac{\widehat{R}_x(n, \lambda_l)}{V^2(n, \lambda_l)}, \overline{U}\right], \overline{L}\right\} \tag{5.6}$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$. Division by squares of eigenbasis elements $V^2(n, \lambda_l)$ implying division by magnitudewise very small, positive or negative numbers yields magnitudewise very large, positive or negative numbers, therefore, $\overline{U}$ and $\overline{L}$, upper and lower bounds, respectively, are used for limiting transfer function values. Numerical values of these bounds should be searched to result in a good denoising performance.

To obtain the denoised signal $\widehat{\mathbf{x}}$, the constant scale factor $\sigma^2 + \|\mathbf{x}\|^2$ in the denominator in (5.4) is corrected by using average or sample mean values of the observed, noisy signal $\mathbf{y}$ and the unscaled output of the filter given by (5.6), denoted as $\widehat{\mathbf{x}}_{\mathbf{u}}$, as

$$\widehat{\mathbf{x}} = \frac{\text{mean}(\mathbf{y})}{\text{mean}(\widehat{\mathbf{x}}_{\mathbf{u}})}\widehat{\mathbf{x}}_{\mathbf{u}} \tag{5.7}$$

since $\text{mean}(\mathbf{y}) \approx \text{mean}(\mathbf{x})$ for the zero-mean WGN.

The proposed vertex-frequency graph Wiener filtering framework includes (i) estimating the GRD of the original, noiseless signal from that of the observed, noisy signal, $R_y(n, \lambda_l)$, (ii) by using this estimate, $\widehat{R}_x(n, \lambda_l)$, establishing the filter vertex-frequency transfer function as in (5.6), (iii) applying this filter with (5.6) to the GFT of the observed, noisy signal $\mathbf{y}$, as in (4.1), to obtain the unscaled filter output $\widehat{\mathbf{x}}_{\mathbf{u}}$, (iv) and finally, correcting the scale of $\widehat{\mathbf{x}}_{\mathbf{u}}$ by (5.7) to obtain the denoised signal $\widehat{\mathbf{x}}$. Two denoising algorithms proposed to implement this filtering scheme will be given in Chapter 8.

6. GRAPH RIHACZEK VERTEX-FREQUENCY DISTRIBUTION

The GRD of a graph signal that is originally proposed in [88, 89] is derived analogous to the definition of the classical RD, in this chapter.

The RD of an analog signal $x(t)$ is defined as [87]

$$R_x(t,w) = \int_{-\infty}^{\infty} x(t)\, x^*(t-\tau)\, e^{-jw\tau}\, d\tau$$

where the Fourier transform is computed from the lag variable to the frequency variable. In a similar way, the GFT operation is applied to the graph shift index in this derivation of the GRD below.

Let $\mathbf{x}_k$ be the shifted (translated) version of a graph signal $\mathbf{x}$ defined on the graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$. Graph shift is applied by employing a graph translation operator defined by generalized convolution with a delta signal $\boldsymbol{\delta}_k$ located at vertex k, as recommended in [3, 8, 72]

$$\mathbf{x}_k = \sqrt{L}\,\mathbf{V}\left[\mathbf{x_f} \circ \left(\mathbf{V}^H \boldsymbol{\delta}_k\right)\right] = \sqrt{L}\,\mathbf{V}\left[\mathbf{x_f} \circ \mathbf{V}(k,:)^H\right] \tag{6.1}$$

for $k = 1, 2, \ldots, L$. From (6.1), nth sample of the k-shifted signal located at vertex n, is written as

$$x_k(n) = \sqrt{L}\,\mathbf{V}(n,:)\left[\mathbf{x_f} \circ \mathbf{V}(k,:)^H\right] = \sqrt{L}\left[\mathbf{x_f}^T \circ \mathbf{V}^*(k,:)\right]\mathbf{V}(n,:)^T \tag{6.2}$$

for $n = 1, 2, \ldots, L$. Collecting nth samples of the shifted versions of the graph signal $\mathbf{x}$, a new graph signal $\mathbf{x_s}(n) = [x_1(n), \ldots, x_k(n), \ldots, x_L(n)]^T$ is obtained, where the discrete independent variable is the graph shift index k. The GFT of this new signal is computed after taking the complex conjugate of it, as

$$\mathbf{x_s}^*(n) = \sqrt{L}\left[\mathbf{V} \circ \left(\mathbf{x_f}, \mathbf{x_f}, \ldots, \mathbf{x_f}\right)^H\right]\mathbf{V}(n,:)^H, \tag{6.3}$$

to derive the GRD of the graph signal $\mathbf{x}$. (6.3) is obtained by concatenating the second equality in (6.2) for $k = 1, 2, \ldots, L$ to attain the column vector $\mathbf{x_s}(n)$, and then by complex conjugation of its elements.

The GRD of the graph signal $\mathbf{x}$ based on the GFT of $\mathbf{x_s}^*(n)$ is defined in a similar way to the form of the classical RD, as

$$R_x(n,\lambda_l) = x(n)\mathbf{V}^H(\lambda_l,:)\,\mathbf{x_s}^*(n)/\sqrt{L} \tag{6.4}$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$. By substituting (6.3) into (6.4), and employing $\mathbf{V}^H(\lambda_l, :) = \mathbf{V}(:, \lambda_l)^H$,

$$R_x(n, \lambda_l) = x(n)\mathbf{V}(:, \lambda_l)^H \left[\mathbf{V} \circ (\mathbf{x_f}, \mathbf{x_f}, \ldots, \mathbf{x_f})^H\right] \mathbf{V}(n, :)^H. \tag{6.5}$$

Expanding the matrix-vector products above,

$$R_x(n, \lambda_l) = x(n) \sum_{k=1}^{L} \sum_{m=0}^{L-1} V^*(k, \lambda_l) V(k, \lambda_m) x_f^*(\lambda_m) V^*(n, \lambda_m). \tag{6.6}$$

Interchanging the order of summations in (6.6), and then substituting

$$\sum_{k=1}^{L} V^*(k, \lambda_l) V(k, \lambda_m) = \delta(l - m)$$

since $\mathbf{V}\mathbf{V}^H = \mathbf{V}^H\mathbf{V} = \mathbf{I}$,

$$R_x(n, \lambda_l) = x(n) \sum_{m=0}^{L-1} x_f^*(\lambda_m) V^*(n, \lambda_m) \delta(l - m) = x(n)\, x_f^*(\lambda_l)\, V^*(n, \lambda_l) \tag{6.7}$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$. The same GRD definition developed in [88, 89], as given in (5.3), is obtained above.

The derivational details of the GRD definition given above are as follows. The graph signal of interest is first translated, conjugated and correlated with the unshifted signal, respectively; then, a GFT is finally computed from the graph shift or lag variable.

It is emphasized that if the order of these operations is reversed, i.e., first conjugation and then translation is applied to the graph signal, the GRD definition in [88, 89] can not be achieved for graph signals and bases, with complex values; resulting that the GRD can not be expressed with a simple definition.

Appendix B presents two inversion methods for the developed GRD $R_x(n, \lambda_l)$, given in (5.3). The graph signal $\mathbf{x}$ is recovered from its GRD $R_x(n, \lambda_l)$ within a fixed phase factor or a fixed sign for the complex and real signal cases, respectively.

Let $\mathbf{x_1}$ and $\mathbf{x_2}$ be two graph signals defined on the same graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{W}\}$, the cross GRD of them is obtained by applying graph translation and conjugation operations to the second graph signal $\mathbf{x_2}$, as

$$R_{x_1,x_2}(n, \lambda_l) = x_1(n)\, x_{2,f}^*(\lambda_l)\, V^*(n, \lambda_l)$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$, where $x_{2,f}(\lambda_l)$'s denote the GFT coefficients of the second graph signal $\mathbf{x_2}$.

6.1. Graph Zadeh Transfer Function and the GRD

Let $\mathbf{y}$ be an observed, noisy signal of a linear graph filter with filter matrix $\mathbf{H}$. The inner product of the observed, noisy signal $\mathbf{y}$ and the filter output signal $\mathbf{Hy}$ is written as

$$\langle \mathbf{y}, \mathbf{Hy} \rangle = \mathbf{y}^H \mathbf{Hy} = \mathbf{y}^H \left(\mathbf{V} \circ \boldsymbol{\mathcal{H}} \right) \mathbf{y_f} \tag{6.8}$$

where the second equality is obtained by using (4.5) for $\mathbf{H}$ and recognizing the GFT $\mathbf{y_f} = \mathbf{V}^H \mathbf{y}$.

Expanding (6.8),

$$\langle \mathbf{y}, \mathbf{Hy} \rangle = \sum_{n=1}^{L} \sum_{l=0}^{L-1} y^*(n) V(n, \lambda_l) \mathcal{H}(n, \lambda_l) y_f(\lambda_l) = \sum_{n=1}^{L} \sum_{l=0}^{L-1} \mathcal{H}(n, \lambda_l) R_y^*(n, \lambda_l) \tag{6.9}$$

recognizing $R_y(n, \lambda_l) = y(n)\, y_f^*(\lambda_l)\, V^*(n, \lambda_l)$.

(6.9) reads

$$\langle \mathbf{y}, \mathbf{Hy} \rangle = \langle \boldsymbol{\mathcal{H}}, \mathbf{R_y} \rangle \tag{6.10}$$

where $\mathbf{R_y} = \left[R_y(n, \lambda_l) \right]_{L \times L}$ is the GRD matrix of the observed signal, yielding an inner product of the graph Zadeh vertex-frequency transfer function and the GRD of the observed, noisy signal in the vertex-frequency domain. Therefore, the vertex-frequency transfer function of the proposed graph Zadeh filter matches the GRD of the observed, noisy signal as a weighting function in the corresponding domain, as has been shown in [78] for classical RD and the Zadeh TF transfer function.

6.2. Mean and Variance of the GRD

Proposition 1: For a real, observed, noisy signal $\mathbf{y}$ given in signal measurement model (2.3), where the original, noiseless signal $\mathbf{x}$ is deterministic and the AWG noise vector $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$, and for a real basis matrix $\mathbf{V}$ obtained from graph Laplacian, mean and variance of the GRD of $\mathbf{y}$ are as follows

$$\mathbb{E}\left\{ R_y(n, \lambda_l) \right\} = R_x(n, \lambda_l) + \sigma^2 V^2(n, \lambda_l), \tag{6.11}$$

$$\text{Var}\left\{R_y(n,\lambda_l)\right\} = \sigma^2 V^2(n,\lambda_l)\left[x^2(n) + x_f^2(\lambda_l)\right] +$$
$$2\sigma^2 V^2(n,\lambda_l) R_x(n,\lambda_l) + \sigma^4 V^2(n,\lambda_l)\left[1 + V^2(n,\lambda_l)\right]. \tag{6.12}$$

Proof: Please see Appendix C.

Proposition 2: For the complex, observed, noisy signal $\mathbf{y}$ given in signal measurement model (2.3), circular Gaussian random variables (2.4) and (2.5), where the original, noiseless complex signal $\mathbf{x}$ is deterministic, and for a complex basis matrix $\mathbf{V}$, mean and variance of the GRD of $\mathbf{y}$ are as follows

$$\mathbb{E}\left\{R_y(n,\lambda_l)\right\} = R_x(n,\lambda_l) + \sigma^2\left|V(n,\lambda_l)\right|^2, \tag{6.13}$$

$$\text{Var}\left\{R_y(n,\lambda_l)\right\} = \sigma^4\left|V(n,\lambda_l)\right|^2 + \sigma^2\left|V(n,\lambda_l)\right|^2\left[|x(n)|^2 + |x_f(\lambda_l)|^2\right]. \tag{6.14}$$

Proof: Please see Appendix D.

7. WIENER FILTER IN THE WGFT DOMAIN

7.1. Derivation

An estimate of the original signal $\mathbf{x}$ is obtained from its noisy version $\mathbf{y}$ by filtering in the WGFT domain;

$$\widehat{x}(n) = \sum_{m=1}^{L} \sum_{k=0}^{L-1} S_y(m, \lambda_k) \mathcal{H}(m, \lambda_k) V(n, \lambda_k) \tag{7.1}$$

for $n = 1, 2, \ldots, L$. (2.7) is modified by masking the WGFT of the input signal $\mathbf{y}$ with the transfer function $\mathcal{H}(m, \lambda_k)$ of a denoising filter before the inversion, to yield (7.1) above. The transfer function is chosen to minimize the MSE,

$$\text{MSE} = \frac{1}{L} \times \mathbb{E} \left\{ \sum_{n=1}^{L} |x(n) - \widehat{x}(n)|^2 \right\}$$

By substituting (7.1) into MSE expression, setting derivatives of the MSE with respect to conjugated coefficients $\mathcal{H}^*(m, \lambda_k)$'s to zero with the assumption that they are independent from $\mathcal{H}(m, \lambda_k)$'s according to Wirtinger calculus [103], using $\mathbb{E}\{w(m)w^*(n)\} = \sigma^2 \delta(m-n)$ [102] for the complex, circular noise model, and, $\sum_{n=1}^{L} V(n, \lambda_{k'}) V^*(n, \lambda_k) = \delta(k' - k)$ for the orthonormal graph basis, the linear system of equations to solve for $\mathcal{H}(m, \lambda_k)$ coefficients are obtained as

$$\sum_{m'=1}^{L} \mathbb{E} \left\{ S_y(m', \lambda_k) S_y^*(m, \lambda_k) \right\} \mathcal{H}(m', \lambda_k) = \sum_{n=1}^{L} \mathbb{E} \left\{ x(n) S_y^*(m, \lambda_k) \right\} V^*(n, \lambda_k) \tag{7.2}$$

for each $m = 1, 2, \ldots, L$, $k = 0, 1, \ldots, L-1$. It is assumed that the graph basis elements $V(n, \lambda_k)$'s are cleaned from noise, and hence, can be viewed as deterministic [20, 28], above.

Assuming that $\mathbf{x}$ is deterministic, substituting for $S_y(m, \lambda_k)$ according to definition (2.6),

$$\mathbb{E} \left\{ x(n) S_y^*(m, \lambda_k) \right\} = x(n) S_x^*(m, \lambda_k) \tag{7.3}$$

for the zero-mean noise $w(n)$. Substituting (7.3) into the righthand side of (7.2),

$$\sum_{n=1}^{L} \mathbb{E} \left\{ x(n) S_y^*(m, \lambda_k) \right\} V^*(n, \lambda_k) = x_f(\lambda_k) S_x^*(m, \lambda_k) \tag{7.4}$$

where $x_f(\lambda_k) = \sum_{n=1}^{L} x(n) V^*(n, \lambda_k)$ is the GFT coefficient of $\mathbf{x}$ for the eigenvalue λ_k.

By substituting for $S_y(m, \lambda_k)$ and using the zero-mean, complex, circular noise model $\mathbb{E}\left\{w(m)w^*(n)\right\} = \sigma^2\delta(m-n)$,

$$\mathbb{E}\left\{S_y(m', \lambda_k)S_y^*(m, \lambda_k)\right\} = S_x(m', \lambda_k)S_x^*(m, \lambda_k) + \sigma^2 \sum_{p=1}^{L} h_{m'}(p)h_m^*(p)\left|V(p, \lambda_k)\right|^2. \tag{7.5}$$

By substituting (7.4) and (7.5) into (7.2), and defining the transfer coefficient vector $\boldsymbol{\mathcal{H}}(:, \lambda_k) = [\mathcal{H}(m, \lambda_k)]_{m=1}^{L}$, the original WGFT vector $\mathbf{S_x}(\lambda_k) = [S_x(m, \lambda_k)]_{m=1}^{L}$, and $\mathbf{h_k}(p) = \sigma V(p, \lambda_k)\left[h_m^*(p)\right]_{m=1}^{L}$, the equation system (7.2) is expressed in the vectorized form

$$\left[\mathbf{S_x}^*(\lambda_k)\,\mathbf{S_x}^T(\lambda_k) + \sum_{p=1}^{L}\mathbf{h_k}(p)\mathbf{h_k}^H(p)\right]\boldsymbol{\mathcal{H}}(:, \lambda_k) = x_f(\lambda_k)\mathbf{S_x}^*(\lambda_k) \tag{7.6}$$

by concatenating equations indexed by m, for each frequency index $k = 0, 1, \ldots, L-1$.

Assuming that $\mathbf{h_k}(p)$ vectors are linearly independent, $\mathbf{A} = \sum_{p=1}^{L}\mathbf{h_k}(p)\mathbf{h_k}^H(p)$ appearing above becomes a full rank, invertible matrix. Then, from the matrix inversion lemma [102],

$$\begin{aligned}\left[\mathbf{A} + \mathbf{S_x}^*(\lambda_k)\,\mathbf{S_x}^T(\lambda_k)\right]^{-1} &= \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{S_x}^*(\lambda_k)\\ &\times\left[1 + \mathbf{S_x}^T(\lambda_k)\,\mathbf{A}^{-1}\mathbf{S_x}^*(\lambda_k)\right]^{-1}\mathbf{S_x}^T(\lambda_k)\,\mathbf{A}^{-1}.\end{aligned} \tag{7.7}$$

By solving (7.6) and then substituting (7.7) therein, the Wiener transfer function vector is obtained, $\boldsymbol{\mathcal{H}}_W(:, \lambda_k)$, that minimizes the MSE, as

$$\boldsymbol{\mathcal{H}}_W(:, \lambda_k) = \frac{x_f(\lambda_k)\mathbf{A}^{-1}\mathbf{S_x}^*(\lambda_k)}{1 + \mathbf{S_x}^T(\lambda_k)\,\mathbf{A}^{-1}\mathbf{S_x}^*(\lambda_k)} \tag{7.8}$$

after simplifying the expression, for $k = 0, 1, \ldots, L-1$.

Let us define $\boldsymbol{a} = \mathbf{A}^{-1}\mathbf{S_x}^*(\lambda_k)$ that appears in (7.8). It is the unique solution of the equation system $\mathbf{A}\boldsymbol{a} = \mathbf{S_x}^*(\lambda_k)$, namely,

$$\sum_{p=1}^{L}\mathbf{h_k}(p)\left[\mathbf{h_k}^H(p)\boldsymbol{a}\right] = \mathbf{S_x}^*(\lambda_k). \tag{7.9}$$

Let us define $b(p) = \mathbf{h_k}^H(p)\boldsymbol{a}$ above. Then, from (7.9), $b(p)$'s are common scalar unknowns of all the equations

$$\sum_{p=1}^{L} \mathbf{h_k}(p)(m)b(p) = S_x^*(m, \lambda_k) \tag{7.10}$$

for $m = 1, 2, \ldots, L$, corresponding to individual equations involved in (7.9). Substituting definitions of $\mathbf{h_k}(p)$ and $S_x(m, \lambda_k)$ (given by (2.6)) into (7.10),

$$\sigma \sum_{p=1}^{L} V(p, \lambda_k) h_m^*(p) b(p) = \sum_{p=1}^{L} x^*(p) h_m^*(p) V(p, \lambda_k). \tag{7.11}$$

$b(p)$'s satisfy (7.11) uniquely, if

$$b(p) = \sum_{m=1}^{L} \mathbf{h_k}(p)^*(m) a(m) = x^*(p)/\sigma \tag{7.12}$$

for $p = 1, 2, \ldots, L$, since they are uniquely defined in terms of $\boldsymbol{a}$ as given by the first equality above. By substituting for the elements of $\mathbf{h_k}(p)$ in (7.12),

$$\sum_{m=1}^{L} h_m(p) a(m) = \frac{x^*(p)}{\sigma^2 V^*(p, \lambda_k)} \tag{7.13}$$

for $p = 1, 2, \ldots, L$. A window matrix $\mathbf{H}$ with elements $H_{p,m} = h_m(p)$ on the pth row and mth column is defined. Then, from (7.13)

$$\boldsymbol{a} = \mathbf{H}^{-1} \left[\frac{x^*(n)}{\sigma^2 V^*(n, \lambda_k)} \right]_{n=1}^{L}. \tag{7.14}$$

The term in the denominator of (7.8) is

$$1 + \mathbf{S_x}^T(\lambda_k)\, \boldsymbol{a} = 1 + \sum_{m=1}^{L} S_x(m, \lambda_k) a(m). \tag{7.15}$$

By substituting (2.6) into (7.15), interchanging the order of summations, and then substituting (7.13),

$$1 + \mathbf{S_x}^T(\lambda_k)\, \boldsymbol{a} = 1 + \|\mathbf{x}\|^2/\sigma^2, \tag{7.16}$$

where $\|\mathbf{x}\|^2 = \mathbf{x}^H \mathbf{x}$.

By substituting (7.14) and (7.16) into (7.8), and using the definition of the GRD of $\mathbf{x}$ given by [88, 89], as in (5.3)

$$R_x(n, \lambda_k) = x(n)\, x_f^*(\lambda_k)\, V^*(n, \lambda_k),$$

the Wiener transfer function vector is obtained as

$$\boldsymbol{\mathcal{H}}_W\left(:, \lambda_k\right) = \frac{1}{\sigma^2 + \|\mathbf{x}\|^2}\mathbf{H}^{-1}\left[\frac{R_x^*(n, \lambda_k)}{|V(n, \lambda_k)|^2}\right]_{n=1}^{L} \tag{7.17}$$

in terms of the GRD of the original graph signal, for each $k = 0, 1, \ldots, L-1$. (7.17) is also valid for the unitary DFT basis.

The proposed Wiener filtering is illustrated for real valued graph data and a real, orthonormal basis $\mathbf{V}$, in this work. Real form of the Wiener transfer function can be derived starting from the real signal model in (2.3), by a similar derivation. It follows directly from (7.17) by noting that real terms are equal to their conjugates and will replace them in (7.17).

7.2. Graph Wide-Sense Stationary Signal Case

If $\mathbf{x}$ is a graph wide-sense stationary (GWSS) signal, its autocorrelation matrix is diagonalizable by eigenvectors of the graph Laplacian matrix [15, 21, 22]; $\mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\} = \mathbf{V}\boldsymbol{\Lambda}_{\mathbf{x}}\mathbf{V}^H$ with diagonal elements of $\boldsymbol{\Lambda}_{\mathbf{x}}$, $\Lambda_x(\lambda_k)$, $k = 0, 1, \ldots, L-1$, serving as graph power spectral density coefficients of $\mathbf{x}$. Then, a similar derivation to the one presented for the deterministic $\mathbf{x}$ above leads to the following vertex invariant Wiener transfer function;

$$\boldsymbol{\mathcal{H}}_W\left(:, \lambda_k\right) = \frac{\Lambda_x(\lambda_k)}{\Lambda_x(\lambda_k) + \sigma^2}\mathbf{H}^{-1}\mathbf{1}, \tag{7.18}$$

$k = 0, 1, \ldots, L-1$, for this GWSS signal case. $\mathbf{1}$ denotes an L-dimensional ones vector above.

8. ALGORITHMS

8.1. Image Denoising

In this thesis, two vertex-frequency Wiener filtering algorithms based on lowpass filtering, and, smoothing and hard thresholding are proposed for image denoising. Firstly, common operations used in both proposed algorithms such as pre-filtering step and constructing the weighted adjacency matrix are mentioned as follows. Gaussian filtering is first applied to vectorized input, noisy image to stabilize the graph weigths by reducing the effect of noise, as recommended in [20, 28, 45, 52], and graphs employed in denoising algorithms are constructed from this filtered image $\tilde{\mathbf{x}}$. The weighted adjacency matrix is then computed in a form reminiscent to that of the BF [45, 49], as follows

$$w_{mn} = \exp\left(-\frac{\|\tilde{\mathbf{x}}_{\mathbf{m}} - \tilde{\mathbf{x}}_{\mathbf{n}}\|^2}{h_x^2}\right) \exp\left(-\frac{\|\mathbf{p}_{\mathbf{m}} - \mathbf{p}_{\mathbf{n}}\|^2}{h_p^2}\right) \tag{8.1}$$

for $m,\, n = 1, 2, \ldots, L$, where h_x and h_p are smoothing parameters in the photometric and pixel (spatial) domains, respectively. $\tilde{\mathbf{x}}_{\mathbf{m}}$ and $\tilde{\mathbf{x}}_{\mathbf{n}}$ denote vectorized forms of patches centered at pixels m and n in the Gaussian pre-filtered image, and $\mathbf{p}_{\mathbf{m}}$ and $\mathbf{p}_{\mathbf{n}}$ denote their locations in the 2-D image, in spatial domain, respectively. Each vertex (pixel) is connected to each other in the 2-D image, with the weights in (8.1). The orthonormal eigenvectors serving as the graph basis $\mathbf{V}$ are obtained with the eigendecomposition of the unnormalized Laplacian $\boldsymbol{\mathcal{L}} = \mathbf{D} - \mathbf{W}$ derived from the weighted adjacency matrix $\mathbf{W}$.

8.1.1. Lowpass filtering based Wiener filter algorithm

The GFT coefficients of the noisy image is first lowpass filtered in the GFT domain [3, 21, 28] as

$$x_{d,f}(\lambda_l) = \frac{1}{1 + \alpha\lambda_l} y_f(\lambda_l), \tag{8.2}$$

$l = 0, 1, \ldots, L-1$. α is a positive regularization parameter whose value should be searched to result in a good denoising performance. Then, inverse GFT is applied to obtain the filter output, as the denoised image $\mathbf{x_d} = \mathbf{V}\mathbf{x_{d,f}}$ which is used to compute the estimated GRD of the original, noiseless image $\widehat{R}_x(n, \lambda_l)$ as

$$\widehat{R}_x(n, \lambda_l) = R_{x_d}(n, \lambda_l) = x_d(n)\, x_{d,f}(\lambda_l)\, V(n, \lambda_l), \tag{8.3}$$

$n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$. By substituting the $\widehat{R}_x(n, \lambda_l)$ into (5.6), an estimate of the Wiener transfer function is obtained.

The Wiener vertex-frequency transfer function obtained in (5.6) can be applied to the GFT coefficients of the noisy image $\mathbf{y}$ or those of the denoised image $\mathbf{x_d}$ by replacing $y_f(\lambda_l)$ with $x_{d,f}(\lambda_l)$ in (4.1). Both options are tried and the second one whose results are slightly better is employed in experiments. The unscaled Wiener filter output $\widehat{\mathbf{x}}_\mathbf{u}$ is subsequently scaled as in (5.7) to obtain the denoised image $\widehat{\mathbf{x}}$.

Besides the one-step version of the algorithm mentioned above, LPFW_1, two-step version of this algorithm LPFW_2 is also developed and applied to noisy images for further denoising. Second lowpass filtering in the GFT domain is applied to $\widehat{\mathbf{x}}_\mathbf{1}$ that is the denoised image obtained by the LPFW_1, in a similar way in (8.2). The new GRD matrix, computed from this filtered image as in (8.3), is substituted into (5.6) to obtain a new Wiener filter estimate that is subsequently applied to the GFT coefficients of $\widehat{\mathbf{x}}_\mathbf{1}$ instead of those of noisy images $\mathbf{y}$. The output image $\widehat{\mathbf{x}}_\mathbf{2}$ is finally obtained by correcting the scale as in (5.7).

8.1.2. Smoothing and hard thresholding based Wiener filter algorithm

The GRD estimate of the noiseless image $\mathbf{x}$, $\widehat{R}_x(n, \lambda_l)$, is obtained by hard thresholding and smoothing the GRD of the input, noisy image $\mathbf{y}$, $R_y(n, \lambda_l)$, as

$$\widetilde{R}_y(n, \lambda_l) = R_y(n, \lambda_l)\, \mathbf{t}\left(|R_y(n, \lambda_l)| \geq t(n, \lambda_l)\right), \tag{8.4}$$

$l = 0, 1, \ldots, L-1$, $n = 1, 2, \ldots, L$, where $\mathbf{1}(\cdot)$ yields 1 if the condition is satisfied and 0 otherwise. The threshold parameter $t(n, \lambda_l)$ with a multiplication term $\sqrt{1 + V^2(n, \lambda_l)}$ that is the last term of the GRD of the input, noisy image $R_y(n, \lambda_l)$ derived in (6.12), is defined as

$$t(n, \lambda_l) = r\, \widehat{\sigma}^2\, |V(n, \lambda_l)| \sqrt{1 + V^2(n, \lambda_l)} \tag{8.5}$$

Hence, any prior information for noiseless image $\mathbf{x}$ is not required for the threshold parameter $t(n, \lambda_l)$. r is a positive parameter that should be searched to result in a good denoising performance and the estimated noise standard deviation $\widehat{\sigma}$ is calculated by a proposed method given in [105].

Smoothing is applied to the thresholded GRD coefficients $\widetilde{R}_y(n, \lambda_l)$ to estimate the mean of $R_y(n, \lambda_l)$ and the estimated GRD of the noiseless image x to be used in (5.6), $\widehat{R}_x(n, \lambda_l)$, is computed by substructing the last term emerging in (6.11) from the smoothed term as follows

$$\widehat{R}_x(n, \lambda_l) = \frac{1}{K^2} \sum_{m=0}^{K-1} \sum_{k=0}^{K-1} \widetilde{R}_y(n-m, \lambda_{l-k}) - \widehat{\sigma}^2 \, V^2(n, \lambda_l) \tag{8.6}$$

where K is smoothing size parameter whose value should be searched to result in a good denoising performance. Hard thresholding and smoothing operations employed for denoising are applied in such an order in experiments, however, they can also be applied in reverse order resulting slighty different denoising performances.

The estimated Wiener vertex-frequency transfer function employing the $\widehat{R}_x(n, \lambda_l)$ is applied to the GFT coefficients of the input, noisy image $\mathbf{y}$. Then, the output image $\widehat{\mathbf{x}}_\mathbf{1}$ is obtained by correcting the scale via (5.7).

Similarly to the LPFW algorithm, two-step version of the SHTW is also developed and applied to noisy images for further improved results. One-step and two-step versions of the SHTW algorithm are designated as SHTW$_1$ and SHTW$_2$, respectively. New noise standard deviation value on the SHTW$_1$ output $\widehat{\mathbf{x}}_\mathbf{1}$ is estimated as $\widehat{\sigma}_1$, by using a method given in [105]. Hard thresholding and smoothing operations employing $\widehat{\sigma}_1$ are subsequently applied to the GRD of $\widehat{\mathbf{x}}_\mathbf{1}$ as in (8.4), (8.5) and (8.6). Then, the estimated Wiener filter is obtained for the second step by substituting the new $\widehat{R}_x(n, \lambda_l)$ into (5.6) and applied to the GFT coefficients of $\widehat{\mathbf{x}}_\mathbf{1}$ instead of those of noisy images $\mathbf{y}$, since they yield better denoising performance. The output image $\widehat{\mathbf{x}}_\mathbf{2}$ is finally obtained by correcting the scale as in (5.7).

8.2. Graph Signal Denoising

Three real graph signal datasets [1] with available adjacency matrices are considered for denoising in this thesis where two different works are presented. The first one is a vertex-frequency Wiener filtering scheme employed for image and graph signal denoising, introduced elaborately in previous chapters. These filtering algorithms proposed for graph signal denoising is presented in the first section. In the other work, a vertex-frequency graph Wiener filtering class in the WGFT domain is proposed for graph signal denoising,

[1] Available at https://github.com/JeffLIrion/MTSG_Toolbox.

as mentioned in Chapter 1. Proposed algorithms of this filtering is given is the next section.

8.2.1. Vertex-frequency Wiener filtering

The orthonormal eigenvectors serving as the graph basis $\mathbf{V}$ for the algorithms are obtained with the eigendecomposition of the unnormalized Laplacian $\boldsymbol{\mathcal{L}} = \mathbf{D} - \mathbf{W}$ derived from the adjacency matrix $\mathbf{W}$ readily available.

Both versions of the proposed algorithms, LPFW_1, LPFW_2, SHTW_1 and SHTW_2, are employed in denoising of these graph signals. The substracted term in (8.6) is not considered while estimating the noiseless graph signal $\mathbf{x}$; which is the only difference in SHTW_2 algorithm while employing it for graph signal denoising.

Furthermore, an alternative version of the SHTW algorithm is proposed for denoising of graph signals. The averaging operation is performed in the frequency and vertex directions separately, instead of performing a 2-D convolution given in (8.6) by considering the GRD of graph signal datasets as a regular 2-D structure (grid). In the frequency direction, each column of the GRD matrix is smoothed by ones vector of size $K \times 1$ as regular 1-D convolution, while the GRD samples at vertices in a K-hop local neighborhood of each vertex is averaged in the vertex direction as a simple form of vertex domain graph filtering described in [3]. One-step and two-step versions of these algorithms are designated as SHTW_1^* and SHTW_2^*, respectively.

8.2.2. Vertex-frequency Wiener filter class in the WGFT domain

As a second work mentioned in Chapter 1, vertex-frequency Wiener filtering class in the WGFT domain is proposed and applied to graph signals for denoising.

The GRD of the noiseless signal $\mathbf{x}$ is estimated by hard thresholding and smoothing the GRD of the input, noisy signal $\mathbf{y}$, $R_y(n, \lambda_k)$, as described by (8.4), (8.5) and (8.6). This GRD estimate, $\widehat{R}_x(n, \lambda_k)$, is substituted into the real form of (7.17), without the constant scale term $\sigma^2 + \|\mathbf{x}\|^2$, to estimate the unscaled Wiener filter transfer function. The unscaled Wiener filter estimate is applied to the input, noisy signal $\mathbf{y}$ via (7.1). The obtained unscaled filter output $\widehat{\mathbf{x}}_\mathbf{u}$ is then scaled as in (5.7) to account for the constant term in the denominator in (7.17).

As the second step of the algorithm for further denoising, similar hard thresholding and smoothing steps are applied to the GRD of the denoised output $\widehat{\mathbf{x}}_{\mathbf{1}}$ of the first step. The obtained improved GRD estimate $\widehat{R}_x(n, \lambda_k)$ as such is substituted into (7.17) to obtain an improved unscaled Wiener filter estimate. This new Wiener filter estimate is applied to the first step output $\widehat{\mathbf{x}}_{\mathbf{1}}$ via (7.1), and the scale of the obtained unscaled Wiener output is corrected similarly as described above, to yield the final denoised signal $\widehat{\mathbf{x}}$.

The proposed Wiener filter algorithms are illustrated for two window functions; one described in the spectral domain with its GFT

$$H(\lambda_k) = C \exp(-\lambda_k \tau)$$

where C and $\tau > 0$ are window amplitude and width parameters [92], and, the other being the Hann window in the vertex domain of width D vertices [92]

$$\mathrm{h_m(n)} = \frac{1}{2}\big(1 + \cos(\pi \mathrm{d_{mn}}/\mathrm{D})\big)$$

where d_{mn} is the distance between vertices m and n. They are both normalized to satisfy (2.8). The proposed algorithm versions using windows defined in spectral and vertex domains are designated as WS and WV, respectively.

An alternative version of above algorithms where the averaging operation is separately performed in the frequency and vertex directions as in the algorithms marked with $(*)$ in Subsection (8.2.1), are also proposed for graph signal denoising. This form of the algorithm is designated as WS* and WV* for spectral and vertex domain windows, respectively.

8.3. Computational Cost

At each step of the proposed algorithms, for an input, noisy graph signal of length L; Gaussian pre-filtering (of noisy image patches) before graph construction and graph weights calculation for the first step only, GFT computation for the input signal, lowpass filtering in the GFT domain, GRD computation, smoothing and thresholding operations applied to the input GRD, Wiener transfer function calculations, and the output signal computation require $O(L^2)$ computations, each, in terms of real or complex multiplications and additions. For SHTW algorithms, smoothing operation in (8.6) takes $O(K^2L^2)$ additions, but $K^2 \ll L$.

However, eigendecomposition of the $L \times L$ graph Laplacian matrix to compute the graph basis matrix $\mathbf{V}$ requires $O(L^3)$ computations, which determines the overall computational cost of both one-step and two-step versions of the two proposed algorithms.

For 4-nearest neighbor (NN) and 8-NN graph configurations, compared with the complete graph configuration in image denoising accuracy in the next chapter, the sparsity of the Laplacian is $O(L)$; thus, its eigendecomposition can be performed with $O(L^2)$ operations. Hence, the overall computational cost remains as $O(L^2)$ operations for such graph configurations.

For image denoising tasks, $L = 64^2 = 4096$ for each 64×64 image patch requiring a separate eigendecomposition. Hence, the computational cost of denoising a 256×256 image by any of the proposed algorithms is $16\,O(4096^3)$ real operations, which is very high in the case of image denoising applications. For k-NN graph configurations, the computational cost becomes $16\,O(4096^2)$ real operations, which is more feasible.

For a noisy image patch of size 64×64, it takes about 1 minutes for two-step LPFW algorithm and about 19 minutes for two-step SHTW algorithm in MATLAB (v2018) on a desktop computer with an Intel Xeon E5-1620 v2 CPU. For irregular graph signals, denoising the noisy Minnesota road data of size 2640×1 takes about 15 seconds on the same computer.

9. EXPERIMENTAL RESULTS

9.1. Image Denoising

Proposed algorithms given in Section 8.1, with their one-step and two-step versions are employed to denoise 256×256 benchmark gray-scale noisy images with additive, zero mean WGN. They are applied to 64×64 noisy image patches in a locally adaptive manner and optimum parameter values are searched for each patch. Denoising results are obtained using PSNR and SSIM index values that are averaged over 20 independent realizations and they are compared with those of the BF, NLM, BM3D, NLGBT and OGLR methods.

PSNR (top) and SSIM index (bottom) results attained by two versions of the proposed algorithm LPFW_1 and LPFW_2, and the other compared methods are presented in Table 9.1. Denoising algorithms are applied to 14 benchmark noisy (with $\sigma = 15$) images of different structure and texture, including natural[1] (*Parrot*, *House*, *Barbara*, etc.), PWS depth[2] (*Cones* and *Teddy*) and PWS synthetic (*Stripes* and *Box*) images. Both LPFW_1 and LPFW_2 algorithms have obtained remarkable results by surpassing the compared methods in average PSNR up to 10 dB for PWS synthetic images. Especially for the *Stripes* image, obtained results are notably close to that of the true (ideal) Wiener filter presented in Section 3.1 when it is applied to patches of size 64×64. Graph based approach OGLR has attained best results for PWS depth images by slightly surpassing the proposed algorithms. For natural images, the proposed algorithms have significantly outperformed the BF and NLM methods (except for the *Barbara* image only, for the NLM) while giving slightly worse results than BM3D and OGLR methods. *Lena* image is of high importance among natural images, since it contains either natural image properties like a feathered and tasseled hat or PWS image properties like flat regions on her face. Denoising performances of the proposed two-step algorithm LPFW for such an image are beyond the NLM and BF methods and are slightly worse than the BM3D and OGLR methods, demonstrating that the proposed graph based algorithms can compete with recent denoising methods.

PSNR (top) and SSIM index (bottom) results obtained by two versions (LPFW_1, LPFW_2 and SHTW_1, SHTW_2) of the proposed algorithms LPFW and SHTW, and the other compared methods are presented in Table 9.2. Denoising algorithms are applied to

[1] Available at http://www.cs.tut.fi/∼foi/GCF-BM3D.
[2] Available at http://vision.middlebury.edu/stereo/data.

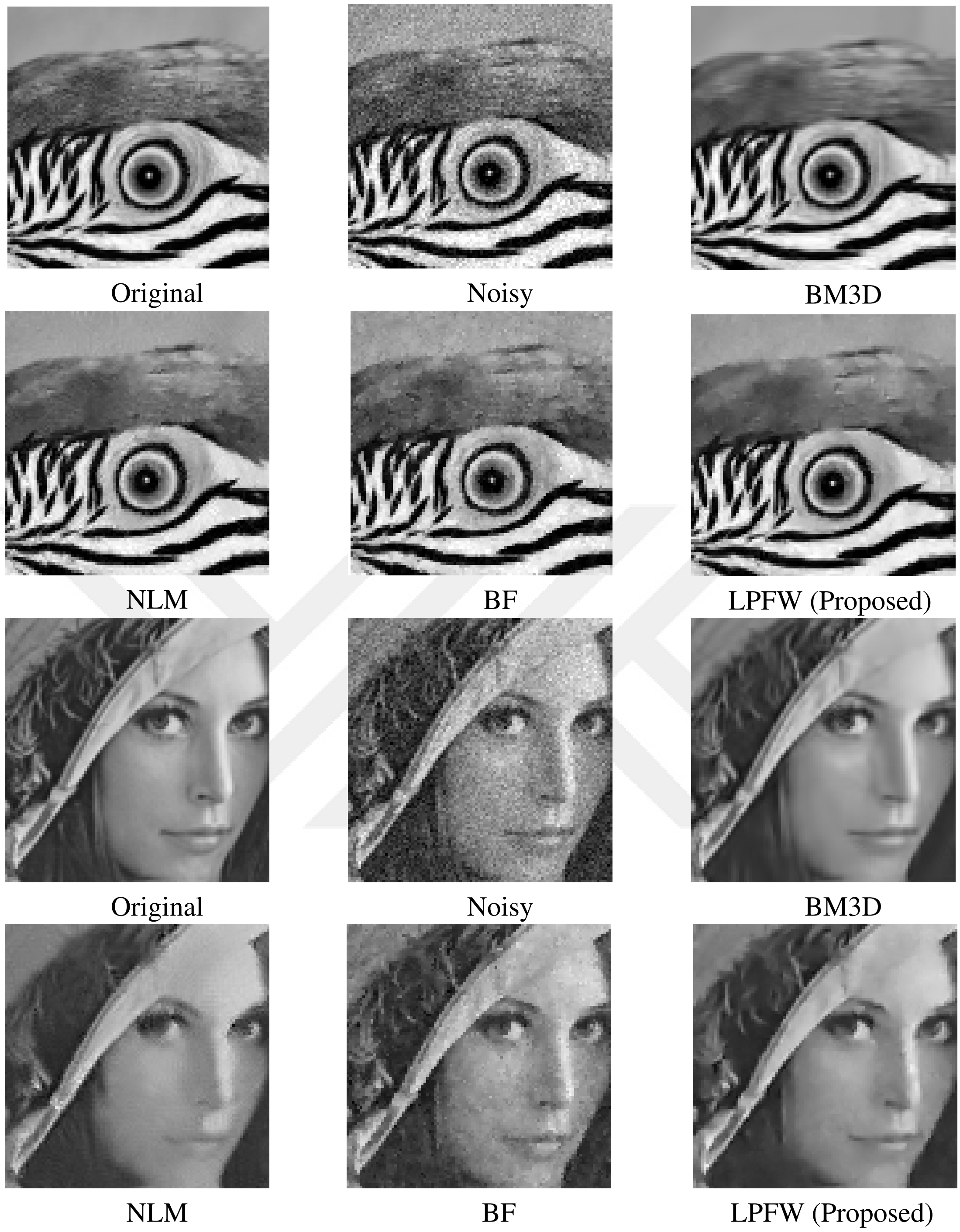

Figure 9.1. *Fragments of the Parrot and Lena natural images are displayed for visual comparison with their original (noiseless), noisy (with $\sigma = 15$), denoised versions by the proposed two-step algorithm LPFW$_2$ and by BF, NLM and BM3D methods. Image fragments of the Parrot image are presented in uppermost two lines, while those of Lena image are presented in lowermost two lines*

PWS depth and synthetic images for different noise standard deviations. All versions of the proposed algorithms surpass the compared methods in terms of PSNR values, among them, LPFW$_2$ outperforms other algorithms for all standard deviation values by achieving

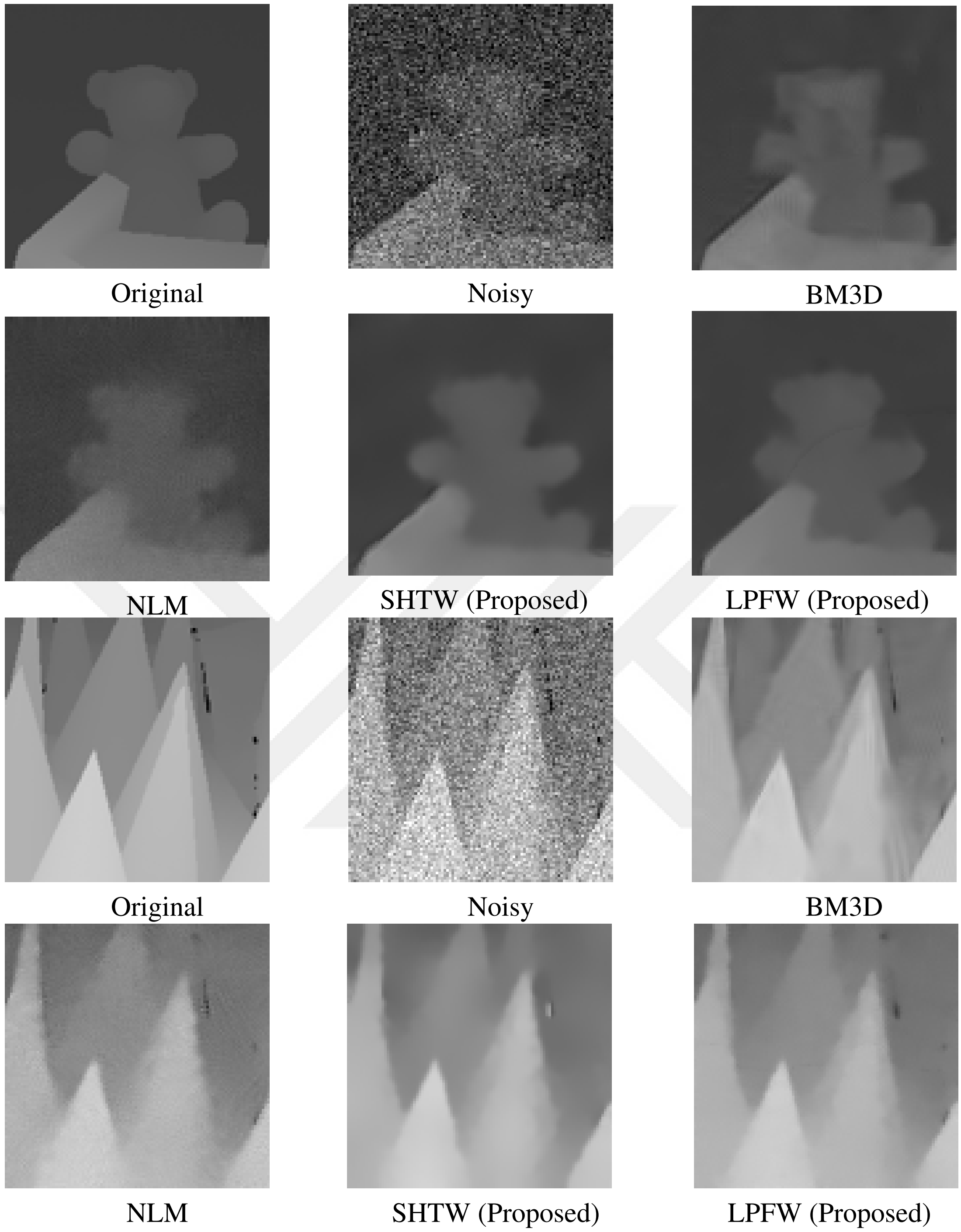

Figure 9.2. *Fragments of the Teddy and Cones PWS depth images are displayed for visual comparison with their original (noiseless), noisy (with $\sigma = 25$), denoised versions by the proposed two-step algorithms LPFW$_2$ and SHTW$_2$, and by NLM and BM3D methods. Image fragments of the Teddy image are presented in uppermost two lines, while those of Cones image are presented in lowermost two lines*

the best PSNR results. However, for depth images *Cones* and *Teddy*, the BM3D attains better results than SHTW$_1$ and SHTW$_2$ for $\sigma = 5$. In particular, for the *Teddy* image, the BM3D surpass the LPFW$_1$ for $\sigma = 5$. For $\sigma = 25$, the BM3D is also better than LPFW$_1$

Table 9.1. *Natural, PWS depth and PWS synthetic image denoising: Performances in average PSNR and SSIM index for two versions of the proposed algorithm LPFW(proposed), and for compared methods BM3D, OGLR, NLM and BF, for noise level of 15. Best results among five algorithms are written in boldface*

Algorithm	***Image***													
	Box	Stripes	Teddy	Cones	C.man	Montage	Peppers	House	Lena	Boats	Barbara	Parrot	Baboon	Grass
LPFW_1	51.87 0.999	58.97 1.000	39.29 0.969	38.93 0.962	31.06 0.869	34.14 0.933	31.99 0.888	33.72 0.865	32.24 0.899	30.73 0.861	30.62 0.864	31.01 0.880	28.64 0.833	29.65 0.870
LPFW_2	**54.17** **0.999**	**60.75** **1.000**	39.51 **0.975**	39.28 0.968	31.09 0.876	34.28 0.938	32.06 0.893	33.79 0.869	32.31 0.904	30.79 0.864	30.67 0.867	31.03 0.885	**28.65** **0.834**	29.71 0.871
BM3D	45.81 0.993	46.59 0.999	36.87 0.972	36.29 0.964	**31.58** 0.888	**35.09** **0.953**	**32.69** **0.909**	**34.93** 0.891	**33.23** **0.923**	**31.17** **0.881**	**32.19** **0.907**	**31.30** **0.893**	27.98 0.806	**30.02** **0.882**
OGLR	45.95 0.997	46.68 0.999	**39.66** 0.969	**39.71** **0.978**	31.33 **0.896**	34.77 0.949	32.32 0.902	34.85 **0.896**	32.79 0.916	30.62 0.871	31.92 0.905	30.86 0.889	28.11 0.822	29.61 0.865
NLM	42.68 0.978	45.42 0.999	35.38 0.951	34.98 0.938	30.86 0.843	32.98 0.922	31.34 0.885	33.24 0.866	31.59 0.889	30.01 0.836	30.89 0.876	30.52 0.855	27.89 0.817	28.94 0.852
BF	30.75 0.641	29.42 0.991	30.47 0.654	30.37 0.667	29.06 0.715	29.94 0.722	29.38 0.771	29.69 0.692	29.27 0.754	28.75 0.773	28.63 0.784	28.91 0.744	27.49 0.805	28.39 0.845

Table 9.2. *PWS depth and PWS synthetic image denoising: Performances in average PSNR and SSIM index for two versions of the proposed algorithms LPFW(proposed) and SHTW(proposed), and for compared methods BM3D, NLM and BF, for different noise levels. Better results than competing algorithms are written in boldface*

σ	$\left[\begin{smallmatrix}\mathrm{LPFW}_1\\ \mathrm{LPFW}_2\end{smallmatrix} \middle\vert \begin{smallmatrix}\mathrm{SHTW}_1\\ \mathrm{SHTW}_2\end{smallmatrix} \middle\vert \mathrm{BM3D} \middle\vert \begin{smallmatrix}\mathrm{NLM}\\ \mathrm{BF}\end{smallmatrix}\right]$ ***Image***															
	Box				Stripes				Teddy				Cones			
5	68.163 1.0000 **70.363** **1.0000**	65.606 0.9999 **67.017** **1.0000**	55.521 0.9992	52.663 0.9975 43.819 0.9759	70.313 1.0000 **72.267** **1.0000**	68.257 1.0000 **69.249** **1.0000**	56.141 0.9999	55.651 0.9999 41.944 0.9995	43.498 0.9848 **43.593** 0.9861	42.011 0.9838 42.046 0.9839	43.542 **0.9919**	43.242 0.9843 40.873 0.9721	43.461 0.9804 **43.607** 0.9816	41.618 0.9742 41.641 0.9743	43.297 **0.9909**	42.786 0.9841 40.486 0.9708
10	59.495 0.9999 **61.666** **1.0000**	57.243 0.9995 **58.714** **0.9996**	49.448 0.9969	47.257 0.9908 36.372 0.8678	61.767 1.0000 **64.637** **1.0000**	61.058 1.0000 **62.225** **1.0000**	50.124 0.9999	49.176 0.9999 34.715 0.9975	41.044 0.9786 **41.169** 0.9811	40.511 0.9808 **40.516** 0.9808	39.370 **0.9823**	38.158 0.9719 35.515 0.8703	40.829 0.9723 **41.153** 0.9755	39.256 0.9698 **39.277** 0.9701	38.883 **0.9784**	37.614 0.9417 35.315 0.8736
15	51.873 0.9994 **54.174** **0.9999**	49.927 0.9991 **52.838** **0.9993**	45.806 0.9932	42.684 0.9785 30.755 0.6412	58.972 1.0000 **60.755** **1.0000**	58.931 1.0000 **59.705** **1.0000**	46.587 0.9998	45.427 0.9997 29.421 0.9912	39.294 0.9694 **39.507** **0.9749**	38.333 0.9747 **38.393** **0.9748**	36.877 0.9721	35.386 0.9502 30.476 0.6541	38.926 0.9621 **39.278** **0.9682**	37.490 0.9628 **37.504** **0.9648**	36.292 0.9643	34.987 0.9383 30.375 0.6672
20	45.745 0.9961 **47.023** **0.9988**	44.164 0.9969 **45.798** **0.9971**	43.285 0.9881	37.514 0.9544 26.479 0.4167	51.619 1.0000 **56.758** **1.0000**	51.191 0.9999 **51.443** **1.0000**	44.102 0.9997	42.896 0.9996 25.329 0.9772	37.644 0.9525 **38.529** **0.9718**	37.466 0.9687 **37.476** **0.9687**	35.084 0.9616	33.473 0.9218 26.363 0.4348	37.408 0.9494 **37.809** **0.9594**	36.276 0.9461 **36.279** **0.9547**	34.550 0.9499	33.193 0.9066 26.296 0.4534
25	39.611 0.9812 **42.045** **0.9925**	38.962 0.9821 **41.181** **0.9894**	41.081 0.9813	34.569 0.9208 23.287 0.2734	42.844 0.9999 **46.067** **1.0000**	42.453 0.9998 **43.844** **1.0000**	42.145 0.9996	40.844 0.9993 22.235 0.9538	36.246 0.9396 **36.832** **0.9634**	34.032 0.9482 **34.102** **0.9511**	33.755 0.9503	32.058 0.8911 23.228 0.2914	36.453 0.9376 **36.953** **0.9514**	34.871 0.9451 **34.877** **0.9452**	33.261 0.9356	31.685 0.8724 23.219 0.3122

and SHTW_1.

SSIM index values of the proposed algorithms LPFW_2 and SHTW_2 are better than other compared methods, however, BM3D attains better results for depth images *Cones* and *Teddy* for lower standard deviation values 5 and 10 only.

As PWS synthetic images *Box* and *Stripes* are considered, the proposed algorithms drastically outperform the BM3D for lower standard deviation values, by attaining almost 12 dB more in PSNR. An obvious improvement over all compared methods is achieved for applied PWS images.

Fig. 9.1 presents two fragments of natural images *Parrot* and *Lena* with their origi-

Table 9.3. *Performance comparison in average PSNR and SSIM index of the 4-NN and 8-NN and complete graph configurations employed in the $LPFW_2$ algorithm for denoising PWS synthetic and depth images for $\sigma = 15$. Best results are written in boldface*

Configuration	***Image***			
	Box	Stripes	Teddy	Cones
4-NN	53.381 0.9996	60.147 1.0000	38.392 0.9563	39.171 0.9589
8-NN	53.785 0.9995	60.215 1.0000	38.292 0.9569	39.079 0.9569
Complete Graph	**54.174** **0.9999**	**60.755** **1.0000**	**39.507** **0.9749**	**39.278** **0.9682**

Table 9.4. *Performance comparison in average PSNR of the $LPFW_2$, OGLR and NLGBT methods in denoising PWS synthetic and depth images, for different noise standard deviations. Best results are written in boldface*

σ	***Image*** [$LPFW_2$ / NLGBT / OGLR] Box			Stripes			Teddy			Cones		
	$LPFW_2$	NLGBT	OGLR	$LPFW_2$	NLGBT	OGLR	$LPFW_2$	NLGBT	OGLR	$LPFW_2$	NLGBT	OGLR
10	**61.67**	61.50	49.94	**64.64**	51.16	49.21	41.17	42.29	**42.43**	41.15	42.84	**42.93**
15	54.17	**56.40**	45.95	**60.75**	47.46	46.68	39.51	39.38	**39.66**	39.29	39.18	**39.71**
20	**47.02**	46.41	42.22	**56.76**	44.96	44.24	**38.53**	36.71	37.83	**37.81**	36.53	37.37
25	42.04	**45.63**	41.15	**46.05**	43.22	44.08	**36.83**	34.62	36.39	**36.95**	34.43	35.59

nal, noisy (with $\sigma = 15$) and denoised versions, while Fig. 9.2 displays two fragments of PWS depth images *Teddy* and *Cones*, with $\sigma = 25$ for noisy version.

Proposed methods are superior in edge recovery than compared methods since round edges disappear even in the denoisied image by the BM3D. Some deteriorations also occur in denoised versions of other algorithms as in the center of *Teddy* image for the BM3D in Fig. 9.2. However, for the denoised versions of the *Cones* image, output of the BM3D method appears as more vivid or less blurry than those of the proposed algorithms. Thus, to demonstrate performance of the proposed algorithms, contrast measures for PWS depth images are also computed by averaging image intensity variances of all horizontal lines. The attained contrast measures are averaged over 5 independent denoising realizations. The averaged contrast measure results of the proposed algorithms $LPFW_2$ and $SHTW_2$ are higher than that of the BM3D method as 6.5022×10^{-3}, 6.4137×10^{-3} and 6.3951×10^{-3}, respectively, demonstrating that they are more succesfull in denoising according to such a measure, though they seem more blurry in visual comparison.

The contrast measure results of the algorithms mentioned above are 2.9610×10^{-3}, 2.7584×10^{-3} and 2.7185×10^{-3}, respectively, for the denoised versions of the *Teddy* image fragment. Although, BM3D obtains more variation with the deterioration occurred in the denoised image, the proposed algorithms are superior than the BM3D, again.

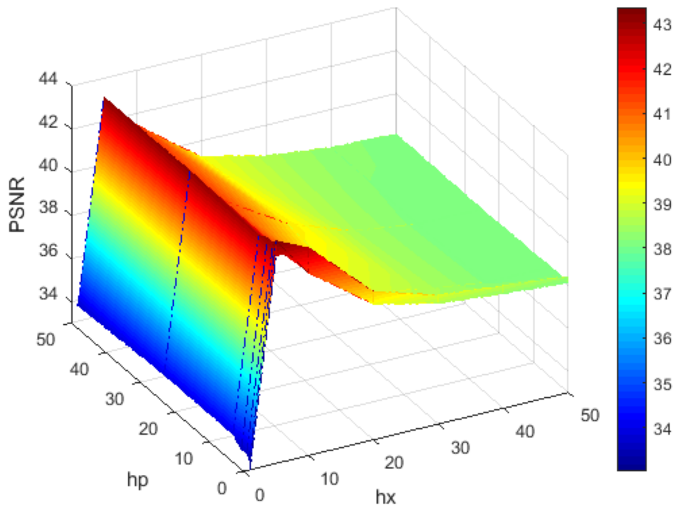

Figure 9.3. *A surface demonstrating the relation between PSNR value and squared smoothing parameters h_x and h_p, employed in weighted adjacency matrix construction formula*

A modified version of the weighted adjacency matrix formula given in (8.1) is also developed and employed in proposed image denoising algorithms. The patches $\tilde{\mathbf{x}}_{\mathbf{m}}$ and $\tilde{\mathbf{x}}_{\mathbf{n}}$, to be employed in measuring the similarity in the photometric domain, are separately normalized and substituted into (8.1), subsequently. Graph Laplacian is computed from this weighted adjacency matrix, and graph bases employed in proposed denoising algorithms are obtained by eigendecomposition of the graph Laplacian. Two-step LPFW algorithm is applied to two different 64×64 noisy patches of *Lena* and *Montage* images with best-performing window size parameters as $w = 7$ and $w = 3$, respectively. PSNR values obtained by LPFW$_1$ and LPFW$_2$ are 29.45 and 29.52 for the *Lena* image patch, and are 40.59 and 40.63 for the the *Montage* image patch, respectively. After normalization process in (8.1) mentioned above, PSNR values attained by LPFW$_1$ and LPFW$_2$ are 25.18 and 24.74 for the *Lena* image patch, and are 26.65 and 26.52 for the *Montage* image patch, respectively. Thus, this normalization process results worse PSNR performances in image denoising. The consistent improvement achieved by the second step of the algorithms is not valid for denoising algorithm employing normalized weighted adjacency matrix. Moreover, this deterioration in PSNR obtained by the second step of the algorithms is increased as the window size parameter increases.

Fig. 9.3 presents the relation between PSNR values and the squared smoothing parameters in the photometric and pixel (spatial) domains, h_x and h_p, respectively. One-step version of the LPFW algoritm, LPFW_1, is applied to a 50×50 noisy image patch of the Teddy image with the noise level of 15, for this comparison. Obtained PSNR values are averaged over 5 independent realizations. h_x and h_p are in the range $[1, 50]$ and $[0.01, 50]$, respectively, and other design parameters are adjusted as their optimum values. According to the PSNR values obtained by different combinations of smoothing parameter values, h_x parameter seems to be more effective on denoising performance than h_p. That is, achieving the pixel similarity information of two patches accurately are more determinative on denoising performance than the distance information of these patches in the 2-D image, for such a weighted adjacency matrix form reminiscent to BF. The optimum values of h_x giving effective PSNR values are in the range $[4, 7]$.

Graph parameters employed in denoising are searched in detail for each separate patch of all noisy images to obtain denoising results given in Tables 9.1 and 9.2 and are presented in Tables F.1 and F.2, in Appendix F. Such parameters have following definitions. h_p and h_x are smoothing parameters in pixel (spatial) and photometric domains. $h_{p,G}$ denotes Gaussian pre-filtering smoothing parameter while w denotes patch size around pixels in determining the weighted adjacency matrix.

Mode, median and range values of graph construction parameters presented for each patch of all images in Table F.1 are as follows. $h_{p,G}^2 \in [0.001, 10]$, with median 0.825 and mode 1.000 having 44 occurences. $h_p^2 \in [0.105, 8.25]$ with median 1.225 and mode 1.5 with 31 occurences. For PWS synthetic images *Stripes* and *Box* with gray levels in the range $[0, 255]$, $h_x^2 \in \{45, 52.5, 60, 67.5, 75\}$ with median and mode 75, both, with 18 occurences. For natural and PWS depth images normalized to the intensity range $[0, 1]$, $h_x^2 \in [0.004, 0.176]$ with median and mode 0.019, both, with 79 occurences. $w \in \{1, 3, 5, 7, 9, 11\}$, with median and mode of 3 samples, both, with 75 occurences among 224 patches.

9.1.1. Comparison with k-nearest neighbor graph configurations

In this work, complete graph configuration with the weight as in (8.1) is used in the proposed algorithms. Denoising results obtained in this way are compared with those of 4-NN and 8-NN graph configurations in which each image pixel is connected to its 4 or 8 nearest neighbors only. As the proposed algorithm that gives the best results, LPFW_2 is employed for this comparison with the weight as in (8.1), in denoising PWS depth and synthetic noisy images with the noise level of 15.

Table 9.3 presents PSNR (top) and SSIM index (bottom) values for different options of graph configuration mentioned above. Best results are obtained for all images when the complete graph configuration is considered. Denoising results of the 4-NN configuration are slightly better than those of 8-NN for PWS depth images, while the opposite of that case is valid for PWS synthetic images.

9.1.2. Comparison with NLGBT and OGLR methods

Graph based image denoising approaches NLGBT and OGLR achieve better results than the BM3D algorithm for PWS depth images not only in PSNR values but also in SSIM index values. Denoising results given in Tables 9.1 and 9.2 demonstrate that the proposed algorithms LPFW and SHTW are useful for denoising of PWS images. Thus, $LPFW_2$ algorithm is compared with graph based methods in denoising PWS depth and synthetic images for different noise levels. NLGBT results presented in Table 9.4 are attained by us for PWS synthetic images *Stripes* and *Box* while results of PWS depth images *Cones* and *Teddy* are directly taken from the paper [32].

The proposed algorithm $LPFW_2$ outperforms graph based methods NLGBT and OGLR in denoising the *Stripes* image for all noise levels, in particular, by almost 14 dB in PSNR obtained especially for lower standard deviation values. For other PWS synthetic *Box* image, OGLR obtains the worst results for all noise levels. NLGBT outperforms the $LPFW_2$ for noise standard deviation values 15 and 25, by almost 4 dB in PSNR, while $LPFW_2$ attains better results for other noise levels.

For PWS depth images *Cones* and *Teddy*, the proposed $LPFW_2$ algorithm outperforms graph based methods for higher noise level values 20 and 25, by almost 2 dB. However, OGLR obtains best results for standard deviations 10 and 15, by almost 2 dB more than the $LPFW_2$. The OGLR always surpasses the NLGBT method for all depth images and noise levels.

These denoising results mentioned above demonstrate that the proposed algorithm $LPFW_2$ can compete with the graph based image denoising methods for PWS depth and synthetic images under different noise levels.

Table 9.5. *Performance comparison in average SNR for the proposed algorithms LPFW, SHTW, SHWT* and graph wavelet based methods HGLET, GHWT*

Algorithm	***Graph Signal***		
	Dentritic Tree	Toronto T.V.	Minnesota R.N.
LPFW_1	25.05	10.63	14.52
LPFW_2	**25.07**	**10.64**	**14.54**
SHTW_1	24.92	8.53	12.90
SHTW_2	**25.07**	**9.61**	**13.66**
SHTW_1^*	25.06	10.22	14.03
SHTW_2^*	**25.07**	**10.23**	**14.40**
HGLET	20.85	8.96	10.16
GHWT	23.03	8.27	11.76

9.2. Graph Signal Denoising

9.2.1. Real graph data

The proposed algorithms are applied to real graph signals such as the thickness data on the dendritic tree, the traffic volume data for Toronto and the Minnesota road network data. Denoising performances of three algorithms proposed for noisy graph signals are compared with graph wavelet shrinkage based denoising methods GHWT and HGLET in [94], using averaged signal-to-noise ratio (SNR) values defined as

$$\text{SNR} = 20\log_{10}\left(\frac{\|\mathbf{x}\|}{\|\mathbf{x}-\widehat{\mathbf{x}}\|}\right)$$

SNR values of the noisy graph signals are adjusted to be 8 dB, 7 dB and 5 dB for dendritic tree, Toronto T.V. and Minnesota R.N, respectively, as in [94].

SNR results averaged over 50 independent trials, attained by LPFW, SHTW and SHTW* and those attained by GHWT and HGLET methods are presented in Table 9.5. The proposed algorithms surpass the compared methods for all three graph signals by significant margins in SNR, especially LPFW_2 is best performing algorithm among them.

SNR values obtained by the other proposed algorithms WS, WV, WS* and WV*, performing in the WGFT domain are presented in Table 9.6. According to the SNR results averaged over 100 trials, these proposed algorithms also surpass the compared methods, especially for the dendritic tree data by almost 3.5 dB more in SNR.

The noiseless, noisy and denoised (by the LPFW_2) versions of Toronto traffic volume data and thickness data on the dendritic tree are presented in Fig. 9.4, while same versions of the Toronto traffic volume data are displayed in Fig. 9.5 when the WS algorithm is used for denoising.

Table 9.6. *Performance comparison in average SNR for the proposed algorithms WS, WV, WS*, WV* and HGLET, GHWT methods*

Algorithm	***Graph Signal***		
	Dentritic Tree	Toronto T.V.	Minnesota R.N.
WS	**23.42**	**10.48**	**13.47**
WV	**23.45**	**9.49**	**12.42**
WS*	**23.35**	**10.41**	**12.34**
WV*	**23.32**	**10.10**	**12.72**
HGLET	20.85	8.96	10.16
GHWT	23.03	8.27	11.76

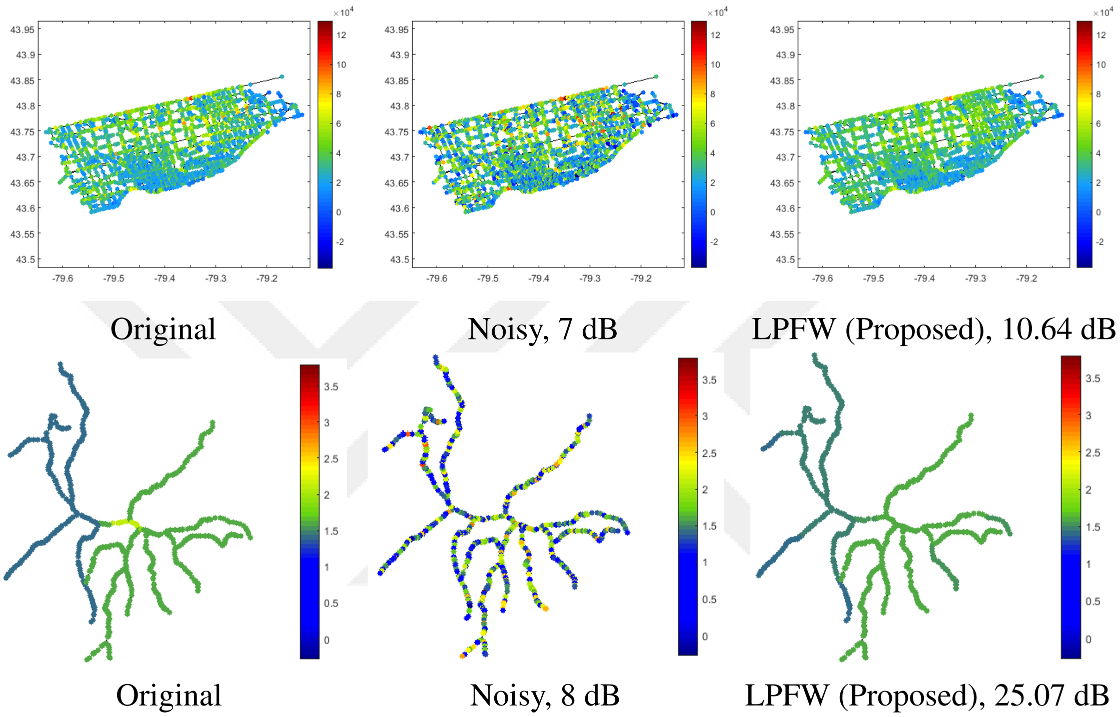

Figure 9.4. *The original (noiseless), noisy and denoised versions of the Toronto traffic volume data (upper line) and the dentritic tree thickness data (lower line) are presented for visual comparison*

9.2.2. Discrete Fourier transform basis and chirp denoising

The proposed graph Wiener filtering scheme is also valid for a complex-valued, unitary graph basis, for which $\mathbf{V}\mathbf{V}^H = \mathbf{I}$. The DFT basis with $V(n,l) = \exp[j(2\pi/L)l(n-1)]/\sqrt{L}$, $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$, is such a basis, which is the eigenbasis of the adjacency (and Laplacian) matrix of a directed, cyclic graph (dicycle graph) with L vertices that models discrete-time periodic signals with period L [19]. Adjacency matrix elements are $w_{mn} = 1$ when $m = \mathrm{mod}(n, L) + 1$, and zero otherwise for such a graph [19].

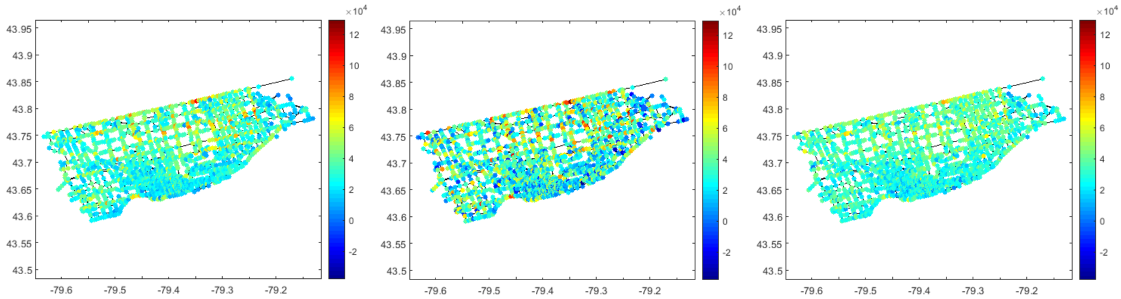

Figure 9.5. *The original (noiseless), noisy and denoised versions of the Toronto traffic volume data are displayed for visual comparison*

By substituting the DFT basis into (5.4) and (5.3),

$$\mathcal{H}_W\left(n, \frac{2\pi}{L}l\right) = \frac{L\, R_x\left(n, \frac{2\pi}{L}l\right)}{\sigma^2 + \|\mathbf{x}\|^2} = \frac{\sqrt{L}\, x(n)X^*(l)e^{-j2\pi ln/L}}{\sigma^2 + \|\mathbf{x}\|^2}$$

for $n, l = 0, 1, \ldots, L-1$, as the TF Wiener transfer function for estimating a discrete-time signal $x(n)$ from its noisy version $y(n)$ given by (2.3). $X(l)$ denotes the L-point DFT sequence of $x(n)$ above. The Wiener filter output is obtained by (4.1) with the DFT basis, as

$$\widehat{x}(n) = \frac{1}{\sqrt{L}} \sum_{l=0}^{L-1} e^{j2\pi ln/L}\, \mathcal{H}_W\left(n, \frac{2\pi}{L}l\right) Y(l)$$

for $n = 0, 1, \ldots, L-1$, where $Y(l)$ is the L-point DFT of the noisy $y(n)$. The TF Wiener filtering scheme for denoising given by the two equations above are novel in the classical TF filtering literature.

In a practical setting, an estimate of the classical RD of the original, noiseless signal, $\widehat{R}_x(n, 2\pi l/L)$, should be obtained from the RD of the noisy signal, $R_y(n, 2\pi l/L) = y(n)Y^*(l)\exp(-j2\pi ln/L)/\sqrt{L}$. This input RD estimate is used, in turn, to obtain an estimate of the Wiener transfer function, $\widehat{\mathcal{H}}_W(n, 2\pi l/L)$.

A noisy version of the chirp signal is considered for denoising $x(n) = \exp[\frac{j\pi n^2}{4L}]$, $n = 0, 1, \ldots, L-1$, according to signal model (2.3), where the additive complex, circular noise $w(n)$ is modeled by (2.4) and (2.5). The ideal (true) form of the proposed TF Wiener filter given above is applied to denoise this noisy chirp and compare the results with those of the interpolated half-band Weyl filter [79]. TF transfer function of the Weyl filter is chosen to isolate the instantaneous frequency line of the chirp $\omega(n) = \pi n/(2L)$, as $H(n, \omega) = \delta(\omega - \pi n/(2L))$, $0 \le n \le L-1$. With least-squares scaling of Weyl filter

Table 9.7. *Output SNR values for the Weyl filter, true Wien_Rihac filter and its estimates in denoising a noisy chirp signal, for different input SNR values*

Method	***SNR_{in}***					
	20	15	10	5	0	-3
Weyl	21.307	21.277	21.183	20.903	20.117	18.980
True Wien_Rihac	43.987	39.284	34.693	28.983	23.827	21.404
Wien_Rihac Algorithms	21.926	17.854	13.104	9.924	5.654	3.062

outputs employed, this Weyl filter is a very good approximation to the ideal TF Wiener filter based on Wigner-Weyl framework [81, 82]. Hence, it is fair to compare with the ideal form of the proposed TF Wiener filter (designated as Wien_Rihac).

Table 9.7 lists filter output SNR values averaged over 100 independent trials, for various input SNR values ($-10\log_{10}(\sigma^2)$ for this unity amplitude chirp). Large improvements achieved by the proposed ideal Wiener filter indicate that the true form of the TF Wiener filter for denoising should be rather as given above, although the Weyl filter also gives excellent results and is a practical realization of the ideal Wiener filter of the Wigner-Weyl approach.

Unfortunately, contrary to the Weyl filter accurately implementing the Wigner-Weyl scheme, it seems to be very difficult to accurately estimate the transfer function of our Wiener filter, although it has the correct form, when the original signal is a wideband signal as in this example.

The LPFW_1 algorithm is implemented by substituting the output of a lowpass filter with passband $[0, \pi/2)$ in radians, $x_d(n)$, its DFT and the DFT basis into the TF Wiener transfer function above. The SHTW_1 algorithm is also implemented by searching the threshold parameter t in (8.4) as real multiples of $|V(n,l)| = 1/\sqrt{L}$ due to variance given by (6.14), and by taking the subtracted term in (8.6) as σ^2/L due to mean given by (6.13). In addition, noisy $|R_y|$ is smoothed and $\arg[R_y]$ is smoothed or kept unchanged to estimate modulus and phase of the original RD, R_x, respectively, to substitute into the Wiener transfer function. This method is designated as the SW algorithm.

For the later two algorithms, the constant scale factor $\sigma^2 + ||\mathbf{x}||^2$ is estimated as $||\mathbf{y}||^2 - (L-1)\sigma^2$, since $||\mathbf{x}||^2 \approx ||\mathbf{y}||^2 - L\sigma^2$. Scale correction in (5.7) can not be used for the chirp signal case, since average values of real and imaginary parts of the chirp are also close to zero, making (5.7) unstable.

Results of these proposed algorithms are usually much worse than those of the Weyl filter. They can not accurately estimate the true Wiener transfer function, although they achieve some amount of denoising. Hence, only best output SNR values provided among these three algorithms are listed in Table 9.7. For $\text{SNR}_{\text{in}} = 20$ and 15 dB, presented results are obtained by the SW, where $\arg[\widehat{R}_x]$ is taken as $\arg[R_y]$ without smoothing and $|\widehat{R}_x|$ for $\omega \in [0, \pi/2)$ is taken constant as the mean value of $|R_y|$ for $\omega \in [0, \pi/2)$. $|\widehat{R}_x|$ is taken as zero outside this frequency range of the chirp. For $\text{SNR}_{\text{in}} = -3$ dB, the presented result is obtained by the SHTW_1. For other input SNRs, best results presented are obtained by the LPFW_1. The Weyl filter is surpassed for $\text{SNR}_{\text{in}} = 20$ dB only, by the SW.

10. CONCLUSION

In this thesis, the Rihaczek-Zadeh TF filtering scheme is extended to graph signals and a graph Zadeh vertex-frequency filter is proposed. Vertex-frequency graph Wiener filtering based on this Zadeh filtering scheme is subsequently proposed for image and graph signal denoising. The vertex-frequency transfer function of the proposed Wiener filter has a quite different form than $\frac{R_x(n,\lambda_l)}{(\sigma^2+R_x(n,\lambda_l))}$ that is expected in the TF filtering literature as reviewed in [81], in terms of a vertex-frequency signal energy distribution.

In Appendix A, the Wiener transfer function is expressed in terms of GRS matrix of the graph signal and the graph basis matrix $\mathbf{V}$ obtained from graph Laplacian. However, in order to express Wiener transfer function coefficients analitically, the condition of the graph signal of interest to be deterministic as in (5.4) or to be GWSS as in (A.1.1) should be satisfied. The proposed Wiener filtering scheme is reduced to stationary graph Wiener filter for GWSS signals as presented in Appendix A, demonstrating that the proposed vertex-frequency Wiener filtering framework is a generalization of such filters [15, 18, 21, 22, 63] to nonstationary graph signals.

Moreover, the GRD [88, 89] based on a GSO defined by generalized convolution with a delta signal [3, 8, 72] is derived in Chapter 6. The graph shift operation should be applied first to the graph signal than the complex conjugation process in this derivation.

The proposed Wiener filter algorithms yield considerably better PSNR and SSIM index values for PWS images than BF, NLM and BM3D methods and compete with graph based denoising methods NLGBT and OGLR. For natural images, obtained results are slightly worse than, but get close to those obtained by these compared methods. In denoising of real graph signal datasets, proposed algorithms significantly outperfom the graph wavelet shrinkage based methods GHWT and HGLET in SNR values for each graph data.

It is important to note that proposed spectral graph based algorithms achieve significant performances in image denoising as well as irregular graph signal denoising, since such methods are adaptive to the image of interest. Graph bases are obtained by the eigendecomposition of the Laplacian matrix including the similarity information of pixels in the image. In particular, denoising performances of such proposed algorithms are remarkable for PWS images, since the information of such an image can be obtained accurately from its first few eigenvalues.

As another work presented in this thesis, a vertex-frequency graph Wiener filter family in the WGFT domain indexed by used graph window functions is also derived to denoise graph signals. This family encompasses the graph Zadeh based Wiener filter as a limiting case. Proposed algorithms give better SNR results than compared graph wavelet shrinkage based methods for tried real graph data.

The vertex-frequency graph Wiener filtering proposed in this thesis is formulated for a fixed graph structure underlying the given noisy graph signal. That is, the weights of graph edges, hence the graph Laplacian matrix and its eigenvectors that determine the GFT operation, are assumed to be given, in this development of the Wiener filter. Their optimal choice or learning is outside the scope of this work. Graph learning issues or optimal choice of graph edge weights for a given graph signal are not encompassed in the proposed optimization framework. However, in simulations, pre-filtering parameters, smoothing parameters and window sizes that determine graph weights are thoroughly searched for the best performance in these presented image denoising results.

A more principled or automated approach might be integrating the optimal graph Laplacian regularization approach proposed in [31] for image denoising, or graph learning techniques, to the Wiener filtering scheme developed in this article in order to use optimally selected graph basis vectors in the proposed Wiener filter. For a comprehensive list of graph learning methods, please see [1, 26].

REFERENCES

[1] Ortega, A., Frossard, P., Kovačević, J., Moura, J.M.F. and Vandergheynst, P. (2018). Graph signal processing: Overview, challenges, and applications. In: *Proceedings of the IEEE*, 106, 808–828.

[2] Sandryhaila, A. and Moura, J.M.F. (2014). Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Processing Magazine*, 31 (5), 80–90.

[3] Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A. and Vandergheynst, P. (2013). 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), 83–98.

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

[5] Yağan, A.C. and Özgen, M.T. (2017). A spectral graph based image coding method. In: *2017 IEEE 25th Signal Processing and Communications Applications Conference (SIU)*.

[6] Chen, S., Sandryhaila, A., Moura, J.M.F. and Kovačević, J. (2014). Signal denoising on graphs via graph filtering. In: *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*.

[7] Chen, S., Sandryhaila, A., Moura, J.M.F. and Kovačević, J. (2015). Signal recovery on graphs: Variation minimization. *IEEE Transactions on Signal Processing*, 63 (17), 4609–4624.

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

[9] Deri, J.A. and Moura, J.M.F. (2017). Spectral projector-based graph Fourier transforms. *IEEE Journal of Selected Topics in Signal Processing*, 11, 785–795.

[10] Sardellitti, S., Barbarossa, S. and Lorenzo, P.D. (2017). On the graph Fourier transform for directed graphs. *IEEE Journal of Selected Topics in Signal Processing*, 11, 796–811.

[11] Mhaskar, H. (2018). A unified framework for harmonic analysis of functions on directed graphs and changing data. *Applied and Computational Harmonic Analysis*, 44, 611–644.

[12] Shafipour, R., Khodabakhsh, A., Mateos, G. and Nikolova, E. (2017). A digraph Fourier transform with spread frequency components. In: *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*.

[13] Girault, B., Ortega, A. and Narayanan, S.S. (2018). Irregularity-aware graph Fourier transforms. *IEEE Transactions on Signal Processing*, 66, 5746–5761.

[14] Sandryhaila, A. and Moura, J.M.F. (2014). Discrete signal processing on graphs: Frequency analysis. *IEEE Transactions on Signal Processing*, 62, 3042–3054.

[15] Perraudin, N. and Vandergheynst, P. (2017). Stationary signal processing on graphs. *IEEE Transactions on Signal Processing*, 65 (13), 3462–3477.

[16] Loukas, A. and Foucard, D. (2016). Frequency analysis of temporal graph signals. [Online]. Available: https://arxiv.org/abs/1602.04434.

[17] Perraudin, N., Loukas, A., Grassi, F. and Vandergheynst, P. (2017). Towards stationary time-vertex signal processing. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3914–3918.

[18] Marques, A.G., Segarra, S., Leus, G. and Ribeiro, A. (2017). Stationary graph processes and spectral estimation. *IEEE Transactions on Signal Processing*, 65 (22), 5911–5926.

[19] Gavili, A. and Zhang, X.-P. (2017). On the shift operator, graph frequency, and optimal filtering in graph signal processing. *IEEE Transactions on Signal Processing*, 65 (23), 6303–6318.

[20] Zhang, F. and Hancock, E.R. (2008). Graph spectral image smoothing using the heat kernel. *Pattern Recognition*, 41, 3328 – 3342.

[21] Isufi, E., Loukas, A., Simonetto, A. and Leus, G. (2017). Autoregressive moving average graph filtering. *IEEE Transactions on Signal Processing*, 65 (2), 274–288.

[22] Girault, B. (2015). Stationary graph signals using an isometric graph translation. In: *23rd European Signal Processing Conference (EUSIPCO)*, 1516–1520.

[23] Lorenzo, P.D., Barbarossa, S., Banelli, P. and Sardellitti, S. (2016). Adaptive least mean squares estimation of graph signals. *IEEE Transactions on Signal and Information Processing over Networks*, 2, 555–568.

[24] Lorenzo, P.D., Banelli, P. Barbarossa, S. and Sardellitti, S. (2017). Distributed adaptive learning of graph signals. *IEEE Transactions on Signal Processing*, 65, 4193–4208.

[25] Lorenzo, P.D., Banelli, P., Isufi, E., Barbarossa, S. and Leus, G. (2018). Adaptive graph signal processing: Algorithms and optimal sampling strategies. *IEEE Transactions on Signal Processing*, 66, 3584–3598.

[26] Cheung, G., Magli, E., Tanaka, Y. and Ng, M.K. (2018). Graph spectral image processing. In: *Proceedings of the IEEE*, 106, 907–930.

[27] Elmoataz, A., Lezoray, O. and Bougleux, S. (2008). Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing. *IEEE Transactions on Image Processing*, 17 (7), 1047–1060.

[28] Kheradmand, A. and Milanfar, P. (2013). A general framework for kernel similarity-based image denoising. In: *2013 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 415–418.

[29] Liu, X., Zhai, D., Zhao, D., Zhai, G. and Gao, W. (2014). Progressive image denoising through hybrid graph laplacian regularization: A unified framework. *IEEE Transactions on Image Processing*, 23 (4), 1491–1503.

[30] Hu, W., Cheung, G., Li, X. and Au, O.C. (2014). Graph-based joint denoising and super-resolution of generalized piecewise smooth images. In: *2014 IEEE International Conference on Image Processing (ICIP)*, 2056–2060.

[31] Pang, J. and Cheung, G. (2017). Graph laplacian regularization for image denoising: Analysis in the continuous domain. *IEEE Transactions on Image Processing*, 26 (4), 1770–1785.

[32] Hu, W., Li, X., Cheung, G. and Au, O.C. (2013). Depth map denoising using graph-based transform and group sparsity. In: *2013 IEEE 15th International Workshop on Multimedia Signal Processing (MMSP)*, 1–6.

[33] Meyer, F.G. and Shen, X. (2014). Perturbation of the eigenvectors of the graph laplacian: Application to image denoising. *Applied and Computational Harmonic Analysis*, 36, 326–334.

[34] Iizuka, Y. and Tanaka, Y. (2014). Depth map denoising using collaborative graph wavelet shrinkage on connected image patches. In: *2014 IEEE International Conference on Image Processing (ICIP)*, 828–832.

[35] Gadde, A., Narang, S. and Ortega, A. (2013). Bilateral filter: Graph spectral interpretation and extensions. In: *2013 IEEE International Conference on Image Processing*, 1222–1226.

[36] Onuki, M. and Tanaka, Y. (2014). Trilateral filter on graph spectral domain. In: *2014 IEEE International Conference on Image Processing (ICIP)*, 2046–2050.

[37] Onuki, M., Ono, S., Yamagishi, M. and Tanaka, Y. (2016). Graph signal denoising via trilateral filter on graph spectral domain. *IEEE Transactions on Signal and Information Processing over Networks*, 2 (2), 137–148.

[38] Berger, P., Hannak, G. and Matz, G. (2017). Graph signal recovery via primal-dual algorithms for total variation minimization. *IEEE Journal of Selected Topics in Signal Processing*, 11 (6), 842–855.

[39] Couprie, C., Grady, L., Najman, L., Pesquet, J.-C. and Talbot, H. (2013). Dual constrained TV-based regularization on graphs. *SIAM Journal on Imaging Sciences*, 6, 1246–1273.

[40] Shuman, D.I., Vandergheynst, P., Kressner, D. and Frossard, P. (2018). Distributed signal processing via chebyshev polynomial approximation. *IEEE Transactions on Signal and Information Processing over Networks*, 4, 736–751.

[41] Gadde, A., Xu, M. and Ortega, A. (2017). Sparse inverse bilateral filters for image processing. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[42] Susnjara, A., Perraudin, N., Kressner, D. and Vandergheynst, P. (2015). Accelerated filtering on graphs using lanczos method. [Online]. Available: https://arxiv.org/abs/1509.04537.

[43] Tian, D., Mansour, H., Knyazev, A. and Vetro, A. (2014). Chebyshev and conjugate gradient filters for graph image denoising. In: *2014 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*.

[44] Knyazev, A. (2015). Edge-enhancing filters with negative weights. In: *2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 260–264.

[45] Milanfar, P. (2013). A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine*, 30 (1), 106–128.

[46] Lebrun, M., Buades, A. and Morel, J.-M. (2013). A nonlocal bayesian image denoising algorithm. *Image Processing On Line*, 6, 1665–1688.

[47] Lu, L., Jin, W. and Wang, X. (2015). Non-local means image denoising with a soft threshold. *IEEE Signal Processing Letters*, 22 (7), 833–837.

[48] Gu, S., Xie, Q., Meng, D., Zuo, W., Feng, X. and Zhang, L. (2016). Weighted nuclear norm minimization and its applications to low level vision. *International Journal of Computer Vision*, 121, 183–208.

[49] Tomasi, C. and Manduchi, R. (1998). Bilateral filtering for gray and color images. In: *Sixth International Conference on Computer Vision*, 839–846.

[50] Elad, M. (2002). On the origin of the bilateral filter and ways to improve it. *IEEE Transactions on Image Processing*, 11, 1141–1151.

[51] Buades, A. Coll, B. and Morel, J.-M. (2005). A non-local algorithm for image denoising. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2, 60–65.

[52] Buades, A., Coll, B. and Morel, J.-M. (2005). A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4, 490–530.

[53] Kervrann, C. and Boulanger, J. (2006). Optimal spatial adaptation for patch-based image denoising. *IEEE Transactions on Image Processing*, 15 (10), 2866–2878.

[54] Takeda, H., Farsiu, S. and Milanfar, P. (2007). Kernel regression for image processing and reconstruction. *IEEE Transactions on Image Processing*, 16 (2), 349–366.

[55] Talebi, H. and Milanfar, P. (2016). Fast multilayer laplacian enhancement. *IEEE Transactions on Computational Imaging*, 2, 496–509.

[56] Chatterjee, P. and Milanfar, P. (2012). Patch-based near-optimal image denoising. *IEEE Transactions on Image Processing*, 21 (4), 1635–1649.

[57] Dabov, K., Foi, A., Katkovnik, V. and Egiazarian, K. (2007). Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8), 2080–2095.

[58] Katkovnik, V., Foi, A., Egiazarian, K. and Astola, J. (2009). From local kernel to nonlocal multiple-model image denoising. *International Journal of Computer Vision*, 86, 1–32.

[59] Lee, J-S. (1980). Digital image enhancement and noise filtering by use of local statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2 (2), 165–168.

[60] Kuan, D.T., Sawchuk, A.A., Strand, T.C. and Chavel, P. (1985). Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7 (2), 165–177.

[61] Jiang, S.-S. and Sawchuk, A.A. (1986). Noise updating repeated Wiener filter and other adaptive noise smoothing filters using local image statistics. *Applied Optics*, 25, 2326–2337.

[62] Bindilatti, A.A., Vieira, M.A.C. and Mascarenhas, N.D.A. (2018). Poisson Wiener filtering with non-local weighted parameter estimation using stochastic distances. *Signal Processing*, 144, 68–76.

[63] Girault, B., Gonçalves, P., Fleury, E. and Mor, A.S. (2014). Semi-supervised learning for graph to signal mapping: A graph signal Wiener filter interpretation. In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1115–1119.

[64] Julesz, B. (1962). Visual pattern discrimination. *IRE Transactions on Information Theory*, 8 (2), 84–92.

[65] Chatterjee, P. and Milanfar, P. (2010). Is denoising dead?. *IEEE Transactions on Image Processing*, 19 (4), 895–911.

[66] Ghael, S.P., Sayeed, A.M. and Baraniuk, R.G. (1997). Improved wavelet denoising via empirical Wiener filtering. In: *SPIE Wavelet Applications in Signal and Image Processing V*, 389–399.

[67] Yaroslavsky, L. (1997). Nonlinear adaptive image processing in transform domain. In: *IEEE Workshop Nonlinear Signal Image Processing*.

[68] Talebi, H. and Milanfar, P. (2014). Global image denoising. *IEEE Transactions on Image Processing*, 23 (2), 755–768.

[69] Yağan, A.C. and Özgen, M.T. (2016). A spectral graph Wiener filter in graph Fourier domain for improved image denoising. In: *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 450–454.

[70] Yağan, A.C. and Özgen, M.T. (2020). Spectral graph based vertex-frequency Wiener filtering for image and graph signal denoising. *IEEE Transactions on Signal and Information Processing over Networks*, 6 (1), 226–240.

[71] Shuman, D.I., Ricaud, B. and Vandergheynst, P. (2012). A windowed graph Fourier transform. In: *2012 IEEE Statistical Signal Processing Workshop (SSP)*, 133–136.

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

[73] Jestrović, I., Coyle, J.L. and Sejdić, E. (2017). A fast algorithm for vertex-frequency representations of signals on graphs. *Signal Processing*, 131, 483–491.

[74] Stanković, L., Daković, M. and Sejdić, E. (2017). Vertex-frequency analysis: A way to localize graph spectral components. *IEEE Signal Processing Magazine*, 34 (4), 176–182.

[75] Thanh, L.T., Linh-Trung, N., Dung, N.V. and Abed-Meraim, K. (2017). A new windowed graph Fourier transform. In: *2017 4th NAFOSTED Conference on Information and Computer Science*, 150–155.

[76] Tepper, M. and Sapiro, G. (2016). A short-graph Fourier transform via personalized pagerank vectors. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4806–4810.

[77] Zadeh, L.A. (1950). Frequency analysis of variable networks. In: *Proceedings of the IRE*, 38, 291–299.

[78] Kozek, W. (1992). Time-frequency signal processing based on the Wigner-Weyl framework. *Signal Processing*, 29 (1), 77–92.

[79] Matz, G. and Hlawatsch, F. (2018). Linear time–frequency filters: On-line algorithms and applications. In: *Applications in Time-Frequency Signal Processing*, 205–272.

[80] Kozek,W. and Hlawatsch, F. (1992). A comparative study of linear and nonlinear time-frequency filters. In: *Proceedings of the IEEE-SP International Symposium on Time-Frequency and Time-Scale Analysis*, 163–166.

[81] Hlawatsch, F., Matz, G., Kirchauer, H. and Kozek, W. (2000). Time-frequency formulation, design, and implementation of time-varying optimal filters for signal estimation. *IEEE Transactions on Signal Processing*, 48 (5), 1417–1432.

[82] Stanković, L.J., Stanković, S. and Djurović, I. (2000). Space/spatial-frequency analysis based filtering. *IEEE Transactions on Signal Processing*, 48 (8), 2343–2352.

[83] Priestley, M.B. (1965). Evolutionary spectra and non-stationary processes. *Journal of the Royal Statistical Society: Series B (Methodological)*, 27 (2), 204–229.

[84] Abdrabbo, N.A. and Priestley, M.B. (1969). Filtering non-stationary signals. *Journal of the Royal Statistical Society: Series B (Methodological)*, 31 (1), 150–159.

[85] Sills, J.A. and Kamen, E.W. (1995). Wiener filtering of nonstationary signals based on spectral density functions. In: *34th IEEE Conference on Decision and Control*, 3, 2521–2526.

[86] Khan, H.A. and Chaparro, L.F. (1997). Non-stationary Wiener filtering based on evolutionary spectral theory. In: *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 5, 3677–3680.

[87] Rihaczek, A.W. (1968). Signal energy distribution in time and frequency. *IEEE Transactions on Information Theory*, 14 (3), 369–374.

[88] Stanković, L., Sejdić, E. and Daković, M. (2018). Vertex-frequency energy distributions. *IEEE Signal Processing Letters*, 25 (3), 358–362.

[89] Stanković, L., Sejdić, E. and Daković, M. (2018). Reduced interference vertex-frequency distributions. *IEEE Signal Processing Letters*, 25 (9), 1393–1397.

[90] Lander, P. and Berbari, E. (1994). Enhanced ensemble averaging using the time-frequency plane. In: *IEEE-SP International Symposium on Time- Frequency and Time-Scale Analysis*, 241–243.

[91] Sayeed, A.M., Lander, P. and Jones, D.L. (1995). Improved time-frequency filtering of signal-averaged ECGs. *Journal of Electrocardiology*, 28, 53–58.

[92] Stanković, L., Mandić, D.M., Daković, M., Scalzo, B., Brajović, M., Sejdić, E. and Constantinides, A.G. (2019). Vertex-Frequency Graph Signal Processing. [Online]. Available: https://arxiv.org/abs/1907.03471v1.

[93] Hammond, D.K., Jacques, L. and Vandergheynst, P. (2017). Image denoising with nonlocal spectral graph wavelets. In: *Image Processing and Analysis with Graphs: Theory and Practise*, 8, 207–236.

[94] Irion, J. and Saito, N. (2017). Efficient approximation and denoising of graph signals using the multiscale basis dictionaries. *IEEE Transactions on Signal and Information Processing over Networks*, 3 (3), 607–616.

[95] Bruna, J., Zaremba, W., Szlam, A. and LeCun, Y. (2014). Spectral networks and locally connected networks on graphs. In: *International Conference on Learning Representations (ICLR)*.

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

[97] Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A. and Vandergheynst, P. (2017). Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34, 18–42.

[98] Boscaini, D., Masci, J., Melzi, S., Bronstein, M.M., Castellani, U. and Vandergheynst, P. (2015). Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks. *Computer Graphics Forum*, 34, 13–23.

[99] Segarra, S., Marques, A.G. and Ribeiro, A. (2017). Optimal graph-filter design and applications to distributed linear network operators. *IEEE Transactions on Signal Processing*, 65 (15), 4117–4131.

[100] Chen, L., Cheng, S., Stankovic, V. and Stankovic, L. (2018). Shift-enabled graphs: Graphs where shift-invariant filters are representable as polynomials of shift operations. *IEEE Signal Processing Letters*, 25, 1305–1309.

[101] Girault, B., Gonçalves, P. and Fleury, E. (2015). Translation on graphs: An isometric shift operator. *IEEE Signal Processing Letters*, 22 (12), 2416–2420.

[102] Kailath, T., Sayed, A.H. and Hassibi, B. (2000). *Linear Estimation*. Pearson Education (US), Prentice-Hall, 3, 115–116, App.A, pp. 729, 3, pp. 104.

[103] Adali, T., Schreier, P. and Scharf, L. (2011). Complex-valued signal processing: The proper way to deal with impropriety. *IEEE Transactions on Signal Processing*, 59 (11), 5101–5125.

[104] Van Trees, H.L. (2004). *Detection, Estimation, and Modulation Theory, Part I: Detection, Estimation, and Linear Modulation Theory, Wiley*, 2, 156-157.

[105] Immerkær, J. (1996). Fast noise variance estimation. *Computer Vision and Image Understanding*, 64 (2), 300–302.

APPENDICES

A. Wiener Vertex-Frequency Transfer Function

Two general forms of the vertex-frequency transfer function of the proposed graph Wiener filter are derived for the case of random graph signal of interest $\mathbf{x}$, in terms of its GRS.

From (5.3), the GRD matrix of $\mathbf{x}$,

$$\mathbf{R_x} = [R_x(n, \lambda_l)]_{L\times L} = \left(\mathbf{x}\mathbf{x_f}^H\right) \circ \mathbf{V}^*.$$

For the random $\mathbf{x}$, the expected value of its GRD is defined as the GRS of $\mathbf{x}$ in order to express the Wiener transfer function in terms of it. Thus, the GRS matrix, denoted as $\overline{\mathbf{R}}_\mathbf{x} = \mathbb{E}\{\mathbf{R_x}\}$,

$$\overline{\mathbf{R}}_\mathbf{x} = \mathbb{E}\left\{\mathbf{x}\mathbf{x_f}^H\right\} \circ \mathbf{V}^*. \tag{A.1}$$

Substituting the first equality in (3.1), as the Wiener filter matrix, into (4.6), the Wiener transfer function matrix is obtained as

$$\boldsymbol{\mathcal{H}}_W = \mathbf{V}^{-\circ} \circ \mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\}\left[\sigma^2\mathbf{I} + \mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\}\right]^{-1}\mathbf{V}. \tag{A.2}$$

Inserting $\mathbf{V}\mathbf{V}^H = \mathbf{I}$ between the expectation and inversion terms above, moving $\mathbf{V}$ inside the expectation, moving $\mathbf{V}^H$ and $\mathbf{V}$ first inside the inversion and then inside the inner expectation, using $\mathbf{x_f} = \mathbf{V}^H\mathbf{x}$, and substituting into (A.2)

$$\mathbb{E}\left\{\mathbf{x}\mathbf{x_f}^H\right\} = \overline{\mathbf{R}}_\mathbf{x} \circ \left(\mathbf{V}^*\right)^{-\circ} \tag{A.3}$$

obtained from (A.1), together with

$$\mathbb{E}\left\{\mathbf{x_f}\mathbf{x_f}^H\right\} = \mathbf{V}^H\left[\overline{\mathbf{R}}_\mathbf{x} \circ \left(\mathbf{V}^*\right)^{-\circ}\right]$$

obtained by multiplying both sides of (A.3) by $\mathbf{V}^H$ from the left, (A.2) is written as

$$\boldsymbol{\mathcal{H}}_W = \mathbf{V}^{-\circ} \circ \left[\overline{\mathbf{R}}_\mathbf{x} \circ \left(\mathbf{V}^*\right)^{-\circ}\right]\left\{\sigma^2\mathbf{I} + \mathbf{V}^H\left[\overline{\mathbf{R}}_\mathbf{x} \circ \left(\mathbf{V}^*\right)^{-\circ}\right]\right\}^{-1} \tag{A.4}$$

that expresses the Wiener transfer function matrix in terms of the GRS matrix of $\mathbf{x}$ and the graph basis matrix $\mathbf{V}$.

Alternatively, substituting the second equality in (3.1) into (4.6), using $\mathbf{x_f} = \mathbf{V}^H\mathbf{x}$, and then substituting (A.3) and

$$\mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\} = \left[\overline{\mathbf{R}}_{\mathbf{x}} \circ (\mathbf{V}^*)^{-\circ}\right] \mathbf{V}^H$$

obtained by multiplying both sides of (A.3) by $\mathbf{V}^H$ from the right, into the resulting expression,

$$\boldsymbol{\mathcal{H}}_W = \mathbf{V}^{-\circ} \circ \left\{\sigma^2 \mathbf{I} + \left[\overline{\mathbf{R}}_{\mathbf{x}} \circ (\mathbf{V}^*)^{-\circ}\right] \mathbf{V}^H\right\}^{-1} \left[\overline{\mathbf{R}}_{\mathbf{x}} \circ (\mathbf{V}^*)^{-\circ}\right]. \tag{A.5}$$

In (A.2), (A.4) and (A.5), matrix product operations have priority over the elementwise product.

(A.4) and (A.5) relate the Wiener transfer function matrix to the GRS matrix of $\mathbf{x}$. However, they do not yield explicit analytical relations between the elements of these matrices, i.e., the explicit analytical expression for the Wiener vertex-frequency transfer function coefficients $\mathcal{H}_W(n, \lambda_l)$ in terms of the GRS coefficients of $\mathbf{x}$ is not available or feasible in the general case of nonstationary random graph signal $\mathbf{x}$ to be estimated.

A.1. Reduction to the Stationary Form

If $\mathbf{x}$ is a GWSS signal, its autocorrelation matrix is diagonalizable by eigenvectors of the graph Laplacian matrix [15, 21, 22]; $\mathbb{E}\left\{\mathbf{x}\mathbf{x}^H\right\} = \mathbf{V}\boldsymbol{\Lambda}_{\mathbf{x}}\mathbf{V}^H$ with diagonal elements of $\boldsymbol{\Lambda}_{\mathbf{x}}$ serving as graph power spectral density coefficients of $\mathbf{x}$.

Substituting for the signal autocorrelation matrix in (A.2) and simplifying the expression using $\mathbf{V}\mathbf{V}^H = \mathbf{V}^H\mathbf{V} = \mathbf{I}$,

$$\boldsymbol{\mathcal{H}}_W = \mathbf{V}^{-\circ} \circ \left[\mathbf{V}\boldsymbol{\Lambda}_{\mathbf{x}} \left(\sigma^2 \mathbf{I} + \boldsymbol{\Lambda}_{\mathbf{x}}\right)^{-1}\right] =$$

$$\left[\frac{\Lambda_x(\lambda_0)}{\sigma^2 + \Lambda_x(\lambda_0)}\mathbf{1}, \frac{\Lambda_x(\lambda_1)}{\sigma^2 + \Lambda_x(\lambda_1)}\mathbf{1}, \cdots, \frac{\Lambda_x(\lambda_{L-1})}{\sigma^2 + \Lambda_x(\lambda_{L-1})}\mathbf{1}\right]$$

where $\mathbf{1}$ denotes an $L \times 1$ dimensional ones vector. The second equality follows from the first one, because multiplying $\mathbf{V}$ by the diagonal matrix from the right scales columns of $\mathbf{V}$ with the corresponding diagonal elements, and then elementwise division by elements of $\mathbf{V}$ leaves constant columns carrying these diagonal elements.

Thus,

$$\mathcal{H}_W(n, \lambda_l) = \frac{\Lambda_x(\lambda_l)}{\sigma^2 + \Lambda_x(\lambda_l)} \tag{A.1.1}$$

for $n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$. (A.1.1) indicates that the vertex-frequency transfer function of the proposed graph Wiener filter reduces to vertex-invariant transfer function of the stationary graph Wiener filter as in [15, 18, 21, 22, 63], when $\mathbf{x}$ is a stationary graph signal.

B. Inversion Methods for the Graph Rihaczek Distribution

Method 1: For the unnormalized graph Laplacian matrix $\boldsymbol{\mathcal{L}} = \mathbf{D} - \mathbf{W}$, the eigenvector associated with the first eigenvalue $\lambda_0 = 0$ is a constant vector, $V(n, \lambda_0 = 0) = 1/\sqrt{L}$, $n = 1, 2, \ldots, L$. Then, the GFT definition in (2.1) yields

$$x_f(\lambda_0 = 0) = \sum_{n=1}^{L} x(n)/\sqrt{L}. \tag{B.1}$$

Substituting (B.1) and $V(n, \lambda_0 = 0) = 1/\sqrt{L}$ into (5.3),

$$R_x(n, \lambda_0 = 0) = x(n)x_f^*(0)V^*(n, 0) = x(n)\sum_{n=1}^{L} x^*(n)/L. \tag{B.2}$$

Summing (B.2) over n, and taking square root of both sides,

$$\sqrt{L\sum_{n=1}^{L} R_x(n, 0)} = \left|\sum_{n=1}^{L} x(n)\right|. \tag{B.3}$$

Dividing (B.2) by (B.3), side by side,

$$x(n) = e^{j\phi}\frac{\sqrt{L}\, R_x(n, \lambda_0 = 0)}{\sqrt{\sum_{n=1}^{L} R_x(n, 0)}} \tag{B.4}$$

for $n = 1, 2, \ldots, L$, where $\phi = \arg\left[\sum_{n=1}^{L} x(n)\right]$ is an indeterminate phase common for all signal samples. For a real valued graph signal $\mathbf{x}$, a similar procedure results in a common indeterminate sign, $\operatorname{sign}\left[\sum_{n=1}^{L} x(n)\right]$, in place of the phase factor, in (B.4).

Hence, graph signal samples can be recovered from the GRD of the graph signal, within a common phase or sign for complex or real signal cases, respectively.

Method 2: For real valued, nonnegative signals, such as images, the vertex-marginal property of the GRD [88, 89] is used to obtain

$$x(n) = |x(n)| = \sqrt{\sum_{l=0}^{L-1} R_x(n, \lambda_l)} \tag{B.5}$$

for $n = 1, 2, \ldots, L$.

C. Mean and Variance of the GRD for the Real Case

Substituting for $y_f(\lambda_l)$ in (5.5), then taking expected value of both sides of (5.5), and substituting, subsequently,

$$\mathbb{E}\left\{y(n)y(k)\right\} = x(n)x(k) + \sigma^2\delta(n-k)$$

resulting from the signal observation model given by (2.3) with zero-mean AWGN uncorrelated with $\mathbf{x}$:

$$\begin{aligned}\mathbb{E}\left\{R_y(n,\lambda_l)\right\} &= V(n,\lambda_l)\sum_{k=1}^{L} V(k,\lambda_l)\left[x(n)x(k) + \sigma^2\delta(n-k)\right] \\ &= x(n)V(n,\lambda_l)\sum_{k=1}^{L} V(k,\lambda_l)x(k) + \sigma^2 V^2(n,\lambda_l).\end{aligned} \tag{C.1}$$

Identifying $x_f(\lambda_l)$ given by (2.1) for the real basis $\mathbf{V}$ in the first term of (C.1), that term above is recognized as $R_x(n,\lambda_l)$ and (6.11) follows.

From (5.5) and (C.1),

$$\begin{aligned}R_y(n,\lambda_l) - \mathbb{E}\left\{R_y(n,\lambda_l)\right\} = \\ V(n,\lambda_l)\sum_{k=1}^{L} V(k,\lambda_l)[x(n)w(k) + w(n)x(k) + w(n)w(k)] - \sigma^2 V^2(n,\lambda_l)\end{aligned} \tag{C.2}$$

Substituting (C.2) into

$$\text{Var}\left\{R_y(n,\lambda_l)\right\} = \mathbb{E}\left\{\left[R_y(n,\lambda_l) - \mathbb{E}\left\{R_y(n,\lambda_l)\right\}\right]^2\right\},$$

and, using $\mathbb{E}\{w(n)w(k)\} = \sigma^2\delta(n-k)$ and the uncorrelatedness of $\mathbf{x}$ and $\mathbf{w}$;

$$\begin{aligned}\text{Var}\left\{R_y(n,\lambda_l)\right\} &= \sigma^4 V^4(n,\lambda_l) - 2\sigma^4 V^3(n,\lambda_l)\sum_{k=1}^{L} V(k,\lambda_l)\delta(n-k) + E \\ &= E - \sigma^4 V^4(n,\lambda_l),\end{aligned} \tag{C.3}$$

where E, above, denotes the mean square value of the first term in (C.2), and hence, it can be expanded as a double sum.

By substituting $\mathbb{E}\{w(n)w(k)w(m)\} = 0$, $\mathbb{E}\{w^2(n)w(k)\} = 0$, $\mathbb{E}\{w^2(n)w(m)\} =$

0 for $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$,

$$\begin{aligned} E =& V^2(n,\lambda_l)\sum_{k=1}^{L}\sum_{m=1}^{L} V(k,\lambda_l)V(m,\lambda_l)[x^2(n)\sigma^2\delta(k-m)+\sigma^2 x(k)x(m) \\ &+ x(k)x(n)\sigma^2\delta(n-m) + x(n)x(m)\sigma^2\delta(n-k) + \mathbb{E}\{w^2(n)w(k)w(m)\}]. \end{aligned} \tag{C.4}$$

Using $\sum_{k=1}^{L} V^2(k,\lambda_l) = 1$ for the first term, recognizing the GFT $x_f(\lambda_l)$ in simplifying second, third and fourth terms, and using $\mathbb{E}\{w^2(n)w(k)w(m)\} = 0$ for $k \neq m$,

$$\mathbb{E}\{w^2(n)w(k)w(m)\} = \mathbb{E}\{w^2(n)\}\mathbb{E}\{w^2(k)\} = \sigma^4$$

for $k = m \neq n$ and $\mathbb{E}\{w^4(n)\} = 3\sigma^4$ [102] for the last term, (C.4) becomes

$$\begin{aligned} E = \sigma^2 V^2(n,\lambda_l)\left[x^2(n) + x_f^2(\lambda_l)\right] + 2\sigma^2 V^3(n,\lambda_l)x(n) \\ \times x_f(\lambda_l) + \sigma^4 V^2(n,\lambda_l)\sum_{k\neq n} V^2(k,\lambda_l) + 3\sigma^4 V^4(n,\lambda_l). \end{aligned} \tag{C.5}$$

Recognizing that $R_x(n,\lambda_l) = x(n)x_f(\lambda_l)V(n,\lambda_l)$ in the second term above, combining the last two terms and using again $\sum_{k=1}^{L} V^2(k,\lambda_l) = 1$, (C.5) becomes

$$\begin{aligned} E = \sigma^2 V^2(n,\lambda_l)\left[x^2(n) + x_f^2(\lambda_l)\right] + 2\sigma^2 V^2(n,\lambda_l)R_x(n,\lambda_l) \\ + L\sigma^4 V^2(n,\lambda_l)\left[1 + 2V^2(n,\lambda_l)\right]. \end{aligned} \tag{C.6}$$

By substituting (C.6) into (C.3), (6.12) is obtained.

D. Mean and Variance of the GRD for the Complex Case

Substituting for the GFT, $y_f(\lambda_l)$, in the GRD of the observed signal and taking the expected value,

$$R_y(n,\lambda_l) = y(n)y_f^*(\lambda_l)V^*(n,\lambda_l)$$

$$\mathbb{E}\{R_y(n,\lambda_l)\} = V^*(n,\lambda_l)\sum_{k=1}^{L} V(k,\lambda_l)\mathbb{E}\{y(n)y^*(k)\}. \tag{D.1}$$

For the complex signal model given by (2.3) with the zero-mean, complex, circular Gaussian noise described by (2.4) and (2.5) that is uncorrelated with $\mathbf{x}$,

$$\mathbb{E}\{y(n)y^*(k)\} = x(n)x^*(k) + \sigma^2\delta(n-k). \tag{D.2}$$

By substituting (D.2) into (D.1),

$$\mathbb{E}\left\{R_y(n,\lambda_l)\right\} = x(n)V^*(n,\lambda_l)\sum_{k=1}^{L}V(k,\lambda_l)x^*(k) + \sigma^2\left|V(n,\lambda_l)\right|^2. \tag{D.3}$$

From (2.1), $x_f^*(\lambda_l)$ is identified in the first term of (D.3). Then, that term above is recognized as $R_x(n,\lambda_l) = x(n)x_f^*(\lambda_l)V^*(n,\lambda_l)$ and (6.13) follows.

From $R_y(n,\lambda_l) = y(n)y_f^*(\lambda_l)V^*(n,\lambda_l)$ with (2.3) substituted, and (D.3);

$$\begin{aligned} R_y(n,\lambda_l) - \mathbb{E}\left\{R_y(n,\lambda_l)\right\} = V^*(n,\lambda_l)\sum_{k=1}^{L}V(k,\lambda_l)[x(n)w^*(k) + w(n)x^*(k) \\ + w(n)w^*(k)] - \sigma^2\left|V(n,\lambda_l)\right|^2. \end{aligned} \tag{D.4}$$

Substituting (D.4) into

$$\text{Var}\left\{R_y(n,\lambda_l)\right\} = \mathbb{E}\left\{\left|R_y(n,\lambda_l) - \mathbb{E}\left\{R_y(n,\lambda_l)\right\}\right|^2\right\},$$

and, using $\mathbb{E}\{w(n)w^*(k)\} = \sigma^2\delta(n-k)$ for the complex, circular noise and the uncorrelatedness of **x** and **w**; Var $\{R_y(n,\lambda_l)\}$ simplifies to

$$\begin{aligned} \text{Var}\left\{R_y(n,\lambda_l)\right\} = E - \sigma^4\left|V(n,\lambda_l)\right|^2\Bigg[V^*(n,\lambda_l)\sum_{k=1}^{L}V(k,\lambda_l)\delta(n-k) \\ + V(n,\lambda_l)\sum_{k=1}^{L}V^*(k,\lambda_l)\delta(n-k)\Bigg] + \sigma^4\left|V(n,\lambda_l)\right|^4 = E - \sigma^4\left|V(n,\lambda_l)\right|^4, \end{aligned} \tag{D.5}$$

where E, above, denotes the mean value of the absolute squared first term in (D.4), and hence, it can be expanded as a double sum.

By substituting these equations $\mathbb{E}\{w^*(n)w^*(k)w(m)\} = 0$,$\mathbb{E}\{w(n)w^*(k)w(m)\} = 0$, $\mathbb{E}\{|w(n)|^2w^*(k)\} = 0$, $\mathbb{E}\{|w(n)|^2w(m)\} = 0$, $\mathbb{E}\{w^*(n)w^*(k)\} = 0$, $\mathbb{E}\{w(n)w(m)\} = 0$, for the complex, circular Gaussian noise described by (2.4) and (2.5), E simplifies to

$$\begin{aligned} E = \left|V(n,\lambda_l)\right|^2\sum_{k=1}^{L}\sum_{m=1}^{L}V(k,\lambda_l)V^*(m,\lambda_l)\left[|x(n)|^2\sigma^2\delta(k-m)\right. \\ \left. + \sigma^2x^*(k)x(m) + \mathbb{E}\{|w(n)|^2w^*(k)w(m)\}\right]. \end{aligned} \tag{D.6}$$

Using $\sum_{k=1}^{L}\left|V(k,\lambda_l)\right|^2 = 1$ for the first term, recognizing the GFT $x_f(\lambda_l)$ given by (2.1) in simplifying the second term; and for the last term, using $\mathbb{E}\{|w(n)|^2w^*(k)w(m)\} = 0$

for $k \neq m$,

$$\mathbb{E}\{|w(n)|^2 w^*(k) w(m)\} = \mathbb{E}\{|w(n)|^2\}\mathbb{E}\{|w(k)|^2\} = \sigma^4$$

for $k = m \neq n$ and

$$\mathbb{E}\{|w(n)|^4\} = \mathbb{E}\left\{\left[w_R^2(n) + w_I^2(n)\right]^2\right\} = 2\sigma^4$$

since $\mathbb{E}\left\{w_{R,I}^2(n)\right\} = \sigma^2/2$ from (2.4) and $\mathbb{E}\left\{w_{R,I}^4(n)\right\} = 3\sigma^4/4$ for Gaussian $w_{R,I}(n)$ [102], (D.6) becomes

$$\begin{aligned} E = {} & \sigma^2 \left|V(n,\lambda_l)\right|^2 \left[|x(n)|^2 + |x_f(\lambda_l)|^2\right] \\ & + \sigma^4 \left|V(n,\lambda_l)\right|^2 \sum_{k \neq n} \left|V(k,\lambda_l)\right|^2 + 2\sigma^4 \left|V(n,\lambda_l)\right|^4 . \end{aligned} \tag{D.7}$$

Combining the last two terms and using again $\sum_{k=1}^{L} \left|V(k,\lambda_l)\right|^2 = 1$, (D.7) becomes

$$E = \sigma^2 \left|V(n,\lambda_l)\right|^2 \left[|x(n)|^2 + |x_f(\lambda_l)|^2\right] + \sigma^4 \left|V(n,\lambda_l)\right|^2 \left[1 + |V(n,\lambda_l)|^2\right]. \tag{D.8}$$

By substituting (D.8) into (D.5), (6.14) is obtained.

E. Graph Rihaczek Distribution for Isometric Shift Operators

E.1. An Isometric Graph Shift Operator

The isometric GSO proposed in [101] is $\mathbf{T} = \exp\left(-j\pi\sqrt{\boldsymbol{\mathcal{L}}/\rho}\right)$ where ρ denotes an upper bound on moduli of eigenvalues of the Laplacian matrix. It can be diagonalized as

$$\mathbf{T} = \mathbf{V} \exp\left(-j\pi\sqrt{\boldsymbol{\Lambda}/\rho}\right) \mathbf{V}^H. \tag{E.1.1}$$

Then, the shifted version of a graph signal $\mathbf{x}$ by k vertices is defined as [101]

$$\mathbf{x}_k = \mathbf{T}^k \mathbf{x} = \mathbf{V} \exp\left(-jk\pi\sqrt{\boldsymbol{\Lambda}/\rho}\right) \mathbf{x_f}, \tag{E.1.2}$$

for $k = 0, 1, \ldots, L-1$. From (E.1.2), nth sample of the k-shifted signal, at vertex n, is given by

$$x_k(n) = \mathbf{V}(n,:) \exp\left(-jk\pi\sqrt{\boldsymbol{\Lambda}/\rho}\right) \mathbf{x_f} = \sum_{m=0}^{L-1} V(n,\lambda_m)\, e^{-jk\pi\sqrt{\lambda_m/\rho}}\, x_f(\lambda_m) \tag{E.1.3}$$

for $n = 1, 2, \ldots, L$. Then, $\mathbf{x_s}(n) = [x_0(n), x_1(n), \ldots, x_{L-1}(n)]^T$ collects nth samples of the shifted versions of $\mathbf{x}$, as a new graph signal with independent variable being the graph shift index k. Another definition of the GRD for this shift operator can be developed by applying the GFT operation to the graph shift index, based on the GFT of $\mathbf{x_s}^*(n)$:

$$R_x^{(i1)}(n, \lambda_l) = x(n)\mathbf{V}^H(\lambda_l, :)\,\mathbf{x_s}^*(n)/\sqrt{L} = x(n)\sum_{k=1}^{L} V^*(k, \lambda_l)\,x_{k-1}^*(n)/\sqrt{L}. \quad \text{(E.1.4)}$$

Substituting (E.1.3) into (E.1.4), and, interchanging inner and outer summations;

$$\begin{aligned} R_x^{(i1)}(n, \lambda_l) =& \frac{x(n)}{\sqrt{L}} \sum_{m=0}^{L-1} x_f^*(\lambda_m) V^*(n, \lambda_m) \left[\sum_{k=1}^{L} V^*(k, \lambda_l) e^{j(k-1)\pi\sqrt{\lambda_m/\rho}}\right] \\ =& \frac{x(n)}{\sqrt{L}} \sum_{m=0}^{L-1} x_f^*(\lambda_m) V^*(n, \lambda_m) \widetilde{V}^*\left(\pi\sqrt{\lambda_m/\rho}, \lambda_l\right), \end{aligned} \quad \text{(E.1.5)}$$

$n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$, where $\widetilde{V}(\omega, \lambda_l)$ stands for the discrete-time Fourier transform of the sequence $V(k, \lambda_l)$.

To check for the frequency-marginal property of this version of the GRD;

$$\sum_{n=1}^{L} R_x^{(i1)}(n, \lambda_l) = \sum_{m=0}^{L-1} |x_f(\lambda_m)|^2\ \widetilde{V}^*\left(\pi\sqrt{\lambda_m/\rho}, \lambda_l\right)/\sqrt{L}$$

by summing both sides of (E.1.5) and using (2.1). The above term does not reduce to $|x_f(\lambda_l)|^2$ for an arbitrary graph basis $\mathbf{V}$, since $\widetilde{V}(\pi\sqrt{\lambda_m/\rho}, \lambda_l) \neq \sqrt{L}\,\delta(l-m)$ in general. Hence, this version of the GRD does not satisfy the frequency-marginal property.

To check for the vertex-marginal property,

$$\sum_{l=0}^{L-1} R_x^{(i1)}(n, \lambda_l) = x(n) \sum_{m=0}^{L-1} x_f^*(\lambda_m) V^*(n, \lambda_m) \left[\sum_{l=0}^{L-1} \widetilde{V}^*\left(\pi\sqrt{\lambda_m/\rho}, \lambda_l\right)\right]/\sqrt{L}$$

from (E.1.5). Since the inner sum can not yield unity for all m, the above term does not reduce to $|x(n)|^2$ for an arbitrary basis $\mathbf{V}$. Hence, the vertex-marginal property is not satisfied either.

A real valued, nonnegative graph signal $\mathbf{x}$ can be recovered from its GRD $R_x^{(i1)}(n, \lambda_l)$ as follows.

From (E.1.4), $\mathbf{r_x}^{(i1)}(n) = \left[R_x^{(i1)}(n, \lambda_0), \ldots, R_x^{(i1)}(n, \lambda_{L-1})\right]^T = x(n)\mathbf{V}^H\mathbf{x_s}^*(n)/\sqrt{L}$

Then,

$$\mathbf{V}\mathbf{r_x}^{(i1)}(n) = \frac{x(n)}{\sqrt{L}}\mathbf{V}\mathbf{V}^H\mathbf{x_s}^*(n) = x(n)\,\mathbf{x_s}^*(n)/\sqrt{L} \quad \text{(E.1.6)}$$

since $\mathbf{V}\mathbf{V}^H = \mathbf{I}$. Since $\mathbf{x}_0 = \mathbf{x}$ from (E.1.2), (E.1.6) yields

$$x(n) = |x(n)| = \sqrt{\sqrt{L}\left[\mathbf{V}\mathbf{r_x}^{(i1)}(n)\right](1)}, \tag{E.1.7}$$

$n = 1, 2, \ldots, L$, from the first element of $\mathbf{V}\mathbf{r_x}^{(i1)}(n)$.

E.2. Another Isometric Graph Shift Operator

Another isometric GSO proposed in [19] can be adapted to be based on the Laplacian matrix, as

$$\mathbf{A}_e = \mathbf{V}\,\mathrm{diag}\left[e^{-j\frac{2\pi}{L}0}, \ldots, e^{-j\frac{2\pi}{L}l}, \ldots, e^{-j\frac{2\pi}{L}(L-1)}\right]\mathbf{V}^H \tag{E.2.1}$$

$\mathbf{V}$ being the eigenvector matrix of the Laplacian. k-shifted versions of a graph signal $\mathbf{x}$ are then obtained as $\mathbf{x}_k = \mathbf{A}_e^k\mathbf{x}$, $k = 0, 1, \ldots, L-1$. By collecting nth samples of k-shifted signals to form a shift vector and applying the GFT operation to its conjugate, as done in the previous section for the first isometric shift operator, another definition of the GRD of $\mathbf{x}$ for this shift operator given by (E.2.1) can be obtained, as follows:

$$\begin{aligned} R_x^{(i2)}(n, \lambda_l) &= x(n)\sum_{m=0}^{L-1} x_f^*(\lambda_m)V^*(n, \lambda_m)\left[\sum_{k=1}^{L} V^*(k, \lambda_l)e^{j\frac{2\pi}{L}m(k-1)}\right]/\sqrt{L} \\ &= x(n)\sum_{m=0}^{L-1} x_f^*(\lambda_m)V^*(n, \lambda_m)\widetilde{V}^*(m, \lambda_l), \end{aligned} \tag{E.2.2}$$

$n = 1, 2, \ldots, L$, $l = 0, 1, \ldots, L-1$, where $\widetilde{V}(m, \lambda_l)$ denotes the L-point DFT of the sequence $V(k, \lambda_l)$. For the DFT basis $V(k, \lambda_l) = \exp[j(2\pi/L)l(k-1)]/\sqrt{L}$, $k = 1, 2, \ldots, L$, (E.2.2) reduces to the original GRD definition of (5.3), since the inner sum above yields $\delta(l-m)$.

From (E.2.2),

$$\sum_{n=1}^{L} R_x^{(i2)}(n, \lambda_l) = \sum_{m=0}^{L-1} |x_f(\lambda_m)|^2\ \widetilde{V}^*(m, \lambda_l)$$

by using (2.1). For the DFT basis, $\widetilde{V}(m, \lambda_l) = \delta(l-m)$ and the above sum yields $|x_f(\lambda_l)|^2$. Hence, the frequency-marginal property is satisfied for the DFT basis. For other graph bases, this version of the GRD does not satisfy the frequency-marginal property.

To check for the vertex-marginal property,

$$\sum_{l=0}^{L-1} R_x^{(i2)}(n,\lambda_l) = x(n) \sum_{m=0}^{L-1} x_f^*(\lambda_m) V^*(n,\lambda_m) \left[\sum_{l=0}^{L-1} \widetilde{V}^*(m,\lambda_l)\right]$$

from (E.2.2) again. For the DFT basis, $\widetilde{V}(m,\lambda_l) = \delta(l-m)$ and the inner sum yields unity for all m. Hence, the above term reduces to $|x(n)|^2$ by using (2.2), and the vertex-marginal property is satisfied for the DFT basis. For an arbitrary graph basis $\mathbf{V}$, the inner sum can not yield a constant value for all m; hence, the vertex-marginal property is not satisfied.

Inversion Method 1: By substituting $\lambda_0 = 0$ and $V(k,\lambda_0 = 0) = 1/\sqrt{L}$, $k = 1,2,\ldots,L$ into (E.2.2), $R_x^{(i2)}(n,\lambda_0 = 0)$ is obtained as in (B.2). Hence, graph signal samples $x(n)$ are recovered from $R_x^{(i2)}(n,\lambda_0 = 0)$ by (B.4), as for the original GRD definition of (5.3), within a common phase or sign for complex or real signal cases, respectively.

Inversion Method 2: To recover a real valued, nonnegative graph signal $\mathbf{x}$, $\mathbf{r_x}^{(i2)}(n)$ $= \left[R_x^{(i2)}(n,\lambda_0),\ldots,R_x^{(i2)}(n,\lambda_{L-1})\right]^T$ can be defined for each n, as done for the first isometric shift operator in the previous section. Since $R_x^{(i2)}(n,\lambda_l)$ is also defined by (E.1.4), for the shift operator of (E.2.1), (E.1.6) and (E.1.7) also hold for $\mathbf{r_x}^{(i2)}(n)$. Hence, $x(n)$ is obtained from $\mathbf{r_x}^{(i2)}(n)$ by (E.1.7) for each n.

F. Graph Parameter Values for Image Denoising Results

Gaussian pre-filtering parameters, smoothing parameters in the photometric and spatial domains, and window sizes yielding graph weights are listed for all patches of all images in Tables F.1 and F.2, for the results given in Tables 9.1 and 9.2, respectively.

Table F.1. *Best parameters employed in graph construction and Gaussian pre-filtering are presented for $\sigma = 15$. Patches are numbered from left to right in each patch row and numbering continues as such in the upper row. h_x^2, h_p^2 are squared smoothing parameters in the photometric and pixel (spatial) domains, respectively, while $h_{p,G}^2$ denotes smoothing parameter used in Gaussian pre-filtering step, and w denotes patch size around pixels in* (8.1)

Image	***Parameter***	***Patch Number***															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Box	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	$h_{p,G}^2$	0.875	1	0.875	0.875	1	1	1	1	0.875	1	1	1	1	1	1	1
	h_x^2	75	75	60	75	52.5	45	75	45	75	75	52.5	75	60	75	75	60
	h_p^2	0.5	0.5	0.5	0.675	0.5	0.5	0.5	0.5	0.625	0.5	0.75	0.75	0.5	0.5	0.5	0.5
Stripes	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	$h_{p,G}^2$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	h_x^2	75	75	60	75	67.5	45	75	75	75	60	52.5	75	60	75	75	60
	h_p^2	0.5	0.5	0.5	0.675	0.5	0.5	0.5	0.5	0.5	0.625	0.875	0.75	0.5	0.5	0.5	0.5
Teddy	w	1	1	1	1	7	7	3	3	5	5	5	5	5	5	5	5
	$h_{p,G}^2$	0.875	2	0.7	1	1.5	1.25	1	1	2.375	2.25	2.25	2.5	2.375	2.625	2.25	2.375
	h_x^2	0.019	0.039	0.019	0.019	0.039	0.049	0.009	0.009	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
	h_p^2	2.5	4.75	4.5	6	7	7	8	8	6	6	8	8.25	6	5.25	7.75	8
Cones	w	3	3	5	5	3	3	5	5	5	5	7	7	5	5	7	7
	$h_{p,G}^2$	1.5	1.45	2.375	2.25	1.5	1.5	2.375	2.5	2.25	2.375	2.5	2.5	2.375	2.375	2.625	2.5
	h_x^2	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
	h_p^2	6	6	8	7.875	6	6.25	8	8	6	6	7.75	8	6	5.875	8	8.125
Cameraman	w	1	3	5	5	5	3	7	3	3	3	1	9	1	5	3	1
	$h_{p,G}^2$	0.75	0.625	0.5	0.5	1	0.5	0.875	0.5	0.5	0.05	0.001	1.25	1	0.5	0.1	1.25
	h_x^2	0.019	0.059	0.117	0.117	0.059	0.078	0.117	0.098	0.059	0.176	0.039	0.117	0.078	0.117	0.117	0.117
	h_p^2	1.25	0.5	0.5	0.5	0.525	0.75	1	0.5	1.125	0.525	1.2	3.4	0.3	0.105	0.5	0.562
Montage	w	3	1	5	5	3	5	5	7	3	3	3	3	3	1	3	3
	$h_{p,G}^2$	0.6	1.25	1	1.25	2	0.5	0.75	0.625	0.5	0.525	0.65	10	0.525	0.3	0.75	0.6875
	h_x^2	0.029	0.004	0.059	0.059	0.039	0.098	0.098	0.078	0.068	0.039	0.019	0.078	0.039	0.078	0.019	0.019
	h_p^2	3.5	1	0.875	0.875	3	1	1	1.5	1.5	2	2	1.375	2	3.5	2	2
Peppers	w	1	1	5	3	7	5	7	3	5	7	3	3	3	3	3	3
	$h_{p,G}^2$	1.125	1.125	0.6	0.475	0.8	0.6	0.8	0.5	0.6	0.8	0.65	0.75	1	0.75	0.75	1
	h_x^2	0.004	0.004	0.068	0.059	0.059	0.098	0.117	0.078	0.098	0.059	0.019	0.019	0.019	0.019	0.019	0.019
	h_p^2	0.675	0.675	0.9	0.725	0.875	0.7	2.5	0.675	0.7	0.875	1.6	1.6	1.6	1.6	1.6	1.6
House	w	1	1	5	3	5	5	7	5	7	3	1	11	5	7	3	1
	$h_{p,G}^2$	0.75	1.25	0.5	0.45	0.725	0.45	0.8	0.5	0.5	0.375	0.75	1.125	0.875	0.5	0.25	1.125
	h_x^2	0.019	0.004	0.068	0.059	0.059	0.098	0.117	0.147	0.059	0.117	0.019	0.117	0.117	0.117	0.068	0.117
	h_p^2	3	0.675	0.9	0.725	0.875	0.8	1.25	0.515	0.75	0.525	1.2	3.4	0.3	0.105	0.75	0.75
Lena	w	7	7	3	5	7	3	7	3	7	3	3	3	7	3	3	3
	$h_{p,G}^2$	0.8	0.8	1	0.6	0.8	0.75	0.8	1	0.8	0.75	0.75	0.75	0.9	0.75	1	1
	h_x^2	0.059	0.117	0.019	0.098	0.059	0.019	0.059	0.019	0.059	0.019	0.019	0.019	0.059	0.019	0.019	0.019
	h_p^2	1	2	1.5	0.625	0.875	1.75	0.875	1.625	0.875	1.375	1.25	2	1	1.5	1.625	2
Boats	w	7	3	7	7	3	3	3	3	3	3	3	3	5	3	3	3
	$h_{p,G}^2$	0.9	0.75	0.8	0.8	0.75	0.625	0.75	0.75	0.75	0.875	0.675	0.75	0.75	0.75	0.75	0.675
	h_x^2	0.059	0.019	0.059	0.059	0.019	0.019	0.019	0.019	0.019	0.019	0.023	0.019	0.098	0.019	0.019	0.019
	h_p^2	1	1.5	0.95	0.95	1.5	1.5	1.45	1.375	1.375	1.5	1.5	1.75	1.5	1.5	1.5	1.625
Barbara	w	3	3	3	5	5	3	3	5	3	3	5	5	3	3	3	3
	$h_{p,G}^2$	0.625	0.7	0.825	0.6	0.6	1	0.675	0.5	0.75	0.875	0.5	0.5	0.75	0.75	0.75	0.825
	h_x^2	0.019	0.019	0.019	0.098	0.098	0.019	0.019	0.098	0.019	0.019	0.098	0.098	0.019	0.019	0.019	0.019
	h_p^2	1.75	1.5	1.5	1.5	1.375	1.375	1.5	1.5	1.375	1.5	1.5	1.75	1.5	1.5	1.6	1.25
Parrot	w	3	3	3	7	3	3	3	5	3	3	1	5	3	3	3	7
	$h_{p,G}^2$	0.825	0.675	0.925	0.9	0.75	0.675	0.675	0.8	0.75	0.75	1.125	0.5	0.675	0.75	0.675	0.8
	h_x^2	0.019	0.019	0.019	0.059	0.023	0.019	0.019	0.098	0.019	0.023	0.004	0.098	0.019	0.019	0.019	0.059
	h_p^2	1.5	1.5	1.25	1	1.675	1.6	1.5	1.5	1.375	1.5	0.675	1.75	1.5	1.5	1.75	1
Baboon	w	7	7	7	7	7	7	7	7	5	7	7	5	5	5	5	7
	$h_{p,G}^2$	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.9	0.5	0.6	0.6	0.6	0.8
	h_x^2	0.059	0.059	0.059	0.059	0.059	0.059	0.067	0.059	0.098	0.059	0.059	0.098	0.098	0.098	0.091	0.067
	h_p^2	0.875	1	1	1.1	0.9	1	0.9	1	1.5	0.9	1	1.75	1.5	1.5	1.75	1
Grass	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	$h_{p,G}^2$	1.25	1	1.375	1.125	1.25	1.125	1.125	1.125	1.25	1.325	1.25	1	1	1.25	1.125	1.125
	h_x^2	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004
	h_p^2	0.625	0.675	0.675	0.675	0.675	0.675	0.675	0.675	0.675	0.675	0.625	0.675	0.675	0.675	0.625	0.675

Table F.2. *Best parameters employed in graph construction and Gaussian pre-filtering are presented for Depth and Synthetic images, for different standard deviations (σ). Best parameters for $\sigma = 15$ are already presented in Table F.1*

σ	*Image*	*Para.*	*Patch Number*															
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
5	Box	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	0.875	1	0.875	0.875	1	1	1	1	0.875	1	1	1	0.875	1	1	1
		h^2_x	75	75	60	75	52.5	45	75	45	75	75	52.5	75	60	75	75	60
		h^2_p	0.5	0.5	0.5	0.675	0.5	0.5	0.5	0.5	0.625	0.5	0.75	0.75	0.5	0.625	0.5	0.5
	Stripes	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		h^2_x	75	67.5	60	75	67.5	45	67.5	75	67.5	60	52.5	75	60	67.5	75	60
		h^2_p	0.5	0.625	0.5	0.675	0.5	0.625	0.5	0.5	0.5	0.5	0.875	0.75	0.5	0.625	0.5	0.5
	Teddy	w	9	9	13	13	9	9	11	13	15	15	17	17	13	15	17	15
		$h^2_{p,G}$	0.75	0.75	1.375	1.25	0.625	0.75	1.375	1.125	2.25	2.375	2.5	2.375	2.375	2.5	2.625	2.75
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	6	6.125	8.5	8.75	5.75	6	8.875	8.625	7.5	7.25	8	7.75	7.5	7.5	7.875	8
	Cones	w	13	13	21	19	13	15	21	21	15	15	13	13	15	13	13	15
		$h^2_{p,G}$	1.25	1.25	2.75	2.875	1.125	1.25	2.75	2.625	2.25	2.375	2	2.25	2.125	2.25	2.125	2
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	7	7.125	7.625	7.75	7	7	7.75	7.5	6.25	6.5	4	5	6.25	6.625	4.75	3.875
10	Box	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	0.875	1	0.875	0.875	1	1	1	1	0.875	1	1	1	1.125	1	1	1
		h^2_x	75	75	60	75	52.5	45	75	37.5	75	67.5	52.5	75	60	75	75	60
		h^2_p	0.5	0.625	0.5	0.675	0.5	0.5	0.75	0.5	0.625	0.5	0.75	0.75	0.5	0.5	0.5	0.5
	Stripes	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		h^2_x	67.5	75	60	75	75	45	67.5	75	75	60	52.5	75	60	60	75	60
		h^2_p	0.5	0.625	0.625	0.675	0.5	0.625	0.5	0.5	0.75	0.5	0.875	0.75	0.625	0.625	0.5	0.5
	Teddy	w	5	5	7	7	3	5	7	7	7	7	9	9	7	9	9	9
		$h^2_{p,G}$	1	1.125	1.375	1.5	1.125	1.125	1.5	1.625	2.5	2.25	2	2.125	2.625	2.5	2	2
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	6	6.125	9	9.125	5.75	5.875	8.75	9	7.5	7.75	8	8	7.375	7.5	7.875	8
	Cones	w	7	7	9	9	7	9	9	9	9	9	7	9	9	9	7	9
		$h^2_{p,G}$	1.5	1.25	2.5	2.25	1.75	1.375	2.5	2.5	2.375	2.25	2.375	2.375	2.5	2.625	2.375	2.25
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	7	6.75	7.75	8	7	7	8	7.625	6.25	6.5	8	7.5	6.625	6.5	8	7.75
20	Box	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	0.875	1	1.125	0.875	1	1	1	1	0.95	1	1	1.125	1	1	1	1
		h^2_x	75	45	60	75	37.5	45	75	45	60	75	52.5	75	60	75	67.5	60
		h^2_p	0.5	0.5	0.5	0.675	0.35	0.5	0.5	0.5	0.625	0.5	0.75	0.675	0.5	0.5	0.5	0.375
	Stripes	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		h^2_x	75	67.5	67.5	75	67.5	52.5	67.5	75	60	60	52.5	75	67.5	67.5	75	60
		h^2_p	0.375	0.625	0.5	0.75	0.5	0.625	0.5	0.625	0.5	0.5	0.75	0.75	0.625	0.625	0.5	0.5
	Teddy	w	3	3	3	3	3	5	3	3	5	5	5	5	5	3	5	5
		$h^2_{p,G}$	1.125	1	2.75	2.625	1.125	1.25	2.5	2.75	2.5	2.375	3	2.75	2.5	3	2.875	3
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	6.875	7	8	7.75	6.75	7.125	8	7.875	7	6.75	12	11.5	7	7.5	11.25	12
	Cones	w	3	3	1	1	3	5	1	1	5	5	5	7	5	5	3	5
		$h^2_{p,G}$	2	1.75	3.5	3.25	2.25	2	3.375	3.5	4	3.75	2.25	2.5	4	4.125	2.5	2.75
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	6	5.75	6	6.5	6.125	6	6.75	6.5	5.5	5.75	8	7.5	5.875	5.5	8	8.25
25	Box	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	0.875	1	0.875	0.875	1	1	1	1	0.875	1	1	1	1	0.875	1	1
		h^2_x	75	75	60	75	52.5	45	75	52.5	75	52.5	75	60	67.5	75	67.5	60
		h^2_p	0.5	0.5	0.5	0.675	0.5	0.5	0.5	0.5	0.625	0.5	0.75	0.75	0.5	0.5	0.5	0.5
	Stripes	w	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		$h^2_{p,G}$	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
		h^2_x	75	67.5	60	67.5	67.5	60	67.5	75	67.5	67.5	52.5	75	60	67.5	67.5	60
		h^2_p	0.5	0.625	0.625	0.675	0.5	0.625	0.625	0.5	0.5	0.5	0.75	0.75	0.5	0.625	0.5	0.5
	Teddy	w	3	3	3	3	3	3	5	3	3	3	3	3	3	3	3	3
		$h^2_{p,G}$	2	1.5	1.625	1.75	1.875	2	1.5	1.75	2.25	2	2.5	2.25	2	2	2.5	2.25
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	6.625	6.625	7.25	7.5	6.75	6.625	7.75	7.5	7.5	7.25	7.5	7.5	7.25	7.375	7.5	7.375
	Cones	w	3	3	3	5	3	3	3	3	3	3	5	5	3	3	5	5
		$h^2_{p,G}$	3.5	3.25	3.25	3	3.5	3.625	3.375	3.25	2.625	2.75	3.5	3.25	2.5	2.625	3.5	3.125
		h^2_x	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019	0.019
		h^2_p	6	6.125	8.5	8	6.25	5.75	8.25	8.625	6	6.25	8.5	9	6.125	6	8	8.375

CURRICULUM VITAE

Name:	Ali Can YAĞAN
Foreign Languages:	Turkish, English
Birth date and place:	04/03/1991, Gaziantep
E-mail:	alicanyagan@eskisehir.edu.tr
ORCID ID:	0000-0003-4752-6862

Education

- PhD (2017–2020), Eskişehir Technical University, Department of Electrical and Electronics Engineering
- MSc (2014–2016), Anadolu University, Department of Electrical and Electronics Engineering
- BSc (2008–2013), Mustafa Kemal University, Department of Electrical and Electronics Engineering

Experience

- 2014– , Research Assistant, Eskişehir Technical University
- 2013–2014, Electrical Engineer, Alp Erdem Electric Inc.

Publications

- Yağan, A.C. and Özgen, M.T. (2020). Spectral graph based vertex-frequency Wiener filtering for image and graph signal denoising. *IEEE Transactions on Signal and Information Processing over Networks*, 6 (1), 226–240.
- Yağan, A.C. and Özgen, M.T. (2020). Vertex-frequency Wiener filter in the windowed graph Fourier domain for graph signal denoising. *IEEE Signal Processing Letters*, (in peer review).
- Yağan, A.C. and Özgen, M.T. (2017). A spectral graph based image coding method. In: *2017 IEEE 25th Signal Processing and Communications Applications Conference (SIU)*.
- Yağan, A.C. and Özgen, M.T. (2016). A spectral graph Wiener filter in graph Fourier domain for improved image denoising. In: *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 450–454.