

A Harmonic Journey through Wavelets on Graphs

Wolfgang Erb^[0000–0003–3541–5401]

Abstract Graph wavelets provide a powerful framework for multiscale decompositions of data defined on graphs and networks. In comparison to classical wavelets, which are built upon translation and dilation in Euclidean spaces, graph wavelets must adapt to the intrinsically irregular geometry of the underlying graph. These lecture notes take the reader on a journey through modern approaches to wavelet constructions on graphs, presenting the fundamental ideas of spectral, spatial, and diffusion-based methodologies, as well as novel signal-adaptive approaches based on wedgelet decompositions.

Wolfgang Erb
Università degli Studi di Padova, Dipartimento di Matematica “Tullio Levi-Civita”, Via Trieste 63,
35121 Padova, Italy, e-mail: wolfgang.erb@unipd.it

Contents

1	Introduction	3
2	Graph structures	6
3	Spectral Graph Theory	10
	3.1 The graph Laplacian	10
	3.2 Basic properties of the graph Laplacian	12
4	The graph Fourier transform (GFT)	16
	4.1 The Fourier transform of a graph signal	16
	4.2 Numerical calculation of the GFT	19
	4.3 Graph convolution and filtering	20
	4.4 Polynomial approximation of the graph convolution	21
	4.5 Denoising of graph signals	23
5	Spectral Graph Wavelet Transform	25
	5.1 Spectral graph wavelets and scaling functions	25
	5.2 Frames and reconstruction from spectral wavelet coefficients	27
	5.3 Efficient calculation of wavelet coefficients	31
6	Diffusion wavelets on graphs	31
	6.1 Construction of diffusion scaling functions	33
	6.2 Wavelet spaces and diffusion wavelets	34
7	Multiresolution analysis with binary graph partitionings (BGPs)	36
	7.1 Haar wavelets based on BGPs	36
	7.2 m -term approximation for wavelets on BGPs	42
	7.3 Advanced basis systems based on BGPs	49
8	Graph wedgelets on adaptive graph partitionings	53
	8.1 Wedge splits and binary wedge partitioning (BWP) trees	53
	8.2 Signal-adaptive greedy generation of BWP trees	55
	8.3 Geometric wavelets based on wedge splits	58
	8.4 Memory requirements for wedgelet encoding	60
	8.5 Examples on graphs and images	61
9	Software resources	65

1 Introduction

The rapid growth of digitally available and transmitted data has created a strong demand for efficient representation and compression techniques in signal processing and data analysis. In many modern applications, data is not defined on regular Euclidean spaces, but rather on irregular structures that are able to describe complex relations between different data points. Such structures can naturally be modeled in terms of graphs, in which the vertices represent the data points and the edges describe interactions or similarities between the points. Examples include social networks, transportation systems, or biological interaction graphs.

Classical signal processing tasks such as noise reduction or data compression have natural counterparts for graph signals, i.e., functions defined on the vertices of the graph. Many classical harmonic analysis tools can be generalized to graphs by exploiting the spectral decomposition of the graph Laplacian. In particular, the eigenvalues and eigenvectors of the graph Laplacian give rise to graph analogues of fundamental operations such as the graph Fourier transform, convolutional filters, and different multiscale representations. A general overview of these developments in graph signal processing can be found in [44, 45, 57, 60].

Among the available tools, wavelet constructions on graphs are of particular interest because they provide multiscale decompositions that contain both global and local features of graph signals. These representations are particularly useful for applications such as data compression and learning tasks, where sparse representations of signals play a central role. Over the past two decades, several different approaches have been proposed to construct wavelet-type systems on graphs.

These lecture notes provide an introduction to different construction principles for wavelets and multiscale representations on general graphs. The notes start with a brief overview of spectral graph theory, which provides the mathematical foundation for many constructions in graph signal processing. In particular, the graph Laplacian, its spectral decomposition, and its main algebraic properties will be discussed. These discussions form the basis for several more advanced constructions, starting with the graph Fourier transform and the convolution on graphs.

One important class of wavelets is directly based on the graph Fourier transform. In this class, wavelets are defined through spectral filters applied to the eigenvalues of the graph Laplacian, leading to representations which are localized in the spectral domain, but also in the vertex domain. Such spectral constructions include spectral graph wavelets, but also more general frames that are able to decompose a graph signal in the vertex-frequency domain [35, 56, 58, 59]. In a similar way, graph wavelet packets [2, 11] and wavelet filterbanks [46] can be constructed, which generalize the respective concepts to irregular graphs using techniques from spectral graph theory.

A second major construction principle for graph wavelets discussed in these notes relies on a multiresolution analysis generated with the help of a diffusion operator on the graph. Increasing powers of this diffusion operator act as a dilation operation on the graph and generate a family of scaling spaces containing scaling functions with different spatial localization. The orthogonal elements of the resulting detail spaces are known as diffusion wavelets [3, 14, 62].

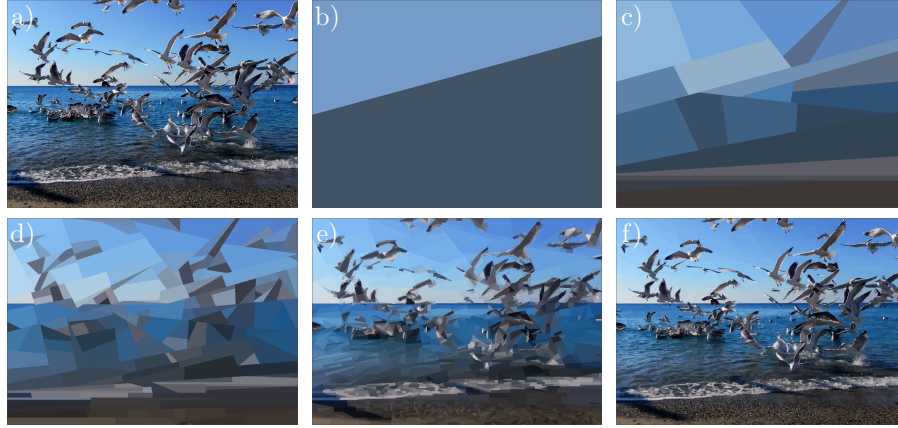


Fig. 1 Wedgelet decomposition of an image showing a beach at Cetraro (Cosenza, Italy). a) Original 1075×806 image "Cetraro - Comune di Cetraro - 2023-09-24 10-35-43 001.jpg" downloaded from wikimedia commons; b)c)d)e)f) Graph wedgelet compression using an approximation of the image in 2, 20, 200, 2000 and 20000 piecewise constant wedgelets, respectively.

A conceptually different third class of wavelet constructions is based on hierarchical graph partitionings. In this approach, the graph is recursively clustered into subsets of vertices, generating partitioning trees that induce a multiresolution decomposition of the signal. The associated basis elements are Haar-type scaling functions and wavelets supported on the subsets defined by the partitioning tree [13, 30, 43]. Partition-based constructions are attractive for applications, as a wide range of clustering algorithms is available for graphs. Moreover, the resulting orthogonal wavelet decomposition is computationally efficient and can be implemented easily using the underlying partitioning trees. Specific constructions based on hierarchical spectral clustering, k-medoids clustering, or spanning trees are described in [36, 37, 38, 54]. More general wavelet-type orthogonal systems on weighted partitioning trees have been studied in [9]. Partitioning based constructions have found applications in machine learning, for example in scattering networks [8] and semi-supervised learning [30]. Furthermore, improved Haar-type constructions for smooth signal classes have been obtained through learning-based approaches [51]. A further conceptually similar strategy is the usage of lifting-based wavelet constructions. This strategy extends the lifting scheme from classical wavelet theory to graph signals [39, 55].

In most partition-based approaches the hierarchical decomposition tree is determined only by the topological structure of the graph and does not include properties of graph signals. As shown in [30], the structure of the partitioning tree has a strong impact on the sparsity of the resulting wavelet representation. For data compression or sparse approximation, it is therefore of main importance to construct partitions that adapt not only to the graph topology but also to the geometric structure of the signal itself. To this end, in the final chapter of these notes we study wedgelet decompositions [26, 28] of graph signals, which rely on constructing signal-adaptive partitioning trees using elementary wedge splits.

The development of such signal-adaptive constructions can be motivated by classical image compression methods. In these methods efficient compression is achieved by combining adaptive geometric partitions with simple local approximation models as, for instance, constant functions or low-degree polynomials. Important examples in image processing include adaptive triangulations [12, 18, 19, 20], quadtree decompositions [42, 53], tetralets [41], wedgelets [22, 29, 65], and binary space partitioning (BSP) trees [48, 49]. In particular for binary space partitioning trees there exists a well-developed theory on m -term approximation with geometric wavelets [17, 40] that can be also extended to graphs [26]. On the other hand, as images can be interpreted as particular graphs, all the wavelet decompositions presented in this work can be applied also for the compression and approximation of images, as for instance shown in Fig. 1.

The material presented in these notes is based on lectures given in the course "Mathematical Models and Numerical Methods for Big Data" in the data science master degree program at the University of Padova, as well as on a series of lectures given in July 2025 at the C.I.M.E. summer school "Modern Perspectives in Approximation Theory: Graphs, Networks, quasi-interpolation and Sampling Theory" in Cetraro, Italy. The purpose of these notes is to provide a concise and accessible introduction to the main ideas for the most relevant wavelet constructions on graphs. In the manuscript, further references to relevant literature and to available software implementations are provided to facilitate further research and experimentation on this topic.

2 Graph structures

Throughout this lecture, we will consider a graph G as a triplet $G = (V, E, \mathbf{A})$ consisting of the following principal components:

- A finite set $V = \{v_1, \dots, v_n\}$ of n **vertices**;
- A set $E \subset V \times V$ of **edges** of the graph G containing all edges $e_{i,j} = (v_i, v_j)$ from a node v_i to a node v_j . For simplicity, we will assume that the graph is **undirected**, i.e., if $e_{i,j} \in E$ then also $e_{j,i} \in E$;
- A symmetric **adjacency matrix** $\mathbf{A} \in \mathbb{R}^{n \times n}$ with non-negative weights

$$\begin{cases} \mathbf{A}_{i,j} > 0 & \text{if } e_{i,j} = (v_i, v_j) \in E, \\ \mathbf{A}_{i,j} = 0 & \text{if there is no edge from } v_i \text{ to } v_j. \end{cases}$$

To simplify our considerations, we implicitly also assume that

- the graph G is **simple**, i.e. that there are no loops and no multiple edges.
- there are no singletons, i.e., every vertex is connected to at least one other vertex.

In the following, if no weights $\mathbf{A}_{i,j}$ are specified, by default we assume that all edges have weight equal to 1. Further, for differing weights, the values $\mathbf{A}_{i,j}$ are interpreted in such a way that the larger the value $\mathbf{A}_{i,j}$ is, the stronger the link between the node v_i and v_j . To provide some illustrative examples for which we can validate our results, we will consider the following simple graphs.

- 1) The **path graph** P_n with n vertices: if the vertices are indexed by $\{1, \dots, n\}$, the (undirected) edges of the path graph are $e_{i,i+1} = (i, i+1)$ for $i \in \{1, \dots, n-1\}$.
- 2) The **circle graph** C_n with n vertices: with the vertices represented as $\{1, \dots, n\}$, the (undirected) edges of the circle graph are those of the path graph plus the additional edge $e_{n,1} = (n, 1)$.
- 3) The **complete graph** K_n with n vertices: every vertex $j \in \{1, \dots, n\}$ is connected to any other vertex $i \in \{1, \dots, n\}$, $i \neq j$ with an edge.

To set up a spectral theory for graphs, we require in addition to the adjacency matrix $\mathbf{A}$ further $n \times n$ -matrices linked to the topological structure of the graph. One important example is the **degree matrix** $\mathbf{D} \in \mathbb{R}^{n \times n}$ given as the diagonal matrix with the entries

$$\begin{cases} \mathbf{D}_{i,i} = \sum_{j=1}^n \mathbf{A}_{i,j} \\ \mathbf{D}_{i,j} = 0 & \text{if } i \neq j. \end{cases}$$

The diagonal entry $\mathbf{D}_{i,i}$ is referred to as **degree** of the vertex v_i . In the standard case that all edge weights are equal to 1 the number $\mathbf{D}_{i,i}$ corresponds to the number of edges starting from the node v_i .

As we assume that G contains no isolated vertices, we have $\mathbf{D}_{i,i} > 0$. If every vertex of a graph has the same degree, the graph is called **regular**. The circle graph C_n and the complete graph K_n are regular graphs, whereas the path graph P_n is not.

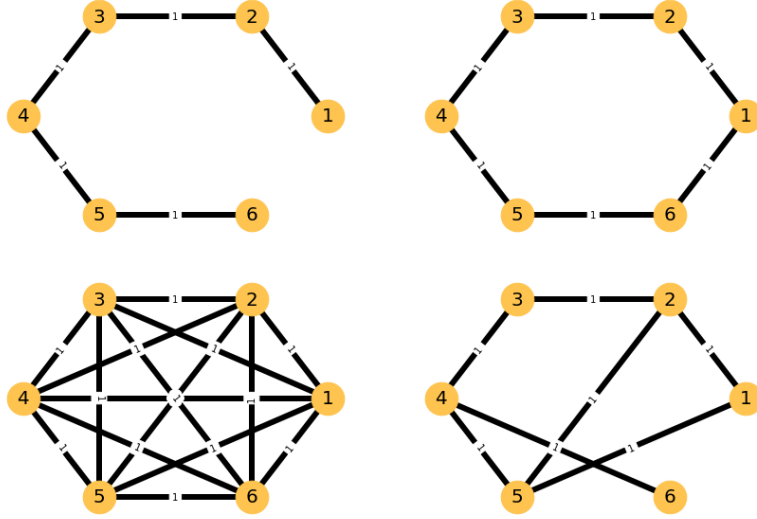
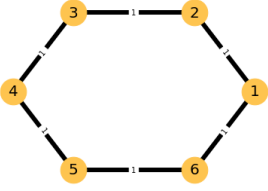
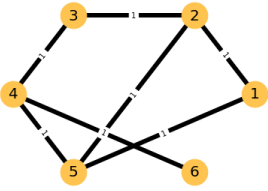
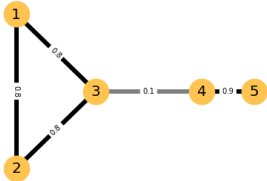


Fig. 2 4 simple graphs. Above left: path graph P_6 . Above right: circle graph C_6 . Below left: complete graph K_6 . Below right: irregular graph.

Example 2.1 We provide the adjacency and the degree matrix for the cycle graph C_6 , a slightly more irregular graph with 6 nodes, and a weighted graph with 5 nodes.

Graph	Adjacency matrix	Degree matrix
	$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

Weighted graph	Adjacency matrix	Degree matrix
	$\begin{bmatrix} 0 & 0.8 & 0.8 & 0 & 0 \\ 0.8 & 0 & 0.8 & 0 & 0 \\ 0.8 & 0.8 & 0 & 0.1 & 0 \\ 0 & 0 & 0.1 & 0 & 0.9 \\ 0 & 0 & 0 & 0.9 & 0 \end{bmatrix}$	$\begin{bmatrix} 1.6 & 0 & 0 & 0 & 0 \\ 0 & 1.6 & 0 & 0 & 0 \\ 0 & 0 & 1.7 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0.9 \end{bmatrix}$

The weight $\mathbf{A}_{i,j} > 0$ of an edge $e_{i,j}$ is a measure of the strength of the relation between the two neighboring nodes v_i and v_j . As a distance between v_i and v_j we can therefore take the inverse $\mathbf{A}_{i,j}^{-1}$ and define the **length** $\ell(e_{i,j})$ of the edge $e_{i,j}$ as

$$\ell(e_{i,j}) = \ell(v_i, v_j) = \frac{1}{\mathbf{A}_{i,j}} \quad \text{if } v_i \text{ is connected with } v_j.$$

A **path** in G is a finite sequence $c = (w_1, \dots, w_k)$ of vertices of G in which any pair (w_j, w_{j+1}) is connected by an edge. The **length** $\ell(c)$ of the path c is given by the sum of the edge lengths along the edges of the paths, i.e.,

$$\ell(c) = \sum_{i=1}^{k-1} \ell(w_i, w_{i+1}).$$

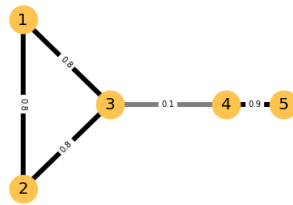
The **minimal length** of a path connecting two nodes v and w is referred to as the **graph distance** $d(v, w)$ between v and w . It defines a metric distance on the set V of vertices. The maximal graph distance between two nodes of G is referred to as **diameter** of G .

A **subgraph** H of a given graph $G = (V, E, \mathbf{A})$ is a graph formed by a subset $W \subset V$ of vertices and the subset of edges from E connecting pairs of vertices in W . The adjacency matrix of H is the principal submatrix of $\mathbf{A}$ with respect to the subset W .

A subgraph H of a graph is called **connected** if any two vertices in H can be joined by a path such that all intermediate points also lie in H .

A subgraph H is called a **connected component** if it is connected and if there are no edges between H and its complement H^c formed by the complementary vertices $V \setminus W$ of V . A graph G is connected if and only if it consists of only one connected component.

Example 2.2 The weighted graph displayed below has only one connected component, and is therefore a connected graph.



The three nodes $\{1, 2, 3\}$ on the left (and the three edges connecting them) form a connected subgraph which is not a connected component of the entire graph. The graph distance between the node 1 on the left and the outer right node 5 is

$$d(1, 5) = \ell(1, 3) + \ell(3, 4) + \ell(4, 5) = \frac{1}{0.8} + \frac{1}{0.1} + \frac{1}{0.9} \approx 12.36111.$$

This corresponds already to the diameter of this graph.

Example 2.3 To have larger irregular graphs at disposition, particularly for the visualization of harmonic expansions and wavelets, we introduce two additional graphs.

- 1) The **Minnesota graph** visualized in Fig. 2 consists of a 2D road network of the US state of Minnesota. The data set can be retrieved from [50] and is commonly used as a benchmark in network analysis and for numerical experiments. The graph consists of $n = 2642$ vertices and 3304 edges, and is sparse, planar and connected. We consider the graph as undirected and unweighted, with adjacency matrix entries given by $A_{i,j} \in \{0, 1\}$. As a distance metric d on the Minnesota graph, we use the standard shortest-path distance, which approximates the travel distance along the road network.
- 2) As a similarity graph constructed upon a two-dimensional point cloud, we consider the **bunny graph** consisting of 1035 nodes and 9468 edges. The nodes are obtained by a projection of the 3D points of the Stanford bunny (taken from the Stanford 3D Scanning repository <http://graphics.stanford.edu/data/3Dscanrep/>) into to xy plane, and an additional thinning of the points with a separation radius of 0.0025. The nodes that are within a ball of radius $\varepsilon = 0.01$ are then linked by an edge. This generates a similarity graph in which all edge weights are set equal to one, giving a symmetric adjacency matrix A with entries $A_{i,j} \in \{0, 1\}$. The so-constructed bunny graph visualized in Fig. 2 is connected and undirected. This strategy for the generation of a similarity graph out of a point cloud is referred to as ε -ball strategy.

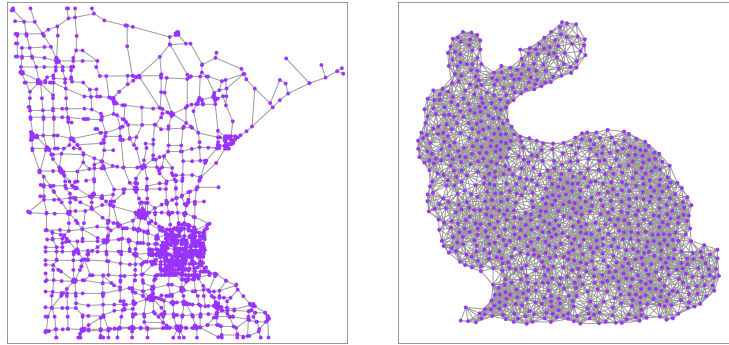
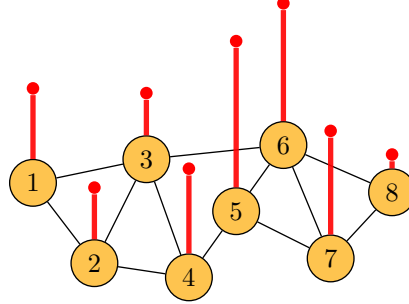


Fig. 3 Two examples of medium sized irregular graphs. Left: Minnesota graph. Right: bunny graph.

Fig. 4 Illustration of a graph signal x on a graph G with 8 vertices. The length of the red lines indicates the values of x on the single nodes.



3 Spectral Graph Theory

3.1 The graph Laplacian

A **graph signal** is a function $x : V \rightarrow \mathbb{R}$ that assigns to each node v_i an attribute $x(v_i) \in \mathbb{R}$. If the graph has n vertices, we can represent x also as a vector

$$x = \begin{bmatrix} x(v_1) \\ \vdots \\ x(v_n) \end{bmatrix} \in \mathbb{R}^n.$$

A simple example of such a graph signal is given in Figure 4. To emphasize that we are working with functions on the vertex set V , we will denote the space of all signals by $\mathcal{L}(V)$. We can endow the space $\mathcal{L}(V)$ with the standard inner product

$$\langle x, y \rangle := y^* x = \sum_{i=1}^n x(v_i) y(v_i), \quad (1)$$

where, in vector notation, y^* denotes the transpose of the vector y . The Hilbert space with the norm $\|x\|_2^2 := x^* x$, will be denoted as $\mathcal{L}^2(V)$. The system

$$\{\delta_{v_1}, \dots, \delta_{v_n}\}, \quad \delta_{v_{i'}}(v_i) := \delta_{i,i'} \quad \text{for } i, i' \in \{1, \dots, n\},$$

of canonical basis vectors forms an orthonormal basis of $\mathcal{L}^2(V)$. In addition, we consider the $\mathcal{L}^r(V)$ spaces equipped with the quasi-norms

$$\|x\|_r = \left(\sum_{i=1}^n |x(v_i)|^r \right)^{\frac{1}{r}}, \quad r > 0.$$

For $r \geq 1$ the latter quantity satisfies a triangle inequality and, thus, defines a norm.

Remark 3.1 While in this manuscript we consider only real-valued signals, in harmonic analysis it is sometimes more convenient to use complex-valued signals. In this case, the inner product for the Hilbert space $\mathcal{L}^2(V)$ is given as

$$y^*x = \sum_{i=1}^n x(v_i)\overline{y(v_i)},$$

and y^* denotes the conjugate transpose of the vector y .

In terms of spectral graph theory the following $n \times n$ -matrix, referred to as the graph Laplacian, is of main importance. The standard **graph Laplacian** $\mathbf{L} \in \mathbb{R}^{n \times n}$, sometimes also called the combinatorial graph Laplacian, is defined as the matrix

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \quad \text{with} \quad \mathbf{L}_{i,j} = \begin{cases} \mathbf{D}_{i,i} & \text{if } i = j \\ -\mathbf{A}_{i,j} & \text{if } i \neq j \end{cases}.$$

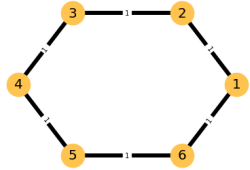
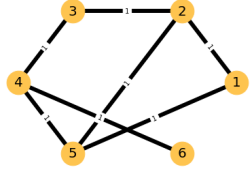
If the adjacency matrix $\mathbf{A}$ is symmetric, also the graph Laplacian $\mathbf{L}$ is symmetric. We will see shortly that $\mathbf{L}$ is also a positive semi-definite matrix, meaning that all eigenvalues are zero or positive.

We additionally introduce the **normalized graph Laplacian** $\mathbf{L}_N \in \mathbb{R}^{n \times n}$ as

$$\mathbf{L}_N = \mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}} = \mathbf{I}_n - \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}.$$

Also the matrix $\mathbf{L}_N$ is symmetric and positive semi-definite. The normalization ensures that the spectrum of $\mathbf{L}_N$ is in $[0, 2]$.

Example 3.2 For the simple graphs with 6 nodes in Example 2.1, we calculate the standard and the normalized graph Laplacian.

Graph	graph Laplacian $\mathbf{L}$	normalized Laplacian $\mathbf{L}_N$
	$\begin{bmatrix} 2 & -1 & 0 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & 0 & -1 & 2 \end{bmatrix}$	$\begin{bmatrix} 1 & -\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & 0 & 0 & -\frac{1}{2} & 1 \end{bmatrix}$
	$\begin{bmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & -\frac{1}{\sqrt{6}} & 0 & 0 & -\frac{1}{\sqrt{6}} & 0 \\ -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{\sqrt{6}} & 0 & -\frac{1}{3} & 0 \\ 0 & -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{\sqrt{6}} & 0 & 0 \\ 0 & 0 & -\frac{1}{\sqrt{6}} & 1 & -\frac{1}{3} & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{6}} & -\frac{1}{3} & 0 & -\frac{1}{3} & 1 & 0 \\ 0 & 0 & 0 & -\frac{1}{\sqrt{3}} & 0 & 1 \end{bmatrix}$

For the weighted graph of Example 2.1 we get the following graph Laplacians.

Weighted graph	graph Laplacian $\mathbf{L}$	normalized Laplacian $\mathbf{L}_N$
	$\begin{bmatrix} 1.6 & -0.8 & -0.8 & 0 & 0 \\ -0.8 & 1.6 & -0.8 & 0 & 0 \\ -0.8 & -0.8 & 1.7 & -0.1 & 0 \\ 0 & 0 & -0.1 & 1 & -0.9 \\ 0 & 0 & 0 & -0.9 & 0.9 \end{bmatrix}$	$\begin{bmatrix} 1 & -0.5 & -0.49 & 0 & 0 \\ -0.5 & 1 & -0.49 & 0 & 0 \\ -0.49 & -0.49 & 1 & -0.08 & 0 \\ 0 & 0 & -0.08 & 1 & -0.95 \\ 0 & 0 & 0 & -0.95 & 1 \end{bmatrix}$

3.2 Basic properties of the graph Laplacian

We derive some basic properties of the graph Laplacians $\mathbf{L}$ and $\mathbf{L}_N$. Let $x \in \mathcal{L}(V)$. Then we can express the matrix-vector product $\mathbf{L}x$ as

$$\mathbf{L}x(v_i) = \mathbf{D}_{i,i}x(v_i) - \sum_{j=1}^n \mathbf{A}_{i,j}x(v_j) = \sum_{j=1}^n \mathbf{A}_{i,j}(x(v_i) - x(v_j)),$$

i.e., $\mathbf{L}x$ can be interpreted as a discrete finite difference scheme that, in some cases, approximates a continuous Laplace operator. Similarly, for $\mathbf{L}_N$, we get

$$\mathbf{L}_N x(v_i) = x(v_i) - \frac{1}{\sqrt{\mathbf{D}_{i,i}}} \sum_{j=1}^n \frac{\mathbf{A}_{i,j}}{\sqrt{\mathbf{D}_{j,j}}} x(v_j) = \frac{1}{\sqrt{\mathbf{D}_{i,i}}} \sum_{j=1}^n \mathbf{A}_{i,j} \left(\frac{x(v_i)}{\sqrt{\mathbf{D}_{i,i}}} - \frac{x(v_j)}{\sqrt{\mathbf{D}_{j,j}}} \right).$$

Example 3.3 Let x be a twice continuously differentiable function on the interval $[0, 1]$. We consider the discrete points $v_i = (i-1)h, i \in \{0, \dots, n\}$, with $h = 1/n$ in the interval $[0, 1]$ as a path graph P_{n+1} with the non-zero edge weights $\mathbf{A}_{i,i+1} = h^{-1}$. Then, we get

$$\mathbf{L}x(v_i) = \frac{2x(v_i) - x(v_{i+1}) - x(v_{i-1}))}{h}, \quad i \in \{1, \dots, n-1\},$$

for the respective graph Laplacian. This corresponds to a second order difference quotient that discretizes the second derivative $-x''(t)$ of the differentiable function x .

Theorem 3.4 For the graph Laplacian $\mathbf{L} \in \mathbb{R}^{n \times n}$ of a graph $G = (V, E, \mathbf{A})$ we have:

- 1) $\mathbf{L}$ is symmetric.
- 2) All the rows (and columns) sum up to 0, i.e. $\mathbf{L}e = 0$ for $e = (1, \dots, 1)^*$. This implies that $\mathbf{L}$ has an eigenvalue $\lambda_1 = 0$ with eigenvector $e \in \mathbb{R}^n$.
- 3) For every $x \in \mathbb{R}^n$, we have

$$x^* \mathbf{L} x = \frac{1}{2} \sum_{(v_i, v_j) \in E} \mathbf{A}_{i,j} (x(v_i) - x(v_j))^2.$$

This implies that $\mathbf{L}$ is positive semi-definite and its eigenvalues are all non-negative such that they can be ordered as $0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$.

Proof. The two properties 1) and 2) follow directly from the definition of the graph Laplacian. Property 3) is shown by the following short calculation:

$$\begin{aligned} \sum_{i,j=1}^n \mathbf{A}_{i,j} (x(v_i) - x(v_j))^2 &= \sum_{i,j=1}^n \mathbf{A}_{i,j} x(v_i)^2 + \sum_{i,j=1}^n \mathbf{A}_{i,j} x(v_j)^2 - 2 \sum_{i,j=1}^n \mathbf{A}_{i,j} x(v_i) x(v_j) \\ &= \sum_{i=1}^n \mathbf{D}_{i,i} x(v_i)^2 + \sum_{j=1}^n \mathbf{D}_{j,j} x(v_j)^2 - 2 \sum_{i,j=1}^n \mathbf{A}_{i,j} x(v_i) x(v_j) \\ &= 2x^* \mathbf{D} x - 2x^* \mathbf{A} x = 2x^* (\mathbf{D} - \mathbf{A}) x = 2x^* \mathbf{L} x. \end{aligned}$$

□

Similarly, for the normalized Laplacian $\mathbf{L}_N$ we obtain the following result.

Theorem 3.5 For the normalized graph Laplacian $\mathbf{L}_N \in \mathbb{R}^{n \times n}$, we have

a) $\mathbf{L}_N$ is symmetric and

$$x^* \mathbf{L}_N x = \frac{1}{2} \sum_{(v_i, v_j) \in E} \mathbf{A}_{i,j} \left(\frac{x(v_i)}{\mathbf{D}_{i,i}^{1/2}} - \frac{x(v_j)}{\mathbf{D}_{j,j}^{1/2}} \right)^2.$$

In particular, $\mathbf{L}_N$ is positive semi-definite and all its eigenvalues are non-negative.

b) The non-negative eigenvalues of $\mathbf{L}_N$ can be ordered such that

$$0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n \leq 2.$$

c) $\sum_{k=1}^n \lambda_k = n$

d) $\mathbf{L}_N$ has an eigenvalue $\lambda_1 = 0$ with the eigenvector $\mathbf{D}^{1/2} \mathbf{e}$.

e) We have $\lambda_2 \leq \frac{n}{n-1}$ and $\lambda_n \geq \frac{n}{n-1}$.

Proof. a) Since $\mathbf{L}_N = \mathbf{I}_n - \mathbf{D}^{-1/2} \mathbf{A} \mathbf{D}^{-1/2}$ and $\mathbf{A}$ is symmetric, also $\mathbf{L}_N$ is symmetric. The formula for $x^* \mathbf{L}_N x$ follows similarly to the proof of Theorem 3.4. Since $x^* \mathbf{L}_N x \geq 0$, the matrix $\mathbf{L}_N$ is positive semi-definite, i.e., all eigenvalues of $\mathbf{L}_N$ are positive or zero. b) Because of a), we only have to show that the maximal eigenvalue satisfies $\lambda_n \leq 2$. For this, we consider the matrix $\mathbf{I}_n - \mathbf{D}^{-1} \mathbf{A} = \mathbf{D}^{-1/2} \mathbf{L}_N \mathbf{D}^{1/2}$, which is similar to the matrix $\mathbf{L}_N$ and contains therefore the same eigenvalues. The involved matrix $\mathbf{D}^{-1} \mathbf{A}$ is a (row) stochastic matrix, meaning that the sum of each row of $\mathbf{D}^{-1} \mathbf{A}$ is equal to one. This implies for the ∞ -norm of the matrix that

$$\|\mathbf{D}^{-1} \mathbf{A}\|_\infty = \max_{j \in \{1, \dots, n\}} \sum_{i=1}^n \frac{\mathbf{A}_{i,j}}{\mathbf{D}_{j,j}} = 1.$$

For an arbitrary eigenpair (λ, x) of $\mathbf{D}^{-1} \mathbf{A}$ we thus have

$$|\lambda| \|x\|_\infty = \|\lambda x\|_\infty = \|\mathbf{D}^{-1} \mathbf{A} x\|_\infty \leq \|\mathbf{D}^{-1} \mathbf{A}\|_\infty \|x\|_\infty = \|x\|_\infty.$$

Therefore, all eigenvalues of $\mathbf{D}^{-1} \mathbf{A}$ satisfy $|\lambda| \leq 1$. This on the other hand implies that the eigenvalues of $\mathbf{I}_n - \mathbf{D}^{-1} \mathbf{A}$ (and thus of $\mathbf{L}_N$) are between 0 and 2.

c) The trace of $\mathbf{L}_N$ is equal to n , therefore $\sum_{k=1}^n \lambda_k = n$.

d) We have $\mathbf{L}_N \mathbf{D}^{1/2} e = (\mathbf{D}^{1/2} - \mathbf{D}^{-1/2} \mathbf{A}) e = \mathbf{D}^{-1/2} \mathbf{L} e = 0$. The vector $\mathbf{D}^{1/2} e$ is hence an eigenvector of $\mathbf{L}_N$ with respect to the eigenvalue $\lambda_1 = 0$.

e) As $\lambda_1 = 0$, we also have that

$$\lambda_2 \leq \frac{n}{n-1}, \quad \lambda_n \geq \frac{n}{n-1}.$$

□

Example 3.6 We consider the weighted graph of Example 2.1 with the adjacency matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 0.8 & 0.8 & 0 & 0 \\ 0.8 & 0 & 0.8 & 0 & 0 \\ 0.8 & 0.8 & 0 & 0.1 & 0 \\ 0 & 0 & 0.1 & 0 & 0.9 \\ 0 & 0 & 0 & 0.9 & 0 \end{bmatrix}.$$

In this case, the eigenvalues of the graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ can be calculated as

$$0 < 0.0788 < 1.8465 < 2.4000 < 2.4747.$$

Further, the eigenvalues of the normalized Laplacian $\mathbf{L}_N$ are

$$0 < 0.0693 < 1.4773 < 1.5000 < 1.9534.$$

This means that both Laplacians have only one eigenvalue λ_1 equal to zero.

We consider in addition a slightly modified graph that has two connected components and the adjacency matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 0.8 & 0.8 & 0 & 0 \\ 0.8 & 0 & 0.8 & 0 & 0 \\ 0.8 & 0.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.9 \\ 0 & 0 & 0 & 0.9 & 0 \end{bmatrix}$$

The characteristic polynomial of the corresponding graph Laplacian is given as

$$\det(\lambda \mathbf{I} - \mathbf{L}) = \lambda^2 (\lambda - 2.4)^2 (\lambda - 1.8).$$

Thus, the graph Laplacian $\mathbf{L}$ of the modified graph has a repeated eigenvalue $\lambda_1 = \lambda_2 = 0$ with multiplicity 2 which is equal to the number of connected components.

Example 3.7 For the simple example graphs introduced at the beginning of this section the spectra of the graph Laplacian $\mathbf{L}$ are well-known. We have the following explicit descriptions that can be proven easily (we leave this as an exercise).

1) For the path graph P_n , the eigenvalues of $\mathbf{L}$ are

$$\lambda_k = 1 - \cos\left(\frac{\pi(k-1)}{n-1}\right), \quad k \in \{1, \dots, n\}.$$

2) For the circle graph C_n , the eigenvalues of $\mathbf{L}$ are

$$\lambda_k = 1 - \cos\left(\frac{2\pi(k-1)}{n}\right), \quad k \in \{1, \dots, n\}.$$

Note that since $\lambda_{k+1} = \lambda_{n-k+1}$ most of the eigenvalues are double eigenvalues.

3) For the complete graph K_n on n vertices, the eigenvalues of $\mathbf{L}$ are 0 and n (with multiplicity $n-1$). The eigenvalues of $\mathbf{L}_N$ are 0 and $n/(n-1)$, respectively.

Theorem 3.8 *The multiplicity of the zero eigenvalue of the Laplacians $\mathbf{L}$ and $\mathbf{L}_N$ equals the number of connected components of the graph G .*

Proof. We conduct the proof only for the standard Laplacian $\mathbf{L}$, for the normalized variant $\mathbf{L}_N$ the steps are similar. We first show that the number of zero eigenvalues is at least the number of connected components of G . Indeed, assume that G has k connected components, corresponding to the partition of V into disjoint sets $V_1, \dots, V_k$. Define k vectors $w_1, \dots, w_k$ s.t. $w_i(v_j) = 1/\sqrt{|V_i|}$ if $v_j \in V_i$, and 0 otherwise. For $i \in \{1, \dots, k\}$, we have $\|w_i\|_2 = 1$. Additionally, for $i \neq j$, we have $\langle w_i, w_j \rangle = 0$. Finally, $\mathbf{L}w_i = 0$ holds true. Hence there is a set of k orthonormal vectors that are all eigenvectors of $\mathbf{L}$, with respect to the eigenvalue 0.

To see that the algebraic multiplicity of the 0 eigenvalue is at most the number of connected components of G , we consider the identity

$$x^* \mathbf{L} x = \frac{1}{2} \sum_{(v_i, v_j) \in E} \mathbf{A}_{i,j} (x(v_i) - x(v_j))^2.$$

This expression can only be zero if x is constant on every connected component. To see that there is no way of finding a $(k+1)$ -st vector x that is a zero eigenvector, orthogonal to $w_1, \dots, w_k$, observe that any eigenvector x must be nonzero on some node. Hence, we can assume that x is nonzero on a node in V_i , and, thus, nonzero and constant on all nodes in V_i , in which case x can not be orthogonal to w_i . Therefore, there can be no $k+1$ st eigenvector with eigenvalue 0. This implies that the geometric, and thus also the algebraic multiplicity of the 0 eigenvalue must be k . $\square$

Remark 3.9 This theoretical result has a strong impact on classification methods based on the spectral information of the graph Laplacian. It indicates that the graph Laplacian contains relevant information about the geometric structure of a graph. In particular, the multiplicity of the zero eigenvalue automatically gives the number of connected components of the graph. Moreover, even if only one eigenvalue of $\mathbf{L}$ is zero and there are $k-1$ eigenvalues of $\mathbf{L}$ that are very close to zero, this still provides a numerical indication that the graph G consists of k components that are only weakly connected to each other. This offers a theoretical explanation of why spectral graph clustering based on the eigenvectors of the graph Laplacian works.

4 The graph Fourier transform (GFT)

4.1 The Fourier transform of a graph signal

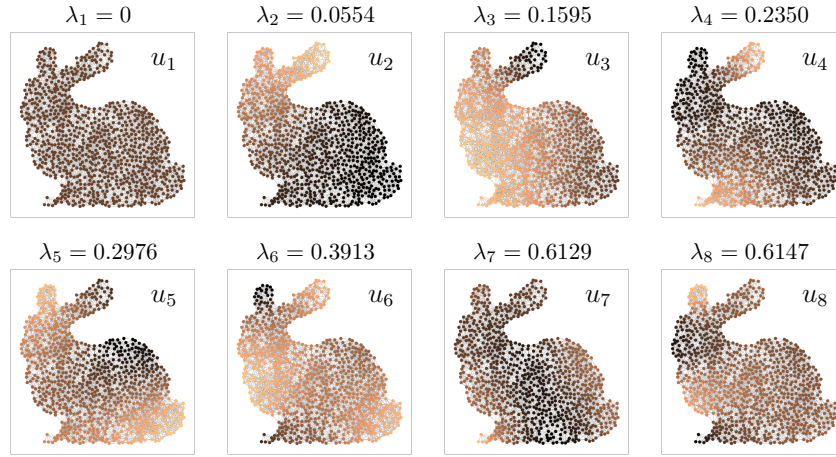


Fig. 5 The first 8 eigenvalues λ_k and eigenvectors u_k of the graph Laplacian $\mathbf{L}$ for the bunny graph introduced in Example 2.3. The eigenvectors u_k can be interpreted as a Fourier basis on the graph. The applied colormap uses light colors to describe negative values of the eigenvectors u_k , while dark colors indicate positive entries.

So far, we have seen that the graph Laplacians $\mathbf{L}$ and $\mathbf{L}_N = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ are symmetric and positive semi-definite matrices, and that their eigenvalues are real and may be ordered as

$$0 = \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n.$$

It is well-known that symmetric matrices can be diagonalized with help of a real orthonormal matrix $\mathbf{U}$, see [34, Corollary 2.5.14]. This yields the following spectral theorem for the graph Laplacians $\mathbf{L}$ and $\mathbf{L}_N$.

Theorem 4.1 (Spectral theorem for the graph Laplacian) *For the symmetric positive semi-definite graph Laplacians $\mathbf{L}$ and $\mathbf{L}_N$ there exists a set*

$$\{u_1, u_2, \dots, u_n\}$$

of real-valued orthonormal eigenvectors with respect to the eigenvalues

$$0 = \lambda_1 \leq \dots \leq \lambda_n.$$

This set is an orthonormal basis of the space $\mathcal{L}^2(V)$.

Using a matrix terminology, the just given spectral theorem is equivalent to the fact that the graph Laplacian $\mathbf{L}$ has a **spectral decomposition** of the form

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^*,$$

where

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix}$$

is a diagonal matrix with the increasingly ordered eigenvalues of $\mathbf{L}$ on the diagonal and

$$\mathbf{U} = [u_1 \ u_2 \ \cdots \ u_n]$$

is a real orthonormal matrix containing the orthonormal eigenvectors u_k of $\mathbf{L}$ as columns. An analogous decomposition holds true for the normalized Laplacian $\mathbf{L}_N$:

$$\mathbf{L}_N = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^*.$$

To keep the notation simple, we will use the same symbols to denote the eigenvalues and eigenvectors of $\mathbf{L}_N$ (although they differ from $\mathbf{L}$).

Definition 4.2 Given the orthonormal basis $\{u_1, \dots, u_n\}$ of eigenvectors of the graph Laplacian $\mathbf{L}$ (or similarly for $\mathbf{L}_N$), the **graph Fourier transform (GFT)** of a graph signal $x : V \rightarrow \mathbb{R}$ is defined as

$$\hat{x} = \begin{bmatrix} \hat{x}_1 \\ \vdots \\ \hat{x}_n \end{bmatrix} := \begin{bmatrix} u_1^* x \\ \vdots \\ u_n^* x \end{bmatrix} = \mathbf{U}^* x.$$

The coefficients

$$\hat{x}_k = u_k^* x = \sum_{i=1}^n x(v_i) u_k(v_i) \quad \left(\sum_{i=1}^n x(v_i) \overline{u_k(v_i)} \text{ if } u_k \in \mathbb{C}^n \right)$$

are also called the **Fourier coefficients** of the signal x . We will refer to the basis $\{u_1, \dots, u_n\}$ also as **Fourier basis** of $\mathcal{L}^2(V)$.

Because of the orthonormality of the Fourier basis $\{u_1, \dots, u_n\}$, every signal x on the graph can be recovered from the Fourier coefficients $\hat{x}$ by the sum

$$x(v_i) = \sum_{k=1}^n \hat{x}_k u_k(v_i).$$

This is referred to as **inverse graph Fourier transform** and can be also written as

$$x = \begin{bmatrix} x(v_1) \\ \vdots \\ x(v_n) \end{bmatrix} = \mathbf{U}\hat{x}.$$

Remark 4.3 The definition of the GFT is motivated by analogous harmonic structures in the Euclidean space. For an integrable function $x : \mathbb{R} \rightarrow \mathbb{R}$, the Fourier transform of x is defined as

$$\hat{x}(\omega) = \int_{\mathbb{R}} x(t) e^{-2\pi i t \omega} dt.$$

For this Fourier transform on the real line, the complex-valued functions $u_{\omega}(t) = e^{2\pi i t \omega}$ are the eigenfunctions of the second order differential operator $-\Delta = -\frac{d^2}{dt^2}$ (also known as Laplace operator), i.e., we have

$$-\Delta u_{\omega}(t) = \omega^2 u_{\omega}(t).$$

The signal $x(t)$ can then be recovered from the Fourier coefficients $\hat{x}(\omega)$, $\omega \in \mathbb{R}$, using the inverse Fourier transform

$$x(t) = \int_{\mathbb{R}} \hat{x}(\omega) e^{2\pi i t \omega} d\omega.$$

Instead of the complex-valued functions $u_{\omega}(t) = e^{2\pi i t \omega}$, it is also possible to use the real-valued eigenfunctions $\sin(2\pi\omega t)$ and $\cos(2\pi\omega t)$ of the Laplacian for the definition of the Fourier transform.

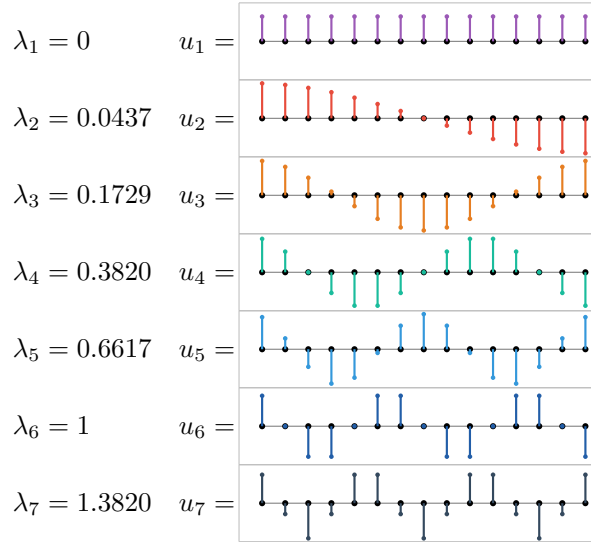


Fig. 6 The first 7 eigenvalues λ_k and eigenvectors u_k of the graph Laplacian $\mathbf{L}$ for the path graph P_{15} .

The GFT can be interpreted in a similar fashion as the classical Fourier transform given in Remark 4.3. In this perspective, the eigenvalues of the graph Laplacians $\mathbf{L}$ and $\mathbf{L}_N$ may be viewed as natural frequencies of the graph. As the eigenvalues λ_k increase, the respective eigenvectors u_k display an increasingly oscillating behavior. Eigenvectors associated with large eigenvalues typically oscillate rapidly across the graph, whereas those associated with small eigenvalues vary only slowly and represent smooth graph signals. In particular, for a connected graph, the eigenvector u_1 associated to the eigenvalue 0 of the graph Laplacian $\mathbf{L}$ is constant.

For this, the GFT $\hat{x} = [\hat{x}_1 \cdots \hat{x}_n]^*$ can be interpreted as the decomposition of a graph signal x into its single frequency components

$$x(v_i) = \sum_{k=1}^n \hat{x}_k u_k(v_i).$$

Low-frequency components correspond to slowly varying, smooth structures of the graph signal, while the high-frequency components capture rapidly changing behavior and are often associated with noise in the signal x .

Example 4.4 (i) For the path graph P_n consisting of n concatenated vertices, the eigenvalues and eigenfunctions of the matrix $\mathbf{L}$ can be computed explicitly as

$$\lambda_k = 2 - 2 \cos \left(\frac{(k-1)\pi}{n} \right), \quad k \in \{1, \dots, n\}$$

and

$$u_1 = \frac{1}{\sqrt{n}}, \quad u_k(v_i) = \sqrt{\frac{2}{n}} \cos \left(\frac{(k-1)\pi(i-0.5)}{n} \right), \quad k \in \{2, \dots, n\}.$$

This means that the graph Fourier transform on the path graph P_n corresponds to the **Discrete Cosine Transform** (DCT II) of a discrete time signal x with n entries. For $n = 15$, the first 7 eigenpairs (λ_k, u_k) are plotted in Fig. 4.1.

(ii) As an example of a Fourier basis on larger and more irregular graph we consider the eigenvectors of the graph Laplacian $\mathbf{L}$ on the bunny graph which has been introduced in Example 2.3. The first 8 eigenvectors of the Fourier basis together with the respective eigenvalues are plotted in Fig. 5.

4.2 Numerical calculation of the GFT

If the entire GFT has to be calculated, the QR algorithm [63, Chap. V] can be applied in order to get the spectral decomposition of the symmetric graph Laplacians $\mathbf{L}$ or $\mathbf{L}_N$. The respective computational costs for the GFT amount to:

- 1) $O(n^3)$ for the calculation of the spectral decomposition of $\mathbf{L}$ or $\mathbf{L}_N$.
- 2) $O(n^2)$ for the GFTs $\hat{x} = \mathbf{U}^*x$ and inverse GFTs $x = \mathbf{U}\hat{x}$ (matrix-vector products).

For large size graphs ($n > 100000$), the calculation of the full spectral decomposition of the graph Laplacian is not sustainable anymore. Some strategies to circumvent the relatively large costs of the full decomposition are the following:

- 1) If the graph Laplacian $\mathbf{L}$ is sparse and only the eigenpairs with respect to the smallest eigenvalues are required, iterative methods can be used for the calculation of these eigenpairs. These methods are referred to as **Krylov subspace methods** and rely on a projection into a small-dimensional Krylov space spanned by polynomials of the matrix $\mathbf{L}$ applied to a starting vector.
- 2) For particular operations on the Fourier coefficients $\hat{x}$ of a signal x , it is possible to avoid the calculation of the spectral decomposition of $\mathbf{L}$. For instance, the graph convolution of two signals discussed in the following section can be computed without the GFT, if one of the involved signals has a particular polynomial structure.
- 3) For graphs with particular structures, as the path graph in Example 4.4, we know the spectral decomposition of the graph Laplacian explicitly. For these explicit transforms fast algorithms are available, as for instance the **fast Fourier transform** or the **discrete cosine transform**.

4.3 Graph convolution and filtering

On the real line, the convolution of two integrable functions $x(t)$ and $y(t)$ is defined as

$$(x * y)(t) = \int_{\mathbb{R}} x(s)y(t-s)ds.$$

This operation is a fundamental tool in signal processing as it allows to filter and process signals, for instance, by removing low-frequency or high-frequency components. Regarding this operation, the convolution theorem states that the convolution of two signals can be calculated easily by multiplying the respective Fourier transforms

$$\widehat{(x * y)}(\omega) = \hat{x}(\omega)\hat{y}(\omega).$$

On general graphs, we don't have the possibility to define the translation of a signal. In order to generalize the convolution operation for graph signals, we therefore have to stick to the statement of the convolution theorem and define the **graph convolution** of two graph signals $x, y \in \mathcal{L}(V)$ as

$$x * y := \mathbf{U}\mathbf{M}_{\hat{x}}\hat{y} = \mathbf{U}\mathbf{M}_{\hat{x}}\mathbf{U}^*y, \quad \text{where} \quad \mathbf{M}_{\hat{x}} = \begin{bmatrix} \hat{x}_1 & & \\ & \ddots & \\ & & \hat{x}_n \end{bmatrix}.$$

Further, we can define a convolution matrix $\mathbf{C}_x \in \mathbb{R}^{n \times n}$ as

$$\mathbf{C}_x := \mathbf{U}\mathbf{M}_{\hat{x}}\mathbf{U}^*,$$

such that the convolution can be written more compactly as

$$x * y = \mathbf{C}_x y.$$

In both formulations, it is visible that the definition of the graph convolution is based on the convolution theorem, and that the convolution of two signals corresponds to the pointwise multiplication of their Fourier coefficients in the Fourier domain. Note that the so-defined graph convolution depends on the choice of the Fourier basis $\{u_1, \dots, u_n\}$. To avoid disambiguities, it is therefore useful to fix the used Fourier basis a priori.

We can use the graph convolution to define **filter functions** on graphs. For a filter $f \in \mathcal{L}(V)$ and a signal $x \in \mathcal{L}(V)$ the filtered signal $f * x$ is given as

$$(f * x)(v_i) = \mathbf{C}_f x(v_i) = \sum_{k=1}^n \hat{f}_k \hat{x}_k u_k(v_i).$$

Here, a filter function f with $\hat{f}_k = 0$ for $k \geq N < n$ is called **low-pass filter**, whereas a filter function f with $\hat{f}_k = 0$ for $k < N < n$ is referred to as **high-pass filter**.

4.4 Polynomial approximation of the graph convolution

The computation of the filtered signal $f * x$ using the GFT is too cost-intensive for large graphs. For particular filter functions, it is possible to obtain the filtered signal without the eigenvectors of $\mathbf{L}$. For this, we define polynomial filters in terms of the GFT as

$$p_m(\lambda_k) = \sum_{j=0}^m \alpha_j \lambda_k^j \approx \hat{f}_k = \hat{f}(\lambda_k),$$

such that $p_m(\lambda)$ is a polynomial of degree $m \leq n$ that approximates the filter $\hat{f}$ on the eigenvalues of $\mathbf{L}$. Then, the convolution $f * x$ can be approximately calculated as

$$\begin{aligned} f * x &= \mathbf{U} \mathbf{M}_{\hat{f}} \mathbf{U}^* x \approx \mathbf{U} \begin{bmatrix} p_m(\lambda_1) & & \\ & \ddots & \\ & & p_m(\lambda_n) \end{bmatrix} \mathbf{U}^* x = p_m \left(\mathbf{U} \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} \mathbf{U}^* \right) x \\ &= p_m(\mathbf{L})x = \sum_{j=0}^m \alpha_j \mathbf{L}^j x, \end{aligned}$$

i.e., $f * x$ is approximated by a linear combination of the vectors $\{x, \mathbf{L}x, \dots, \mathbf{L}^m x\}$ and can be obtained in terms of m matrix-vector products and a vector summation. If the matrix $\mathbf{L}$ is sparse, the matrix-vector products can be calculated very efficiently. Numerical methods involving the spaces generated by the system $\{x, \mathbf{L}x, \dots, \mathbf{L}^m x\}$ are referred to as Krylov subspace methods. These methods form a general class of algorithms that allow to solve large and sparse systems of equations very efficiently.

Usually, a filter function f needs to be applied to several different signals x . It is therefore advantageous to approximate the filter f such that the polynomial $p_m(\lambda)$ is independent of the particular signal x . Further, rather than expressing $p_m(\lambda)$ in the monomial basis, it is preferable to represent it in a more numerically stable basis. A well-suited choice in this context is provided by the Chebyshev polynomials.

The Chebyshev polynomials of the first kind $\{T_k(t)\}$ are defined on the interval $[-1, 1]$ and satisfy the recurrence relation

$$T_0(t) = 1, \quad T_1(t) = t, \quad T_k(t) = 2tT_{k-1}(t) - T_{k-2}(t), \quad k \in \mathbb{N}, k \geq 2.$$

To apply these basis polynomials to the graph Laplacian, the Chebyshev polynomials must be shifted and rescaled to lie within the spectrum $[0, \lambda_n]$ of $\mathbf{L}$. This is achieved by considering the polynomials

$$\tilde{T}_k(t) = T_k\left(\frac{2}{\lambda_n}t - 1\right), \quad k \in \mathbb{N}_0,$$

instead, so that any polynomial $p_m(\lambda)$ of degree m can be decomposed as

$$p_m(\lambda) = \sum_{k=0}^m \beta_k \tilde{T}_k(\lambda),$$

with some coefficients $\beta_k \in \mathbb{R}$, $k \in \{0, \dots, m\}$.

Once the desired filtering behavior of $\hat{f}$ (e.g., low-pass, high-pass, band-pass) has been specified, a simple way to construct the polynomial approximation $p_m(\lambda)$ and to determine the corresponding Chebyshev coefficients β_k is by interpolation. More precisely, if the polynomial $p_m(\lambda)$ is intended to approximate a continuous spectral filter function $\hat{f}$ over the spectrum $[0, \lambda_n]$ of the graph Laplacian, the polynomial $p_m(\lambda)$ can be constructed so as to satisfy the interpolation conditions

$$p_m(t_j) = \hat{f}(t_j), \quad j = \{0, \dots, m\},$$

where

$$t_j = \frac{\lambda_n}{2} \left(\cos\left(\frac{2j+1}{2m+2}\pi\right) + 1 \right), \quad j = \{0, \dots, m\}, \quad (2)$$

are the shifted and dilated roots of the Chebyshev polynomial $T_{m+1}(\lambda)$. With this choice of the interpolation nodes, the coefficients β_k can be calculated efficiently and in a numerically stable way from the samples $\hat{f}(t_j)$ using a discrete cosine transform.

Once the Chebyshev coefficients β_k are computed, the filtered signal $f * x$ can be approximated by

$$f * x \approx p_m(\mathbf{L})x = \sum_{k=0}^m \beta_k \tilde{T}_k(\mathbf{L})x.$$

Note that the coefficients β_k are independent of the signal x . Thus, once the vectors $\tilde{T}_k(\mathbf{L})x$ have been calculated, different filter functions can be applied to the same signal x simply by changing the coefficients β_k , while reusing the already computed vectors.

Remark 4.5 Spectral graph convolutional networks (spectral GCNs) are neural network architectures on graphs, in which graph convolutional filters defined on the spectrum of the graph Laplacian are combined with nonlinear activation functions to form multi-layer models. Since the computation of the full eigendecomposition is too cost-expensive for large graphs, spectral filtering operations are approximated by polynomials of the graph Laplacian [16, 47]. In this setting, the convolutional filters are parametrized by the coefficients of this polynomial in the chosen basis, e.g., monomial or Chebyshev polynomials. These coefficients are then learned from the data during training.

4.5 Denoising of graph signals

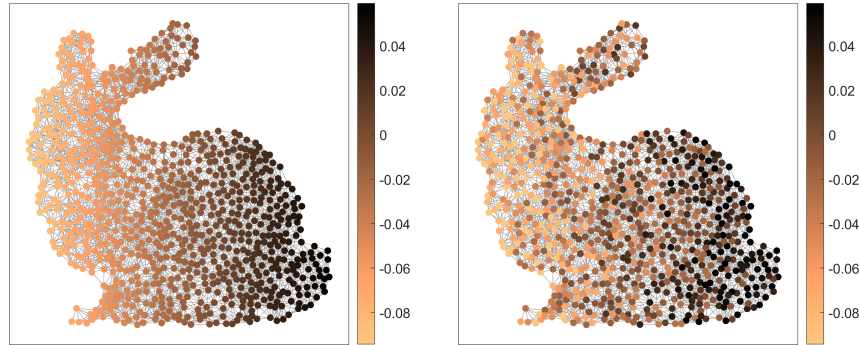


Fig. 7 Left: The original signal x encoding the x -coordinates of the bunny vertices. Right: a noisy variant x_{noisy} of x on the bunny graph.

As a first application of the GFT, we consider the problem of denoising graph signals using a suitable convolutional filter. We suppose that the noise component of the signal is primarily reflected by its high-frequency Fourier coefficients. As an illustrative example, we consider the bunny graph of Example 2.3. As a smooth signal x on the graph, we take the x -coordinate of the planar node positions of the bunny. A noisy signal x_{noisy} is then obtained by adding uniformly distributed white noise to the original signal x . Both x and x_{noisy} are visualized in Fig. 7.

To identify the noise component of x_{noisy} , we consider the GFT of x and x_{noisy} . The Fourier coefficients of both signals are illustrated in Fig. 8 (left). It is visible that starting at around $k = 20$, the frequency components $(\hat{x}_{\text{noisy}})_k$ of the noisy signal get significantly larger than the respective components $\hat{x}_k$ of the smooth signal.

As a suitable filter on the graph, we construct a filter function f that removes the high-frequency components of the noisy signal. More precisely, the filter is chosen such that all Fourier coefficients $(\hat{x}_{\text{noisy}})_k$ with $k \geq 21$ are eliminated, while the lower-frequency components are retained. A denoised signal x_{denoised} is then obtained by computing the graph convolution

$$x_{\text{denoised}} = f * x_{\text{noisy}},$$

where f acts as a low-pass filter that suppresses all frequency components with indices $k \geq 20$. The corresponding Fourier coefficients of f are displayed in Fig. 8 (right).

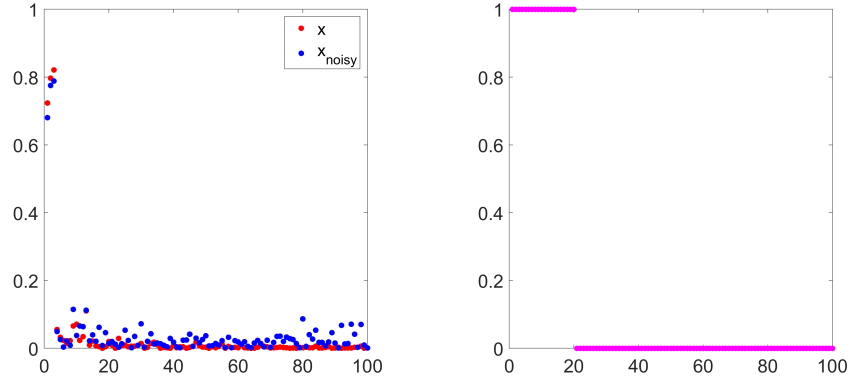


Fig. 8 Left: the absolute value of the first 100 Fourier coefficients of the original signal x and the noisy signal x_{noisy} . The added noise increases principally the high-frequency components of the signal. Right: the Fourier coefficients of the low-pass filter f .

The denoised signal x_{denoised} itself is illustrated in Fig. 9 (left). We can observe that the application of the low-pass filter f had a clear denoising effect for the noisy signal x_{noisy} . The absolute error $|x - x_{\text{denoised}}|$ depicted in Fig. 9 (right) makes however also clear that the original signal x can not be completely restored by the described filtering procedure. This can however also be expected since the addition of noise distorts also the low-frequency part of the signal x , as visible also in Fig. 8 (left).

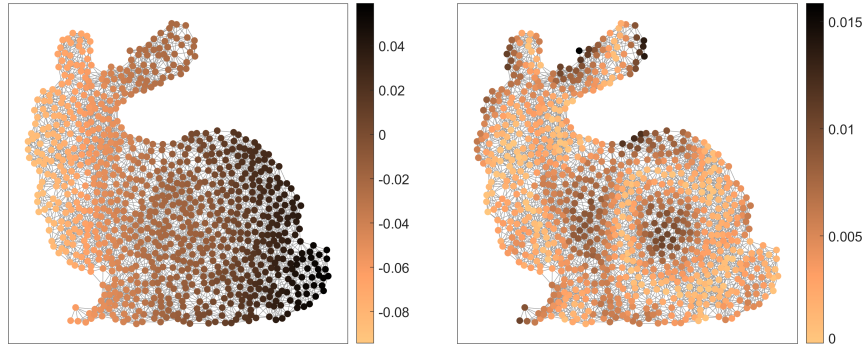


Fig. 9 Left: the denoised signal x_{denoised} . Right: the absolute error $|x - x_{\text{denoised}}|$ between the original and the denoised signal.

5 Spectral Graph Wavelet Transform

5.1 Spectral graph wavelets and scaling functions

The graph Laplacian $\mathbf{L}$ (or similarly $\mathbf{L}_N$) and the respective GFT allow to introduce a **spectral graph wavelet transform** (SGWT) based on the spectrum of the Laplacian. This transform is determined by the selection of two window functions, a function

$$h : [0, \infty) \rightarrow \mathbb{R}, \quad h(0) > 0, \quad \lim_{x \rightarrow \infty} h(x) = 0$$

to define a spectral scaling operator, and a second function

$$g : [0, \infty) \rightarrow \mathbb{R}, \quad g(0) = 0, \quad \lim_{x \rightarrow \infty} g(x) = 0$$

for the spectral wavelets. Chosen properly, the functions h and g will act as low-pass and band-pass filters on the spectrum of a given signal $x \in \mathcal{L}(V)$, respectively.

Definition 5.1 Based on the window functions g and h , and a scaling parameter $t > 0$, we define the **spectral scaling operator** $\mathbf{T}_h$ and the **spectral wavelet operator** $\mathbf{T}_{g,t}$ for a graph signal x in the frequency domain as

$$(\widehat{\mathbf{T}_h x})_k := h(\lambda_k) \hat{x}_k, \quad \text{and} \quad (\widehat{\mathbf{T}_{g,t} x})_k := g(t\lambda_k) \hat{x}_k. \quad (3)$$

The function h for the scaling operator $\mathbf{T}_h x$ is selected in such a way that the low-frequency content of the signal x is approximated, while the higher frequency parts of x are represented by the wavelet components $\mathbf{T}_{g,t} x$ for different choices of the scaling parameter t . As the function g is defined on the half-axis $[0, \infty)$, the application of the scaling $t > 0$ allows to localize the filter g in different parts of the spectrum of the Laplacian $\mathbf{L}$. Under suitable conditions on g , h , and the scaling parameter t , the signal x can be reconstructed from its wavelet components. Applying the inverse GFT to the Fourier coefficients in (3), we obtain respective representations of the operators in the vertex domain:

$$(\mathbf{T}_h x)(v) = \sum_{k=1}^n h(\lambda_k) \hat{x}_k u_k(v), \quad (\mathbf{T}_{g,t} x)(v) = \sum_{k=1}^n g(t\lambda_k) \hat{x}_k u_k(v). \quad (4)$$

Both operations can be considered as convolutional filters applied to the signal x . In particular, in matrix-vector notation, these operators can be written as

$$\mathbf{T}_h x = \mathbf{U} h(\mathbf{\Lambda}) \mathbf{U}^* x = h(\mathbf{L}) x, \quad \mathbf{T}_{g,t} x = \mathbf{U} g(t\mathbf{\Lambda}) \mathbf{U}^* x = g(t\mathbf{L}) x.$$

Definition 5.2 The **spectral graph scaling functions** ϕ_i as well as the **spectral graph wavelets** $\psi_{t,i}$ are defined through the application of the operators $\mathbf{T}_h$ and $\mathbf{T}_{g,t}$ to the canonical basis vectors δ_{v_i} , localized on the single vertices v_i of the graph. We define

$$\phi_i(v) := \mathbf{T}_h \delta_{v_i}(v), \quad \psi_{t,i}(v) := \mathbf{T}_{g,t} \delta_{v_i}(v). \quad (5)$$

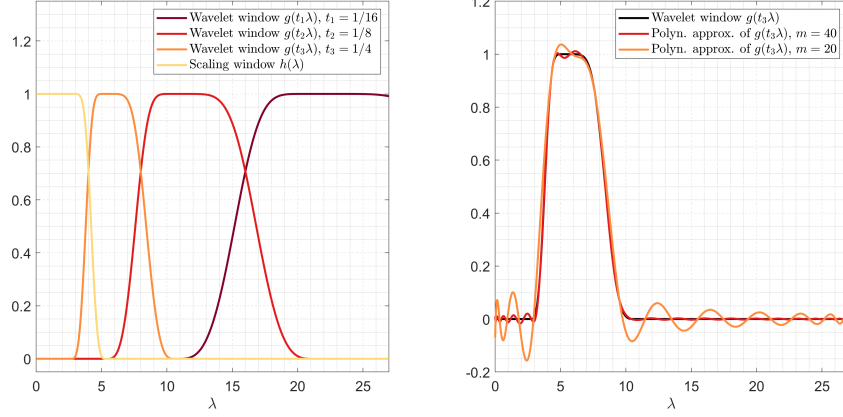


Fig. 10 Left: dilations $g(t_j \lambda)$ of the Meyer wavelet and the scaling window h given in Example 5.6 on the spectrum of $\mathbf{L}$ for the bunny graph. Right: approximation of the wavelet window g with polynomials of degree m . For the approximation, the interpolation procedure described in Section 4.4 is used.

The corresponding expansions in terms of the Fourier basis read as

$$\phi_i(\mathbf{v}) = \sum_{k=1}^n h(\lambda_k) u_k(\mathbf{v}_i) u_k(\mathbf{v}), \quad \psi_{t,i}(\mathbf{v}) = \sum_{k=1}^n g(t\lambda_k) u_k(\mathbf{v}_i) u_k(\mathbf{v}). \quad (6)$$

Examples of ϕ_i and $\psi_{t,i}$ on the bunny graph are visualized in Fig. 11. The wavelet coefficients of a signal x are given as the inner products $\langle x, \psi_{t,i} \rangle$. By the orthonormality of the Fourier basis $\{u_1, \dots, u_n\}$ and the symmetry of the wavelet operators $\mathbf{T}_{g,t}$, the wavelet coefficients can also be written as

$$\langle x, \psi_{t,i} \rangle = \langle \mathbf{T}_{g,t} x, \delta_{\mathbf{v}_i} \rangle = (\mathbf{T}_{g,t} x)(\mathbf{v}_i) = \sum_{k=1}^n g(t\lambda_k) \hat{x}_k u_k(\mathbf{v}_i). \quad (7)$$

Similarly, the coefficients for the spectral scaling functions can be calculated as

$$\langle x, \phi_i \rangle = \langle \mathbf{T}_h x, \delta_{\mathbf{v}_i} \rangle = (\mathbf{T}_h x)(\mathbf{v}_i) = \sum_{k=1}^n h(\lambda_k) \hat{x}_k u_k(\mathbf{v}_i). \quad (8)$$

Remark 5.3 By the assumption $g(0) = 0$ on the wavelet window function g , the spectral graph wavelets $\psi_{t,i}$ satisfy

$$\langle u_1, \psi_{t,i} \rangle = (\mathbf{T}_{g,t} u_1)(\mathbf{v}_i) = (\mathbf{U} g(t\Lambda) \mathbf{U}^* u_1)(\mathbf{v}_i) = 0,$$

i.e., the wavelets $\psi_{t,i}$ are orthogonal to u_1 and to all other eigenvectors of $\mathbf{L}$ with respect to the eigenvalue $\lambda_1 = 0$. This is the reason why for the reconstruction of the signal x also a scaling part $\mathbf{T}_h x$ encoding the low-frequency content of the signal x is required. The

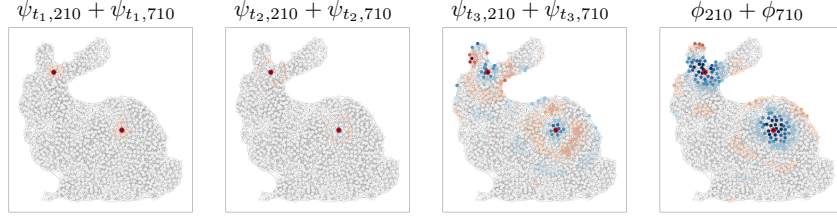


Fig. 11 Illustration of spectral graph wavelets $\psi_{t,i}(v)$ and the scaling functions $\phi_i(v)$ on the bunny graph. The spectral wavelets are based on the scaling function h and the window function g given in Fig. 10. The red rings indicate the center nodes v_{210} and v_{710} of the wavelets and scaling functions.

reconstruction of x from its spectral wavelet coefficients will be ensured by $h(0) > 0$ and a suitable selection of g and the scaling parameters t . In contrast to classical wavelet theory, the function h does not generate the wavelets $\psi_{t,i}$ in terms of a two-scale relation. In particular, the windows g and h can be chosen almost independently from each other. The only dependencies are given by the recovery guarantees discussed below.

5.2 Frames and reconstruction from spectral wavelet coefficients

Our goal is now to formulate and prove a reconstruction formula for graph signals x from a sufficiently large collection of spectral wavelet coefficients. In the setting of spectral graph wavelets, it is in general not possible to construct a wavelet basis for $\mathcal{L}^2(V)$. Instead, we will work with a **redundant wavelet frame** and derive an explicit reconstruction formula associated with this frame.

To this end, we consider the scaling functions ϕ_i , defined via a window function h , together with the spectral graph wavelets $\psi_{t_j,i}$, constructed from a window function g at a finite set of scales $0 < t_1 < \dots < t_J$. As a redundant collection of functions for the decomposition of a signal x , we consider the system

$$\mathcal{F} = \{\mathbf{T}_h \delta_{v_i} : i \in \{1, \dots, n\}\} \cup \{\mathbf{T}_{g,t_j} \delta_{v_i} : i \in \{1, \dots, n\}, j \in \{1, \dots, J\}\}. \quad (9)$$

Based on this system, we are able to analyse a signal x in terms of the coefficients

$$\{\langle x, \phi_i \rangle : i \in \{1, \dots, n\}\} \cup \{\langle x, \psi_{t_j,i} \rangle : i \in \{1, \dots, n\}, j \in \{1, \dots, J\}\}.$$

In order to guarantee recovery of the signal x from these coefficients, it is essential that

$$G(\lambda) = h^2(\lambda) + \sum_{j=1}^J g(t_j \lambda)^2 > 0$$

is strictly positive on the entire spectrum of the graph Laplacian. In this case, we are able to prove that the system (9) is a frame, and to derive a recovery formula.

Theorem 5.4 *Given the window functions h and g , and a finite set of scales $0 < t_1 < \dots < t_J$, we assume that the function $G(\lambda) > 0$ is strictly positive on the interval $[0, \lambda_n]$. Then, the set*

$$\mathcal{F} = \{\phi_i : i \in \{1, \dots, n\}\} \cup \{\psi_{t_j, i} : i \in \{1, \dots, n\}, j \in \{1, \dots, J\}\}$$

forms a frame in $\mathcal{L}^2(V)$ by satisfying the two inequalities

$$A\|x\|_2^2 \leq \sum_{i=1}^n \left(|\langle x, \phi_i \rangle|^2 + \sum_{j=1}^J |\langle x, \psi_{t_j, i} \rangle|^2 \right) \leq B\|x\|_2^2$$

with the frame bounds

$$A = \min_{\lambda \in [0, \lambda_n]} G(\lambda), \quad B = \max_{\lambda \in [0, \lambda_n]} G(\lambda).$$

Further, if we define the canonical dual scaling functions and wavelets as

$$\tilde{\phi}_i = \mathbf{T}_{1/G} \mathbf{T}_h \delta_{v_i} = \mathbf{T}_{h/G} \delta_{v_i}, \quad \tilde{\psi}_{t_j, i}(v) = \mathbf{T}_{1/G} \mathbf{T}_{g, t} \delta_{v_i},$$

the signal x can be recovered from the spectral wavelet coefficients as

$$x(v) = \sum_{i=1}^n \langle x, \phi_i \rangle \tilde{\phi}_i(v) + \sum_{i=1}^n \sum_{j=1}^J \langle x, \psi_{t_j, i} \rangle \tilde{\psi}_{t_j, i}(v).$$

Proof. For a given graph signal x , we consider the coefficients $\langle x, \phi_i \rangle$ and $\langle x, \psi_{t_j, i} \rangle$. Using the identity (7), the Pythagorean theorem yields

$$\begin{aligned} \sum_{i=1}^n |\langle x, \psi_{t_j, i} \rangle|^2 &= \sum_{i=1}^n \left| \sum_{k=1}^n g(t_j \lambda_k) \hat{x}_k u_k(v_i) \right|^2 \\ &= \left\| \sum_{k=1}^n g(t_j \lambda_k) \hat{x}_k u_k \right\|_2^2 = \sum_{k=1}^n |g(t \lambda_k)|^2 |\hat{x}_k|^2. \end{aligned} \tag{10}$$

Analogously, by (8) we obtain for the scaling functions the identity

$$\sum_{i=1}^n |\langle x, \phi_i \rangle|^2 = \sum_{k=1}^n |h(\lambda_k)|^2 |\hat{x}_k|^2. \tag{11}$$

Combining (10) and (11), we get

$$\begin{aligned} \sum_{i=1}^n \left(|\langle x, \phi_i \rangle|^2 + \sum_{j=1}^J |\langle x, \psi_{t_j, i} \rangle|^2 \right) &= \sum_{k=1}^n \left(|h(\lambda_k)|^2 + \sum_{j=1}^J |g(t_j \lambda_k)|^2 \right) |\hat{x}_k|^2 \\ &= \sum_{k=1}^n G(\lambda_k) |\hat{x}_k|^2. \end{aligned} \quad (12)$$

Then, by definition of the constants A and B , we get the inequalities

$$A \sum_{k=1}^n |\hat{x}_k|^2 \leq \sum_{i=1}^n \left(|\langle x, \phi_i \rangle|^2 + \sum_{j=1}^J |\langle x, \psi_{t_j, i} \rangle|^2 \right) \leq B \sum_{k=1}^n |\hat{x}_k|^2. \quad (13)$$

The Pythagorean theorem $\|x\|_2^2 = \sum_{k=1}^n |\hat{x}_k|^2$ implies now the statement for the frame. Also the inversion formula can be shown directly. With the identities (6) and (7), and the orthonormality of the basis functions u_k , we get

$$\begin{aligned} \sum_{i=1}^n \langle x, \psi_{t_j, i} \rangle \tilde{\psi}_i(v) &= \sum_{i=1}^n \sum_{k=1}^n g(t_j \lambda_k) \hat{x}_k u_k(v_i) \sum_{k'=1}^n \frac{g(t_j \lambda_{k'})}{G(\lambda_k)} u_{k'}(v_i) u_{k'}(v) \\ &= \sum_{k=1}^n \hat{x}_k \frac{g(t \lambda_k)^2}{G(\lambda_k)} u_k(v). \end{aligned} \quad (14)$$

In the same way, (6) and (8) provide for the scaling function the identity

$$\sum_{i=1}^n \langle x, \phi_i \rangle \tilde{\phi}_i(v) = \sum_{k=1}^n \hat{x}_k \frac{h(\lambda_k)^2}{G(\lambda_k)} u_k(v). \quad (15)$$

Combining (14) and (15) and the inverse GFT, we finally get

$$\begin{aligned} \sum_{i=1}^n \langle x, \phi_i \rangle \tilde{\phi}_i(v) + \sum_{i=1}^n \sum_{j=1}^J \langle x, \psi_{t_j, i} \rangle \tilde{\psi}_i(v) &= \sum_{k=1}^n \hat{x}_k \frac{h(\lambda_k)^2 + \sum_{j=1}^J g(t \lambda_k)^2}{G(\lambda_k)} u_k(v) \\ &= \sum_{k=1}^n \hat{x}_k u_k(v) = x(v). \end{aligned}$$

□

Remark 5.5 In the literature, the operator $\mathbf{T}_G$ defined as

$$\mathbf{T}_G x(v) = \sum_{i=1}^n \langle x, \phi_i \rangle \phi_i(v) + \sum_{i=1}^n \sum_{j=1}^J \langle x, \psi_{t_j, i} \rangle \psi_i(v)$$

is referred to as **frame operator** and corresponds to the concatenation of an analysis operator, giving the wavelet coefficients of a signal x , with a synthesis operator generating the respective expansion series in terms of the frame elements. Applying the inverse frame operator $\mathbf{T}_{1/G}$ to both sides of this identity recovers exactly the reconstruction

formula stated in Theorem 5.4. Furthermore, the elements $\tilde{\psi}_{t_j,i}$ and $\tilde{\phi}_i$ of the canonical dual frame are obtained by applying the inverse frame operator $\mathbf{T}_{1/G}$ to the frame elements $\psi_{t_j,i}$ and ϕ_i . Detailed introductions to frame theory can be found in [7, 33].

Example 5.6 As an explicit example for window functions h and g on the spectrum of the Laplacian, we consider a construction based on the Meyer scaling function and the Meyer wavelets. For this, we require a transition function

$$\nu : \mathbb{R} \rightarrow [0, 1]$$

satisfying $\nu(x) = 0$ if $x \leq 0$ and $\nu(x) = 1$ if $x \geq 1$. A standard choice for $\nu(x)$ is

$$\nu(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ x^4(35 - 84x + 70x^2 - 20x^3) & \text{if } 0 < x < 1, \\ 1 & \text{if } x \geq 1. \end{cases}$$

The so-constructed function $\nu(x)$ is piecewise smooth. Then, the Meyer scaling function h is defined by the smooth low-pass filter

$$h(\lambda) = \begin{cases} 1, & \lambda \leq \frac{2}{3t_J}, \\ \cos\left(\frac{\pi}{2}\nu\left(\frac{3t_J}{2}\lambda - 1\right)\right), & \frac{2}{3t_J} < \lambda < \frac{4}{3t_J}, \\ 0, & \lambda \geq \frac{4}{3t_J}, \end{cases}$$

where $t_J > 0$ is the largest scaling parameter chosen for the spectral wavelets. The so-defined scaling window h is supported in the interval $[0, \frac{4}{3t_J}]$, which in general is part of the spectrum $[0, \lambda_n]$ of the graph Laplacian. The Meyer wavelet window g is given as

$$g(\lambda) = \begin{cases} \sin\left(\frac{\pi}{2}\nu\left(\frac{3}{2}\lambda - 1\right)\right), & \frac{2}{3} < \lambda < \frac{4}{3}, \\ \cos\left(\frac{\pi}{2}\nu\left(\frac{3}{4}\lambda - 1\right)\right), & \frac{4}{3} < \lambda < \frac{8}{3}, \\ 0, & \text{otherwise.} \end{cases}$$

This defines a smooth band-pass spectral filter supported in the interval $[\frac{2}{3}, \frac{8}{3}]$. Now, defining the dyadic scales $t_j = 2^{j-J}t_J$ for $j \in \{1, \dots, J\}$ we get the scaled window functions $g(t_j\lambda) = g(2^{j-J}t_J\lambda)$ which are supported in the interval $[\frac{2^{j-J+1}}{3t_J}, \frac{2^{j-J+3}}{3t_J}]$. The Meyer construction is designed so that

$$G(\lambda) = h(\lambda)^2 + \sum_{j=1}^J g(2^{j-J}t_J\lambda)^2 = 1, \quad \lambda \in \left[0, \frac{2^{J+1}}{3t_J}\right].$$

With a proper selection of the largest scale J and the scale parameter t_J , it is therefore possible to obtain $G(\lambda) = 1$ on the entire spectrum of the graph Laplacian. This guarantees that the system in Theorem 5.4 forms a tight frame with constants $A = B = 1$ and allows for a stable reconstruction of the graph signals. A visualization of the Meyer wavelet window is given in Fig. 10 (left).

5.3 Efficient calculation of wavelet coefficients

Similar to the GFT, the spectral wavelet transform can be written compactly in matrix-vector form. From this perspective, the wavelet coefficients are obtained in terms of the matrix-vector products

$$c_{t_j} = \begin{bmatrix} \langle x, \psi_{t_j,1} \rangle \\ \vdots \\ \langle x, \psi_{t_j,n} \rangle \end{bmatrix} = \begin{bmatrix} \mathbf{T}_{g,t_j} x(v_1) \\ \vdots \\ \mathbf{T}_{g,t_j} x(v_n) \end{bmatrix} = \mathbf{T}_{g,t_j} x, \quad j \in \{1, \dots, J\}.$$

Similarly, the coefficients of the scaling functions can be expressed as

$$d = \begin{bmatrix} \langle x, \phi_1 \rangle \\ \vdots \\ \langle x, \phi_n \rangle \end{bmatrix} = \begin{bmatrix} \mathbf{T}_h x(v_1) \\ \vdots \\ \mathbf{T}_h x(v_n) \end{bmatrix} = \mathbf{T}_h x.$$

On the other hand, for given coefficients c_{t_j} and d , the reconstruction formula can be compactly written as

$$x = \mathbf{T}_{h/G} d + \sum_{j=1}^J \mathbf{T}_{1/G} \mathbf{T}_{g,t_j} c_{t_j}.$$

In particular, this means that all involved operators for the calculation of the coefficients and the reconstruction are of convolution-type and can be efficiently approximated with polynomials of the graph Laplacian. This can be, for instance, done with a polynomial interpolation scheme based on a Chebyshev expansion, as described in Section 4.4 and in [35]. In some cases, even more accurate approximations can be obtained using Krylov subspace methods based on the Lanczos iteration [27, 61].

6 Diffusion wavelets on graphs

The success and efficiency of classical wavelets in Euclidean spaces is strongly linked to the existence of translation and dilation operators which are naturally defined in these settings. For graph signals, however, these classical definitions are no longer available. Graphs are typically irregular and do not admit uniform dilation or shift operations. While spectral wavelets circumvent these issues by defining a dilation analog on the spectrum of the Laplacian, other wavelet constructions focus more on the vertex domain of the graph.

Introduced by Coifman and Maggioni [14], **diffusion wavelets** represent one such approach in which the classical dilation of Euclidean spaces is replaced by the repeated application of a diffusion operator $\mathbf{T}$. In this way, dilation on graphs is described through the dynamics of diffusion. It allows us to define wavelets with a vertex-local graph-adapted structure. A further advantage compared to spectral wavelets is the resulting orthogonality of the wavelets and an available multi-resolution analysis.

A **diffusion operator** on G is defined as a linear operator $\mathbf{T} : \mathcal{L}(V) \rightarrow \mathcal{L}(V)$ (or equivalently as a matrix in $\mathbb{R}^{n \times n}$) with the following additional properties suitable to obtain diffusion wavelets:

- (i) It is symmetric.
- (ii) Its spectral norm is bounded by $\|\mathbf{T}\|_2 \leq 1$
- (iii) It is a local operator with respect to the graph distance, meaning that $(\mathbf{T}x)(v_i)$ depends only on values of x at vertices in a small neighborhood around v_i .
- (iv) Increasing powers $\mathbf{T}^j$ have a decreasing ε -rank.

Here, the ε -rank of a matrix is defined as follows.

Definition 6.1 Let $\varepsilon > 0$. Given vectors $\{x_1, \dots, x_k\} \subset \mathbb{R}^n$, their ε -span is the set

$$\text{span}_\varepsilon\{x_1, \dots, x_k\} = \left\{ x \in \mathbb{R}^n : \exists c_1, \dots, c_k \text{ such that } \left\| x - \sum_{i=1}^k c_i x_i \right\|_2 < \varepsilon \right\}.$$

For a matrix $\mathbf{A} \in \mathbb{R}^{n \times p}$, its ε -rank is defined as

$$\text{rank}_\varepsilon(\mathbf{A}) := \min \left\{ k \in \mathbb{N} : \begin{array}{l} \exists x_1, \dots, x_k \in \mathbb{R}^n \text{ such that the columns of } \mathbf{A} \\ \text{are in the } \varepsilon\text{-span of } \{x_1, \dots, x_k\} \end{array} \right\}.$$

The ε -rank assumption for increasing powers of the diffusion operator $\mathbf{T}$ reflects the idea that the repeated application of $\mathbf{T}$ leads to a smoothing of a signal. As the power increases, progressively more and more components of the signal are attenuated. Typical choices for $\mathbf{T}$ include the normalized adjacency matrix given as $\mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}$, or operators based on the graph Laplacian, such as the heat diffusion operator $\mathbf{T} = e^{-tL}$.

Once $\mathbf{T}$ is selected, its repeated application generates increasingly smoother signals

$$x, \mathbf{T}x, \mathbf{T}^2x, \mathbf{T}^4x, \dots, \mathbf{T}^{2^{j-1}}x.$$

This observation is at the core of the numerical construction of diffusion wavelets. The powers $\mathbf{T}^{2^{j-1}}$ correspond to different diffusions acting at dyadic scales on a signal, making it possible to differentiate between different resolutions.

Diffusion wavelets are calculated numerically in terms of a sequence of nested **scaling spaces** associated with the powers $\mathbf{T}^{2^{j-1}}$. For this, we denote by $\mathcal{V}^{(0)} = \mathcal{L}^2(V)$ the space of all graph signals, and then calculate J sequentially smaller spaces

$$\mathcal{V}^{(0)} \supseteq \mathcal{V}^{(1)} \supseteq \mathcal{V}^{(2)} \supseteq \dots \supseteq \mathcal{V}^{(J)},$$

such that the range of $\mathbf{T}^{2^{j-1}}$ lies in the ε -span of the space $\mathcal{V}^{(j)}$. As the scale j increases, the scaling spaces $\mathcal{V}^{(j)}$ include increasingly smoother signals. The dimensions of $\mathcal{V}^{(j)}$ decrease with the scale j , and allow for a compact representation of $\mathbf{T}$ at coarser scales. Therefore, unlike classical multi-resolution analysis, the spaces $\mathcal{V}^{(j)}$ are not generated by translating and dilating a single scaling function, but calculated numerically through the action of a diffusion operator $\mathbf{T}$.

6.1 Construction of diffusion scaling functions

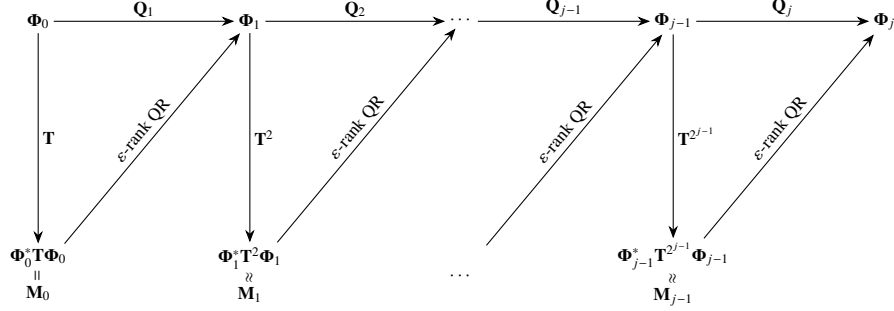


Fig. 12 Diagram of the principal scheme generating the diffusion scaling functions

In the following, we give a brief overview of how the diffusion scaling functions are calculated. We assume that the spaces $\mathcal{V}^{(j)}$ have dimension n_j and that a respective orthonormal basis $\{\phi_{j,k} : k \in \{1, \dots, n_j\}\}$ of $\mathcal{V}^{(j)}$ is encoded in the columns of a matrix $\Phi_j \in \mathbb{R}^{n \times n_j}$ satisfying $\Phi_j^* \Phi_j = \mathbf{I}_{n_j}$.

The calculation of the diffusion scaling functions $\phi_{j,k}$, $k \in \{1, \dots, n_j\}$, spanning the spaces $\mathcal{V}^{(j)}$ is initialized at level $j = 0$ by setting $\Phi_0 = \mathbf{I}_n$, i.e., the initial basis functions $\phi_{0,k} = \delta_{v_k}$, $k \in \{1, \dots, n\}$ correspond precisely to the canonical orthonormal basis in $\mathcal{L}^2(V)$. The initial operator $\mathbf{M}_0 = \mathbf{T} \in \mathbb{R}^{n \times n}$ is already expressed in terms of the first basis Φ_0 , and we have $n_0 = n$.

We describe now the main calculational step for the scale $j \in \{1, \dots, J\}$, illustrated also in the diagram of Fig. 12. For this, we assume that the basis Φ_{j-1} and the matrix $\mathbf{M}_{j-1} \in \mathbb{R}^{n_{j-1} \times n_{j-1}}$ have been calculated, and that $\mathbf{M}_{j-1}$ is an approximate representation of the matrix $\Phi_{j-1}^* \mathbf{T}^{2^{j-1}} \Phi_{j-1}$ describing the power $\mathbf{T}^{2^{j-1}}$ in the basis Φ_{j-1} of the space $\mathcal{V}^{(j-1)}$. Using a special QR decomposition of the matrix $\mathbf{M}_{j-1}$ with column pivoting, we can then extract a new orthonormal basis

$$\mathbf{Q}_j \in \mathbb{R}^{n_{j-1} \times n_j}$$

such that the columns of $\mathbf{M}_{j-1}$ are in the ε -span of the space spanned by the columns of $\mathbf{Q}_j$. The scaling functions spanning the space $\mathcal{V}^{(j)}$ are then given as the columns of

$$\Phi_j = \Phi_{j-1} \mathbf{Q}_j \in \mathbb{R}^{n \times n_j}.$$

In practice, the applied QR decomposition to obtain the matrix Φ_j is a localized QR decomposition with a tolerance ε that removes directions of $\mathbf{M}_{j-1}$ associated with small singular values (cf. [14]). The resulting basis functions $\phi_{j,k}$, $k \in \{1, \dots, n_j\}$, encoded in the columns of Φ_j are referred to as **diffusion scaling functions**. The QR-decomposition is performed in such a way that these functions are localized on the

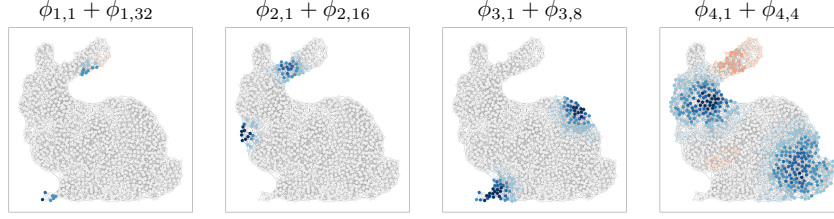


Fig. 13 Illustration of diffusion scaling functions $\phi_{j,k}(v)$ on the bunny graph for different scales j and different positions k . The diffusion is based on the operator $\mathbf{T} = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$.

graph, with supports that grow as the scale increases. Their shapes and support adapt to the diffusion properties of the operator $\mathbf{T}$. The new basis allows to compress the matrix $\mathbf{M}_{j-1}$, and we obtain the matrix

$$\mathbf{T}_j = \mathbf{Q}_j^* \mathbf{M}_{j-1} \mathbf{Q}_j \in \mathbb{R}^{n_j \times n_j}$$

as a representation of $\mathbf{T}^{2^{j-1}}$ in the space spanned by the new basis Φ_j . Finally, the matrix $\mathbf{M}_j \in \mathbb{R}^{n_j \times n_j}$ required for the next computational step is calculated as the square

$$\mathbf{M}_j = \mathbf{T}_j^2 \in \mathbb{R}^{n_j \times n_j}$$

of the matrix $\mathbf{T}_j$. It can be regarded as an approximate representation of the next dyadic power $\mathbf{T}^{2^j}$ in the new basis Φ_j . Different scales of the diffusion scaling functions $\phi_{j,k}$ on the bunny graph are illustrated in Fig. 13.

6.2 Wavelet spaces and diffusion wavelets

The wavelet spaces are defined as the orthogonal complements

$$\Delta \mathcal{V}^{(j)} = \mathcal{V}^{(j-1)} \ominus \mathcal{V}^{(j)}.$$

These spaces contain the information that gets lost when passing from a scale $j-1$ to a coarser scale j . The orthonormal basis functions $\{\psi_{j,k} : k \in \{1, \dots, n_{j-1} - n_j\}\}$ of the detail space $\Delta \mathcal{V}^{(j)}$ are referred to as a **diffusion wavelets**. They represent the variations of a signal that are not smoothed out by the diffusion operator at scale j , but are present at scale $j-1$. The full diffusion wavelet decomposition can be written as

$$\mathcal{L}^2(V) = \mathcal{V}_J \oplus \bigoplus_{j=1}^J \Delta \mathcal{V}^{(j)},$$

where J denotes the coarsest scale considered.

For the computation of the diffusion wavelets, we require the orthogonal complement $\mathbf{Q}_j^\perp \in \mathbb{R}^{n_{j-1} \times (n_{j-1} - n_j)}$ of the basis $\mathbf{Q}_j$. This complement can be calculated together with the basis $\mathbf{Q}_j$ through the QR decomposition of the matrix $\mathbf{M}_j$. The diffusion wavelets $\psi_{j,k}$, $k \in \{1, \dots, n_{j-1} - n_j\}$ at scale j are then given as the columns of the matrices

$$\Psi_j = \Phi_{j-1} \mathbf{Q}_j^\perp, \quad j \in \{1, \dots, J\}.$$

The complete procedure for the combined calculation of the diffusion scaling functions and the diffusion wavelets is summarized in Algorithm 6.1.

Algorithm 6.1: Generation of diffusion wavelets

Input: $\Phi_0 = \mathbf{I}_n$, $n_0 = n$, number J of wavelets levels.

for $j = 1$ **to** J **do**

Calculate square matrix

$$\mathbf{M}_{j-1} = \mathbf{T}_{j-1}^2 \in \mathbb{R}^{n_{j-1} \times n_{j-1}} \quad (\text{use } \mathbf{M}_0 = \mathbf{T} \text{ if } j = 1)$$

 as a compact representation of $\Phi_{j-1}^* \mathbf{T}^{2^{j-1}} \Phi_{j-1}$.

Calculate QR factorization of $\mathbf{M}_{j-1}$ (with column pivoting) such that the first n_j columns in the orthonormal matrix $\mathbf{Q} \in \mathbb{R}^{n_{j-1} \times n_{j-1}}$ $\mathcal{E}$ -span the range of $\mathbf{M}_{j-1}$. Split the matrix $\mathbf{Q}$ in $\mathbf{Q}_j$ (the first n_j columns) and $\mathbf{Q}_j^\perp$ (the last $n_{j-1} - n_j$ columns).

Calculate the matrix

$$\mathbf{T}_j = \mathbf{Q}_j^* \mathbf{M}_{j-1} \mathbf{Q}_j \in \mathbb{R}^{n_j \times n_j}$$

 as a compact representation of $\Phi_j^* \mathbf{T}^{2^{j-1}} \Phi_j$.

Calculate the new scaling functions and diffusion wavelets

$$\Phi_j = \Phi_{j-1} \mathbf{Q}_j, \quad \Psi_j = \Phi_{j-1} \mathbf{Q}_j^\perp.$$

Output: The diffusion scaling functions $\phi_{j,k}$ and the diffusion wavelets $\psi_{j,k}$ as columns of the matrices

$$\Phi_j \in \mathbb{R}^{n \times n_j}, \quad \Psi_j \in \mathbb{R}^{n \times (n_{j-1} - n_j)}, \quad j \in \{1, \dots, J\}.$$

Remark 6.2 Diffusion wavelets have localization properties that are linked to the locality of the diffusion operator $\mathbf{T}$. The wavelets on the finer scales (small j) are highly localized in small regions of the graph, while coarse-scale wavelets have a wider support and describe global structures. This is not explicitly clear from the computational procedure, but arises from the diffusion behavior of the operator $\mathbf{T}$.

Diffusion wavelets implicitly also adapt to the structure of the graph. In densely connected regions, diffusion mixes rapidly and wavelets tend to have a broader support. In sparse or weakly connected regions, diffusion is slower and the respective wavelets more localized. This adaptivity makes diffusion wavelets particularly interesting for graphs with a non-homogeneous structure.

7 Multiresolution analysis with binary graph partitionings (BGPs)

7.1 Haar wavelets based on BGPs

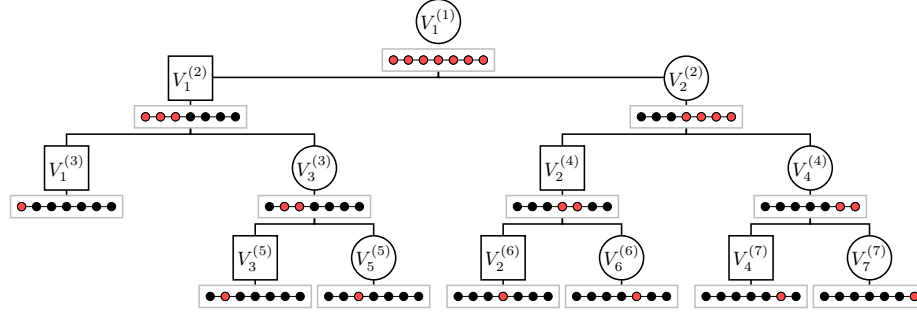


Fig. 14 A complete BGP tree for the path graph P_7 with 7 nodes. The red nodes indicate the elements of the sets $V_k^{(m)}$ in the partitions $\mathcal{P}^{(m)}$.

As a third major family of wavelets on graphs, we consider a vertex-based wavelet construction based on recursive partitionings of the vertex set V . Here, the partitions organized within a binary tree encode the decomposition of the vertices in (presumably) relevant subsets of the graph. We start with a general theory on binary partitioning trees.

Definition 7.1 A **binary graph partitioning (BGP) tree** $\mathcal{T}$ of the graph G is a binary tree consisting of subsets of the vertex set V that are ordered recursively in partitions $\mathcal{P}^{(m)}$, $m \leq M$ ($M \leq n$), of V by the following rules:

1. The vertex set V is the **root** of the BGP tree $\mathcal{T}$ and provides the first trivial partition

$$\mathcal{P}^{(1)} := \{V\}.$$

2. Let $\mathcal{P}^{(m)} = \{V_1^{(m)}, \dots, V_m^{(m)}\}$ be a partition of V consisting of $m < M$ elements in the BGP tree $\mathcal{T}$, that does not contain only leaves of $\mathcal{T}$. Then, there is a **selection rule**

$$j = j_m \in \{1, \dots, m\}$$

that determines the next set $V_j^{(m)}$ of size $|V_j^{(m)}| \geq 2$ to be split.

3. Starting from $\mathcal{P}^{(m)}$, the next partition $\mathcal{P}^{(m+1)}$ of V in $\mathcal{T}$ is obtained by applying a **dyadic split** to the selected set $V_j^{(m)}$, in order to obtain two disjoint non-empty sets W' and W'' such that $V_j^{(m)} = W' \cup W''$. The next partition $\mathcal{P}^{(m+1)}$ is then given as

$$\mathcal{P}^{(m+1)} := \{V_1^{(m+1)}, \dots, V_{m+1}^{(m+1)}\}$$

with $V_k^{(m+1)} = V_k^{(m)}$ if $k \in \{1, \dots, m\} \setminus \{j\}$, $V_j^{(m+1)} = W'$ and $V_{m+1}^{(m+1)} = W''$.

If W' is an element of the partition $\mathcal{P}^{(m+1)}$ obtained from a dyadic split of a set $W \in \mathcal{P}^{(m)}$, then $W' \subset W$ is referred to as **child** of W in the tree $\mathcal{T}$. We call two elements $W', W'' \in \mathcal{T}$ **siblings** if both sets are children of the same **parent** $W \in \mathcal{T}$. If $\mathcal{T}$ is a binary tree, a set $W \in \mathcal{T}$ can only have two children or no children at all. In the latter case we call W a **leaf** of the tree $\mathcal{T}$. The final partition $\mathcal{P}^{(M)}$ consists of $M \leq n$ elements containing the M leaves of the tree $\mathcal{T}$.

Example 7.2 A simple way to construct a BGP tree is through hierarchical 2-center clustering. This approach is a recursive partitioning method linked to the 2-center problem. Given a connected set of nodes, the objective is to obtain two center nodes such that every node is assigned to its nearest center, while minimizing the maximum distance to a center. In practice, heuristic methods such as Gonzalez's algorithm [31] are used to get an approximate solution. In this heuristic, starting from an initial center, the second center is chosen as the node farthest from the first.

In the hierarchical setting, this procedure is applied iteratively. Starting with the vertex set V as the root of the BGP tree, the 2-center problem induces a partition of V into two subsets. Each subset is then subdivided in the same way, yielding a binary partitioning tree. As a selection rule, one can, for instance, choose at each step the subset with the largest number of nodes for the next refinement.

Further, there are several possibilities for selecting the centers. Applying Gonzalez's algorithm recursively, one natural choice is to retain the center of the parent set as one of the two centers and select the second as the node farthest from it. Similar but more adaptive strategies will be used in Section 8 for the construction of graph wedgelets.

The introduced ordering of the tree $\mathcal{T}$ based on the selection rule j_m is non-standard in the literature, but turns out to be extremely helpful when implementing a BGP tree. It will further help us to simplify the notation for Haar wavelets. From a theoretical point of view, also the balancedness of trees will be important.

Definition 7.3 We call a BGP tree $\mathcal{T}$ **balanced** if there exists $\frac{1}{2} \leq \rho < 1$ such that for every child W' of an element $W \in \mathcal{T}$ we have

$$(1 - \rho)|W| \leq |W'| \leq \rho|W|.$$

We call a BGP tree $\mathcal{T}$ **complete** if it has $M = n$ leaves, each containing a single vertex of the graph G . A complete and balanced BGP tree will be called **BGP(ρ) tree**.

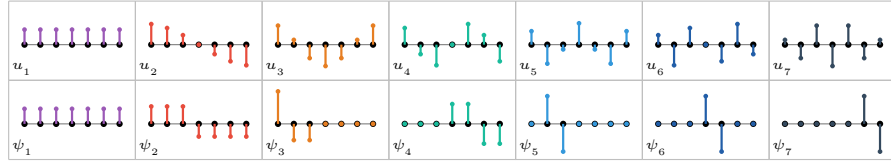


Fig. 15 Comparison of Fourier basis function u_m with Haar wavelets ψ_m on the path graph P_7 . The Haar wavelets are based on the BGP tree in Fig. 14.

Definition 7.4 The **Haar scaling functions** with respect to a BGP tree $\mathcal{T}$ are defined as the characteristic functions of the sets W in the tree $\mathcal{T}$, i.e., we define

$$\omega_W(v) := \mathbb{1}_W(v) = \begin{cases} 1 & \text{if } v \in W, \\ 0 & \text{if } v \notin W, \end{cases} \quad \text{for } W \in \mathcal{T}.$$

For a more precise indexing of the Haar scaling functions, we make use of the ordered partitions $\mathcal{P}^{(m)}$ in the tree $\mathcal{T}$ and set

$$\omega_k^{(m)}(v) := \omega_{V_k^{(m)}}(v) = \mathbb{1}_{V_k^{(m)}}(v),$$

where $m \in \{1, \dots, M\}$ denotes the partition size and $k \in \{1, \dots, m\}$ the index of the element $V_k^{(m)}$ in the partition $\mathcal{P}^{(m)}$. The Haar scaling functions based on the partitions $\mathcal{P}^{(m)}$ generate a sequence of increasing **scaling spaces**

$$\mathcal{H}^{(m)} := \text{span} \left\{ \omega_k^{(m)} : k \in \{1, \dots, m\} \right\}, \quad m \in \{1, \dots, M\}.$$

such that

$$\mathcal{H}^{(1)} \subset \mathcal{H}^{(2)} \subset \dots \subset \mathcal{H}^{(M)}.$$

The space $\mathcal{H}^{(m)}$ corresponds to the space of all signals that are piecewise constant on the elements of the partition $\mathcal{P}^{(m)}$. The additional nestedness of the spaces $\mathcal{H}^{(m)}$ leads to a multiresolution analysis for signals on graphs based on the **detail spaces**

$$\begin{aligned} \Delta\mathcal{H}^{(1)} &:= \mathcal{H}^{(1)}, \\ \Delta\mathcal{H}^{(m)} &:= \mathcal{H}^{(m)} \ominus \mathcal{H}^{(m-1)}, \quad m \in \{2, \dots, M\}, \end{aligned}$$

defined upon the inner product $\langle \cdot, \cdot \rangle$ of the space $\mathcal{L}^2(V)$. If the tree $\mathcal{T}$ is complete, i.e., it contains $M = n$ leaves, we have $\mathcal{H}^{(n)} = \mathcal{L}^2(V)$ and the detail spaces provide an orthogonal decomposition of $\mathcal{L}^2(V)$ of the form

$$\mathcal{L}^2(V) = \bigoplus_{m=1}^n \Delta\mathcal{H}^{(m)}.$$

Due to the ordered binary construction of the BGP trees, the detail spaces $\Delta\mathcal{H}^{(m)}$ are one-dimensional. They are related to the following wavelets.

Definition 7.5 Let $\mathcal{T}$ be a BGP tree for the graph G . Then the **Haar wavelets** with respect to the ordered tree $\mathcal{T}$ are defined as

$$\begin{aligned} \psi_1(v) &:= \omega_1^{(1)}(v), \\ \psi_m(v) &:= \omega_{j_{m-1}}^{(m)}(v) - \frac{|V_{j_{m-1}}^{(m)}|}{|V_m^{(m)}|} \omega_m^{(m)}(v), \quad m \in \{2, \dots, n\}, \end{aligned}$$

where j_{m-1} denotes the index of the set $V_{j_{m-1}}^{(m-1)} \in \mathcal{P}^{(m-1)}$ in the BGP tree which is split into the children sets $V_{j_{m-1}}^{(m)}$ and $V_m^{(m)}$ of the partition $\mathcal{P}^{(m)}$.

For subsequent theoretical results on best m -term approximation, we additionally use the following more compact notation for projections related to the Haar wavelets. Let $W', W \in \mathcal{T}$ such that W' is a child of W . Then, the **wavelet component** $\Psi_{W'}(x)$ of a signal x with respect to the child set W' is defined as the signal

$$\Psi_{W'}(x)(v) = \left(\frac{\langle x, \omega_{W'} \rangle}{|W'|} - \frac{\langle x, \omega_W \rangle}{|W|} \right) \omega_{W'}(v), \quad (16)$$

For the root $V \in \mathcal{T}$, we additionally set

$$\Psi_V(x)(v) = \frac{\langle x, \omega_V \rangle}{|V|}.$$

We collect all basic properties of the Haar wavelets in the following proposition.

Proposition 7.6 *Let $\mathcal{T}$ be a complete BGP tree on a graph G of size n . Then:*

- (i) *The BGP tree $\mathcal{T}$ contains $2n - 1$ elements, $n - 1$ parents and n leaves. Further, the n scaling functions $\{\omega_1^{(n)}, \dots, \omega_n^{(n)}\}$ linked to the final partition $\mathcal{P}^{(n)}$ of $\mathcal{T}$ form a canonical orthonormal basis for the space $\mathcal{L}^2(V) = \mathcal{H}^{(n)}$.*
- (ii) *The Haar wavelets ψ_m , $m \in \{1, \dots, n\}$, generate the one-dimensional detail spaces $\Delta\mathcal{H}^{(m)}$. In particular, the Haar wavelets $\{\psi_1, \dots, \psi_n\}$ form an orthogonal basis for the space $\mathcal{L}^2(V)$. Their $\mathcal{L}^2(V)$ -norm can be calculated as*

$$\|\psi_1\|_2^2 = n, \quad \text{and} \quad \|\psi_m\|_2^2 = \frac{|V_{j_{m-1}}^{(m)}|}{|V_m^{(m)}|} |V_{j_{m-1}}^{(m-1)}|, \quad \text{for } m \in \{2, \dots, n\}.$$

Therefore, for every signal $x \in \mathcal{L}^2(V)$ we get the orthogonal decomposition

$$x(v) = \sum_{m=1}^n \frac{\langle x, \psi_m \rangle}{\|\psi_m\|_2^2} \psi_m(v).$$

- (iii) *For every signal $x \in \mathcal{L}^2(V)$ we have*

$$x = \sum_{W \in \mathcal{T}} \Psi_W(x),$$

i.e., x can be decomposed in terms of $2n - 1$ wavelet components $\Psi_W(x)$. Further, if $W = V_{j_{m-1}}^{(m-1)}$ has the two children $W' = V_{j_{m-1}}^{(m)}$ and $W'' = W \setminus W' = V_m^{(m)}$, then

$$\Psi_{W'}(x)(v) + \Psi_{W''}(x)(v) = \frac{\langle x, \psi_m \rangle}{\|\psi_m\|_2^2} \psi_m(v),$$

i.e., $\Psi_{W'}(x) + \Psi_{W''}(x)$ corresponds to the orthogonal projection of the signal x into the detail space $\Delta\mathcal{H}^{(m)}$.

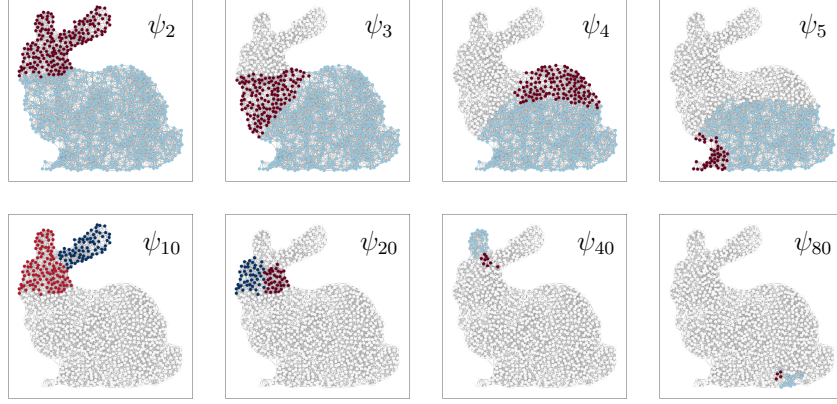


Fig. 16 Illustration of 8 Haar wavelets ψ_m on the bunny graph. The Haar wavelets are based on a BGP tree constructed via hierarchical 2-center clustering (see Example 7.2). The positive entries on the domain $V_{j_{m-1}}^{(m)}$ are colored in red, while the negative entries on $V_m^{(m)}$ are colored in blue. The ratio between the sizes of the two domains is indicated by the brightness of the colors.

Proof. (i) As $\mathcal{T}$ is complete and binary, it contains n leaves, and at least two of them are siblings (assuming that $n > 1$, for $n = 1$ the tree contains just a single root). Pruning the tree $\mathcal{T}$ by removing this pair of siblings, we obtain a reduced tree with $n - 1$ leaves. An induction argument over the number n of leaves therefore tells us that $|\mathcal{T}| = 2n - 1$.

(ii) By Definition 7.5, the Haar wavelets ψ_m are in the scaling space $\mathcal{H}^{(m)}$. Further,

$$\langle \psi_m, \omega_k^{(\ell)} \rangle = \langle \omega_{j_{m-1}}^{(m)}, \omega_k^{(\ell)} \rangle - \frac{|V_{j_{m-1}}^{(m)}|}{|V_m^{(m)}|} \langle \omega_m^{(m)}, \omega_k^{(\ell)} \rangle = 0$$

for all $\ell \in \{1, \dots, m-1\}$ and $k \in \{1, \dots, \ell\}$. This implies that $\psi_m \perp \mathcal{H}^{(m-1)}$, and, thus, that ψ_m is a non-trivial element of the detail space $\Delta\mathcal{H}^{(m)}$. As the space $\Delta\mathcal{H}^{(m)}$ is one-dimensional, the Haar wavelet ψ_m thus generates $\Delta\mathcal{H}^{(m)}$. Therefore, the set $\{\psi_1, \dots, \psi_n\}$ provides an orthogonal basis of $\mathcal{L}^2(V)$ with the respective norms calculated as

$$\|\psi_1\|_2^2 = \|\mathbb{1}_V\|_2^2 = n,$$

and

$$\begin{aligned} \|\psi_m\|_2^2 &= \|\omega_{j_{m-1}}^{(m)}\|_2^2 + \frac{|V_{j_{m-1}}^{(m)}|^2}{|V_m^{(m)}|^2} \|\omega_m^{(m)}\|_2^2 \\ &= \frac{|V_{j_{m-1}}^{(m)}|}{|V_m^{(m)}|} (|V_m^{(m)}| + |V_{j_{m-1}}^{(m)}|) = \frac{|V_{j_{m-1}}^{(m)}|}{|V_m^{(m)}|} |V_{j_{m-1}}^{(m-1)}|. \end{aligned}$$

For any signal $x \in \mathcal{L}^2(V)$, the decomposition in terms of the Haar wavelets can therefore be written as

$$x(v) = \frac{1}{n} \langle x, \psi_1 \rangle \psi_1(v) + \sum_{m=2}^n \frac{|V_m^{(m)}|}{|V_{j_{m-1}}^{(m)}|} \frac{\langle x, \psi_m \rangle}{|V_{j_{m-1}}^{(m-1)}|} \psi_m(v).$$

(iii) The completeness of $\mathcal{T}$ implies that we can decompose any signal x in terms of the n leaves of $\mathcal{T}$ as

$$x = \sum_{i=1}^n x(v_i) \mathbb{I}_{\{v_i\}} = \sum_{W \in \mathcal{P}(n)} \frac{\langle x, \omega_W \rangle}{|W|} \omega_W.$$

Now, by definition of the wavelet components $\Psi_W(x)$ in (16), we recursively get

$$\begin{aligned} x &= \sum_{W \in \mathcal{P}(n) \setminus \mathcal{P}(n-1)} \Psi_W(x) + \sum_{W \in \mathcal{P}(n-1)} \frac{\langle x, \omega_W \rangle}{|W|} \omega_W \\ &= \sum_{m=2}^n \sum_{W \in \mathcal{P}^{(m)} \setminus \mathcal{P}^{(m-1)}} \Psi_W(x) + \Psi_V(x) = \sum_{W \in \mathcal{T}} \Psi_W(x). \end{aligned}$$

For $m \geq 2$, the set $\mathcal{P}^{(m)} \setminus \mathcal{P}^{(m-1)}$ contains the two elements $V_{j_{m-1}}^{(m)}$ and $V_m^{(m)}$, being the two children of $V_{j_{m-1}}^{(m-1)}$. For these three sets we have

$$|V_{j_{m-1}}^{(m-1)}| = |V_{j_{m-1}}^{(m)}| + |V_m^{(m)}|, \quad \text{and} \quad \omega_{j_{m-1}}^{(m-1)}(v) = \omega_{j_{m-1}}^{(m)}(v) + \omega_m^{(m)}(v).$$

This implies the identity

$$\begin{aligned} \sum_{W \in \mathcal{P}^{(m)} \setminus \mathcal{P}^{(m-1)}} \Psi_W(x) &= \Psi_{V_{j_{m-1}}^{(m)}}(x) + \Psi_{V_m^{(m)}}(x) \\ &= \left(\frac{\langle x, \omega_{j_{m-1}}^{(m)} \rangle}{|V_{j_{m-1}}^{(m)}|} - \frac{\langle x, \omega_{j_{m-1}}^{(m-1)} \rangle}{|V_{j_{m-1}}^{(m-1)}|} \right) \omega_{j_{m-1}}^{(m)} + \left(\frac{\langle x, \omega_m^{(m)} \rangle}{|V_m^{(m)}|} - \frac{\langle x, \omega_{j_{m-1}}^{(m-1)} \rangle}{|V_{j_{m-1}}^{(m-1)}|} \right) \omega_m^{(m)} \\ &= \frac{1}{|V_{j_{m-1}}^{(m-1)}|} \left(\left\langle x, \frac{|V_m^{(m)}|}{|V_{j_{m-1}}^{(m)}|} \omega_{j_{m-1}}^{(m)} - \omega_m^{(m)} \right\rangle \omega_{j_{m-1}}^{(m)} - \left\langle x, \omega_{j_{m-1}}^{(m)} - \frac{|V_{j_{m-1}}^{(m)}|}{|V_m^{(m)}|} \omega_m^{(m)} \right\rangle \omega_m^{(m)} \right) \\ &= \frac{|V_m^{(m)}|}{|V_{j_{m-1}}^{(m)}| |V_{j_{m-1}}^{(m-1)}|} \langle x, \psi_m \rangle \psi_m = \frac{\langle x, \psi_m \rangle}{\|\psi_m\|_2^2} \psi_m. \end{aligned}$$

□

The operators $\Psi_{W'}(x)$ can be interpreted as projection operators in which the function x is projected into a detail space $\Delta \mathcal{H}^{(m)}$ and then, in a second step, projected to the set W' such that it vanishes on the complement $W \setminus W'$.

Remark 7.7 If the BGP tree $\mathcal{T} = \mathcal{T}(x)$ depends on the function x , as later on the case for the graph wedgelets, the respective wavelets are referred to as **geometric wavelets**.

7.2 m -term approximation for wavelets on BGPs

To see whether a graph signal x can be approximated sparsely by piecewise constant functions on the elements of a BGP tree $\mathcal{T}$, we will analyze the $\mathcal{L}^2$ -error $\|x - \mathcal{S}_m(x)\|_2$, where $\mathcal{S}_m(x)$ denotes the best m -term approximation

$$\mathcal{S}_m(x) = \sum_{i=1}^m \Psi_{W_i}(x)(v) \quad (17)$$

of the function x in terms of m wavelet components $\Psi_{W_i}(x)$, $i \in \{1, \dots, m\}$. These wavelet components are determined by the elements W_i of the tree $\mathcal{T}$, and sorted descendingly in terms of the $\mathcal{L}^2(V)$ -norm:

$$\|\Psi_{W_1}(x)\|_2 \geq \|\Psi_{W_2}(x)\|_2 \geq \|\Psi_{W_3}(x)\|_2 \geq \dots$$

To analyze the convergence of the m -term approximation, the following energy functional plays a central role (see [21] for a general overview). It can be regarded as the discrete counterpart of a related functional introduced in [17] in the context of binary space partitionings of hypercubes. In wavelet theory, such functionals are commonly used in the characterization of Besov spaces, where they quantify the sparsity of the wavelet representation of a signal x . In our setting, this notion of sparsity is closely connected to the partitions induced by a BGP tree $\mathcal{T}$.

Definition 7.8 For $0 < r < \infty$, we define the r -**energy** of the wavelet components of x with respect to a BGP tree $\mathcal{T}$ as

$$\mathcal{N}_r(x, \mathcal{T}) = \left(\sum_{W \in \mathcal{T}} \|\Psi_W(x)\|_2^r \right)^{\frac{1}{r}}.$$

Similar as the r -energy functional $\mathcal{N}_r(x, \mathcal{T})$ also the following Besov-type smoothness term quantifies how well a function x can be approximated with piecewise constant functions on the elements of a BGP tree.

Definition 7.9 For $\alpha > 0$, $0 < r < \infty$, and a given BGP tree $\mathcal{T}$, we define the smoothness measure $\mathcal{M}_r^\alpha(x, \mathcal{T})$ of graph signal $x \in \mathcal{L}(V)$ with respect to the tree $\mathcal{T}$ as

$$\mathcal{M}_r^\alpha(x, \mathcal{T}) = \left(|V|^{-\alpha r} \|x\|_r^r + \sum_{W \in \mathcal{T}} |W|^{-\alpha r} \sup_{w \in W} \sum_{v \in W} |x(v) - x(w)|^r \right)^{\frac{1}{r}}.$$

Further, we define the **geometric Besov-type smoothness measure** $|x|_{\mathcal{GB}_r^\alpha}$ of a signal $x \in \mathcal{L}(V)$ as the infimum

$$|x|_{\mathcal{GB}_r^\alpha} = \inf_{\mathcal{T} \in \text{BGP}(\rho)} \mathcal{M}_r^\alpha(x, \mathcal{T})$$

over all $\text{BGP}(\rho)$ trees.

In [17], and similarly in [40], the function spaces associated with the Besov-type smoothness measure $|x|_{\mathcal{GB}_r^\alpha}$ are referred to as **geometric B-spaces**. In contrast to the r -energy introduced in Definition 7.8, the functional $|x|_{\mathcal{GB}_r^\alpha}$ is not tied to a single BGP tree. Rather, it quantifies the sparsity of x with respect to a broad family of BGP(ρ) trees. In principle, a similar perspective could be adopted for the r -energy $\mathcal{N}_r(x, \mathcal{T})$ by taking the infimum over all admissible BGP(ρ) trees.

In practice, it might not be possible to determine the infimum over all trees, and only an approximate solution might be feasible. Therefore, in our theoretical setting we will mainly consider near best BGP trees. This is also the setting we have in mind when we design greedy algorithms in the next section to create adaptive partitionings for the compression of graph signals.

Definition 7.10 We say that an x -adapted BGP(ρ) tree $\mathcal{T}_r(x)$ is a **near best BGP(ρ) tree** if there exists a constant $C > 0$ such that

$$\mathcal{N}_r(x, \mathcal{T}_r(x)) \leq C \inf_{\mathcal{T} \in \text{BGP}(\rho)} \mathcal{N}_r(x, \mathcal{T}).$$

With a proper selection of the smoothness parameter α in terms of r , we obtain the following general inequality between the r -energy $\mathcal{N}_r(x, \mathcal{T})$ and the smoothness measure $\mathcal{M}_r^\alpha(x, \mathcal{T})$.

Proposition 7.11 Let $0 < r < \infty$, $\alpha = \frac{1}{r} - \frac{1}{2}$ and $\mathcal{T}$ a BGP tree. Then, we have

$$\mathcal{N}_r(x, \mathcal{T}) \leq C \mathcal{M}_r^\alpha(x, \mathcal{T}). \quad (18)$$

Further, if $\mathcal{T}_r(x)$ is a near best BGP(ρ) tree, we get

$$\mathcal{N}_r(x, \mathcal{T}_r(x)) \leq C' |x|_{\mathcal{GB}_r^\alpha}. \quad (19)$$

The positive constants C and C' depend only on r .

Proof. We adapt the proof of [17, Theorem 3.5], in which a similar relation has been shown for binary space partitions in a continuous setting. To obtain (19) for a near best BGP(ρ) tree, it is in fact sufficient that (18) holds true for all BGP trees with a constant $C > 0$ depending only on r .

To show (18), we consider in more detail the $\mathcal{L}^r(V)$ -norms $\|\Psi_{W'}(x)\|_r$ of the wavelet components $\Psi_{W'}(x)$. If W' is a child of W in $\mathcal{T}$, we can express $\|\Psi_{W'}(x)\|_r$ for any number $0 < r < \infty$ as

$$\|\Psi_{W'}(x)\|_r = \left\| \left(\frac{\langle x, \mathbb{1}_{W'} \rangle}{|W'|} - \frac{\langle x, \mathbb{1}_W \rangle}{|W|} \right) \mathbb{1}_{W'} \right\|_r = |\bar{x}_{W'} - \bar{x}_W| |W'|^{\frac{1}{r}}, \quad (20)$$

where $\bar{x}_{W'}$ and $\bar{x}_W$ denote the mean values of x over the sets W' and W , respectively. For the root V of $\mathcal{T}$, we additionally get $\|\Psi_V(x)\|_r = |\bar{x}_V|^r |V|^{\frac{1}{r}}$. We can further estimate the $\mathcal{L}^r(V)$ -norms $\|\Psi_{W'}(x)\|_2^r$ as

$$\begin{aligned}
\|\Psi_{W'}(x)\|_r^r &\leq \max\{1, 2^{r-1}\} (\|(\bar{x}_{W'} - x)\mathbb{I}_{W'}\|_r^r + \|(\bar{x}_W - x)\mathbb{I}_W\|_r^r) \\
&\leq \max\{1, 2^{r-1}\} (\|(\bar{x}_{W'} - x)\mathbb{I}_{W'}\|_r^r + \|(\bar{x}_W - x)\mathbb{I}_W\|_r^r), \\
\|\Psi_V(x)\|_r^r &\leq \max\{1, 2^{r-1}\} (\|\bar{x}_V - x\|_r^r + \|x\|_r^r).
\end{aligned} \tag{21}$$

(i) In the convex case $1 \leq r < \infty$, we can continue with Minkowski's inequality and get

$$\begin{aligned}
\|(\bar{x}_{W'} - x)\mathbb{I}_{W'}\|_r &= \left(\sum_{w \in W'} \left| \frac{1}{|W'|} \sum_{v \in W'} (x(v) - x(w)) \right|^r \right)^{\frac{1}{r}} \\
&\leq \frac{1}{|W'|} \sum_{v \in W'} \left(\sum_{w \in W'} |x(v) - x(w)|^r \right)^{\frac{1}{r}} \leq \left(\sup_{v \in W'} \sum_{w \in W'} |x(v) - x(w)|^r \right)^{\frac{1}{r}}
\end{aligned}$$

(ii) In the case $0 < r < 1$, we will show the same qualitative bound (with an additional factor 2) using an additional argumentation based on the subadditivity of the concave function t^r . For this, we select the nodes $w_0, w_1 \in W'$ such that $x(w_0) = \min_{w \in W'} x(w)$ and $x(w_1) = \max_{w \in W'} x(w)$. Then,

$$|\bar{x}_{W'} - x(w)|^r \leq |x(w_1) - x(w_0)|^r$$

holds true for all $w \in W'$. By the subadditivity of the function t^r , $0 < r < 1$, we get

$$|x(w_1) - x(w_0)|^r \leq |x(w_1) - x(w)|^r + |x(w_0) - x(w)|^r, \quad w \in W'.$$

This implies now that

$$\begin{aligned}
\|(\bar{x}_{W'} - x)\mathbb{I}_{W'}\|_r^r &= \sum_{w \in W'} |\bar{x}_{W'} - x(w)|^r \\
&\leq \sum_{w \in W'} |x(w_1) - x(w)|^r + \sum_{w \in W'} |x(w_0) - x(w)|^r \\
&\leq 2 \sup_{v \in W'} \sum_{w \in W'} |x(v) - x(w)|^r.
\end{aligned}$$

Combining (i) and (ii), we get now for $C = 3^{\frac{1}{r}} \max\{2^{\frac{1}{r}}, 2\}$ the aimed at bound

$$\begin{aligned}
\mathcal{N}_r(x, \mathcal{T}) &= \left(\sum_{W \in \mathcal{T}} \|\Psi_W(x)\|_2^r \right)^{\frac{1}{r}} = \left(\sum_{W \in \mathcal{T}} |W|^{-\alpha r} \|\Psi_W(x)\|_r^r \right)^{\frac{1}{r}} \\
&\leq 2^{-\frac{1}{r}} C \left(|V|^{-\alpha r} \|x\|_r^r + \sum_{W \in \mathcal{T}} |W|^{-\alpha r} \|(\bar{x}_W - x)\mathbb{I}_W\|_r^r \right)^{\frac{1}{r}} \\
&\leq C \left(|V|^{-\alpha r} \|x\|_r^r + \sum_{W \in \mathcal{T}} |W|^{-\alpha r} \sup_{v \in W'} \sum_{w \in W'} |x(v) - x(w)|^r \right)^{\frac{1}{r}} \\
&= C \mathcal{M}_r(x, \mathcal{T}).
\end{aligned}$$

For the identity in the first line, we used equation (20) and the relation $\alpha = 1/r - 1/2$. For the first inequality, we used the estimates in (21), resulting in the constant $2^{-\frac{1}{r}}C = 3^{\frac{1}{r}} \max\{1, 2^{1-\frac{1}{r}}\}$ depending only on the parameter r . Finally, the last inequality follows from the argumentation lines in (i) and (ii). $\square$

For a restricted range $0 < r < 2$, and balanced BGP(ρ)-trees, the r -energy $\mathcal{N}_r(x, \mathcal{T})$ and the smoothness measure $\mathcal{M}_r^\alpha(x, \mathcal{T})$ are in fact equivalent if α and r are related by $\alpha = 1/r - 1/2$.

Theorem 7.12 *Let $0 < r < 2$, $\alpha = \frac{1}{r} - \frac{1}{2} > 0$ and $\mathcal{T}$ a BGP(ρ) tree with $\frac{1}{2} \leq \rho < 1$. Then, we have the inequalities*

$$C_1 \mathcal{N}_r(x, \mathcal{T}) \leq \mathcal{M}_r^\alpha(x, \mathcal{T}) \leq C_2 \mathcal{N}_r(x, \mathcal{T}). \quad (22)$$

Further, if $\mathcal{T}_r(x)$ is a near best BGP(ρ) tree, we get the equivalences

$$C'_1 \mathcal{N}_r(x, \mathcal{T}_r(x)) \leq |x|_{\mathcal{G}_{\mathcal{B}}^\alpha} \leq C'_2 \mathcal{N}_r(x, \mathcal{T}_r(x)). \quad (23)$$

The positive constants C_1, C'_1 depend only on r , while C_2, C'_2 depend on r and ρ .

Proof. The left hand inequalities in (22) and (23) follow directly from Proposition 7.11 with the constants $C_1 = 1/C$ and $C'_1 = 1/C'$. Therefore, we only have to show the right hand side of (22). The right hand inequality in (23) follows directly from the right hand inequality in (22)

(i) We prove the right hand inequality in (22) first for the case $0 < r \leq 1$. Based on the expansion of x in its wavelet components, $x = \sum_{W \in \mathcal{T}} \Psi_W(x)$, and the fact that the components $\Psi_W(x)$ are piecewise constant, we get

$$\begin{aligned} \sup_{w \in W} \sum_{v \in W} |x(v) - x(w)|^r &= \sup_{w \in W} \sum_{v \in W} \left| \sum_{W' \in \mathcal{T}, W' \subseteq W} \Psi_{W'}(x)(v) - \Psi_{W'}(x)(w) \right|^r \\ &\leq 2 \sum_{v \in W} \left(\sum_{W' \in \mathcal{T}, W' \subseteq W} |\Psi_{W'}(x)(v)| \right)^r \\ &\leq 2 \sum_{W' \in \mathcal{T}, W' \subseteq W} \|\Psi_{W'}(x)\|_r^r. \end{aligned} \quad (24)$$

Here, the second inequality follows from the r -triangle inequality, valid for $0 < r \leq 1$, and by the identity (20). Further, using the same arguments, we also have

$$\|x\|_r^r \leq \sum_{v \in W} \left(\sum_{W' \in \mathcal{T}, W' \subseteq W} |\Psi_{W'}(x)(v)| \right)^r \leq \sum_{W' \in \mathcal{T}, W' \subseteq W} \|\Psi_{W'}(x)\|_r^r. \quad (25)$$

With the inequalities (24) and (25) at hand, we thus get for $\mathcal{M}_r^\alpha(x, \mathcal{T})$ the bound

$$\begin{aligned}
\mathcal{M}_r^\alpha(x, \mathcal{T})^r &\leq 3 \sum_{W \in \mathcal{T}} |W|^{-\alpha r} \sum_{W' \in \mathcal{T}, W' \subseteq W} \|\Psi_{W'}(x)\|_r^r \\
&= 3 \sum_{W \in \mathcal{T}} \sum_{W' \subseteq W} \left(\frac{|W'|}{|W|} \right)^{\alpha r} |W'|^{-\alpha r} \|\Psi_{W'}(x)\|_r^r \\
&= 3 \sum_{W' \in \mathcal{T}} |W'|^{-\alpha r} \|\Psi_{W'}(x)\|_r^r \sum_{W \supseteq W'} \left(\frac{|W'|}{|W|} \right)^{\alpha r}.
\end{aligned}$$

The balancedness of $\mathcal{T}$ implies that $|W'| \leq \rho|W|$ for every parent W of W' , and thus $|W'| \leq \rho^k |W^{(k)}|$ for every parent $W^{(k)}$ in the k -th generation before W' . Thus,

$$\sum_{W \supseteq W'} \left(\frac{|W'|}{|W|} \right)^{\alpha r} \leq \sum_{k=0}^{\infty} \rho^{k\alpha r} \leq \frac{1}{1 - \rho^{\alpha r}}, \quad (26)$$

and we can conclude that

$$\begin{aligned}
\mathcal{M}_r^\alpha(x, \mathcal{T})^r &\leq \frac{3}{1 - \rho^{\alpha r}} \sum_{W' \in \mathcal{T}} |W'|^{-\alpha r} \|\Psi_{W'}(x)\|_r^r \\
&\leq C_2^r \sum_{W' \in \mathcal{T}} \|\Psi_{W'}(x)\|_2^r = C_2^r \mathcal{N}_r(x, \mathcal{T})^r,
\end{aligned}$$

with the constant $C_2 = \left(\frac{3}{1 - \rho^{\alpha r}} \right)^{\frac{1}{r}}$ that depends only on ρ and r .

(ii) In the convex case $1 < r < 2$, a slightly different argumentation based on Hölder's inequality is necessary. In this case, in (24) and (25) only the first of the two inequalities holds true. Nevertheless, inserting them in the definition of $\mathcal{M}_r^\alpha(x, \mathcal{T})$, we get

$$\begin{aligned}
\mathcal{M}_r^\alpha(x, \mathcal{T})^r &\leq 3 \sum_{W \in \mathcal{T}} |W|^{-\alpha r} \sum_{v \in W} \left(\sum_{W' \in \mathcal{T}, W' \subseteq W} |\Psi_{W'}(x)(v)| \right)^r \\
&= 3 \sum_{v \in W} \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} |W|^{-\alpha} \|\Psi_{W'}(x)\|_\infty \right)^r \\
&= 3 \sum_{v \in W} \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} \left(\frac{|W'|}{|W|} \right)^\alpha |W'|^{-\alpha} \|\Psi_{W'}(x)\|_\infty \right)^r.
\end{aligned}$$

We apply now Hölder's inequality to the interior sum using the dual parameters r and $r' = \frac{r}{r-1}$. This gives

$$\begin{aligned}
\mathcal{M}_r^\alpha(x, \mathcal{T})^r &\leq 3 \sum_{v \in W} \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} |W'|^{-\alpha r} \|\Psi_{W'}(x)\|_\infty^r \right) \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} \left(\frac{|W'|}{|W|} \right)^{\frac{\alpha r}{r-1}} \right)^{r-1} \\
&\leq 3 \sum_{v \in W} \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} |W'|^{-\alpha r} \|\Psi_{W'}(x)\|_\infty^r \right) \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} \left(\frac{|W'|}{|W|} \right)^{\alpha r} \right).
\end{aligned}$$

Applying (26), we can conclude our estimate of $\mathcal{M}_r^\alpha(x, \mathcal{T})$ and obtain

$$\begin{aligned}
\mathcal{M}_r^\alpha(x, \mathcal{T})^r &\leq \frac{3}{1 - \rho^{\alpha r}} \sum_{v \in W} \left(\sum_{\substack{W' \in \mathcal{T} \\ \{v\} \subseteq W' \subseteq W}} |W'|^{-\alpha r} \|\Psi_{W'}(x)\|_\infty^r \right) \\
&= \frac{3}{1 - \rho^{\alpha r}} \left(\sum_{W' \in \mathcal{T}, W' \subseteq W} |W'|^{-\alpha r + 1} \|\Psi_{W'}(x)\|_\infty^r \right) \\
&= \frac{3}{1 - \rho^{\alpha r}} \left(\sum_{W' \in \mathcal{T}, W' \subseteq W} \|\Psi_{W'}(x)\|_2^r \right) = C_2^r \mathcal{N}_r(x, \mathcal{T})^r.
\end{aligned}$$

Thus, we get also in the convex case $1 < r < 2$ the same upper estimate as in (i) with the same constant C_2 that depends only on r . $\square$

As a further application of Proposition 7.11, we get the following Jackson-type estimate. It states that if x is smooth with respect to the Besov-type smoothness measure given in Definition 7.9, it suffices to find a near best BGP(ρ) tree to obtain the following m -term approximation rates in terms of wavelet coefficients.

Theorem 7.13 (Jackson estimate) *Let $\alpha > 0$ and $0 < r < 2$ be related by $\alpha = 1/r - 1/2$. Then, for a graph signal $x \in \mathcal{L}(V)$ and geometric wavelets with respect to a near best BGP(ρ) tree $\mathcal{T}_r(x)$, we obtain the m -term approximation error*

$$\|x - \mathcal{S}_m(x)\|_2 \leq C m^{-\alpha} |x|_{\mathcal{G}_{\mathcal{B}_r^\alpha}}$$

with a constant $C > 0$ that depends only on r and ρ .

Proof. The subsequent proof is an adaptation of the techniques developed for the derivation of [40, Theorem 3.4] in which a Jackson estimate for piecewise polynomial approximation on nested triangulations in $\mathbb{R}^2$ is formulated. The same techniques have been used in [17, Theorem 3.6] to derive a respective Jackson estimate for geometric wavelets with respect to binary space partitionings of convex domains.

To subdivide the geometric wavelets of the tree $\mathcal{T}_r(x)$ into dyadic blocks, we define for $\mu \in \mathbb{Z}$ the index sets

$$I_\mu := \left\{ i \mid \frac{\mathcal{N}_r(x, \mathcal{T}_r(x))}{2^\mu} \leq \|\Psi_{W_i}(x)\|_2 < \frac{\mathcal{N}_r(x, \mathcal{T}_r(x))}{2^{\mu-1}} \right\}.$$

For the union of these index sets, we get

$$\bigcup_{\nu \leq \mu} I_\nu = \{i \mid \|\Psi_{W_i}(x)\|_2 \geq 2^{-\mu} \mathcal{N}_r(x, \mathcal{T}_r(x))\}.$$

Thus, by Definition 7.8 of the r -energy $\mathcal{N}_r(x, \mathcal{T}_r(x))$, we have

$$|I_\mu| \leq \sum_{\nu \leq \mu} |I_\nu| = \left| \bigcup_{\nu \leq \mu} I_\nu \right| \leq 2^{\mu r}.$$

Now, setting $m := \sum_{\nu \leq \mu} |I_\nu|$, we obtain the estimate

$$\|x - \mathcal{S}_m(x)\|_2 \leq \left\| \sum_{\nu > \mu} \sum_{i \in I_\nu} |\Psi_{W_i}(x)| \right\|_2 \leq \sum_{\nu > \mu} \left\| \sum_{i \in I_\nu} |\Psi_{W_i}(x)| \right\|_2.$$

The wavelets $\Psi_{W_i}(x)$ are constant on the sets W_i , thus $\|\Psi_{W_i}(x)\|_2 = \|\Psi_{W_i}(x)\|_\infty |W_i|^{1/2}$. This together with the definition of the index sets I_ν yields the bound

$$\begin{aligned} \|x - \mathcal{S}_m(x)\|_2 &\leq \sum_{\nu > \mu} \left\| \sum_{i \in I_\nu} \|\Psi_{W_i}(x)\|_\infty \mathbb{1}_{W_i} \right\|_2 = \sum_{\nu > \mu} \left\| \sum_{i \in I_\nu} \frac{\|\Psi_{W_i}(x)\|_2}{|W_i|^{1/2}} \mathbb{1}_{W_i} \right\|_2 \\ &\leq \sum_{\nu > \mu} 2^{-\nu+1} \mathcal{N}_r(x, \mathcal{T}_r(x)) \left\| \sum_{i \in I_\nu} \frac{\mathbb{1}_{W_i}}{|W_i|^{1/2}} \right\|_2 \\ &= \sum_{\nu > \mu} 2^{-\nu+1} \mathcal{N}_r(x, \mathcal{T}_r(x)) |I_\nu|^{1/2}. \end{aligned}$$

Combining this estimate and the bound on the complexity $|I_\nu|$, we can conclude that

$$\begin{aligned} \|x - \mathcal{S}_m(x)\|_2 &\leq \sum_{\nu > \mu} 2^{-\nu} \mathcal{N}_r(x, \mathcal{T}_r(x)) |I_\nu|^{1/2} \\ &\leq \mathcal{N}_r(x, \mathcal{T}_r(x)) \sum_{\nu > \mu} 2^{-\nu(1-r/2)} \\ &= C' \mathcal{N}_r(x, \mathcal{T}_r(x)) 2^{-\mu(1-r/2)} \\ &\leq C' m^{-1/(r+1/2)} \mathcal{N}_r(x, \mathcal{T}_r(x)) \\ &= C' m^{-\alpha} \mathcal{N}_r(x, \mathcal{T}_r(x)), \end{aligned}$$

with $C' = (1 - 2^{r/2-1})^{-1}$. Now, the statement follows from Proposition 7.11. $\square$

Both, Theorem 7.12 and 7.13 are discrete versions of respective continuous results proven for geometric wavelets on binary space partitionings [17] and piecewise polynomial approximation on nested triangulations [40]. We note that the results of this section hold true not only for graphs but also more generally for finite metric spaces.

Proposition 7.6 (iii) implies that for a complete BGP tree $\mathcal{T}$ we have the identity

$$\|x\|_2 = \left(\sum_{W \in \mathcal{T}} \|\Psi_W(x)\|_2^2 \right)^{\frac{1}{2}} = \mathcal{N}_2(x, \mathcal{T}).$$

The just proven Theorem 7.13 provides us with the following additional bound of the $\mathcal{L}^2(V)$ -norm in terms of $\mathcal{N}_r(x, \mathcal{T})$ and $0 < r < 2$.

Corollary 7.14 *Let G be a graph with n nodes, and $\mathcal{T}$ a BGP(ρ) tree on G , i.e., $\mathcal{T}$ is complete and balanced. Then, for $0 < r < 2$, we have*

$$\|x\|_2 \leq C \mathcal{N}_r(x, \mathcal{T})$$

with a constant $C > 0$ depending only on r .

Proof. In the proof of Theorem 7.13, we have seen that for every BGP(ρ) tree $\mathcal{T}$ the inequality

$$\|x - \mathcal{S}_m(x)\|_2 \leq C m^{-\alpha} \mathcal{N}_r(x, \mathcal{T})$$

holds true with a constant $C > 0$ that depends only on r . This implies that

$$\begin{aligned} \|x\|_2 &\leq \|x - \mathcal{S}_1(x)\|_2 + \|\mathcal{S}_1(x)\|_2 \leq C \mathcal{N}_r(x, \mathcal{T}) + \|\psi_{W_1}(x)\|_2 \\ &\leq C \mathcal{N}_r(x, \mathcal{T}) + \|\psi_{W_1}(x)\|_r \leq (C + 1) \mathcal{N}_r(x, \mathcal{T}), \end{aligned}$$

and, thus, the given statement. $\square$

7.3 Advanced basis systems based on BGPs

In addition to the Haar wavelets ψ_m , the hierarchical partitioning of graphs based on a BGP tree $\mathcal{T}$ allows to introduce more refined basis functions suited to provide orthonormal basis systems on the subsets $V_k^{(m)}$ of the partitions $\mathcal{P}^{(m)}$. For this, we denote by

$$\mathcal{V}_k^{(m)} = \{x \in \mathcal{L}(V) : x(v) = 0 \text{ for } v \in V \setminus V_k^{(m)}\}, \quad m \in \{1, \dots, n\}, \quad k \in \{1, \dots, m\},$$

the space of graph signals with support in $V_k^{(m)}$. By the construction of a BGP tree $\mathcal{T}$, we have the orthogonal decomposition

$$\mathcal{L}^2(V) = \bigoplus_{k=1}^m \mathcal{V}_k^{(m)}, \quad m \in \{1, \dots, n\}. \quad (27)$$

Hierarchical graph Laplacian eigenbasis. A very simple way to obtain an orthonormal basis for the spaces $\mathcal{V}_k^{(m)}$ is by considering the subgraph $G_k^{(m)}$ of G built upon the node set $V_k^{(m)}$. If we restrict the graph Laplacian $\mathbf{L}$ to this node set, we obtain a reduced

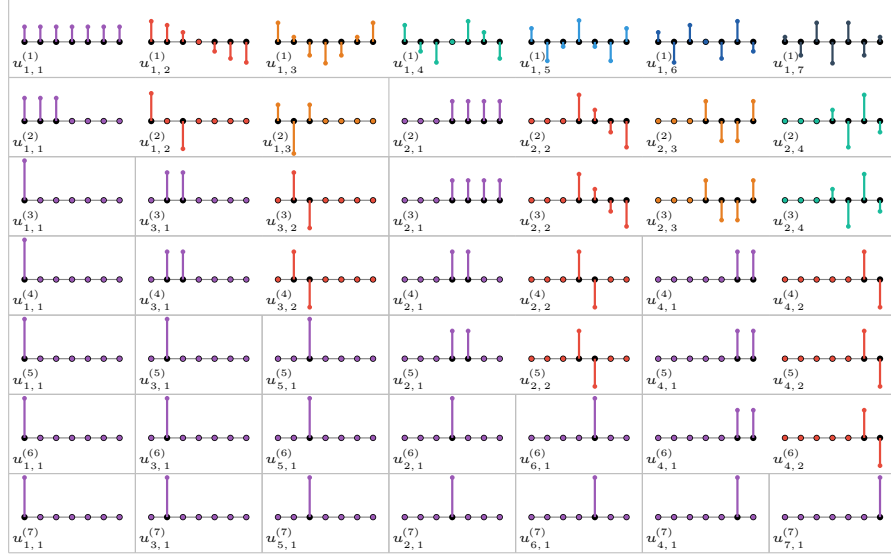


Fig. 17 Hierarchical graph Laplacian eigenbasis $u_{k,i}^{(m)}$ on the path graph P_7 . The partitions of the graph are based on the BGP tree in Fig. 14. The first row contains the Fourier basis of the graph P_7 .

Laplacian matrix $\mathbf{L}^{(V_k^m)}$ for the subgraph $G_k^{(m)}$ which is symmetric and positive semi-definite. This allows us to calculate an orthonormal eigenbasis

$$\left\{ u_{k,1}^{(m)}, \dots, u_{k,|V_k^{(m)}|}^{(m)} \right\}$$

of the Laplacian $\mathbf{L}^{(V_k^m)}$ which is ordered in terms of the increasing eigenvalues of $\mathbf{L}^{(V_k^m)}$ and consists of $|V_k^{(m)}|$ elements. To keep the notation simple, we will regard $u_{k,i}^{(m)}$ also as functions in $\mathcal{L}(V)$ by extending them to the entire node set V by zero. In this sense, the functions $\{u_{k,i}^{(m)}\}$ form an orthonormal basis for the space $\mathcal{V}_k^{(m)}$ and we obtain the dictionary of functions

$$\left\{ \left\{ u_{1,i}^{(m)} \right\}_{i=1}^{|V_1^{(m)}|}, \dots, \left\{ u_{m,i}^{(m)} \right\}_{i=1}^{|V_m^{(m)}|} \right\}, \quad m \in \{1, \dots, n\}. \quad (28)$$

By construction, for any fixed $m \in \{1, \dots, n\}$ this system provides an orthonormal basis for $\mathcal{L}^2(V)$. This follows by the orthonormal decomposition in (27) and the fact that the functions $\{u_{k,i}^{(m)}\}_{i=1}^{|V_k^{(m)}|}$ form a basis for the space $\mathcal{V}_k^{(m)}$. These basis systems are referred to as **hierarchical graph Laplacian eigenfunctions**, see [36, 38]. An example for the path graph P_7 is given in Fig. 17.

In many scenarios, it can be advantageous to replace the Fourier basis with the localized basis system $\{u_{k,i}^{(m)}\}$ for a fixed $m \geq 1$. When the partition size m is chosen large

enough, this basis has an improved spatial localization compared to the global graph Fourier basis. Moreover, if the subgraphs $G_k^{(m)}$ are of moderate size, the computational cost of calculating the corresponding Laplacian eigenbases can be significantly reduced.

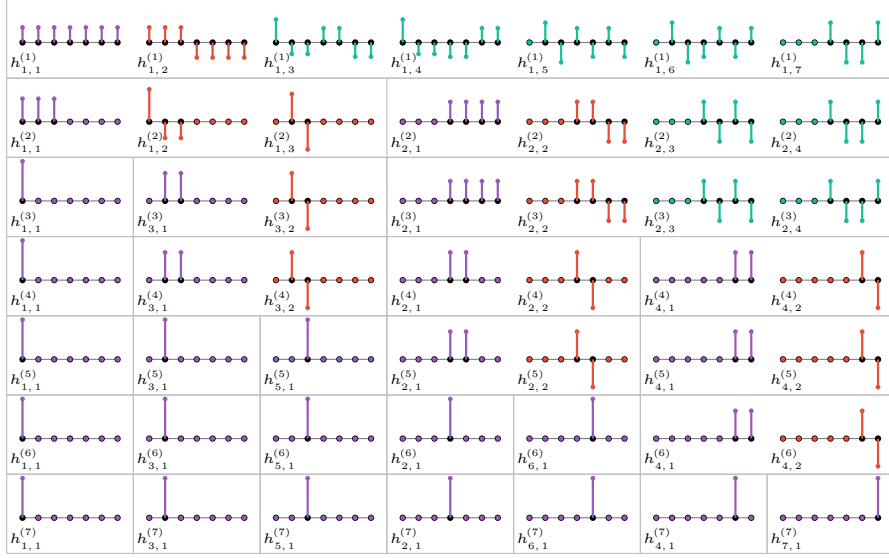


Fig. 18 Haar-Walsh basis systems $h_{k,i}^{(m)}$ on the path graph P_7 . The partitions of the graph are based on the BGP tree in Fig. 14. Haar scaling functions are indicated with purple color, while red color indicates Haar wavelets. Green color indicates more complex Haar-Walsh basis functions.

Haar-Walsh basis systems based on BGPs. A global alternative to the locally defined Haar wavelets is given by the **Haar-Walsh basis systems**

$$h_{k,i}^{(m)}, \quad i \in \{1, \dots, |V_k^{(m)}|\}, \quad k \in \{1, \dots, m\}, \quad m \in \{1, \dots, n\},$$

introduced in [37, 52]. For this dictionary, we use the same indexing as before for the hierarchical graph Laplacian eigenfunctions. Similarly, this system will be constructed in such a way that for any fixed m the corresponding Haar-Walsh basis functions provide an orthonormal basis for $\mathcal{L}^2(V)$. Moreover, the subsystem $\{h_{k,i}^{(m)}\}_{i=1}^{|V_k^{(m)}|}$ forms an orthonormal basis for the subspace $\mathcal{V}_k^{(m)}$. This dictionary is constructed recursively using the following bottom-up strategy:

- 1) The first basis element

$$h_{k,1}^{(m)}(v) = \frac{1}{|V_k^{(m)}|^{1/2}} \omega_k^{(m)}(v)$$

for each space $\mathcal{V}_k^{(m)}$ consists of the normalized Haar scaling function. This defines already the canonical basis elements for the one-dimensional spaces $\mathcal{V}_k^{(n)}$ whose elements are supported on a single node of the graph.

2) We assume that for the partition $\mathcal{P}^{(m+1)}$ all basis functions $h_{k,i}^{(m+1)}$ have been calculated. The Haar-Walsh basis functions $h_{k,i}^{(m)}$ for the sets $V_k^{(m)}$ in the partition $\mathcal{P}^{(m)}$ can then be calculated as follows:

- a) If $k \neq j_m$, we set $h_{k,i}^{(m)} := h_{k,i}^{(m+1)}$.
- b) If $k = j_m$, the set $V_{j_m}^{(m)}$ is the disjoint union of $V_{j_m}^{(m+1)}$ and $V_{m+1}^{(m+1)}$. The functions $h_{j_m,i}^{(m)}$ are calculated from the functions $h_{j_m,i}^{(m+1)}$ and $h_{m+1,i}^{(m+1)}$ as follows:
 - The first element $h_{j_m,1}^{(m)}(v) = \frac{1}{|V_{j_m}^{(m)}|^{1/2}} \omega_{j_m}^{(m)}(v)$ is given as the normalized Haar scaling function;
 - The second element $h_{j_m,2}^{(m)}(v) = \frac{1}{\|\psi_{m+1}\|_2} \psi_{m+1}(v)$ is given as the $m+1$ -th. normalized Haar wavelet;
 - Let $M = \min\{|V_{j_m}^{(m+1)}|, |V_{m+1}^{(m+1)}|\}$. For all $i \in \{2, \dots, M\}$, we set

$$\begin{aligned} h_{j_m,2i-1}^{(m)} &= \frac{1}{\sqrt{2}} h_{j_m,i}^{(m+1)} + \frac{1}{\sqrt{2}} h_{m+1,i}^{(m+1)}, \\ h_{j_m,2i}^{(m)} &= \frac{1}{\sqrt{2}} h_{j_m,i}^{(m+1)} - \frac{1}{\sqrt{2}} h_{m+1,i}^{(m+1)}. \end{aligned}$$

If $V_{m+1}^{(m+1)}$ is the set with the smaller number M of elements, we further set

$$h_{j_m,M+i}^{(m)}(v) = h_{j_m,i}^{(m+1)}(v), \quad \text{for } i \in \{M+1, \dots, |V_{j_m}^{(m+1)}|\},$$

and otherwise

$$h_{j_m,M+i}^{(m)}(v) = h_{m+1,i}^{(m+1)}(v), \quad \text{for } i \in \{M+1, \dots, |V_{m+1}^{(m+1)}|\}.$$

This particular construction guarantees that the dictionary

$$\left\{ \left\{ h_{1,i}^{(m)} \right\}_{i=1}^{|V_1^{(m)}|}, \dots, \left\{ h_{m,i}^{(m)} \right\}_{i=1}^{|V_m^{(m)}|} \right\}, \quad m \in \{1, \dots, n\}. \quad (29)$$

is an orthonormal basis for the space $\mathcal{L}^2(V)$ for any fixed $m \in \{1, \dots, n\}$. Moreover, the subspace $\mathcal{V}_k^{(m)}$ is spanned by the orthonormal system $\{h_{k,i}^{(m)}\}_{i=1}^{|V_k^{(m)}|}$. An example of a Haar-Walsh dictionary for the path P_7 is visualized in Fig. 18.

Utilizing the partitioning structure encoded in the BGP tree, Haar-Walsh systems usually provide computationally more efficient and more localized decompositions of graph signals than the graph Fourier basis. This makes such systems particularly interesting for large graphs. On the other hand, Haar-Walsh basis elements generally do not admit the same direct frequency interpretation as Laplacian eigenvectors.

8 Graph wedgelets on adaptive graph partitionings

8.1 Wedge splits and binary wedge partitioning (BWP) trees



Fig. 19 An adaptive BWP tree based on hierarchical wedge splits of the Minnesota graph for the adaptive approximation of the test signal x_2 illustrated in Fig. 20 (right).

After the general introduction of BGP trees and Haar wavelets, we are interested in finding explicit constructions of **adaptive partitioning trees** on graphs that allow to incorporate the structure of a graph signal x into the partitioning tree. For an efficient implementation and storing of such signal-adaptive trees we require a simple splitting strategy on graphs that can be adapted to the geometry of the signal. For this, we propose as building elements the following elementary wedge splits.

Definition 8.1 We call a dyadic partition $\{V', V''\}$ of the vertex set V a **wedge split** of V if there exist two distinct nodes v' and v'' of V such that V' and V'' have the form

$$\begin{aligned} V' &= \{v \in V \mid d(v, v') \leq d(v, v'')\}, \\ V'' &= \{v \in V \mid d(v, v') > d(v, v'')\}. \end{aligned}$$

As a distance metric $d(v, v')$ for the graph nodes, we will generally use the graph distance based on the length of the shortest path connecting the two nodes v and v' .

Nevertheless, also the usage of other metrics on the vertex set V can principally be taken into consideration. A key advantage of the just defined wedge splits is that they can be encoded very compactly in terms of the two nodes v' in V' and v'' in V'' . They have the following basic properties:

Lemma 8.2 *If $\{V', V''\}$ is a wedge split of V based on the graph distance d , then*

- (i) V' and V'' are uniquely determined by v' and v'' .
- (ii) $V' \cap V'' = \emptyset$ and $V' \cup V'' = V$.
- (iii) If V is connected, then also V' and V'' are connected subsets of the graph G .

While the first two properties follow immediately from the Definition 8.1, the third property is a consequence of the fact that d is the shortest path distance on G .

Using wedge splits, we can define **elementary wedgelets** $\omega_{(v', v'')}^+$ and $\omega_{(v', v'')}^-$ as the indicator functions of the children sets V' and V'' , i.e.,

$$\begin{aligned}\omega_{(v', v'')}^+(v) &= \mathbb{1}_{V'}(v) = \begin{cases} 1, & \text{if } d(v, v') \leq d(v, v''), \\ 0, & \text{otherwise,} \end{cases} \\ \omega_{(v', v'')}^-(v) &= \mathbb{1}_{V''}(v) = \begin{cases} 1, & \text{if } d(v, v') > d(v, v''), \\ 0, & \text{otherwise.} \end{cases}\end{aligned}$$

Using wedge splits, we can now generate the following special BGP trees:

Definition 8.3 A **binary wedge partitioning (BWP) tree** $\mathcal{T}_Q$ of G with respect to the ordered set $Q = \{q_1, \dots, q_M\} \subset V$ is a BGP tree constructed recursively as follows:

1. The root of $\mathcal{T}_Q$ is the set V . It forms the trivial partition $\mathcal{P}^{(1)} = \{V_{q_1}^{(1)}\} = \{V\}$ and is associated to the first node $q_1 \in Q$.
2. For $m \in \{2, \dots, M\}$, the elements of the partition $\mathcal{P}^{(m)} = \{V_{q_1}^{(m)}, \dots, V_{q_m}^{(m)}\}$ of V are associated to nodes $q_i \in V_{q_i}^{(m)}$, $i \in \{1, \dots, m\}$. For $m < M$, a selection rule $j = j_m \in \{1, \dots, m\}$ determines the next set $V_{q_j}^{(m)}$ of size $|V_{q_j}^{(m)}| \geq 2$ to be split.
3. Starting from $\mathcal{P}^{(m)}$ and the selected index $j = j_m$, the set $V_{q_j}^{(m)}$ is divided by a wedge split based on the node q_j and the new node $q_{m+1} \in V_{q_j}^{(m)}$. The resulting children of $V_{q_j}^{(m)}$ are denoted as $V_{(q_j, q_{m+1})}^{(m)+}$ (contains q_j) and $V_{(q_j, q_{m+1})}^{(m)-}$ (with q_{m+1}). The new partition

$$\mathcal{P}^{(m+1)} = \{V_{q_1}^{(m+1)}, \dots, V_{q_{m+1}}^{(m+1)}\}$$

is obtained by setting $V_{q_i}^{(m+1)} = V_{q_i}^{(m)}$ if $i \neq \{j, m+1\}$, $V_{q_j}^{(m+1)} = V_{(q_j, q_{m+1})}^{(m)+}$ and $V_{q_{m+1}}^{(m+1)} = V_{(q_j, q_{m+1})}^{(m)-}$.

A BWP tree $\mathcal{T}_Q$ as introduced in Definition 8.3 is uniquely determined by the ordered node set Q . This allows to store $\mathcal{T}_Q$ compactly in terms of the M nodes of Q .

Different choices for the centers in Q represent the degrees of freedom of the partitioning tree and enable the iterative construction of adaptive refinements. More precisely, in a top-down construction of the BWP tree, each refinement step requires the adaptive selection of a node $q_j \in Q$ and the creation of a new node q_{m+1} .

In the following, we summarize several properties of BWP trees that follow directly from Definition 8.3, Proposition 7.6 and the general Definition 7.1 for BGPs.

Proposition 8.4 *Let $\mathcal{T}_Q$ be a BWP tree determined by the ordered set $Q = \{q_1, \dots, q_M\}$.*

- (i) *A BWP tree $\mathcal{T}_Q$ contains $2M - 1$ elements: 1 root and $2M - 2$ children.*
- (ii) *The M leaves of the binary tree $\mathcal{T}_Q$ are given by the elements of the M -th. partition $\mathcal{P}^{(M)} = \{V_{q_1}^{(M)}, \dots, V_{q_M}^{(M)}\}$.*
- (iii) *A BWP tree $\mathcal{T}_Q$ is complete if and only if $|Q| = |V|$.*
- (iv) *A BWP tree $\mathcal{T}_Q$ is balanced with $\frac{1}{2} \leq \rho \leq \frac{n-1}{n}$.*
- (v) *The characteristic function of the subset $V_{q_i}^{(m)}$ can be written as a product of m elementary wedgelets $\omega_{(q_i, q_j)}^\pm$, with $i, j \in \{1, \dots, m\}$, $i < j$.*

Definition 8.5 The characteristic functions

$$\omega_{q_i}^{(m)}(v) = \mathbb{I}_{V_{q_i}^{(m)}}(v), \quad 1 \leq i \leq m, \quad 1 \leq m \leq M,$$

of the sets $V_{q_i}^{(m)}$ will be referred to as **wedgelets** with respect to the BWP tree $\mathcal{T}_Q$. As particular Haar scaling functions, the wedgelets $\{\omega_{q_i}^{(m)} : i \in \{1, \dots, m\}\}$ form an orthogonal basis for the piecewise constant functions on the partition $\mathcal{P}^{(m)}$.

Remark 8.6 One can define more general families of BWP trees than those introduced in Definition 8.3 and, respectively, larger classes of wedgelets than in Definition 8.5, by allowing wedge splits inside the subsets that are not associated with the centers q_i . However, in this more general setting, the tree can no longer be represented only by a simple ordered node set Q . It leads also to a larger computational cost in the selection of a proper wedge split when calculating the adaptive tree.

8.2 Signal-adaptive greedy generation of BWP trees

To construct BWP trees $\mathcal{T}_Q = \mathcal{T}_Q(x)$ that are adapted to the structure of a given graph signal x , each refinement step requires two fundamental decisions. At each partition level m , one must first select a set $V_{q_j}^{(m)}$, $j \in \{1, \dots, m\}$, to be refined. Subsequently, a new node $q_{m+1} \in V_{q_j}^{(m)}$ must be chosen in order to perform the corresponding elementary wedge split of $V_{q_j}^{(m)}$. In general, both selections can be carried out in an x -adapted way or independently of the signal x . As adaptive refinement strategies, we consider the following three greedy methods:

Max-distance (MD) greedy wedge splitting. At stage m , the domain $V_{q_j}^{(m)}$ is chosen x -adaptively by selecting $j = j_m$ such that

$$j = \operatorname{argmax}_{i \in \{1, \dots, m\}} \left\| (x - \bar{x}_{V_{q_i}^{(m)}}) \mathbb{I}_{V_{q_i}^{(m)}} \right\|_2, \quad (30)$$

where

$$\bar{x}_{V_{q_i}^{(m)}} = \frac{\langle x, \omega_{q_i}^{(m)} \rangle}{|V_{q_i}^{(m)}|} = \frac{1}{|V_{q_i}^{(m)}|} \sum_{v \in V_{q_i}^{(m)}} x(v)$$

is the mean value of x over the set $V_{q_i}^{(m)}$. This first selection rule in (30) ensures that the chosen set $V_{q_j}^{(m)}$ is the one with the maximal $\mathcal{L}^2$ -error. As soon as j , or equivalently, the node q_j are determined, a non-adaptive way to choose the subsequent node q_{m+1} is by the selection rule

$$q_{m+1} = \arg \max_{v \in V_{q_j}^{(m)}} d(q_j, v),$$

i.e., by choosing q_{m+1} to be the vertex in $V_{q_j}^{(m)}$ furthest away from q_j . This choice and the corresponding split can be interpreted as a 2-center clustering of $V_{q_j}^{(m)}$ in which the first node q_j is fixed (see also [4, 31]). An advantage of this selection rule is its simplicity and computational efficiency, as it requires only a single node to be determined, namely the one with maximal distance. However, with this rule only the choice of the set $V_{q_j}^{(m)}$ is adaptive, the wedge split itself remains non-adaptive.

Fully-adaptive (FA) greedy wedge splitting. In the FA-greedy strategy, the subset to be split is selected according to (30), but also the node q_{m+1} determining the wedge split is chosen according to an adaptive rule. If $\{V_{(q_j, q)}^{(m)+}, V_{(q_j, q)}^{(m)-}\}$ denotes the partition of $V_{q_j}^{(m)}$ for the wedge split given by q_j and a second node q , we choose q_{m+1} such that

$$\left\| (x - \bar{x}_{V_{(q_j, q)}^{(m)+}}) \mathbb{I}_{V_{(q_j, q)}^{(m)+}} \right\|_2^2 + \left\| (x - \bar{x}_{V_{(q_j, q)}^{(m)-}}) \mathbb{I}_{V_{(q_j, q)}^{(m)-}} \right\|_2^2 \quad (31)$$

is minimized over all $q \in V_{q_j}^{(m)}$. Compared to the semi-adaptive MD-greedy strategy, the FA-greedy method is computationally more expensive. On the other hand, as the wedge splits are now adapted to the particular form of the underlying function x , we expect a better approximation behavior for the FA-greedy scheme. This will be confirmed in the subsequent numerical experiments.

Randomized (R) greedy wedge splitting. If the size of the sets $V_{q_j}^{(m)}$ is large it might be too time-consuming to find the global minimum of the functional (31) in the FA-greedy scheme. A quasi-optimal alternative is a randomized splitting strategy, in which the minimization of (31) is performed on a subset of $1 \leq R \leq |V_{q_j}^{(m)}|$ randomly picked nodes of $V_{q_j}^{(m)}$. In this strategy, the parameter R acts as a control parameter giving a result close or identical to FA-greedy if R is chosen large enough.

The just introduced greedy selection rules for the adaptive BWP trees, as well as the respective wedgelet encoding and decoding are summarized in Algorithm 8.1 and 8.2.

Computational cost. We assume that the distance between two nodes can be calculated in $O(1)$ arithmetic operations. Further, in a worst-case scenario, the calculation of a mean value over $V_{q_j}^{(m)}$ requires $O(n)$ operations. The computational expenses of Algorithm 8.1 can then be bounded by $O(Mn^2)$, $O(MRn)$ and $O(Mn)$ operations for FA-greedy, R-greedy and MD-greedy, respectively.

Algorithm 8.1: Wedgelet encoding of a graph signal

Input: Graph signal x , initial node $q_1 \in V$, first partition $\mathcal{P}^{(1)} = \{V\} = \{V_{q_1}^{(1)}\}$ and final partition size M .

for $m = 1$ **to** $M - 1$ **do**

1) **Greedy selection of subset:** calculate j according to the rule (30) as

$$j = \underset{i \in \{1, \dots, m\}}{\operatorname{argmax}} \left\| (x - \bar{x}_{V_{q_i}^{(m)}}) \mathbb{I}_{V_{q_i}^{(m)}} \right\|_2.$$

2) Conduct one of the following alternatives to calculate the node q_{m+1} :

(i) **Max-distance (MD) greedy procedure:** select new node

$$q_{m+1} = \arg \max_{v \in V_{q_j}^{(m)}} d(q_j, v)$$

furthest away from q_j and add it to the node set $\mathcal{Q}$.

(ii) **Fully-adaptive (FA) greedy procedure:** determine new node q_{m+1} such that the squared $\mathcal{L}^2$ -error term (31) is minimized, and add it to the node set $\mathcal{Q}$.

(iii) **Randomized (R) greedy procedure:** determine q_{m+1} such that (31) is minimized over a subset of R randomly chosen points, and add it to $\mathcal{Q}$.

3) According to Definition 8.3, generate the **new partition** $\mathcal{P}^{(m+1)}$ from $\mathcal{P}^{(m)}$ by a wedge split of the subset $V_{q_j}^{(m)}$ into the children sets $V_{q_j}^{(m+1)}$ and $V_{q_{m+1}}^{(m+1)}$.

4) Compute **mean values** $\bar{x}_{V_{q_i}^{(m+1)}}$, $i \in \{1, \dots, m+1\}$, for the new partition $\mathcal{P}^{(m+1)}$ by an update from $\mathcal{P}^{(m)}$.

Output: $\mathcal{Q} = \{q_1, \dots, q_M\}$, $\{\bar{x}_{V_{q_1}^{(M)}}, \dots, \bar{x}_{V_{q_M}^{(M)}}\}$.

Algorithm 8.2: Wedgelet decoding of a graph signal

Input: $\mathcal{Q} = \{q_1, \dots, q_M\}$, $\{\bar{x}_{V_{q_1}^{(M)}}, \dots, \bar{x}_{V_{q_M}^{(M)}}\}$.

Calculate the partition $\mathcal{P}^{(M)} = \{V_{q_1}^{(M)}, \dots, V_{q_M}^{(M)}\}$ of V by elementary wedge splits along the BWP tree $\mathcal{T}_{\mathcal{Q}}$ according to the recursive procedure in Definition 8.3.

Output: The wedgelet approximation

$$\mathcal{W}_M x(v) = \sum_{i=1}^M \bar{x}_{V_{q_i}^{(M)}} \omega_{q_i}^{(M)}(v)$$

of x . For $M = n$, $\mathcal{W}_n x = x$ is reconstructed.

Acceleration possibilities. For very large n , FA-greedy and R-greedy (for large R) might be computationally too expensive. In this case, a possibility to increase the calculational speed is to split the graph a priori into J subgraphs. Then, the adaptive BWP methods can be applied (also in a parallelized form) separately to each subgraph. For this procedure, clustering algorithms as J -center clustering [31] are available.

8.3 Geometric wavelets based on wedge splits

Instead of storing the m mean values $\{\bar{x}_{V_{q_1}^{(M)}}, \dots, \bar{x}_{V_{q_M}^{(M)}}\}$ of the wedgelet approximation

$$\mathcal{W}_M x(v) = \sum_{i=1}^M \bar{x}_{V_{q_i}^{(M)}} \omega_{q_i}^{(M)}(v),$$

we can alternatively encode the expansion $\mathcal{W}_M f$ using the wavelet components defined in (16). This alternative representation is particularly useful if a further compression of the signal is desired, for instance, when using an m -term approximation of the signal with $m < M$.

For a BWP tree $\mathcal{T}_Q$ and $m \in \{1, \dots, M-1\}$, we consider the set $V_{q_j}^{(m)}$ based on the selection index $j = j_m \leq m$ such that in the m -th. iteration step $V_{q_j}^{(m)}$ is split into the children $V_{q_j}^{(m+1)}$ and $V_{q_{m+1}}^{(m+1)}$. Then, the wavelet components with respect to the BWP tree $\mathcal{T}_Q$ are given as the signals

$$\begin{aligned} \Psi_{V_{q_j}^{(m+1)}}(x) &= \left(\bar{x}_{V_{q_j}^{(m+1)}} - \bar{x}_{V_{q_j}^{(m)}} \right) \omega_{q_j}^{(m+1)} =: c_j^{(m+1)}(x) \omega_{q_j}^{(m+1)}, \\ \Psi_{V_{q_{m+1}}^{(m+1)}}(x) &= \left(\bar{x}_{V_{q_{m+1}}^{(m+1)}} - \bar{x}_{V_{q_j}^{(m)}} \right) \omega_{q_{m+1}}^{(m+1)} =: c_{m+1}^{(m+1)}(x) \omega_{q_{m+1}}^{(m+1)}, \end{aligned}$$

with the two coefficients $c_j^{(m+1)}(x)$ and $c_{m+1}^{(m+1)}(x)$ depending on the graph signal x . In the particular case of the root set V , we additionally introduce the coefficient $c_1^{(1)}(x) := \bar{x}_V$ giving the representation

$$\Psi_{V_{q_1}^{(1)}}(x) = \bar{x}_V \omega_{q_1}^{(1)} = c_1^{(1)}(x) \omega_{q_1}^{(1)}.$$

In this way, we obtain $2M - 1$ wavelet components for the tree $\mathcal{T}_Q$ together with respective coefficients in order to represent them in terms of the wedgelets. Beside the altered notation, this definition corresponds to the general definition of wavelet components for BGP trees provided in (16).

Algorithm 8.3: Wedgelet encoding with BWP wavelets

Input: Graph signal x , initial node $q_1 \in V$, first partition $\mathcal{P}^{(1)} = \{V\} = \{V_{q_1}^{(1)}\}$ and final partition size M .

Compute $c_1^{(1)}(x) = \bar{x}_V$.

for $m = 1$ **to** $M - 1$ **do**

- 1) 2) & 3) as in Algorithm 8.1 to obtain the partition $\mathcal{P}^{(m+1)}$.
- 4) Compute **wavelet coefficient** $c_{m+1}^{(m+1)}(x) = \bar{x}_{V_{q_{m+1}}^{(m+1)}} - \bar{x}_{V_{q_j}^{(m)}}$.

Output: $Q = \{q_1, \dots, q_M\}$, $\{c_1^{(1)}(x), c_2^{(2)}(x), \dots, c_M^{(M)}(x)\}$.

For the wavelet components $\Psi_{V_{q_j}^{(m+1)}}(x)$ and $\Psi_{V_{q_{m+1}}^{(m+1)}}(x)$, we additionally have the following relation:

$$\begin{aligned} & \sum_{v \in V} \left(\Psi_{V_{q_j}^{(m+1)}}(x)(v) + \Psi_{V_{q_{m+1}}^{(m+1)}}(x)(v) \right) \\ &= \bar{x}_{V_{q_{m+1}}^{(m+1)}} |V_{q_{m+1}}^{(m+1)}| + \bar{x}_{V_{q_j}^{(m+1)}} |V_{q_j}^{(m+1)}| - \bar{x}_{V_{q_j}^{(m)}} |V_{q_j}^{(m)}| \\ &= \sum_{v \in V_{q_{m+1}}^{(m+1)}} x(v) + \sum_{v \in V_{q_j}^{(m+1)}} x(v) - \sum_{v \in V_{q_j}^{(m)}} x(v) = 0. \end{aligned}$$

This implies that

$$\langle \Psi_{V_{q_j}^{(m+1)}}(x), 1 \rangle = -\langle \Psi_{V_{q_{m+1}}^{(m+1)}}(x), 1 \rangle,$$

and that the coefficients $c_j^{(m+1)}(x)$ and $c_{m+1}^{(m+1)}(x)$ are related by

$$c_j^{(m+1)}(x) = -\frac{|V_{q_{m+1}}^{(m+1)}|}{|V_{q_j}^{(m+1)}|} c_{m+1}^{(m+1)}(x).$$

In particular, when computing the coefficients of the geometric wavelets, it suffices to store one of the two, either $c_j^{(m+1)}(x)$ or $c_{m+1}^{(m+1)}(x)$. Using this relation, we can write the wedgelet approximation $\mathcal{W}_M x$ of a graph signal x equivalently in the following forms:

$$\begin{aligned} \mathcal{W}_M x &= \Psi_V(x) + \sum_{m=1}^{M-1} \Psi_{V_{q_j}^{(m+1)}}(x) + \Psi_{V_{q_{m+1}}^{(m+1)}}(x) \\ &= c_1^{(1)}(x) + \sum_{m=1}^{M-1} \left(c_j^{(m+1)}(x) \omega_{q_j}^{(m+1)} + c_{m+1}^{(m+1)}(x) \omega_{q_{m+1}}^{(m+1)} \right) \\ &= c_1^{(1)}(x) + \sum_{m=1}^{M-1} c_{m+1}^{(m+1)}(x) \left(\omega_{q_{m+1}}^{(m+1)} - \frac{|V_{q_{m+1}}^{(m+1)}|}{|V_{q_j}^{(m+1)}|} \omega_{q_j}^{(m+1)} \right) \\ &= c_1^{(1)}(x) + \sum_{m=1}^{M-1} c_j^{(m+1)}(x) \left(\omega_{q_j}^{(m+1)} - \frac{|V_{q_j}^{(m+1)}|}{|V_{q_{m+1}}^{(m+1)}|} \omega_{q_{m+1}}^{(m+1)} \right) = \sum_{m=1}^M c_j^{(m)}(x) \psi_m, \end{aligned} \tag{32}$$

where ψ_m denote the geometric Haar wavelets linked to the BWP tree $\mathcal{T}_Q$, and to the wavelet components $\Psi_{V_{q_j}^{(m+1)}}(x)$ and $\Psi_{V_{q_{m+1}}^{(m+1)}}(x)$. The encoding and decoding of a graph signal x in terms of wedge-based wavelet components is compactly summarized in Algorithm 8.3 and Algorithm 8.4.

Algorithm 8.4: Wedgelet decoding with BWP wavelets

Input: $Q = \{q_1, \dots, q_M\}, \{c_1^{(1)}, c_2^{(2)}, \dots, c_M^{(M)}\}$.

$\mathcal{W}_1 x = c_1^{(1)}(x)$.

for $m = 1$ **to** $M - 1$ **do**

Calculate the partition $\mathcal{P}^{(m+1)} = \{V_{q_1}^{(m+1)}, \dots, V_{q_{m+1}}^{(m+1)}\}$ of V from the partition $\mathcal{P}^{(m)}$ by an elementary wedge split of the set $V_{q_j}^{(m)}$.

Update the wedgelet approximation

$$\mathcal{W}_m x = \mathcal{W}_{m-1} x + c_{m+1}^{(m+1)}(x) \left(\omega_{q_{m+1}}^{(m+1)} - \frac{|V_{q_{m+1}}^{(m+1)}|}{|V_{q_j}^{(m+1)}|} \omega_{q_j}^{(m+1)} \right).$$

Output: The wedgelet approximation

$$\mathcal{W}_M x = c_1^{(1)}(x) + \sum_{m=1}^{M-1} \left(c_j^{(m+1)}(x) \omega_{q_j}^{(m+1)} + c_{m+1}^{(m+1)}(x) \omega_{q_{m+1}}^{(m+1)} \right)$$

of x in one of the equivalent formulations of (32). For $M = n$, we get $\mathcal{W}_n x = x$.

8.4 Memory requirements for wedgelet encoding

To encode a graph signal in terms of M wedgelets, we need to store the mean values $\{\bar{x}_{V_{q_1}^{(M)}}, \dots, \bar{x}_{V_{q_M}^{(M)}}\}$ on the leaves of the BWP tree as well as the geometric information of the wedge splits provided by the set $Q = \{q_1, \dots, q_M\}$. Using a quantization of K different values, every mean value $\bar{x}_{V_{q_i}^{(M)}}$ can be stored with $\log_2(K)$ bits. The same holds true if the BWP wavelet coefficients $c_m^{(m)}(x)$ replace the mean values. As we have n graph nodes, we can further store any vertex v_j using its index $j \in \{1, \dots, n\}$. This requires at most $\log_2(n)M$ bits to store the node set Q . Consequently, we get the following upper bound for the memory requirements of a wedgelet encoding.

Theorem 8.7 *Assume that the mean values $\bar{x}_{V_{q_i}^{(M)}}$ (or the coefficients $c_m^{(m)}$) are given in a quantized form with at most K different values. Then, the wedgelet encodings in Algorithm 8.1 and Algorithm 8.3 require a memory of at most*

$$\frac{[\log_2(n) + \log_2(K)]M}{n} \quad \text{bits per node.}$$

Example 8.8 In the particular case of an image with $512 \times 512 = 2^{18}$ pixels and an image depth of $K = 2^8 = 256$ colors we get by Theorem 8.7 that a representation with $M = 1000$ wedgelets requires a memory of less than 0.1 bits per pixel.

8.5 Examples on graphs and images

Examples of BWPs on graphs

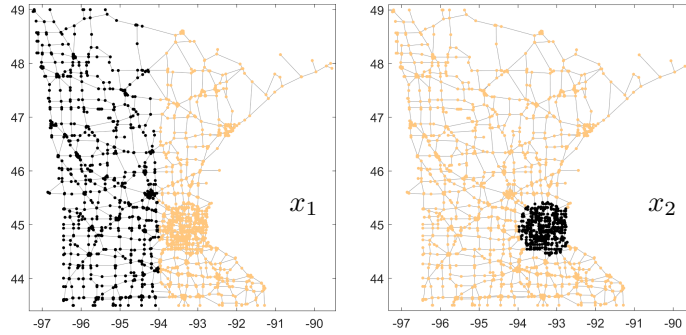


Fig. 20 Piecewise constant test signals x_1 (left) and x_2 (right) on the Minnesota graph.

We will consider the Minnesota road network introduced in Example 2.3 as test scenario, as well as the two binary functions

$$x_1 = 2\mathbb{I}_{V_1} - 1, \quad x_2 = 2\mathbb{I}_{V_2} - 1,$$

based on the characteristic functions of the node sets

$$V_1 = \{v \in V \mid x_v < -94\},$$

$$V_2 = \{v \in V \mid 0.75(x_v + 93.3)^2 + (y_v - 44.95)^2 < 0.35\},$$

where (x_v, y_v) denote the Cartesian coordinates of the node $v \in V$. The functions x_1 , and x_2 are illustrated in Fig. 20.

Starting from a random node q_1 we use Algorithm 8.1 to generate the BWP tree as well as a wedgelet approximation of the two functions. The initial part of the BWP tree for the approximation of x_2 (using FA-greedy) is displayed in Fig. 19.

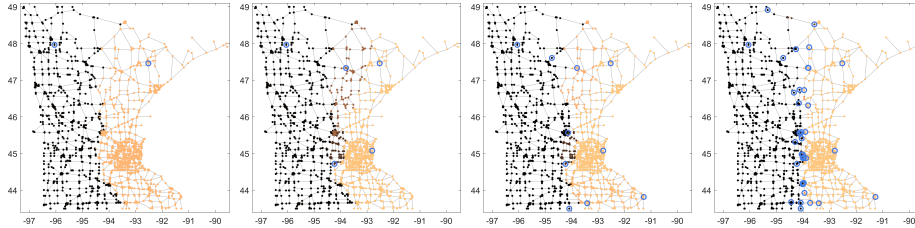


Fig. 21 Approximation of x_1 with 1, 4, 9 and 39 wedge splits (from left to right). The blue rings indicate the nodes in Q . The number of wrongly classified nodes equals 356, 286, 110, and 12, respectively.

In Fig. 21, wedgelet approximations $\mathcal{W}_m x_1$ of the function x_1 are illustrated for different partitioning stages m . The corresponding number of wedge splits equals $m - 1$. The FA-greedy procedure has been used to generate the adaptive BWP tree.

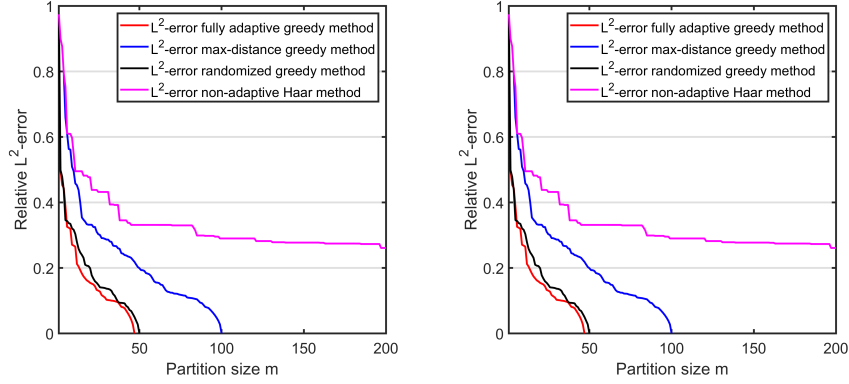


Fig. 22 L^2 -approximation errors for the BWP approximation $\mathcal{W}_m x$ of x_1 (left) and x_2 (right) using FA-greedy, MD-greedy, R-greedy with $R = 40$, and a non-adaptive BGP tree.

For the test functions x_1 and x_2 , the three adaptive variants MD-greedy, R-greedy (with $R = 40$) and FA-greedy, and a non-adaptive Haar basis are compared in Fig. 22. Here, the non-adaptive scheme is based on a BGP tree $\mathcal{T}$ which is generated iteratively based on a 2-center clustering of the currently largest cluster. In these examples, FA-greedy performs best for most partition sizes m , followed by R-greedy and MD-greedy. The FA-greedy variant is also the most cost-intensive of the four. Our tests showed that R-greedy with $R = 100$ randomly chosen nodes provides results which are already very similar to FA-greedy.

Fig. 22 displays that the data-adaptive FA-greedy and R-greedy schemes are able to generate sparser representations of the test functions in which a considerably smaller amount of the wavelet coefficients is required compared to the non-adaptive Haar system. While this higher sparseness is an aimed-at desirable property for data compression, adaptive basis systems require additional memory to store the respective geometric information. Therefore, when using adaptive BWPs for the compression of graph data also the additional storage costs of the BWPs as estimated in Section 8.4 have to be taken into account.

In a final test, we compare the same basis systems on the Minnesota graph for the approximation of piecewise smooth test functions. For this, we modify the piecewise constant signal x_1 by adding a gradient term, i.e., we consider the signal

$$x_3(v) = x_1(v) + \alpha(x_v - \bar{x}), \quad \alpha \in \mathbb{R},$$

where $\bar{x}$ denotes the mean value of x_v over V . As the parameter α increases, the function x_3 gets dominated by the gradient term and a dictionary of piecewise constant signals

will struggle to approximate x_3 in a sparse way. This is confirmed by our simulation shown in Fig. 23. As α gets larger, the $\mathcal{L}^2(V)$ -approximation errors for the expansion in adaptive and non-adaptive Haar basis systems deteriorate. The difference between adaptive and non-adaptive basis systems is however still significant.

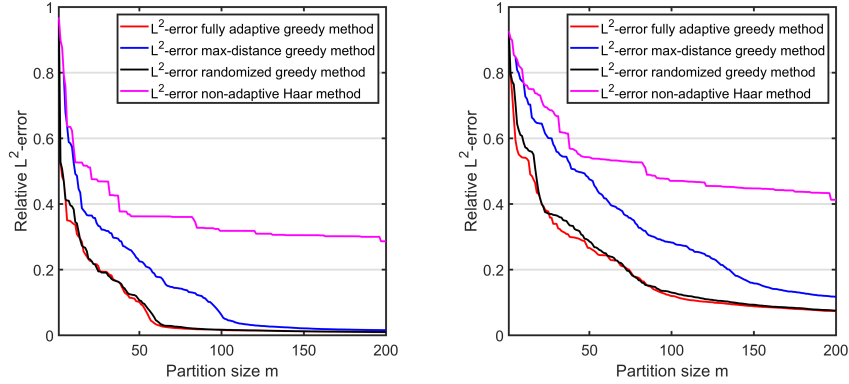


Fig. 23 L^2 -approximation errors for the BWP approximation $\mathcal{W}_m x$ of the piecewise smooth test function x_3 using FA-greedy, MD-greedy, R-greedy with $R = 40$, and a non-adaptive BGP tree, as well as two different gradient parameters $\alpha = 0.1$ (left) and $\alpha = 0.5$ (right).

BWPs for the compression of 2D images



Fig. 24 Role of the distance metric in adaptive wedge partitioning of the image in Fig. 1 a), using R-greedy approximation with $M = 4000$ nodes, $R = 200$ as well as a) the 1-norm, b) the 2-norm and c) the infimum norm for the pixel distance.

As adaptive partitioning tools for discrete domains, BWPs can also be used for the piecewise approximation and compression of images. An image can be naturally thought of as a finite rectangular grid of pixels and interpreted as a graph. Pixels close to each other are therein linked by a weighted edge. The structure and the weights of the single edges determine the local dependencies in the image and have therefore a strong influence on the outcome of the greedy algorithms.

A simple qualitative comparison of the role of the used metric is given in Fig. 24. The 1-norm and the infimum norm for the distance of the pixels lead to partitions with a rather rectangular or rhomboid wedge structure. On the other hand, wedges generated by the 2-norm seem to be more anisotropic and slightly better adapted to the edges of the image.

We further compare also the performance of the FA-greedy and the MD-greedy method for the compression of images. In the example given in Fig. 25, we see that, as expected, the FA-greedy performs considerably better. In particular, the wavelet details in the FA-greedy scheme are smaller and more distinguished than for MD-greedy when using the same number of wedge splits. Regarding the distribution of the center nodes Q , we see moreover in Fig. 26 that the adaptive BWP scheme (in this case a R-greedy scheme) selects the new nodes increasingly closer to the edges of the image such that most refinements of the partitions are performed in those regions where the gradients are large.

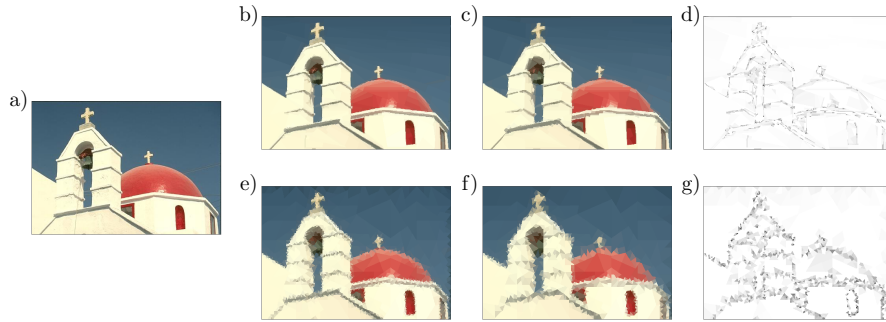


Fig. 25 BWP image compression. a) original 481×321 -image from the BSDS300 dataset; b)c) R-greedy BWP compression for $M = 1600$, $M = 800$; d) wavelet details between b) and c); e)f) MD-greedy BWP compression for $M = 1600$, $M = 800$; g) wavelet details between e) and f).

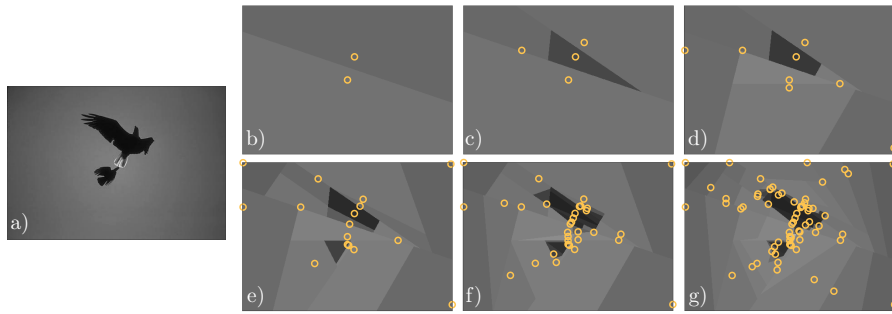


Fig. 26 BWP image encoding. a) original with 481×321 pixels from the BSDS300 dataset; b)-g) R-greedy compression with 2, 4, 8, 16, 32 and 64 nodes, $R = 2000$; The corresponding PSNR values are b) 22.767 dB, c) 23.522 dB, d) 24.853 dB, e) 26.702 dB, f) 28.547 dB, and g) 30.547 dB.

9 Software resources

In these lecture notes, we have provided an introduction to graph signal processing, and in particular to three main wavelet construction principles on graphs, i.e., spectral graph wavelets, diffusion wavelets, and Haar-type wavelets based on hierarchical partitionings.

To complement this theoretical material, we conclude these notes with a non-exhaustive list of software packages, many of which were used to create experiments and figures presented here. These software tools provide a useful starting point for readers who wish to further explore graph signal processing and graph wavelets in practice.

- **PyGSP: Graph Signal Processing in Python**
<https://github.com/epfl-lts2/pygsp>
 A comprehensive Python library for graph signal processing by Defferrard, M., Martin, L., Pena, R., Perraudin, N., including spectral graph wavelets.
- **GSPBox: The Graph Signal Processing Toolbox**
<https://github.com/epfl-lts2/gspbox>
 A MATLAB toolbox providing tools for graph signal processing by Perraudin, N., Paratte, J., Shuman, D., Martin, L., Kalofolias, V., Vandergheynst, P., Hammond, D.K., containing the GFT and the spectral graph wavelet transform.
- **The Spectral Graph Wavelets Toolbox**
<https://wiki.epfl.ch/sgwt>
 A MATLAB toolbox for the spectral graph wavelets transform by Hammond, D.K., Vandergheynst, P., Gribonval, R. [35]
- **Diffusion Wavelets**
<https://github.com/aweinstein/dw>
 Clone of the MATLAB implementation for diffusion wavelets developed in the original work [14] of Coifman, R.R., and Maggioni, M.
- **The Multiscale Transforms for Signals on Graphs (MTSG) Toolbox**
https://github.com/JeffLirion/MTSG_Toolbox
 MATLAB code by Irion, J., Saito, N. [36, 37, 38] to calculate the hierarchical graph Laplacian eigenbasis and the Haar-Walsh basis system.
- **GBFlearn**
<https://github.com/WolfgangErb/GBFlearn>
 A MATLAB toolbox to illustrate how graph basis functions (GBFs) can be used for interpolation, classification and semi-supervised learning on graphs, contains the data sets for the bunny graph and the Minnesota graph [23, 25, 27].
- **GraphBasisFunctions**
<https://github.com/GabrieleSantin/GraphBasisFunctions>
 Python code for the approximation with Graph Basis Functions [15], partially ported from GBFlearn.
- **GBFPUM**
<https://github.com/WolfgangErb/GBFPUM>
 A MATLAB Package for Partition of Unity Based Signal Interpolation and Approximation on Graphs [4, 5].

- **Graph Wedgelets (MATLAB)**
<https://github.com/WolfgangErb/GraphWedgelets>
MATLAB implementation for graph wedgelet approximation, developed in [26, 28]
- **Graph Wedgelets (Python)**
<https://github.com/ZinRicky/graph-wedgelets>
Python implementation of graph wedgelets based on the thesis [6] of Cazzin, R.
- **GUPPY: Graph Uncertainty Principle PlaYground**
<https://github.com/WolfgangErb/GUPPY>
A MATLAB toolbox to illustrate space-frequency localization and uncertainty principles on graphs [24].
- **Empowering Simple Graph Convolutional Networks**
<https://github.com/lpasa/Empowering-Simple-Graph-Convolutional-Networks>
Python code for the generation and analysis of graph convolutional networks based on simple graph convolution operators [47].

Acknowledgements The authors of this article are funded by the Università degli Studi di Padova - Dipartimento di Matematica under the project SID BIRD 2023 entitled “ALISIA - ALgorithms for Immersive Stereoscopic Imaging with Applications to the Daedalus camera system”, and by the European Union – NextGenerationEU under the National Recovery and Resilience Plan (NRRP), Mission 4 Component 2 Investment 1.1 - Call PRIN 2022 No. 104 of February 2, 2022 of Italian Ministry of University and Research; Project 2022FHCNY3 (subject area: PE - Physical Sciences and Engineering) “Computational mEthods for Medical Imaging (CEMI)”. Further support by the Indam research group GNCS (Indam-GNCS project CUP_E53C24001950001) and the Italian Mathematical Union (research group UMI-TAA) is gratefully acknowledged.

Competing Interests The author has no conflicts of interest to declare that are relevant to the content of these lecture notes.

References

1. Birgé, L., Massart, P.: From model selection to adaptive estimation. In: *Festschrift for Lucien Le Cam: Research Papers in Probability and Statistics*, pp. 55–88. Springer, New York (1997)
2. Bulai, I.M., Saliani, S.: Spectral graph wavelet packets frames. *Appl. Comput. Harmon. Anal.* **66**, 1–18 (2023)
3. Bremer, J.C., Coifman, R., Maggioni, M., Szlam, A.D.: Diffusion wavelet packets. *Appl. Comput. Harmon. Anal.* **21**(1), 95–112 (2006)
4. Cavoretto, R., De Rossi, A., Erb, W.: Partition of unity methods for signal processing on graphs. *J. Fourier Anal. Appl.* **27**, Art. 66 (2021)
5. Cavoretto, R., De Rossi, A., Erb, W.: GBFPUM - A MATLAB package for partition of unity based signal interpolation and approximation on graphs. *Dolomites Res. Notes Approx.* **15**(2), 25–34 (2022)
6. Cazzin, R.: Clustering-based techniques for compression of graph signals via geometric graph wedgelets. Master's thesis, University of Padova, Padova (2025), Available at: <https://hdl.handle.net/20.500.12608/102103>
7. Christensen, O.: *An Introduction to Frames and Riesz Bases*. Applied and Numerical Harmonic Analysis. Birkhäuser, Boston (2003)
8. Cheng, X., Chen, X., Mallat, S.: Deep Haar scattering networks. *Inf. Inference* **5**(2), 105–133 (2016)
9. Chui, C.K., Filbir, F., Mhaskar, H.N.: Representation of functions on big data: Graphs and trees. *Appl. Comput. Harmon. Anal.* **38**(3), 489–509 (2015)
10. Chung, F.R.K.: *Spectral Graph Theory*. AMS, Providence (1997)
11. Cloninger, A., Li, H., Saito, N.: Natural graph wavelet packet dictionaries. *J. Fourier Anal. Appl.* **27**, Article 41 (2021)
12. Cohen, A., Dyn, N., Hecht, F., Mirebeau, J.-M.: Adaptive multiresolution analysis based on anisotropic triangulations. *Math. Comput.* **81**(278), 789–810 (2012)
13. Coifman, R., Gavish, M.: Harmonic analysis of digital data bases. In: *Wavelets and Multiscale Analysis*, pp. 161–197. Springer, Berlin (2011)
14. Coifman, R., Maggioni, M.: Diffusion wavelets. *Appl. Comput. Harmon. Anal.* **21**, 53–94 (2006)
15. Cuomo, S., Erb, W., Santin, G.: Kernel-based models for influence maximization on graphs based on Gaussian process variance minimization. *J. Comput. Appl. Math.* **423**, 114951 (2023)
16. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: *Advances in Neural Information Processing Systems* 29, pp. 3844–3852 (2016)
17. Dekel, S., Leviatan, D.: Adaptive multivariate approximation using binary space partitions and geometric wavelets. *SIAM J. Numer. Anal.* **43**(2), 707–732 (2005)
18. Demaret, L., Dyn, N., Iske, A.: Image compression by linear splines over adaptive triangulations. *Signal Process.* **86**(7), 1604–1616 (2006)
19. Demaret, L., Iske, A.: Adaptive image approximation by linear splines over locally optimal Delaunay triangulations. *IEEE Signal Process. Lett.* **13**(5), 281–284 (2006)
20. Demaret, L., Iske, A.: Optimal N -term approximation by linear splines over anisotropic Delaunay triangulations. *Math. Comput.* **84**(293), 1241–1264 (2015)
21. DeVore, R.: Nonlinear approximation. *Acta Numer.* **7**, 51–150 (1998)
22. Donoho, D.L.: Wedgelets: Nearly minimax estimation of edges. *Ann. Statist.* **27**(3), 859–897 (1999)
23. Erb, W.: Semi-Supervised Learning on Graphs with Feature-Augmented Graph Basis Functions. arXiv preprint arXiv:2003.07646 (2020)
24. Erb, W.: Shapes of uncertainty in spectral graph theory. *IEEE Trans. Inf. Theory* **67**(2), 1291–1307 (2021)
25. Erb, W.: Graph signal interpolation with positive definite graph basis functions. *Appl. Comput. Harmon. Anal.* **60**, 368–395 (2022)
26. Erb, W.: Graph wedgelets: Adaptive data compression on graphs based on binary wedge partitioning trees and geometric wavelets. *IEEE Trans. Signal Inf. Process. Netw.* **9**, 24–34 (2023)

27. Erb, W.: Krylov subspace methods to accelerate kernel machines on graphs. *Adv. Comput. Sci. Eng.* **1**(1), 59–81 (2023)
28. Erb, W.: Split-and-merge segmentation of biomedical images using graph wedgelet decompositions. In: Gervasi, O. et al. (eds.) *Computational Science and Its Applications - ICCSA 2025 Workshops, LNCS*, vol. 15899, pp. 252–263. Springer, Cham (2026)
29. Friedrich, F., Demaret, L., Führ, H., Wicker, K.: Efficient moment computation over polygonal domains with an application to rapid wedgelet approximation. *SIAM J. Sci. Comput.* **29**(2), 842–863 (2007)
30. Gavish, M., Nadler, B., Coifman, R.: Multiscale wavelets on trees, graphs and high-dimensional data. In: *Proc. 27th Int. Conf. on Machine Learning (ICML 2010)*, pp. 367–374 (2010)
31. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.* **38**, 293–306 (1985)
32. Godsil, G., Royle, G.: *Algebraic Graph Theory*. Springer, New York (2001)
33. Gröchenig, K.: *Foundations of Time-Frequency Analysis. Applied and Numerical Harmonic Analysis*. Birkhäuser, Boston (2001)
34. Horn, R.A., Johnson, C.R.: *Matrix Analysis*, 2nd edn. Cambridge University Press, Cambridge (2013)
35. Hammond, D.K., Vandergheynst, P., Gribonval, R.: Wavelets on graphs via spectral graph theory. *Appl. Comput. Harmon. Anal.* **30**(2), 129–150 (2011)
36. Irion, J., Saito, N.: Hierarchical graph Laplacian eigen transforms. *Jpn. SIAM Lett.* **6**, 21–24 (2014)
37. Irion, J., Saito, N.: The generalized Haar-Walsh transform. In: *Proc. IEEE Workshop on Statistical Signal Processing*, pp. 472–475 (2014)
38. Irion, J., Saito, N.: Efficient approximation and denoising of graph signals using multiscale basis dictionaries. *IEEE Trans. Signal Inf. Process. Netw.* **3**(3), 607–616 (2017)
39. Jansen, M., Nason, G.P., Silverman, B.W.: Multiscale methods for data on graphs and irregular multidimensional situations. *J. R. Stat. Soc. Ser. B* **71**(1), 97–125 (2009)
40. Karaivanov, B., Petrushev, P.: Nonlinear piecewise polynomial approximation beyond Besov spaces. *Appl. Comput. Harmon. Anal.* **15**(3), 177–223 (2003)
41. Krommweh, J.: Tetrolet transform: A new adaptive Haar wavelet algorithm for sparse image representation. *J. Vis. Commun. Image Represent.* **21**(4), 364–374 (2010)
42. Leonardi, R., Kunt, M.: Adaptive split-and-merge for image analysis and coding. In: *Proc. SPIE*, vol. 594 (1985)
43. Murtagh, F.: The Haar wavelet transform of a dendrogram. *J. Classification* **24**, 3–32 (2007)
44. Ortega, A., Frossard, P., Kovačević, J., Moura, J.M.F., Vandergheynst, P.: Graph signal processing: Overview, challenges, and applications. *Proc. IEEE* **106**(5), 808–828 (2018)
45. Ortega, A.: *Introduction to Graph Signal Processing*. Cambridge University Press, Cambridge (2022)
46. Narang, S.K., Ortega, A.: Perfect reconstruction two-channel wavelet filter banks for graph structured data. *IEEE Trans. Signal Process.* **60**(6), 2786–2799 (2012)
47. Pasa, L., Navarin, N., Erb, W., Sperduti, A.: Empowering simple graph convolutional networks. *IEEE Trans. Neural Netw. Learn. Syst.* **35**(4), 4385–4399 (2024)
48. Radha, H., Leonardi, R., Naylor, B., Vetterli, M.: Image representation using binary space partitioning trees. In: *Proc. SPIE*, vol. 1360, pp. 639–650 (1990)
49. Radha, H., Vetterli, M., Leonardi, R.: Image compression using binary space partitioning trees. *IEEE Trans. Image Process.* **5**(12), 1610–1624 (1996)
50. Rossi, R.A., Ahmed, N.K.: The Network Data Repository with interactive graph analytics and visualization. In: *Proc. AAAI Conf. on Artificial Intelligence* (2015)
51. Rustamov, R., Guibas, L.J.: Wavelets on graphs via deep learning. In: *Adv. Neural Inf. Process. Syst.*, vol. 26 (2013)
52. Saito, N., Shao, Y.: eGHWT: The extended generalized Haar-Walsh transform. *J. Math. Imaging Vis.* **64**(3), 261–283 (2022)
53. Samet, H.: Data structures for quadtree approximation and compression. *Commun. ACM* **28**(9), 973–993 (1985)
54. Sharpnack, J., Singh, A.: Near-optimal and computationally efficient detectors for weak and sparse graph-structured patterns. In: *Proc. GlobalSIP*, pp. 443–446 (2013)

55. Shen, G., Ortega, A.: Transform-based distributed data gathering. *IEEE Trans. Signal Process.* **58**(7), 3802–3815 (2010)
56. Shuman, D.I., Ricaud, B., Vandergheynst, P.: A windowed graph Fourier transform. In: *Proc. IEEE Statistical Signal Processing Workshop*, pp. 133–136 (2012)
57. Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs. *IEEE Signal Process. Mag.* **30**(3), 83–98 (2013)
58. Shuman, D.I., Ricaud, B., Vandergheynst, P.: Vertex-frequency analysis on graphs. *Appl. Comput. Harmon. Anal.* **40**(2), 260–291 (2016)
59. Shuman, D.I.: Localized spectral graph filter frames. *IEEE Signal Process. Mag.* **37**(6), 43–63 (2020)
60. Stanković, L., Daković, M., Sejdić, E.: Introduction to graph signal processing. In: Stanković, L., Sejdić, E. (eds.) *Vertex-Frequency Analysis of Graph Signals*. Signals and Communication Technology. Springer, Cham (2019)
61. Sušnjara, A., Perraudin, N., Kressner, D., Vandergheynst, P.: Accelerated filtering on graphs using the Lanczos method. *arXiv preprint arXiv:1509.04537* (2015)
62. Szlam, A.D., Maggioni, M., Coifman, R.R., Bremer, J.C.: Diffusion-driven multiscale analysis on manifolds and graphs. In: *Proc. SPIE*, vol. 5914, Wavelets XI, 59141D (2005)
63. Trefethen, L.N., Bau, D.: *Numerical Linear Algebra*. SIAM, Philadelphia (1997)
64. von Luxburg, U.: A tutorial on spectral clustering. *Stat. Comput.* **17**, 395–416 (2007)
65. Wakin, M., Romberg, J., Choi, H., Baraniuk, R.: Geometric methods for wavelet-based image compression. In: *Proc. SPIE*, vol. 5207, pp. 507–520 (2003)