Natural Graph Wavelet Packet Dictionaries

Alexander Cloninger[1] · Haotian Li[2] · Naoki Saito[2]

© The Author(s) 2021

Abstract

We introduce a set of novel multiscale basis transforms for signals on graphs that utilize their "dual" domains by incorporating the "natural" distances between graph Laplacian eigenvectors, rather than simply using the eigenvalue ordering. These basis dictionaries can be seen as generalizations of the classical Shannon wavelet packet dictionary to arbitrary graphs, and do not rely on the frequency interpretation of Laplacian eigenvalues. We describe the algorithms (involving either vector rotations or orthogonalizations) to construct these basis dictionaries, use them to efficiently approximate graph signals through the best basis search, and demonstrate the strengths of these basis dictionaries for graph signals measured on sunflower graphs and street networks.

1 Introduction and Motivation

There is an explosion of interest and demand to analyze data sampled on graphs and networks. This has motivated development of more flexible yet mathematically sound *dictionaries* (i.e., an overcomplete collection of atoms or basis vectors) for data analysis and signal processing on graphs. Our main goal here is to build *smooth* multiscale localized basis dictionaries on an input graph, with beneficial reconstruction and sparsity properties, and to fill the "gap" left from our previous graph basis dictionary constructions [18–21,52] as we explain below. Our approach differs from the standard literature as we fully utilize both the similarities between the nodes (through the graph

✉ Naoki Saito
   saito@math.ucdavis.edu

1   Department of Mathematics and Halıcıoğlu Data Science Institute, University of California,
    San Diego, USA

2   Department of Mathematics, University of California, Davis, USA

adjacency matrix) and the similarities between the eigenvectors of the graph Laplacian matrix (through new nontrivial eigenvector distances).

Previous approaches to construct such graph basis dictionaries break down into two main categories. The first category partitions the nodes through recursive graph cuts to generate multiscale basis dictionaries. This includes: the Hierarchical Graph Laplacian Eigen Transform (HGLET) [19]; the Generalized Haar-Walsh Transform (GHWT) [18]; its extension, the eGHWT [52]; and other Haar-like graph wavelets (see, e.g., [6,13,27,38,57]). But their basis vectors either are nonsmooth piecewise constants or have non-overlapping supports. The second category uses spectral filters on the Laplacian (or diffusion kernel) eigenvalues to generate multiscale smooth dictionaries. This includes: the Spectral Graph Wavelet Transform [16]; Diffusion Wavelets [7]; extensions to spectral graph convolutional networks [29]. However, these dictionaries do not fully address the relationships among eigenvectors [5,31,49], which should be utilized for graph dictionary construction; instead, they focus on the eigenvalue distributions to organize the corresponding eigenvectors (although there are some works, e.g., [43,44,55], which recognized the graph structures strongly influence the eigenvector behaviors). These relationships among eigenvectors can result from eigenvector localization in different clusters, differing scales in multi-dimensional data, etc. These notions of similarity and difference between eigenvectors, while studied in the eigenfunction literature [5,31,49], have yet to be incorporated into building localized dictionaries on graphs.

We combine the benefits of both approaches to construct the graph equivalent of spatial-spectral filtering. We have two approaches: one is to utilize the *dual geometry* of an input graph without partitioning the input graph, and the other is to utilize clustering and partition information in both the input graph and its dual domain.

Our first approach, detailed in Sect. 3, fills the "gap" in the cycle of our development of the graph basis dictionaries, i.e., HGLET, GHWT, and eGHWT. This approach is a direct generalization of the classical wavelet packet dictionary [34, Chap. 8] to the graph setting: we hierarchically partition the dual domain to generate a tree-structured "subbands" each of which is an appropriate subset of the graph Laplacian eigenvectors. We also want to note the following correspondence: The HGLET [19] is a graph version of the Hierarchical Block Discrete Cosine Transform (DCT) dictionary [34, Sect. 8.3] (i.e., the non-smooth non-overlapping version of the local cosine dictionary [9], [34, Sect. 8.5]), and the former exactly reduces to the latter if the input graph is P_N, a path graph with N nodes. The former hierarchically partitions the input graph while the latter does the same (with a non-adaptive manner) on the unit interval $[0, 1]$ in the *time* domain. On the other hand, the GHWT [18] is a graph version of the Haar-Walsh wavelet packet dictionary [8], [34, Sect. 8.1], and the former exactly reduces to the latter if the input graph is P_N. The latter hierarchically partitions the interval $[0, N)$ in the *sequency* domain while the former does the same by the graph domain partitioning plus reordering; see [18,20,21] for the details. Our graph basis dictionary using this first approach is a graph version of the *Shannon* wavelet packet dictionary [34, Sect. 8.1.2], which hierarchically partitions the interval $[0, 1/2)$ (or $[0, \pi]$ depending on how one defines the Fourier transform) in the *frequency* domain. Again, the former essentially reduces to the latter if the input graph is P_N.

Our second approach, detailed in Sect. 4, partitions *both* the input graph *and* its dual domain; more precisely, we first hierarchically partition the dual domain, and then partition the input graph with constraints imposed by the dual domain partition. This approach parallels and generalizes classical time-frequency analysis, where the time domain is replaced by a general *node-domain* geometry and the frequency domain is replaced by a general *eigenvector-domain* organization. A version of this approach of node-eigenvector organization that embeds the eigenvectors to a one-dimensional Euclidean domain has also been considered as a visualization technique for low-frequency eigenvectors on clustered graphs [14].

We aim for the significance and impact of this research to be twofold. First, these results will provide the first set of graph wavelet packet bases that adaptively scale to the local structure of the graph. This is especially important for graphs with complicated multiscale structure, whose graph Laplacians have localized eigenvectors, for example. This is an impactful direction, as most of graph wavelet packet bases previously proposed only tile the node-eigenvector "plane" along the node "axis," while Laplacian eigenvectors only tile that plane along the eigenvector "axis". Our approach in Sect. 4 constructs filters in both the node-domain and eigenvector-domain, which is related to the classical *time-frequency adapted* wavelet packets that tile both the time and the frequency domains [17,59].

Second, in the long term, this is a first method of systematically using the novel concept of eigenvector dual geometry [5,31,49]. This direction can set a path for future modification of spectral graph theory applications to incorporate dual geometry.

The structure of this article is organized as follows. Section 2 reviews fundamentals: the basics of graphs, i.e., graph Laplacians and graph Fourier transform as well as graph wavelet transforms and frames that were proposed previously. It also reviews nontrivial metrics of graph Laplacian eigenvectors, which are used to analyze the dual geometry/eigenvector-domain of an input graph. Section 3 presents a natural graph wavelet packet dictionary constructed through hierarchical partition of the eigenvector-domain. Section 4 presents a second version of a natural graph wavelet packet dictionary constructed through a pair of hierarchical partitions, one on the input graph and one on its dual domain. In Sect. 5, we demonstrate the usefulness of our proposed graph wavelet packet dictionaries in graph signal approximation using numerical experiments. Code scripts to reproduce all the figures in this article can be found at [30]. Finally, we discuss our findings gained through these numerical experiments and near-future projects for further improvements of our dictionaries.

2 Background

2.1 Graph Laplacians and Graph Fourier Transform

Let $G = G(V, E)$ be an undirected connected graph. Let $V = V(G) = \{v_1, v_2, \ldots, v_N\}$ denote the set of nodes (or vertices) of the graph, where $N := |V(G)|$. For simplicity, we typically associate each vertex with its index and write i in place of v_i. $E = E(G) = \{e_1, e_2, \ldots, e_M\}$ is the set of edges, where each e_k connects two vertices, say, i and j, and $M := |E(G)|$. In this article we consider only finite graphs

(i.e., $M, N < \infty$). Moreover, we restrict to the case of simple graphs; that is, graphs without loops (an edge connecting a vertex to itself) and multiple edges (more than one edge connecting a pair of vertices). We use $\boldsymbol{f} = [f(1), \ldots, f(N)]^\mathsf{T} \in \mathbb{R}^N$ to denote a graph signal on G, and we define $\boldsymbol{1} := [1, \ldots, 1]^\mathsf{T} \in \mathbb{R}^N$.

We now discuss several matrices associated with graphs. The information in both V and E is captured by the *edge weight matrix* $W(G) \in \mathbb{R}^{N \times N}$, where $W_{ij} \geq 0$ is the edge weight between nodes i and j. In an unweighted graph, this is restricted to be either 0 or 1, depending on whether nodes i and j are adjacent, and we may refer to $W(G)$ as an *adjacency matrix*. In a weighted graph, W_{ij} indicates the *affinity* between nodes i and j. In either case, since G is undirected, $W(G)$ is a symmetric matrix. We then define the *degree matrix* $D(G)$ as the diagonal matrix with entries $D_{ii} = \sum_j W_{ij}$. With this in place, we are now able to define the *(unnormalized) Laplacian* matrix, *random-walk normalized Laplacian* matrix, and *symmetric normalized Laplacian* matrix, respectively, as

$$
\begin{aligned}
L(G) &:= D(G) - W(G) \\
L_{\mathrm{rw}}(G) &:= D(G)^{-1} L(G) \\
L_{\mathrm{sym}}(G) &:= D(G)^{-1/2} L(G) D(G)^{-1/2}.
\end{aligned}
\tag{1}
$$

We use $0 = \lambda_0 \leq \lambda_1 \leq \ldots \leq \lambda_{N-1}$ to denote the sorted Laplacian eigenvalues and $\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{N-1}$ to denote their corresponding eigenvectors, where the specific Laplacian matrix to which they refer will be clear from either context or subscripts. Denoting $\Phi := [\boldsymbol{\phi}_0, \ldots, \boldsymbol{\phi}_{N-1}]$ and $\Lambda := \mathrm{diag}([\lambda_0, \ldots, \lambda_{N-1}])$, the eigendecomposition of $L(G)$ can be written as $L(G) = \Phi \Lambda \Phi^\mathsf{T}$. Since we only consider connected graphs here, we have $0 = \lambda_0 \lneqq \lambda_1$, and $\boldsymbol{\phi}_0 = \boldsymbol{1}/\sqrt{N}$, which is called the direct current component vector or the *DC vector* for short. The second smallest eigenvalue λ_1 is called the *algebraic connectivity* of G and the corresponding eigenvector $\boldsymbol{\phi}_1$ is called the *Fiedler vector* of G. The Fiedler vector plays an important role in graph partitioning and spectral clustering; see, e.g., [61], which suggests the use of the Fiedler vector of $L_{\mathrm{rw}}(G)$ for spectral clustering over that of the other Laplacian matrices.

Remark 2.1 In this article, we use the Fiedler vectors of L_{rw} of an input graph and its subgraphs as a tool to hierarchically bipartition the graph although any other graph partition methods can be used in our proposed algorithms. However, note that we use the unnormalized graph Laplacian eigenvectors of $L(G)$ for simplicity to construct the dual domain of G and consequently our graph wavelet packet dictionaries. In other words, L_{rw} is only used to compute its Fiedler vector for our graph partitioning purposes.

The graph Laplacian eigenvectors are often viewed as generalized Fourier modes on graphs. Therefore, for any graph signal $\boldsymbol{f} \in \mathbb{R}^N$ and coefficient vector $\boldsymbol{g} \in \mathbb{R}^N$, the *graph Fourier transform* and *inverse graph Fourier transform* [54] are defined by

$$
\mathscr{F}_G(\boldsymbol{f}) := \Phi^\mathsf{T} \cdot \boldsymbol{f} \in \mathbb{R}^N \quad \text{and} \quad \mathscr{F}_G^{-1}(\boldsymbol{g}) := \Phi \cdot \boldsymbol{g} \in \mathbb{R}^N.
\tag{2}
$$

Since Φ is an orthogonal matrix, it is not hard to see that $\mathscr{F}_G^{-1} \circ \mathscr{F}_G = I_N$. Thus, we can use $\mathscr{F}_G$ as an analysis operator and $\mathscr{F}_G^{-1}$ as a synthesis operator for graph harmonic analysis.

As an important example and for future reference, let us consider the Laplacian eigenpairs of a path graph P_N, which we also discussed earlier [20,40,48,50,51]. In this case, the eigenvectors of $L(P_N)$ are exactly the *DCT Type II* basis vectors (used in the JPEG standard) [56]:

$$\lambda_k = \lambda_{k;N} := 4\sin^2\left(\frac{\pi k}{2N}\right), \quad \phi_k(x) = \phi_{k;N}(x) := a_{k;N}\cos\left(\frac{\pi k}{N}\left(x - \frac{1}{2}\right)\right), \tag{3}$$

where $k = 0 : N - 1$, $x = 1 : N$, and $a_{k;N}$ is a normalization constant to have $\|\phi_{k;N}\|_2 = 1$. It is clear that the eigenvalue is a monotonically increasing function of the *frequency*, which is the eigenvalue index k divided by 2 in this case.

2.2 Graph Wavelet Transforms and Frames

We now briefly review graph wavelet transforms and frames; see, e.g., [42,54] for more information. Translation and dilation are two important operators for classical wavelet construction. However, unlike $\mathbb{R}^d$ ($d \in \mathbb{N}$) or its finite and discretized lattice graph $P_{N_1} \times \cdots \times P_{N_d}$, we cannot assume the underlying graph has self-symmetric structure in general, i.e., its interior nodes may not always have the same neighborhood structure. Therefore, it is difficult to construct graph wavelet bases or frames by translating and dilating a single mother wavelet function of a fixed shape, e.g., the Mexican hat mother wavelet in $\mathbb{R}$, because the graph structure varies at different locations. Instead, some researchers, e.g., Hammond et al. [16], constructed wavelet frames by shifting smooth graph spectral filters to be centered at different nodes. A general framework of building wavelet frames can then be summarized as follows:

$$\psi_{j,n} := \overbrace{\Phi F_j \Phi^\mathsf{T}}^{\text{Filtering}} \delta_n \quad \text{for } j = 0, 1, \cdots, J \text{ and } n = 1, 2, \cdots, N, \tag{4}$$

where the index j stands for different scale of spectral filtering (the greater j, the finer the scale, and $J \in \mathbb{N}$ represents the finest scale specified by the user), the index n represents the center location of the wavelet, δ_n is the standard basis vector centered at node n, and the diagonal matrices $F_j \in \mathbb{R}^{N \times N}$ satisfies $F_0(l, l) = h(\lambda_{l-1})$ and $F_j(l, l) = g(s_j \lambda_{l-1})$ for $l = 1 : N$, $j = 1 : J$. Here, h is a scaling function (which mainly deals with the small eigenvalues), while g is a graph wavelet generating kernel. For example, the kernel proposed in [16] can be approximated by the Chebyshev polynomial and lead to a fast algorithm. Note that $\{s_j\}_{j=1:J}$ are dilation parameters.

Furthermore, one can show that as long as the generalized partition of unity

$$A \cdot I_N \leq \sum_{j=0}^{J} F_j \leq B \cdot I_N, \quad 0 < A \leq B \tag{5}$$

holds, $\{\boldsymbol{\psi}_{j,n}\}_{j=0:J;n=1:N}$ forms a *graph wavelet frame*, which can be used to decompose and recover any given graph signals [16].

2.3 A Motivating Example

However, one important drawback of the above method is that the construction of the spectral filters F_j solely depends on the eigenvalue distribution (except some flexibility in choosing the filter pair (h, g), and the dilation parameters $\{s_j\}$) and does *not* reflect how the eigenvectors *behave*. For simple graphs such as P_N and C_N (a cycle graph with N nodes), the graph Laplacian eigenvectors are global sinusoids whose frequencies can be simply read off from the corresponding eigenvalues, as discussed in Sect. 2.1. Hence, the usual *Littlewood-Paley wavelet theory* (see, e.g., [11, Sect. 4.2], [22, Sect. 2.4]) applies for those simple graphs. Unfortunately, the graph Laplacian eigenvectors of a general graph — even if it is ever so slightly more complicated than P_N and C_N — can behave in a much more complicated or unexpected manner than those of P_N or C_N, as discussed in [5,20,31,40,48–51].

   In order to demonstrate this serious problem concretely and make this article self-contained, let us examine the following example that was also discussed in [5,31,49]. Let us consider a thin rectangle in $\mathbb{R}^2$, and suppose that this rectangle is discretized as $P_{N_x} \times P_{N_y}$ ($N_x > N_y > 1$). The Laplacian eigenpairs of this lattice graph can be easily derived from Eq. (3) as:

$$\lambda_k = \lambda_{(k_x,k_y)} := \lambda_{k_x;N_x} + \lambda_{k_y;N_y}$$
$$\phi_k(x, y) = \varphi_{k_x,k_y}(x, y) := \phi_{k_x;N_x}(x) \cdot \phi_{k_y;N_y}(y)$$

where $k = 0 : N_x N_y - 1$; $k_x = 0 : N_x - 1$, $k_y = 0 : N_y - 1$, $x = 1 : N_x$, and $y = 1 : N_y$. As always, let $\{\lambda_k\}_{k=0:N_x N_y-1}$ be ordered in the nondecreasing manner. Figure 1a shows the corresponding eigenvectors ordered in this manner (with $N_x = 7$, $N_y = 3$). Note that the layout of 3×7 grid of subplots is for the page saving purpose: the layout of 1×21 grid of subplots would be more natural if we use only the eigenvalue size for eigenvector ordering. For such a 2D lattice graph, the smallest eigenvalue is still $\lambda_0 = \lambda_{(0,0)} = 0$, and the corresponding eigenvector is constant. The second smallest eigenvalue λ_1 is $\lambda_{(1,0)} = 4\sin^2(\pi/2N_x)$, since $\pi/2N_x < \pi/2N_y$, and its eigenvector has half oscillation (i.e., half period) in the x-direction. But, how about λ_2? Even for such a simple situation there are two possibilities for λ_2, depending on N_x and N_y. If $N_x > 2N_y$, then $\lambda_2 = \lambda_{(2,0)} < \lambda_{(0,1)}$. On the other hand, if $N_y < N_x < 2N_y$, then $\lambda_2 = \lambda_{(0,1)} < \lambda_{(2,0)}$. More generally, if $KN_y < N_x < (K+1)N_y$ for some $K \in \mathbb{N}$, then $\lambda_k = \lambda_{(k,0)} = 4\sin^2(k\pi/2N_x)$ for $k = 0, \ldots, K$. Yet we have $\lambda_{K+1} = \lambda_{(0,1)} = 4\sin^2(\pi/2N_y)$ and λ_{K+2} is equal to either $\lambda_{(K+1,0)} = 4\sin^2((K+1)\pi/2N_x)$ or $\lambda_{(1,1)} = 4[\sin^2(\pi/2N_x) + \sin^2(\pi/2N_y)]$ depending on N_x and N_y. Clearly, the mapping between k and (k_x, k_y) is quite nontrivial, and moreover, the eigenpair computation does not tell us how to map from k to (k_x, k_y). In Fig. 1a, one can see this behavior with $K = 2$, i.e., notice that $\boldsymbol{\phi}_2(\equiv \boldsymbol{\varphi}_{2,0})$ has one oscillation in the x-direction and no oscillation in the y-direction whereas $\boldsymbol{\phi}_3(\equiv \boldsymbol{\varphi}_{0,1})$ has no oscillation in the x-direction and half oscillation in the y-direction. In other words, all

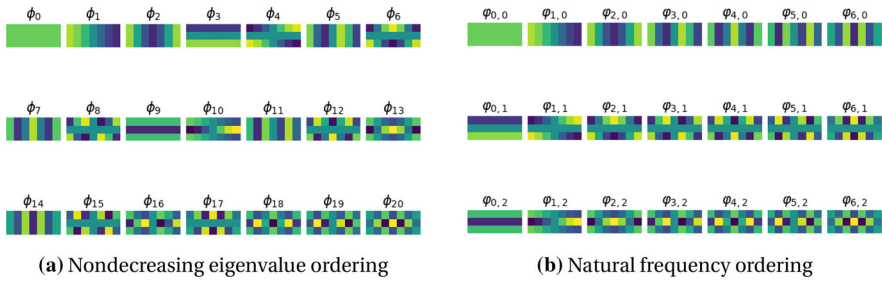(a) Nondecreasing eigenvalue ordering　　　　(b) Natural frequency ordering

Fig. 1 Laplacian eigenvectors of $P_7 \times P_3$ ordered sequentially in terms of nondecreasing eigenvalues (**a**); those ordered in terms of their natural horizontal/vertical frequencies (**b**). The color scheme called *viridis* [46] is used to represent the amplitude of eigenvectors ranging from deep violet (negative) to dark green (zero) to yellow (positive) (Color figure online)

of a sudden the eigenvalue of a completely different type of oscillation *sneaks into* the eigenvalue sequence. Hence, on a general graph, by simply looking at its Laplacian eigenvalue sequence $\{\lambda_k\}_{k=0,1,\ldots}$, it is *almost impossible to organize the eigenvectors into physically meaningful dyadic blocks and follow the Littlewood-Paley approach* unless the underlying graph is of very simple nature, e.g., P_N or C_N. Therefore, it will be problematic to design graph wavelets by using spectral filters built solely upon eigenvalues and we need to find a way to distinguish eigenvector behaviors.

What we really want to do is to *organize* those eigenvectors based on their natural frequencies or their behaviors, as shown in Fig. 1b instead of Fig. 1a, without explicitly knowing the mapping from k to (k_x, k_y) in this example. In order to do so for a general graph, we need to define and compute quantitative similarity or difference between its eigenvectors. However, we cannot use the usual ℓ^2-distances among them since they all have the same value $\sqrt{2}$ due to their orthonormality. Then a natural question is: how can we *quantify the similarity/difference between the eigenvectors*?

2.4 Nontrivial Eigenvector Distances

As a remedy to these issues, we measure the "behavioral" difference between the eigenvectors using the so-called *Difference of Absolute Gradient (DAG) pseudometric* [31], which is also used in all of our numerical experiments in Sect. 5. Note that [5,31, 49] proposed several other affinities and distances between Laplacian eigenvectors. The reasons why we decided to use the DAG pseudometric in this article are: 1) its computational efficiency compared to the other eigenvector metrics; 2) its superior performance for grid-like graphs; and 3) its close relationship with the Hadamard-product affinity proposed in [5]. See [31,49] for the details on the other eigenvector metrics and their performance comparison. Below, we briefly summarize this DAG pseudometric.

Instead of the usual ℓ^2-distance, we use the *absolute gradient* of each eigenvector as its feature vector describing its behavior. More precisely, let $Q(G) \in \mathbb{R}^{N \times M}$ be the *incidence matrix* of an input graph $G(V, E, W)$ whose kth column indicates the head and tail of the kth edge $e_k \in E$. However, we note that we need to orient each edge of

G in an arbitrary manner to form a directed graph temporarily in order to construct its incidence matrix. For example, suppose e_k joins nodes i and j, then we can set either $(Q_{ik}, Q_{jk}) = (-\sqrt{W_{ij}}, \sqrt{W_{ij}})$ or $(\sqrt{W_{ij}}, -\sqrt{W_{ij}})$. Of course, we set $Q_{lk} = 0$ for $l \neq i, j$. It is easy to see that $Q(G) Q(G)^\top = L(G)$.

We now define the *DAG pseudometric* between $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$ by

$$d_{\text{DAG}}(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j) := \||\nabla_G|\boldsymbol{\phi}_i - |\nabla_G|\boldsymbol{\phi}_j\|_2 \quad \text{where } |\nabla_G|\boldsymbol{\phi} := \text{abs}.(Q(G)^\top \boldsymbol{\phi}) \in \mathbb{R}_{\geq 0}^M, \tag{6}$$

where $\text{abs}.(\cdot)$ applies the absolute value in the entrywise manner to its argument. We note that $|\nabla_G|\boldsymbol{\phi}$, the absolute gradient of an eigenvector $\boldsymbol{\phi}$, is invariant with respect to: 1) sign flip, i.e., $|\nabla_G|\boldsymbol{\phi} \equiv |\nabla_G|(-\boldsymbol{\phi})$ and 2) choice of sign of each column (i.e., edge orientation) of the incidence matrix $Q(G)$. We also note that this quantity is not a metric but a *pseudometric* because the identity of discernible of the axioms of metric is not satisfied. In order to see the meaning of this quantity, let us analyze its square as follows.

$$\begin{aligned} d_{\text{DAG}}(\boldsymbol{\phi}_i, \boldsymbol{\phi}_j)^2 &= \langle|\nabla_G|\boldsymbol{\phi}_i - |\nabla_G|\boldsymbol{\phi}_j, |\nabla_G|\boldsymbol{\phi}_i - |\nabla_G|\boldsymbol{\phi}_j\rangle_E \\ &= \langle|\nabla_G|\boldsymbol{\phi}_i, |\nabla_G|\boldsymbol{\phi}_i\rangle_E + \langle|\nabla_G|\boldsymbol{\phi}_j, |\nabla_G|\boldsymbol{\phi}_j\rangle_E - 2\langle|\nabla_G|\boldsymbol{\phi}_i, |\nabla_G|\boldsymbol{\phi}_j\rangle_E \\ &= \lambda_i + \lambda_j - \sum_{x \in V} \sum_{y \sim x} |\boldsymbol{\phi}_i(x) - \boldsymbol{\phi}_i(y)| \\ &\qquad \cdot |\boldsymbol{\phi}_j(x) - \boldsymbol{\phi}_j(y)| \quad \text{thanks to } Q(G) Q(G)^\top = L(G) \end{aligned}$$

where $\langle\cdot, \cdot\rangle_E$ is the inner product over edges. The last term of the formula can be viewed as *a global average of absolute local correlation* between eigenvectors. In this sense, this quantity is related to the *Hadamard-product affinity* between eigenvectors proposed by Cloninger and Steinerberger [5]. Note that the computational cost is $O(M)$ for each $d_{\text{DAG}}(\cdot, \cdot)$ evaluation provided that the eigenvectors have already been computed.

Let us demonstrate the power of the DAG pseudometric using the 2D lattice graph $P_7 \times P_3$ used in the previous subsection. Figure 2 displays the embedding of the 21 eigenvectors shown in Fig. 1a into $\mathbb{R}^2$ by computing the distances among all the eigenvectors via Eq. (6) followed by applying the classical Multidimensional Scaling (MDS) [2, Chap. 12]. Of course, in general, when a graph is given, we cannot assume the best embedding dimension a priori. Here we simply embedded into $\mathbb{R}^2$ because the top two eigenvalues of the Gram matrix of the configurations (i.e., the outputs of the MDS) were about twice the third eigenvalue. Figure 2 clearly reveals the natural two-dimensional organization of the eigenvectors, and is similar to a rotated version of Fig. 1b.

2.5 Graph Wavelet Packets

Instead of building the graph wavelet packet dictionary by graph wavelet frames using spectral filters as summarized in Sect. 2.2, one could also accomplish it by generalizing the classical wavelet packets to graphs. The classical wavelet packet decomposition (or
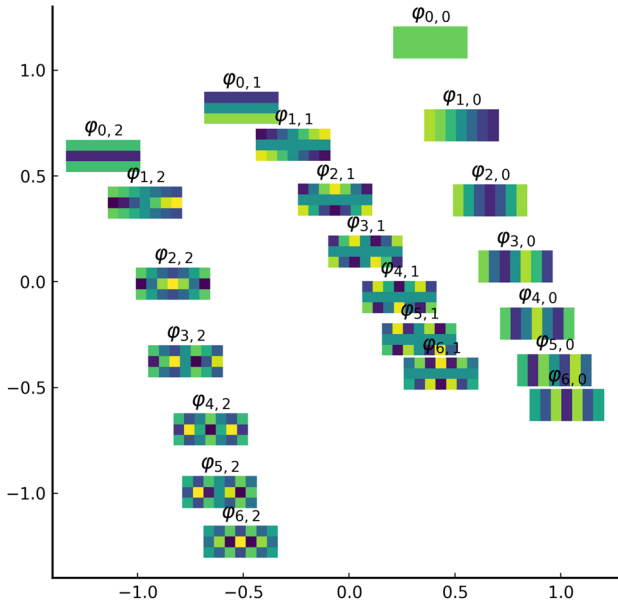
Fig. 2 Embedding of the Laplacian eigenvectors of $P_7 \times P_3$ into $\mathbb{R}^2$ via d_{DAG} and the classical MDS (Color figure online)

dictionary construction) of a 1D discrete signal is obtained by passing it through a full binary tree of filters (each node of the tree represents either low-pass filtered or high-pass filtered versions of the coefficients entering that node followed by the subsampling operation) to get a set of binary-tree-structured coefficients [10], [34, Sect. 8.1]. This basis dictionary for an input signal of length N has up to $N(1 + \log_2 N)$ basis vectors (hence clearly redundant), yet contains more than 1.5^N searchable orthonormal bases (ONBs) [10,59]. For the purpose of efficient signal approximation, the *best-basis algorithm* originally proposed by Coifman and Wickerhauser [10] can find the most desirable ONB (and the expansion coefficients of the input signal) for a given task among such an immense number of ONBs. The best-basis algorithm requires a user-specified cost function, e.g., the ℓ^p-norm ($0 < p \le 1$) of the expansion coefficients for sparse signal approximation, and the basis search starts at the bottom level of the dictionary and proceeds upwards, comparing the cost of the coefficients at the children nodes to the cost of the coefficients at their parents nodes. This best-basis search procedure only costs $O(N)$ operations provided that the expansion coefficients of the input signal have already been computed.

In order to generalize the classical wavelet packets to the graph setting, however, there are two main difficulties: 1) the concept of the frequency domain of a given graph is not well-defined (as discussed in Sect. 2.3); and 2) the relation between the Laplacian eigenvectors and sample locations are much more subtle on general graphs. For 1), we propose to construct a *dual graph*[1] $G^\star$ of the input graph G and view it as

---

[1] Our definition of a dual graph is different from the standard definition in the graph theory; see Remark 3.1 for the details.

the natural spectral domain of G, and use any graph partition method to hierarchically bipartition $G^\star$ instead of building low and high pass filters like the classical case. This can be viewed as the *generalized Littlewood-Paley theory*. For 2), we propose a node-eigenvector organization algorithm called the *pair-clustering algorithm*, which implicitly provides a downsampling process on graphs; see Sect. 4 for the details.

3 Natural Graph Wavelet Packets using *Varimax Rotations*

Given a graph $G = G(V, E, W)$ with $|V| = N$ and the nontrivial distance d between its eigenvectors (e.g., d_{DAG} of Eq. (6)), we build a *dual graph* $G^\star = G^\star(V^\star, E^\star, W^\star)$ by viewing the eigenvectors as its nodes, $V^\star = \{\boldsymbol{\phi}_0, \ldots, \boldsymbol{\phi}_{N-1}\}$, and the nontrivial affinity between eigenvector pairs as its edge weights, $W_{ij}^\star = 1/d(\boldsymbol{\phi}_{i-1}, \boldsymbol{\phi}_{j-1})$, $i, j = 1, 2, \cdots, N$. We note that one can use the alternative and popular Gaussian affinity, i.e., $\exp(-d(\boldsymbol{\phi}_{i-1}, \boldsymbol{\phi}_{j-1})^2/\epsilon)$. This affinity, however, requires a user to select an appropriate scale parameter $\epsilon > 0$, which is not a trivial task as explained in [33], for example. Moreover, our edge weights using the inverse distances tend to connect the eigenvectors more globally compared to the Gaussian affinity with a fixed bandwidth. Using $G^\star$, which is a complete graph, for representing the graph spectral domain and studying relations between the eigenvectors is clearly more *natural* and effective than simply using the eigenvalue magnitudes, as [5,31,49] hinted at. In this section, we will propose one of our graph wavelet packet dictionary constructions solely based on hierarchical bipartitioning of $G^\star$. Basic Steps to generate such a graph wavelet packet dictionary for G are quite straightforward:

Step 1: *Bipartition the dual graph $G^\star$ recursively* via any method, e.g., spectral graph bipartition using the *Fiedler vectors*;

Step 2: *Generate wavelet packet vectors* using the eigenvectors belonging to each subgraph of $G^\star$ that are *well localized on G*.

Note that Step 1 corresponds to bipartitioning the frequency band of an input signal using the *characteristic functions* in the classical setting. Hence, our graph wavelet packet dictionary constructed as above can be viewed as a graph version of the *Shannon* wavelet packet dictionary [34, Sect. 8.1.2].

Remark 3.1 Our definition of the "dual graph" $G^\star$ of a given (primal) graph G is a graph representing the dual geometry/eigenvector domain of the primal graph; in particular, it is not related to the graph-theoretic notion of dual graph (see, e.g., [15, Sect. 1.8]), and does not satisfy the equivalence of the double dual graph and the primal graph. Our definition is also different from that of Leus et al. [28] who defined the dual graph of a primal graph by first assuming that the eigenbasis of a graph shift operator (e.g., the adjacency matrix) of the dual graph is the transpose of the eigenbasis of the graph shift operator of the primal graph. Furthermore, their definition does not work for graph Laplacian matrices.

Remark 3.2 Our dual domain using the DAG pseudometric among eigenvectors is a finite pseudometric space. In order to hierarchically partition the points in that space, our strategy — constructing a complete graph connecting all these points using an
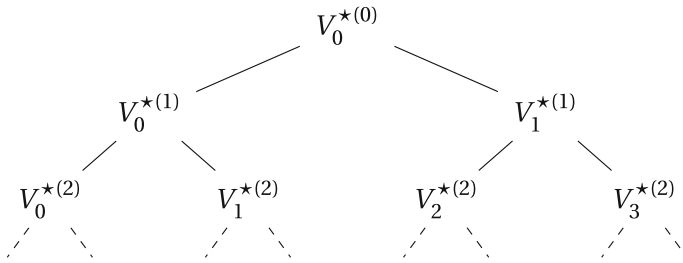
Fig. 3 The hierarchical bipartition tree of the dual graph nodes $V^\star \equiv V_0^{\star(0)}$, which corresponds to the frequency domain bipartitioning used in the classical wavelet packet dictionary

appropriate affinity measure as its edge weights followed by the recursive applications of the spectral graph partitioning, which we will discuss in detail in the next subsection, and which has been used in all of our numerical examples in this article — is the most convenient and efficient approach as far as we know.

We now describe the details of each step of our graph wavelet packet dictionary construction below.

3.1 Hierarchical Bipartitioning of $G^\star$

Let $V_0^{\star(0)} := V^\star$ be the node set of the dual graph $G^\star$, which is simply the set of the eigenvectors of the unnormalized graph Laplacian matrix $L(G)$. Suppose we get the *hierarchical bipartition tree* of $V_0^{\star(0)}$ as shown in Fig. 3. Hence, each $V_k^{\star(j)}$ contains an appropriate subset of the eigenvectors of $L(G)$. As we mentioned earlier, any graph bipartitioning method can be used to generate this hierarchical bipartition tree of $G^\star$. Typically, we use the Fiedler vector of the random-walk normalized graph Laplacian matrix L_{rw} (see Eq. (1)) of each subgraph of $G^\star$, whose use is preferred over that of L or L_{sym} as von Luxburg discussed [61].

Remark 3.3 We recursively apply the above bipartition algorithm until we reach $j = j_{max} > 0$, where each $V_k^{\star(j_{max})}$, $k = 0 : N - 1$, contains a single eigenvector. Note that our previous graph basis dictionaries, i.e., HGLET [19], GHWT [18], and eGHWT [52], also constructed such "full" hierarchical bipartition trees in the primal (input graph) domain, not in the dual domain. Note also that during the hierarchical bipartition procedure, some $V_k^{\star(j)}$ may become a singleton before reaching $j = j_{max}$. If this happens, such a subset is copied to the next lower level $j + 1$. See [21] for the detailed explanation of such situations. We can also stop the recursion at some level $J(< j_{max})$, of course. Below, we denote j_{max} as the deepest possible level at which every subset becomes a singleton for the first time whereas we denote $J(\le j_{max})$ as a more general deepest level specified by a user.

Figure 4 demonstrates the above strategy for the 2D lattice graph discussed in Sect. 2, whose dual domain geometry together with the graph Laplacian eigenvectors belonging to $V_0^{\star(0)}$ was displayed in Fig. 2. The thick red line indicates the first split of
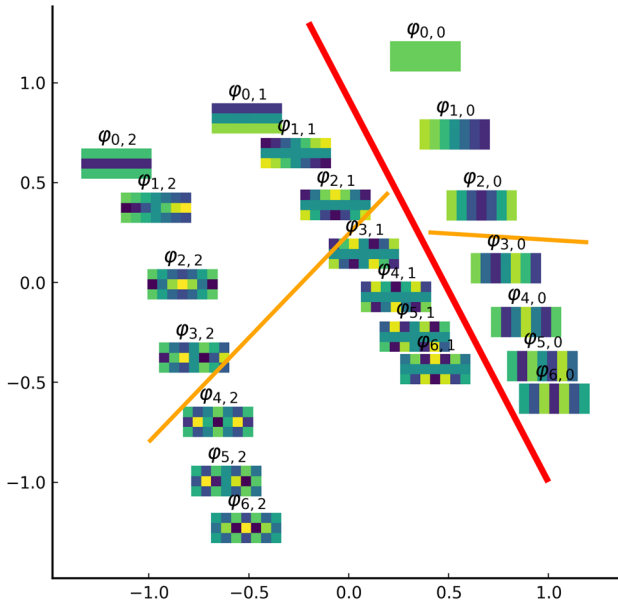
Fig. 4 The result of the hierarchical bipartition algorithm applied to the dual geometry of the 2D lattice graph $P_7 \times P_3$ shown in Fig. 2 with $J = 2$. The thick red line indicates the bipartition at $j = 1$ while the orange lines indicate those at $j = 2$ (Color figure online)

$V_0^{\star(0)}$, i.e., all the eigenvectors above this red line belong to $V_0^{\star(1)}$ while those below it belong to $V_1^{\star(1)}$. Then, our hierarchical bipartition algorithm further splits them into $\left\{ V_0^{\star(2)}, V_1^{\star(2)} \right\}$ and $\left\{ V_2^{\star(2)}, V_3^{\star(2)} \right\}$, respectively. This two-level bipartition pattern is quite reasonable and natural considering the fact that the size of the original rectangle is 7×3, both of which are odd integers.

3.2 Localization on *G* via Varimax Rotation

For realizing Step 2 of the above basic algorithm, we propose to use the *varimax rotation* on the eigenvectors in $V_k^{\star(j)}$ for each j and k. Let $\Phi_k^{(j)} \in \mathbb{R}^{N \times N_k^j}$ be a matrix whose columns are the eigenvectors belonging to $V_k^{\star(j)}$. A varimax rotation is an orthogonal rotation, originally proposed by Kaiser [25] and often used in *factor analysis* (see, e.g., [37, Chap. 11]), to maximize the variances of energy distribution (or a scaled version of the *kurtosis*) of the input column vectors, which can also be interpreted as the *approximate entropy minimization of the distribution of the eigenvector components* [47, Sect. 3.2]. For the implementation of the varimax rotation algorithm, see Appendix A, which is based on the Basic Singular Value (BSV) Varimax Algorithm of [23]. Thanks to the orthonormality of columns of $\Phi_k^{(j)}$, this is equivalent to finding an orthogonal rotation that maximizes the overall *4th order moments*, i.e.,

$$\Psi_k^{(j)} := \Phi_k^{(j)} \cdot R_k^{(j)}, \quad \text{where } R_k^{(j)} = \arg \max_{R \in \mathrm{SO}(N_k^j)} \sum_{p=1}^{N} \sum_{q=1}^{N_k^j} \left[\left(\Phi_k^{(j)} \cdot R \right)^4 \right]_{p,q}. \quad (7)$$

The column vectors of $\Psi_k^{(j)}$ are *more "localized" in the primal domain G* than those of $\Phi_k^{(j)}$. This type of localization is important since the graph Laplacian eigenvectors in $\Phi_k^{(j)}$ are of *global* nature in general. We also note that the column vectors of $\Psi_k^{(j)}$ are orthogonal to those of $\Psi_{k'}^{(j')}$ as long as the latter is neither a direct ancestor nor a direct descendant of the former. Hence, Steps 1 and 2 of the above basic algorithm truly generate the graph wavelet packet dictionary for an input graph signal. We refer to this graph wavelet packet dictionary $\left\{ \Psi_k^{(j)} \right\}_{j=0:J; \, k=0:2^j-1}$ generated by this algorithm as the *Varimax Natural Graph Wavelet Packet* (VM-NGWP) dictionary. One can run the *best-basis algorithm* of Coifman-Wickerhauser [10] on this dictionary to extract the ONB most suitable for a task at hand (e.g., an efficient graph signal approximation) once an appropriate cost function is specified (e.g., the ℓ^p-norm minimization, $0 < p \le 1$). Note also that it is easy to extract a graph Shannon wavelet basis from this dictionary by specifying the appropriate dual graph nodes, i.e., $\Psi_1^{(1)}, \Psi_1^{(2)}, \ldots, \Psi_1^{(J)}$, and the father wavelet vectors $\Psi_0^{(J)}$ where $J (\le j_{\max})$ is the user-specified deepest level of the hierarchical bipartition tree. We point out that the meaning of the level index j in our NGWP dictionaries is different from that in the general graph wavelet frames (4) discussed in Sect. 2.2: in our NGWP dictionaries, a smaller j corresponds to a finer and more localized (in the primal graph domain) basis vector in $V_k^{\star(j)}$.

Let us now demonstrate that our algorithm actually generates the classical *Shannon* wavelet packets dictionary [34, Sect. 8.1.2] when an input graph is the simple path P_N. Note that the varimax rotation algorithm does not necessarily sort the vectors as shown in Fig. 5 because the minimization in Eq. (7) is the same modulo to any permutation of the columns and any sign flip of each column. In other words, to produce Fig. 5, we carefully applied sign flip to some of the columns, and sorted the whole columns so that each subfigure simply shows translations of the corresponding wavelet packet vectors.

Let us also demonstrate how some VM-NGWP basis vectors of the 2D lattice graph $P_7 \times P_3$ look like. Figure 6 shows such VM-NGWP basis vectors with $J = 2$. Those basis vectors are placed at the same locations as the graph Laplacian eigenvectors in the dual domain shown in Fig. 4 for the demonstration purpose. It is quite clear that those VM-NGWP basis vectors are more localized in the primal graph domain than those graph Laplacian eigenvectors shown in Fig. 4. We note that we determined the index l in $\psi_{k,l}$ for each k in such a way that the main features of the VM-NGWP basis vectors translates nicely in the horizontal and vertical directions, and some sign flips were applied as in the case of the 1D Shannon wavelet packets shown in Fig. 5. As one can see, like the classical wavelet packet vectors on a rectangle, $\{\psi_{0,l}\}_{l=0:2}$, are the father wavelets and clearly function as local averaging operators along the horizontal direction while $\{\psi_{1,l}\}_{l=0:3}$, work as localized first order differential operators along the horizontal direction. On the other hand, $\{\psi_{2,l}\}_{l=0:2}$ work as localized first order differ-
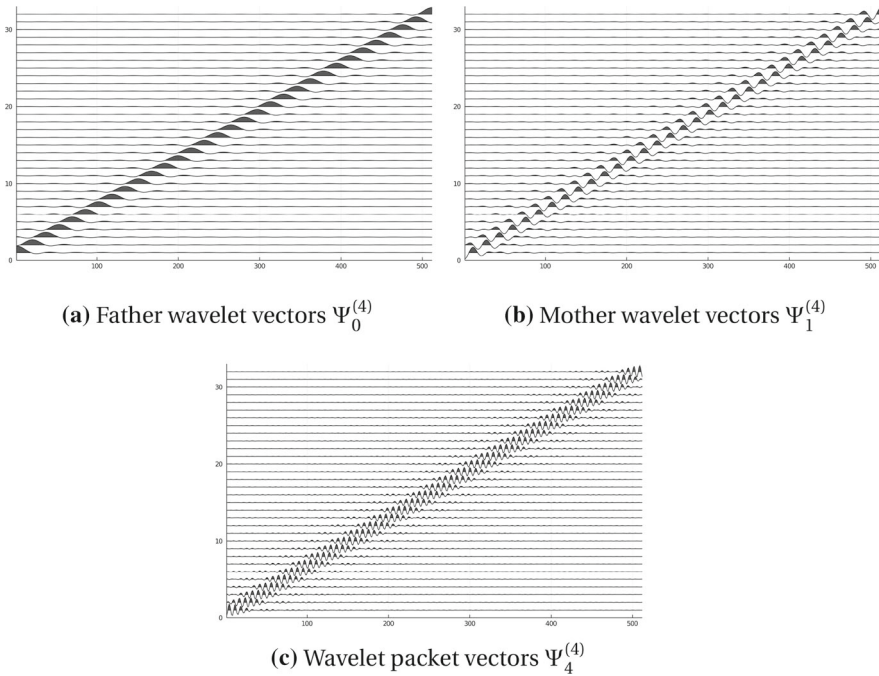
(a) Father wavelet vectors $\Psi_0^{(4)}$

(b) Mother wavelet vectors $\Psi_1^{(4)}$

(c) Wavelet packet vectors $\Psi_4^{(4)}$

Fig. 5 Some of the Shannon wavelet packet vectors on P_{512}

ential operators along the vertical direction; $\{\psi_{2,l}\}_{l=3:6}$ work as localized second order differential operators along the vertical direction; $\{\psi_{3,l}\}_{l=0:3}$ work as localized mixed differential operators; and finally, $\{\psi_{3,l}\}_{l=4:6}$ work as localized Laplacian operators.

3.3 Computational Complexity

The varimax rotation algorithm of Appendix A is of iterative nature and is an example of the BSV algorithms [23]: for each iteration at the dual node set $V_k^{\star(j)}$, it requires computing the full Singular Value Decomposition (SVD) of a matrix of size $N_k^j \times N_k^j$ representing a gradient of the objective function, which itself is computed by multiplying matrices of sizes $N_k^j \times N$ and $N \times N_k^j$. The convergence is checked with respect to the relative error between the current and previous gradient estimates measured in the nuclear norm (i.e., the sum of the singular values). For our numerical experiments in Sect. 5, we set the maximum iteration as 1000 and the error tolerance as 10^{-12}. Therefore, to generate $\Psi_k^{(j)}$ for each (j, k), the computational cost in the worst case scenario is $O\left(c \cdot (N_k^j)^3 + N \cdot (N_k^j)^2\right)$ where $c = 1000$ and the first term accounts for the SVD computation and the second does for the matrix multiplication. For a perfectly balanced and fully developed bipartition tree with $N = 2^{j_{\max}}$, we have $N_k^j = 2^{j_{\max}-j}$, $j = 0 : j_{\max}$, $k = 0 : 2^j - 1$. Hence we have:
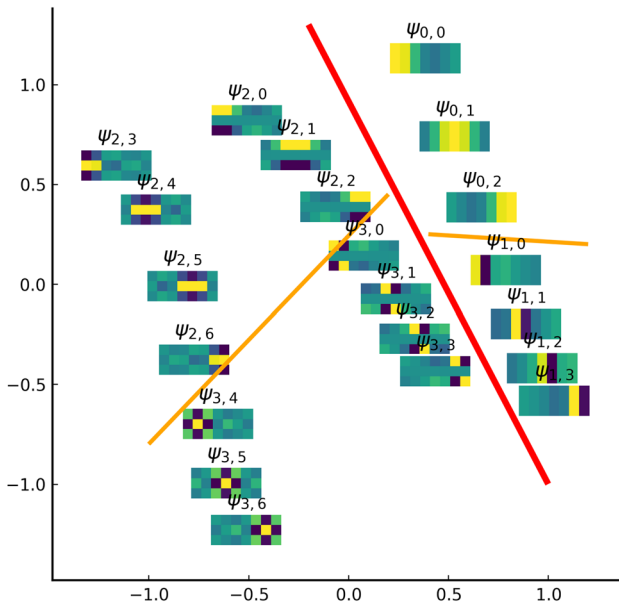
Fig. 6 The VM-NGWP basis vectors of the 2D lattice graph $P_7 \times P_3$ computed by the varimax rotations in the hierarchically partitioned dual domain shown in Fig. 4. Note that the column vectors of the basis matrix $\Psi_k^{(2)}$ are denoted as $\psi_{k,l}, l = 0, 1, \ldots$, in this figure instead of $\psi_{k,l}^{(2)}$ for simplicity (Color figure online)

$$\sum_{k=0}^{2^j-1} (N_k^j)^2 = \sum_{k=0}^{2^j-1} 2^{2(j_{\max}-j)} = 2^{2j_{\max}-2j} \cdot 2^j = N^2 \cdot 2^{-j}, \qquad (8)$$

and

$$\sum_{k=0}^{2^j-1} (N_k^j)^3 = \sum_{k=0}^{2^j-1} 2^{3(j_{\max}-j)} = 2^{3j_{\max}-3j} \cdot 2^j = N^3 \cdot 2^{-2j}.$$

Note that at the bottom level $j = j_{\max}$, each node is a leaf containing only one eigenvector, and there is no need to do any rotation estimation and computation. Note also that at the root level $j = 0$, the columns of $\Phi_0^{(0)}$ span the whole $\mathbb{R}^N$, and we know that the varimax rotation turns $\Phi_0^{(0)}$ into the identity matrix (or its permuted version). Hence, we do not need to run the varimax rotation algorithm on the root node. Finally, summing the cost $O\left(c \cdot (N_k^j)^3 + N \cdot (N_k^j)^2\right)$ from $j = 1$ to $j_{\max} - 1$, the total worst case computational cost becomes $O((1 + c/3)N^3 - 2N^2 - 4c/3N)$. So after all, it is an $O(N^3)$ algorithm. In practice, the convergence is often achieved with less than 1000 iterations at each node except possibly for the nodes with small j where N_k^j is large. For example, when computing the VM-NGWP dictionary for the path graph

P_{512} ($j_{\max} = 9$) shown in Fig. 5, the average number of iterations over all the dual graph nodes $\left\{ V_k^{\star(j)} \right\}_{j=0:9; \, k=0:2^j-1}$ was 68.42 with the standard deviation 98.09.

4 Natural Graph Wavelet Packets Using *Pair-Clustering*

Another way to construct a natural graph wavelet packet dictionary is to mimic the convolution and subsampling strategy of the classical wavelet packet dictionary construction: form a binary tree of spectral filters in the dual domain via $\left\{ V_k^{\star(j)} \right\}_{j=0:J; \, k=0:2^j-1}$ and then perform the filtering/downsampling process based on the relations between the sampling points (primal nodes) and the eigenvectors of $L(G)$. In order to fully utilize such relations, we look for a coordinated pair of partitions on G and $G^\star$, which is realized by our *pair-clustering* algorithm described below. We will first describe the one-level pair-clustering algorithm and then proceed to the hierarchical version.

4.1 One-Level Pair-Clustering

Suppose we partition the dual graph $G^\star$ into $K \geq 2$ clusters using any method including the spectral clustering [61] as we used in the previous section. Let $V_1^\star, \ldots, V_K^\star$ be those mutually disjoint K clusters of the nodes $V^\star$, i.e., $V^\star = \bigsqcup\limits_{k=1}^{K} V_k^\star$, which is also often written as $\bigoplus\limits_{k=1}^{K} V_k^\star$. Denote the cardinality of each cluster as $N_k := |V_k^\star|$, $k = 1 : K$, and we clearly have $\sum\limits_{k=1}^{K} N_k = N$. Then, we also partition the primal graph nodes V into mutually disjoint K clusters, $V_1, \ldots, V_K$ with the constraint that $|V_k| = |V_k^\star| = N_k, k = 1 : K$, and the members of V_k and $V_k^\star$ are as "closely related" as possible. The purpose of partitioning V is to select appropriate primal graph nodes as sampling points around which the graph wavelet packet vectors using the information on $V_k^\star$ are localized. With a slight abuse of notation, let V also represent a collection of the standard basis vectors in $\mathbb{R}^N$, i.e., $V := \{\delta_1, \ldots, \delta_N\}$, where $\delta_k(k) = 1$ and 0 otherwise. In order to formalize this constrained clustering of V, we define the *affinity measure* α between V_k and $V_k^\star$ as follows:

$$\alpha(V_k, V_k^\star) := \sum_{\delta \in V_k, \, \phi \in V_k^\star} |\langle \delta, \phi \rangle|^2, \tag{9}$$

where $\langle \cdot, \cdot \rangle$ is the standard inner product in $\mathbb{R}^N$. Note that $\alpha(V, V^\star) = \sum\limits_{\delta \in V, \, \phi \in V^\star} |\langle \delta, \phi \rangle|^2 = \sum\limits_{\phi \in V^\star} \|\phi\|^2 = N$. Denote the feasible partition set as

$$U(V; N_1, \ldots, N_K) := \left\{ (V_1, \ldots, V_K) \,\middle|\, \bigsqcup_{k=1}^{K} V_k = V; \, |V_k| = N_k, k = 1 : K \right\}.$$

Now we need to solve the following optimization problem for a given partition of $V^\star = \bigsqcup_{k=1}^{K} V_k^{\star}$:

$$(V_1, \ldots, V_K) = \arg \max_{(V_1, \ldots, V_K) \in U(V; N_1, \ldots, N_K)} \sum_{k=1}^{K} \alpha(V_k, V_k^{\star}) \qquad (10)$$

This is a discrete optimization problem. In general, it is not easy to find the global optimal solution except for the case $K = 2$. For $K = 2$, we can find the desired partition of V by the following greedy algorithm: 1) compute score(δ) $:= \alpha(\{\delta\}, V_1^\star) - \alpha(\{\delta\}, V_2^\star)$ for each $\delta \in V$; 2) select N_1 δ's in V that give the largest N_1 values of score($\cdot$), set them as V_1, and set $V_2 = V \setminus V_1$.

When $K > 2$, we can find a *local optimum* by the similar strategy as above: 1) compute the values $\alpha(\{\delta\}, V_1^\star)$ for each $\delta \in V$; 2) select N_1 δ's giving the largest N_1 values, and set them as V_1; 3) compute the values $\alpha(\{\delta\}, V_2^\star)$ for each $\delta \in V \setminus V_1$, select N_2 δ's giving the largest N_2 values, and set them as V_2; 4) repeat the above process to produce $V_3, \ldots, V_K$. While this greedy strategy does not reach the global optimum of Eq. (10), we find that empirically the algorithm attains a reasonably large value of the objective function. We note that our one-level pair-clustering problem is a particular example of the so-called *submodular welfare problem* [62] with cardinality constraints; however, we will not pursue this direction for a general $K > 2$ with the one-level pair clustering. Rather, we will apply it with $K = 2$ in a hierarchical manner, which will be discussed next.

4.2 Hierarchical Pair-Clustering

In order to build a multiscale graph wavelet packet dictionary, we develop a hierarchical (i.e., multilevel) version of the pair-clustering algorithm. First, let us assume that the hierarchical bipartition tree of $V^\star$ is already computed using the same algorithm discussed in Sect. 3.1. We now begin with level $j = 0$ where $V_0^{(0)}$ is simply $V = \{\delta_1, \delta_2, \cdots, \delta_N\}$ and $V_0^{\star(0)}$ is $V^\star = \{\phi_0, \phi_1, \cdots, \phi_{N-1}\}$. Then, we perform one-level pair-clustering algorithm ($K = 2$) to get $\left(V_0^{\star(1)}, V_0^{(1)} \right)$ and then $\left(V_1^{\star(1)}, V_1^{(1)} \right)$. We iterate the above process to generate paired clusters $\left(V_k^{\star(j)}, V_k^{(j)} \right)$, $j = 0 : J$, $k = 0 : 2^j - 1$. Note that the hierarchical pair-clustering algorithm ensures nestedness in both the primal node domain V and the dual/eigenvector domain $V^\star$.

4.3 Generating the NGWP Dictionary

Once we generate two hierarchical bipartition trees $\left\{ V_k^{(j)} \right\}$ and $\left\{ V_k^{\star(j)} \right\}$, we can proceed to generate the NGWP vectors $\left\{ \Psi_k^{(j)} \right\}$ that are necessary to form an NGWP dictionary. For each $\delta_l \in V_k^{(j)}$, we first compute the orthogonal projection of δ_l onto the span of $V_k^{\star(j)}$, i.e., span $\left(\Phi_k^{(j)} \right)$ where $\Phi_k^{(j)}$ are those eigenvectors of $L(G)$ belonging to $V_k^{\star(j)}$. Unfortunately, $\Phi_k^{(j)} \left(\Phi_k^{(j)} \right)^{\mathsf{T}} \delta_l$ and $\Phi_k^{(j)} \left(\Phi_k^{(j)} \right)^{\mathsf{T}} \delta_{l'}$ are not mutually orthogonal for $\delta_l, \delta_{l'} \in V_k^{(j)}$ in general. Hence, we need to perform orthogonalization of the vectors $\left\{ \Phi_k^{(j)} \left(\Phi_k^{(j)} \right)^{\mathsf{T}} \delta_l \right\}_l$. We use the *modified Gram-Schmidt with $\ell^p (0 < p < 2)$ pivoting orthogonalization* (MGSLp) [7] to generate the orthonormal graph wavelet packet vectors associated with $V_k^{\star(j)}$ (and hence also $V_k^{(j)}$). This MGSLp algorithm listed in Appendix B tends to generate localized orthonormal vectors because the ℓ^p-norm[2] pivoting promotes sparsity. We refer to the graph wavelet packet dictionary $\left\{ \Psi_k^{(j)} \right\}_{j=0:J; \, k=0:2^j-1}$ generated by this algorithm as the *Pair-Clustering Natural Graph Wavelet Packet* (PC-NGWP) dictionary.

   Let us now briefly discuss the performance of the PC-NGWP dictionary on the same examples in Sect. 3, i.e., P_{512} and $P_7 \times P_3$, without displaying figures to save pages. We essentially obtained the similar wavelet packet vectors in both cases as those shown in Figs. 5 and 6 using the VM-NGWP dictionaries; yet they are not exactly the same: the localization of those PC-NGWP vectors in the primal node domain is worse (e.g., with larger sidelobes) than that of the VM-NGWP vectors mainly due to the MGSLp orthogonalization procedure (even if it promoted sparsity).

4.4 Computational Complexity

At each $V_k^{\star(j)}$ of the hierarchical bipartition tree of the dual graph $G^\star$, the orthogonal projection of the standard basis vectors in $V_k^{(j)}$ onto span $\left(\Phi_k^{(j)} \right)$ and the MGSLp procedure are the two main computational burden for our PC-NGWP dictionary construction. The orthogonal projection costs $O \left(N \cdot (N_k^j)^2 + N \cdot N_k^j \right)$ while the MGSLp costs $O \left(2N \cdot (N_k^j)^2 \right)$. Hence, the dominating cost for this procedure is $O \left(3N \cdot (N_k^j)^2 \right)$ for each (j, k). And we need to sum up this cost on all the tree nodes. Let us analyze the special case of the perfectly balanced and fully developed bipartition tree with $N = 2^{j_{\max}}$ as we did for the VM-NGWP in Sect. 3.3. In this case, the bipartition tree has $1 + j_{\max}$ levels, and $N_k^j = 2^{j_{\max}-j}$, $k = 0 : 2^j - 1$. So, for the jth level, using Eq. (8), we have $O(3N^3 \cdot 2^{-j})$. Finally, by summing this from $j = 1$ to $j_{\max} - 1$ (again, no computation is needed at the root and the bot-

tom levels), the total cost for PC-NGWP dictionary construction in this ideal case is: $O(3N^3 \cdot (1 - 2/N)) \approx O(3N^3)$. So, it still requires $O(N^3)$ operations; the difference from that of the VM-NGWP is the constants, i.e., 3 (PC-NGWP) vs $1 + 1000/3 \approx 334$ (the worst case VM-NGWP).

5 Applications in Graph Signal Approximation

In this section, we demonstrate the usefulness of our proposed NGWP dictionaries in efficient approximation of graph signals on two graphs, and compare the performance with the other previously proposed methods: the global graph Laplacian eigenbasis; the HGLET best basis [19]; the graph Haar basis; the graph Walsh basis; the GHWT coarse-to-fine (c2f) best basis [18]; the GHWT fine-to-coarse (f2c) best basis [18]; and the eGHWT best basis [52]. Recall that the HGLET is a graph version of the Hierarchical Block DCT dictionary, which is based on the hierarchical bipartition tree of a primal graph, as briefly discussed in Introduction. Although the HGLET can choose three different types of graph Laplacian eigenvectors of L, $L_{\rm rw}$, and $L_{\rm sym}$ at each subgraph (see Eq. (1)), we only use those of the unnormalized L at each subgraph in order to compare its performance in a fair manner with the NGWP dictionaries that are based on the eigenvectors of $L(G)$. Note also that the graph Haar basis is a particular basis choosable from the GHWT f2c dictionary and the eGHWT dictionary while the graph Walsh basis is choosable from both versions of the GHWT dictionaries as well as the eGHWT; see [21,52] for the details. We use the ℓ^1-norm minimization as the best-basis selection criterion for all the best bases in our experiments. The edge weights of the dual graph $G^\star$ are the reciprocals of the DAG pseudometric between the corresponding eigenvectors of $L(G)$ as defined in Eq. (6). For a given graph G and a graph signal f defined on it, we decompose f into those dictionaries and select those bases first. Then, to measure the approximation performance, we sort the expansion coefficients in the nonincreasing order of their magnitude, and use the top k most significant terms to approximate f where k starts from 0 up to about 50% of the total number of terms, or more precisely, $\lfloor 0.5N \rfloor + 1$. All of the approximation performance is measured by the relative ℓ^2 approximation error with respect to the fraction of coefficients retained, which we denote FCR for simplicity.

5.1 Sunflower Graph Signals Sampled on Images

We consider the so-called "sunflower" graph shown in Fig. 7a. This particular graph has 400 nodes and each edge weight is set as the reciprocal of the Euclidean distance between the endpoints of that edge. Consistently counting the number of spirals in such a sunflower graph gives rise to the *Fibonacci numbers*: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, . . . ; see Fig. 7a. We also note that the majority of nodes (374 among 400) have degree 4 while there are eight nodes with degree 2, 17 nodes with degree 3, and the central node has the greatest degree 9. See, e.g., [35,60] and our code SunFlowerGraph.jl in [30], for algorithms to construct such sunflower grids and graphs. We can also view such a distribution of nodes as a simple
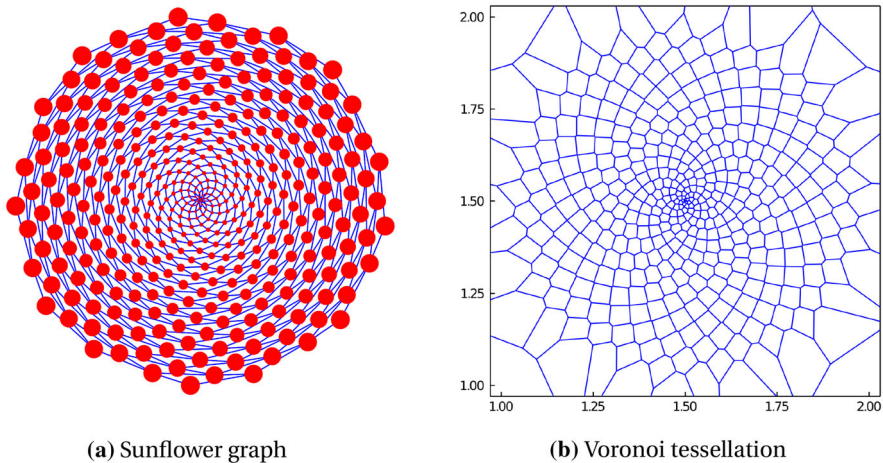
(a) Sunflower graph

(b) Voronoi tessellation

Fig. 7 Sunflower graph ($N = 400$) (**a**); node radii vary for visualization purpose; its Voronoi tessellation (**b**) (Color figure online)

model of the distribution of *photoreceptors in mammalian visual systems* due to cell generation and growth; see, e.g., [45, Chap. 9]. Such a viewpoint motivates us the following sampling scheme: 1) overlay the sunflower graph on several parts of the standard Barbara image; 2) construct the Voronoi tessellation of the bounding square region with the nodes of the sunflower graph as its seeds as shown in Fig. 7b; 3) compute the average pixel value within each Voronoi cell; and 4) assign that average pixel value to the corresponding seed/node.[3] See [63] for more about the relationship between the Voronoi tessellation and the sunflower graph. We also note that for generating the Voronoi tessellation, we used the following open source Julia packages developed by the JuliaGeometry team [58]: `VoronoiDelaunay.jl`; `VoronoiCells.jl`; and `GeometricalPredicates.jl`. For our numerical experiments, we sampled two different regions: her left eye and pants, where quite different image features are represented, i.e., a piecewise-smooth image containing oriented edges and a textured image with directional oscillatory patterns, respectively.

First, let us discuss our approximation experiments on Barbara's eye graph signal, which are shown in Fig. 8. From Fig. 8c, we observe the following: 1) the VM-NGWP best basis performed best closely followed by the PC-NGWP best basis; 2) the HGLET best basis chose the global graph Laplacian eigenbasis, which worked relatively well particularly up to $FCR \approx 0.27$; and 3) those bases chosen from the Haar-Walsh wavelet packet dictionaries did not perform well; among them, the eGHWT best basis performed well in the range $FCR \gtrsim 0.27$. Note also that the GHWT c2f best basis turned out to be the graph Walsh basis for this graph signal. These observations can be attributed to the fact that this Barbara's eye graph signal is not of piecewise-*constant* nature; rather, it is a *locally smooth* graph signal. Hence, the NGWP dictionaries con-

---

[3] If a Voronoi cell does not contain any original image pixels (which occurs at some tiny cells around the center), we bilinearly interpolate the pixel value at the node location using the nearest image pixel values.

(a) The sunflower graph overlaid on Barbara's left eye

(b) Barbara's left eye as an input graph signal
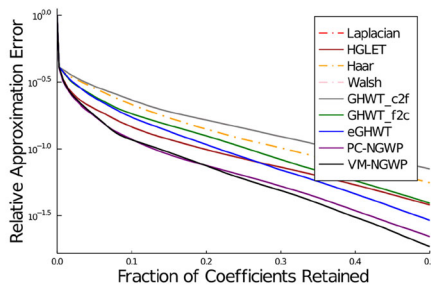
(c) Approximation performance of various methods

Fig. 8 Barbara's left eye region sampled on the sunflower graph nodes (**a**) as a graph signal (**b**); the relative ℓ^2 approximation errors by various methods (**c**) (Color figure online)

taining smooth *and* localized basis vectors made a difference in performance compared to the global graph Laplacian eigenbasis and the eGHWT best basis.

In order to examine what kind of basis vectors were chosen as the best basis to approximate this Barbara's eye signal, we display the 16 most significant VM-NGWP best basis vectors in Fig. 9. The corresponding PC-NGWP best basis vectors are relatively similar; hence they are not shown here. We note that many of these top basis vectors essentially work as oriented edge detectors for Barbara's eye. For example, $\boldsymbol{\psi}_{5,2}^{(7)}$ (Fig. 9g) and $\boldsymbol{\psi}_{5,1}^{(6)}$ (Fig. 9k) try to capture her eyelid while $\boldsymbol{\psi}_{2,18}^{(3)}$ (Fig. 9j), $\boldsymbol{\psi}_{11,1}^{(6)}$ (Fig. 9l), and $\boldsymbol{\psi}_{15,3}^{(6)}$ (Fig. 9n) do the same for her iris and sclera. The other basis vectors take care of shading and peripheral features of her eye region. We also note that seven among these top 16 best basis vectors are the global graph Laplacian eigenvectors; see Fig. 9a, b, e, h, i, m, o.

Now, let us discuss our second approximation experiments: Barbara's pants region as an input graph signal as shown in Fig. 10. The nature of this graph signal is completely different from the eye region: it is dominated by directional oscillatory patterns of her pants. From Fig. 10c, we observe the following: 1) the NGWP best bases and the eGHWT best basis performed very well and competitively; the NGWP best bases performed better than the eGHWT best basis up to $FCR \approx 0.2$ while the latter out-
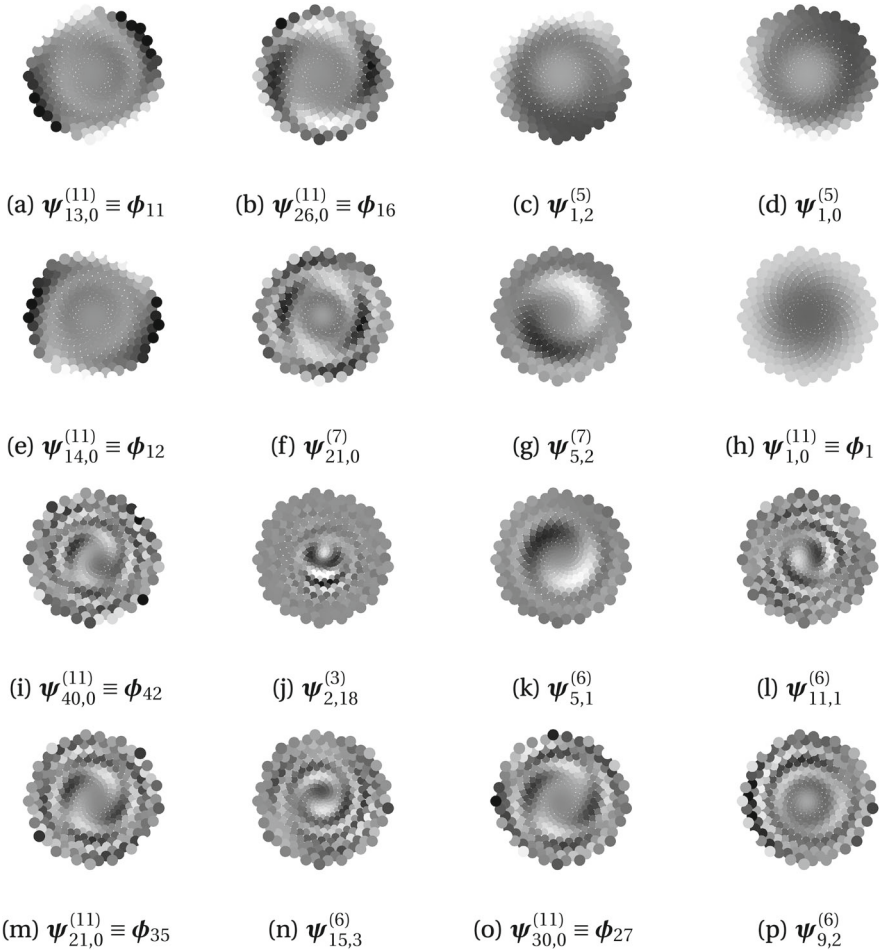
Fig. 9 Sixteen most significant VM-NGWP best basis vectors (the DC vector not shown) for Barbara's eye. The basis vector amplitudes within $(-0.15, 0.15)$ are mapped to the grayscale colormap

performed all the others for $FCR \gtrsim 0.2$; 2) the GHWT f2c best basis performed relatively well behind those three bases; 3) there is a substantial gap in performance between those four bases and the rest: the graph Haar basis; the GHWT c2f best basis; and the HGLET best basis. Note that similarly to the case of Barbara's eye, the latter two best bases coincide with the graph Walsh basis and the global graph Laplacian eigenbasis, respectively. We knew that the eGHWT is known to be quite efficient in capturing oscillating patterns as shown by Shao and Saito for the graph setting [52] and by Lindberg and Villemoes for the classical non-graph setting [32]. Hence, it is a good thing to observe that our NGWPs are competitive with the eGHWT for this type of textured signal.

Figure 11 shows the 16 most significant VM-NGWP best basis vectors for approximating Barbara's pants signal. We note that the majority of these basis vectors are
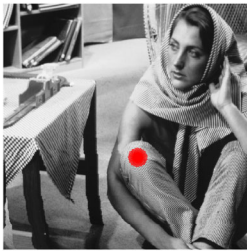
(a) The sunflower graph overlaid on Barbara's pants/knee region
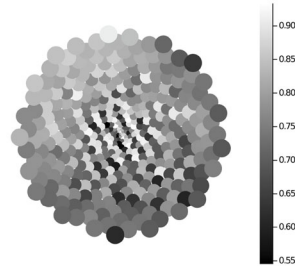
(b) Barbara's pants region as an input graph signal
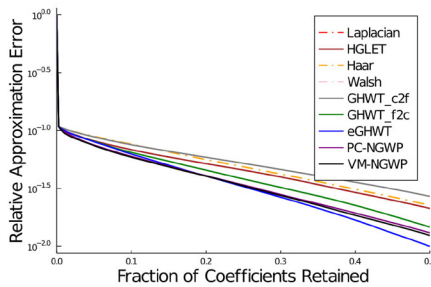
(c) Approximation performance of various methods

Fig. 10 Barbara's pants region sampled on the sunflower graph nodes (**a**) as a graph signal (**b**); the relative ℓ^2 approximation errors by various methods (**c**) (Color figure online)

of high-frequency nature than those for the eye signal shown in Fig. 9, which reflect the oscillating anisotropic patterns of her pants. The basis vectors $\boldsymbol{\psi}_{3,1}^{(6)}$ (Fig. 11a), $\boldsymbol{\psi}_{9,0}^{(11)}$ (Fig. 11j), and $\boldsymbol{\psi}_{1,0}^{(11)}$ (Fig. 11p) take care of shading in this region while the other basis vectors extract oscillatory patterns of various scales. We also note that four among these top 16 best basis vectors are the global graph Laplacian eigenvectors; see Fig. 11h, j, k, p.

5.2 Toronto Street Network

We obtained the street network data of the City of Toronto from its open data portal.[4] Using the street names and intersection coordinates included in the dataset, we construct the graph representing the street network there with $N = 2275$ nodes and $M = 3381$ edges. Figure 12 displays this graph. As before, each edge weight was set as the reciprocal of the Euclidean distance between the endpoints of that edge.

We analyze two graph signals on this street network: 1) spatial distribution of the street intersections and 2) pedestrian volume measured at each intersection. The first graph signal was constructed by counting the number of the nodes within the disk
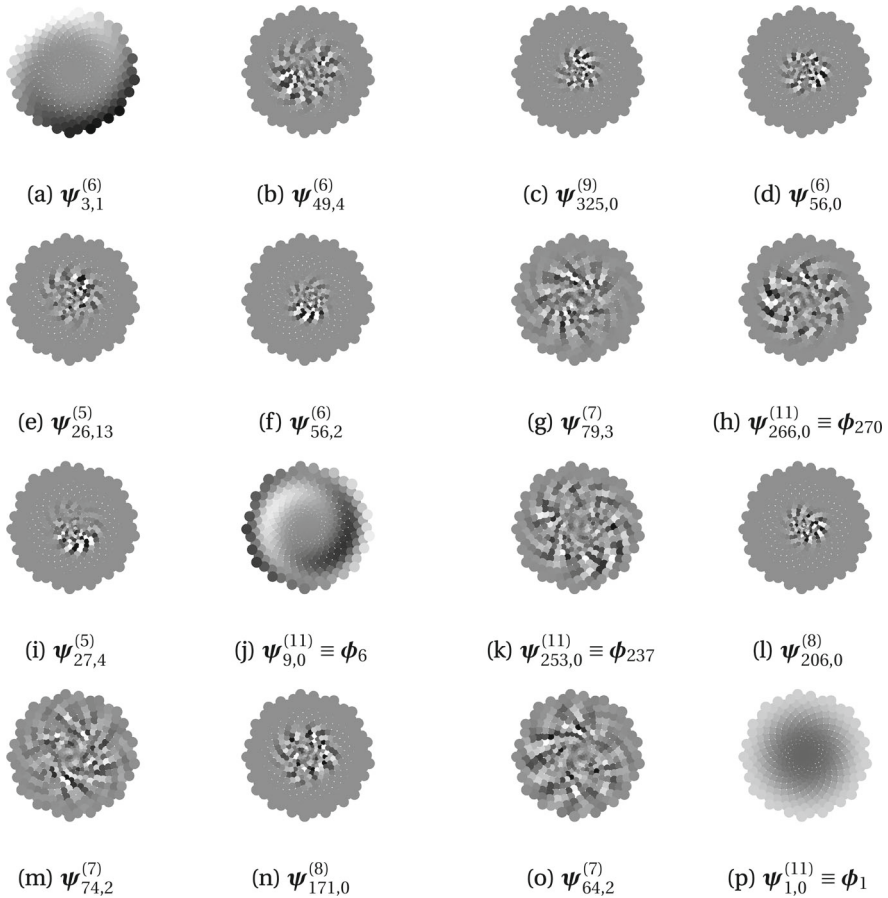
Fig. 11 Sixteen most significant VM-NGWP best basis vectors (the DC vector not shown) for Barbara's pants. The basis vector amplitudes within $(-0.15, 0.15)$ are mapped to the grayscale colormap

of radius 4.7 km centered at each node. In other words, this is a smooth version of histogram of the distribution of street intersections computed with the overlapping circular bins of equal size. The longest edge length measured in the Euclidean distance among all these 3381 edges was chosen as this radius of this disk, which is located at the northeast corner of this graph as one can easily see in Fig. 12a. The second graph signal is the most recent 8 peak-hour pedestrian volume counts collected at intersections (i.e., nodes in this graph) where there are traffic signals. The dataset was collected between the hours of 7:30 am and 6:00 pm, over the period of 03/22/2004–02/28/2018. From Fig. 12b, we observe that qualitative behaviors of these error curves are relatively similar to those of Barbara's eye signal shown in Fig. 8c. More precisely, 1) NGWP best bases outperformed all the others and the difference between the VM-NGWP and the PC-NGWP is negligible; 2) the HGLET best basis chose the global graph Laplacian eigenbasis, which worked quite well following the NGWP best bases;
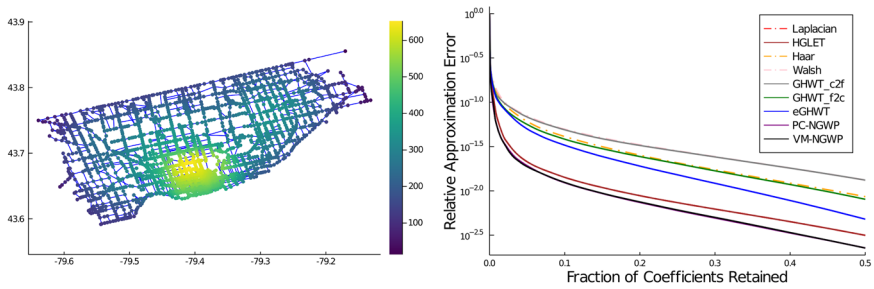
(a) A smooth spatial distribution of the street inter-
sections

(b) Approximation performance of various meth-
ods

Fig. 12 A graph signal representing the smooth spatial distribution of the street intersections on the Toronto street network (**a**). The horizontal and vertical axes of this plot represent the longitude and latitude geo-coordinates of this area, respectively. The results of our approximation experiments (**b**) (Color figure online)

3) those bases based on the Haar-Walsh wavelet packet dictionaries did not perform well.

In order to examine what kind of basis vectors were chosen to approximate this smooth histogram of street intersections, we display the most important 16 VM-NGWP best basis vectors in Fig. 13. We note that these top basis vectors exhibit different spatial scales. The basis vectors with levels $j = 5$ and $j = 6$ are relatively localized to specific regions of Toronto. For example, $\psi_{4,4}^{(5)}$ (Fig. 13h) tries to differentiate the eastern neighbor of the dense downtown region along the north-south direction while $\psi_{2,0}^{(6)}$ (Fig. 13b) tries to do the same along the east-west direction. $\psi_{2,2}^{(6)}$ (Fig. 13j) tries to differentiate the intersection density around the northeast region of Toronto. On the other hand, there are coarse scale basis vectors with $j = j_{\max} = 43$, which are in fact the global Laplacian eigenvectors, i.e., Fig. 13a, e, k, n. It is not surprising that these coarse scale basis vectors were selected as a part of the VM-NGWP best basis considering that the global graph Laplacian eigenbasis performed quite well on this graph signal as shown in Fig. 12b.

Now, let us analyze the pedestrian volume data measured at the street intersections as shown in Fig. 14a, which is highly localized around the specific part of the downtown region (the dense region in the lower middle section) of the street graph. Fig. 14b shows the approximation errors of various methods. From Fig. 14b, we observe the following: 1) the eGHWT best basis clearly outperformed all the other methods; 2) the HGLET best basis and the GHWT c2f best basis followed the eGHWT best basis; 3) the graph Haar basis and the GHWT f2c best basis were next best performers; 4) The VM-NGWP best basis followed those five while the PC-NGWP best basis was the distant seventh performer; and 5) the global bases such as the graph Laplacian eigenbasis and the graph Walsh based were the worst performers. Considering the non-smooth and highly localized nature of the input signal, it is not surprising that the global bases did not perform well and that the non-smooth local bases (the eGHWT, the GHWT best bases, the graph Haar basis) and the basis vectors whose supports strictly follow the partition pattern of the primal graph (the HGLET best basis) had
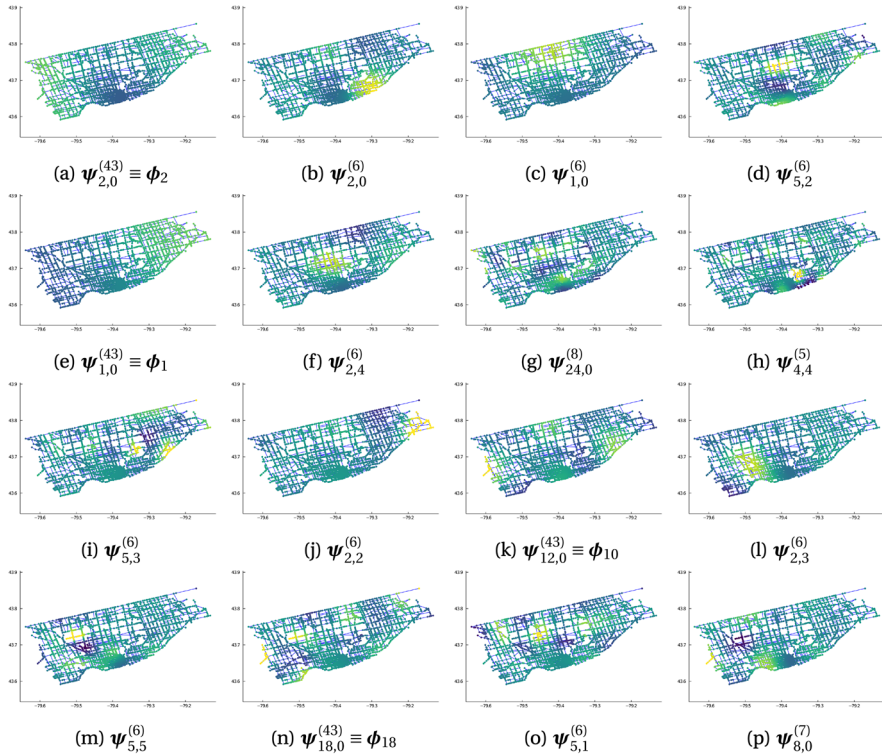
Fig. 13 Sixteen most significant VM-NGWP best basis vectors (the DC vector not shown) for street intersection density data on the Toronto street map. The basis vector amplitudes within $(-0.075, 0.075)$ are mapped to the viridis colormap (Color figure online)

an edge over the NGWP best bases that contain smooth basis vectors whose supports may spread among neighboring regions.

In order to examine the performance difference between the VM-NGWP best basis and the PC-NGWP best basis, we display their 16 most significant basis vectors in Figs. 15 and 16, respectively. We note that the top VM-NGWP best basis vectors exhibit local to intermediate spatial scales. The basis vectors with $j = 1$ (Fig. 15d, l, m, n, o, p) are highly localized at certain nodes within the dense downtown region while the basis vectors with $j = 2, 3, 4$ (Fig. 15b, c, f, g, h, i, j, k) try to characterize the pedestrian volume within the downtown region and its neighbors as oriented edge detectors. The top basis vector $\boldsymbol{\psi}_{0,2}^{(4)}$ in Fig. 15a works as a local averaging operator around the downtown region. On the other hand, the top PC-NGWP best basis vectors are more localized than those of the VM-NGWP best basis vectors. As one can see from Fig. 16, there are neither medium nor coarse scale basis vectors in these top 16 basis vectors. The reason behind these performance difference between the VM-NGWP and the PC-NGWP is the following. The VM-NGWP best basis for this graph signal turned out to be "almost" the graph Shannon wavelet basis with the deepest level $J = 4$, i.e., the basis for the union of the following subspaces: $V_0^{\star(4)}$, $V_8^{\star(7)}$,

(a) Pedestrian volume data measured at the street intersections

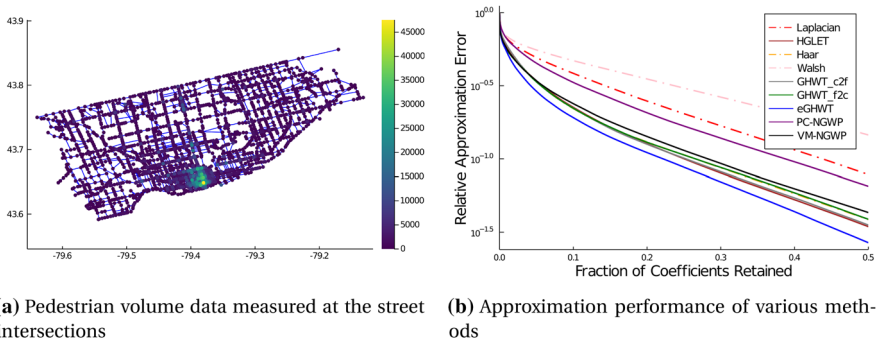(b) Approximation performance of various methods

Fig. 14 The pedestrian volume graph signal on the Toronto street network (**a**); the results of our approximation experiments (**b**) (Color figure online)

$V_{18}^{\star(43)} (= \{\boldsymbol{\phi}_{18}\})$, $V_{19}^{\star(43)} (= \{\boldsymbol{\phi}_{19}\})$, $V_5^{\star(6)}$, $V_3^{\star(5)}$, $V_1^{\star(3)}$, $V_1^{\star(2)}$, and $V_1^{\star(1)}$. Note that $V_8^{\star(7)} \cup V_{18}^{\star(43)} \cup V_{19}^{\star(43)} \cup V_5^{\star(6)} \cup V_3^{\star(5)} = V_1^{\star(4)}$, hence this is "almost" the graph Shannon wavelet basis with $J = 4$, which is the basis for the union $V_0^{\star(4)} \cup V_1^{\star(4)} \cup V_1^{\star(3)} \cup V_1^{\star(2)} \cup V_1^{\star(1)}$. Hence, this VM-NGWP best basis should behave similarly to that graph Shannon wavelet basis (except the mother wavelet vectors at level $j = 4$). In particular, it does not contain oscillatory basis vectors of large scale that are not really necessary to approximate this highly localized pedestrian volume data. On the other hand, the PC-NGWP best basis turned out to be the graph Shannon wavelet basis with $J = 1$, i.e., the basis for the subspaces $V_0^{\star(1)}$ and $V_1^{\star(1)}$. Since the pedestrian volume data is quite non-smooth and localized, $\boldsymbol{\delta}$-like basis vectors with scale $j = 1$ in the PC-NGWP dictionary tend to generate sparser coefficients, i.e., having a small number of large magnitude coefficients with many negligible ones. Therefore, the best basis algorithm with ℓ^1-norm ends up favoring those fine scale basis vectors in the PC-NGWP best basis for this graph signal.

6 Discussion

In this article, we proposed two ways to construct graph wavelet packet dictionaries that fully utilize the natural dual domain of an input graph: the VM-NGWP and the PC-NGWP dictionaries. Then, using two different graph signals on each of the two different graphs, we compared their performance in approximating a given graph signal against our previous multiscale graph basis dictionaries, such as the HGLET, GHWT, and eGHWT dictionaries, which include the graph Haar and the graph Walsh bases. Our proposed dictionaries outperformed the others on locally smooth graph signals, and performed reasonably well for a graph signal sampled on an image containing oriented anisotropic texture patterns. On the other hand, our new dictionaries were beaten by the eGHWT on the non-smooth and localized graph signal. One of the potential reasons for such a behavior is the fact that our dictionaries are a direct generalization of the "Shannon" wavelet packet dictionaries, i.e., their "frequency" domain support is localized and well-controlled while the "time" domain support is not compact. In
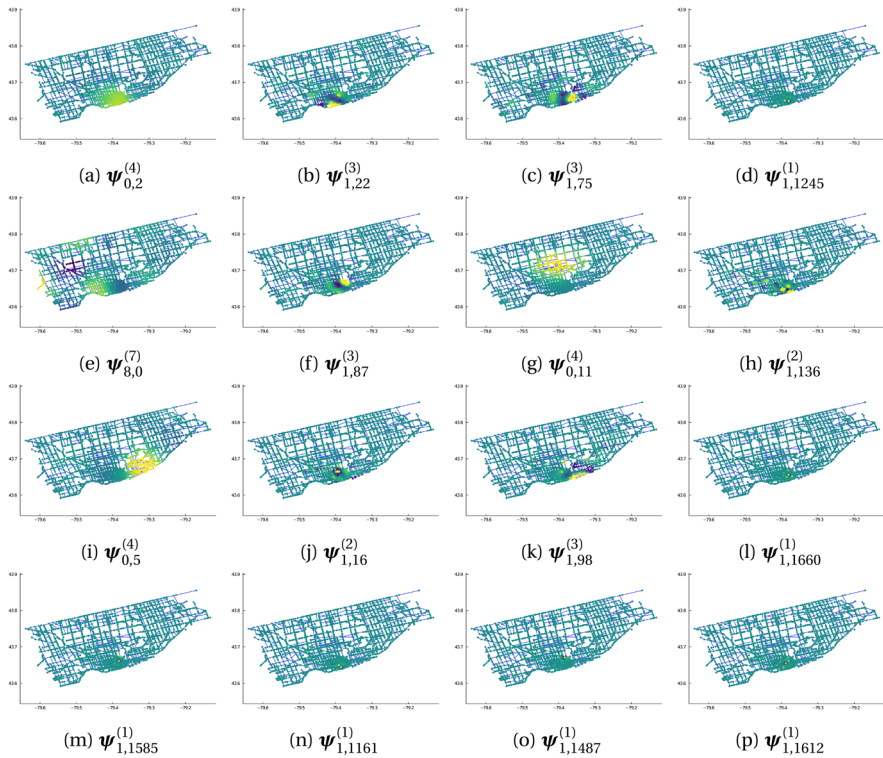
Fig. 15 Sixteen most significant VM-NGWP best basis vectors for pedestrian volume data on the Toronto street map. The basis vector amplitudes within $(-0.075, 0.075)$ are mapped to the viridis colormap (Color figure online)

order to improve the performance of our dictionaries for such non-smooth localized graph signals, we need to bipartition $G^\star$ recursively but *smoothly with overlaps*, which may lead to a graph version of the *Meyer wavelet packet dictionary* [34, Sect. 7.2.2, 8.4.2], whose basis vectors are more localized in the "time" domain than those of the Shannon wavelet packet dictionary. In fact, it is interesting to investigate such smooth partitioning with overlaps not only on $G^\star$ but also on G itself since it may lead to the graph version of the local cosine basis dictionary [9], [34, Sect. 8.4.3].

We also note that the VM-NGWP dictionary performed generally better than the PC-NGWP dictionary for the graph signals we have examined. One of the possible reasons is the use of the explicit localization procedure, i.e., the varimax rotation, in the former; the latter allows one to try to "pinpoint" a particular primal node where the basis vector should concentrate, but the MGSLp procedure unfortunately shuffles and slightly delocalizes the basis vectors after orthogonalization. On the other hand, the difference in their computational costs is just a constant in $O(N^3)$ operations. Hence, it is important to investigate how to reduce the computational complexity in both cases. One such possibility is to use only the first N_0 graph Laplacian eigenvectors with $N_0 \ll N$. Clearly, one cannot represent a given graph signal precisely with N_0
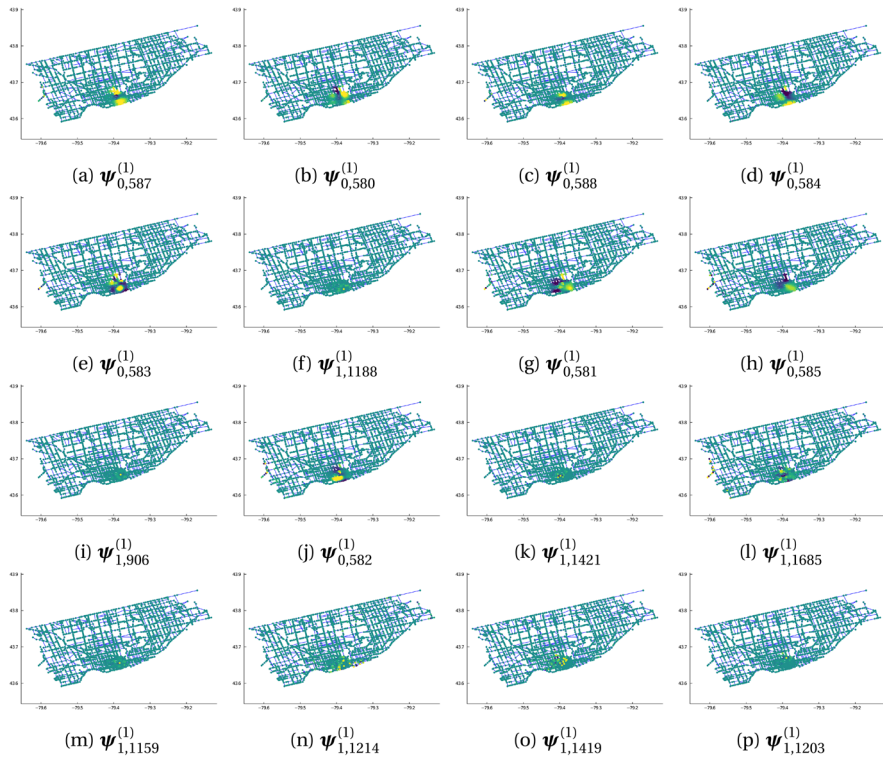
Fig. 16 Sixteen most significant PC-NGWP best basis vectors for pedestrian volume data on the Toronto street map. The basis vector amplitudes within $(-0.075, 0.075)$ are mapped to the viridis colormap (Color figure online)

eigenvectors, but this scenario may be acceptable for certain applications including graph signal clustering, classification, and regression. Of course, it is of our interest to investigate whether we can come up with faster versions of the varimax rotation algorithm and the MGSLp algorithm, which forms one of our future research projects.

Finally, we would like to emphasize that the natural dual domain $G^\star$ can be used in applications beyond dictionary construction. Such applications include: graph cuts and spectral clustering [41,53] to move beyond noted limitations to using the first few eigenvectors [39]; graph visualization and embeddings [1] to represent embeddings with lower distortion [24] as was done in [26]; and anomaly detection through spectral methods [12,36] by going beyond the first few eigenvectors [3,4]. It is interesting to investigate going beyond the initial set of eigenvectors with small eigenvalues in a method informed by $G^\star$, and the effect of $G^\star$ on such methods.

A Varimax Rotation

The algorithm of varimax rotation we adopted and used in this article is the so-called Basic Singular Value (BSV) algorithm proposed by Jennrich [23]. The algorithm in Eq. (7) can be summarized as follows:

(0) Initialize an orthogonal rotation matrix R.
(1) Compute df/dR, where f is the objective function defined in Eq. (7), and df/dR is the matrix of partial derivatives of f at R.
(2) Find the singular value decomposition $U\Sigma V^*$ of df/dR.
(3) Replace R by UV^* and go to (1) or stop.

Algorithm 1 below describes the details. Jennrich also showed that under certain general conditions, this algorithm converges to a stationary point from any initial estimate. The BSV algorithm seems to be the standard varimax rotation algorithm available in many packages, e.g., MATLAB®,[5] R, etc.

Input: Full column rank input matrix whose columns to be rotated $A \in \mathbb{R}^{N \times m}$ ($m \leq N$); maximum number of iteration steps maxit (default value: 1000); relative tolerance tol (default value: 1e-12)
Output: Rotated matrix $B \in \mathbb{R}^{N \times m}$

```
B = A                                    // initialize the output matrix
S = 0                                    // initialize the nuclear norm
for i from 1 to maxit do
    S₀ = S [U, Σ, V] = svd(Aᵀ · (N · B∘3 − B · diag(Bᵀ · B)))     // B∘3 := B ∘ B ∘ B
                       // where ∘ is the Hadamard (entrywise) product
    T = U · V*             // update the orthogonal rotation matrix
    S = trace(Σ) B = A · T              // update the rotated matrix
    if |S − S₀|/S < tol then
        break                 // stop when S does not change much
    end
end
return B
```

Algorithm 1: The Varimax Rotation Algorithm

B Modified Gram-Schmidt with ℓ^p Pivoting Orthogonalization

We have implemented a simplified version of the modified Gram-Schmidt with mixed ℓ^2-ℓ^p ($0 < p < 2$) pivoting algorithm described and used in [7]. Algorithm 2 below describes the details. Our version skips the the step of computing the largest ℓ^2 norm and picking the parameter λ (a notation used in [7]) to increase the numerical stability. Instead, we directly set up a tolerance parameter, tol, for robustness. On the other hand, we keep the ℓ^p ($0 < p < 2$) pivoting procedure in MGS (i.e., always perform the orthogonalization process of the vector with minimum ℓ^p-norm in the candidate pool), which nicely preserves the sparsity of the obtained wavelet-like vectors after the orthogonalization process. The MGSLp algorithm is summarized as follows.

Input: List of unit vectors $v = [v_1, \ldots, v_m] \in \mathbb{R}^{N \times m}$; norm parameter $0 < p < 2$ (default value: 1); error tolerance tol (default value: 1e-12)
Output: List of orthonormal vectors $q = [q_1, \ldots, q_r] \in \mathbb{R}^{N \times r}$ where $r = \text{rank}(v)$
$q = \emptyset$ // initialize the output list
$w = [\|v_1\|_p, \ldots, \|v_m\|_p]$
for i *from* 1 *to* m **do**
    $k = i - 1 + \text{findmin}(w)$ // find the minimum ℓ^p-norm index
    $\text{swap}(v_i, v_k)$ // pivoting
    if $\|v_i\|_2 < \text{tol}$ **then**
        break   // check linear dependency
    end
    $\tilde{v} = v_i / \|v_i\|_2$
    $w = \emptyset$ // re-initialize the ℓ^p-norm vector
    for j *from* $i + 1$ *to* m **do**
        $v_j = v_j - (\tilde{v}^\mathsf{T} \cdot v_j)\tilde{v}$
        $w = \text{append}(w, \|v_j\|_p)$
    end
    $q = \text{append}(q, \tilde{v})$
end
return q

Algorithm 2: Modified Gram-Schmidt Orthogonalization with ℓ^p pivoting

References

1. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**, 1373–1396 (2003)
2. Borg, I., Groenen, P.J.F.: Modern Multidimensional Scaling: Theory and Applications, 2nd edn. Springer, New York (2005)
3. Cheng, X., Mishne, G.: Spectral embedding norm: Looking deep into the spectrum of the graph Laplacian. SIAM J. Imaging Sci. **13**, 1015–1048 (2020)
4. Cloninger, A., Czaja, W.: Eigenvector localization on data-dependent graphs, in 2015 International Conference on Sampling Theory and Applications (SampTA), IEEE, (2015), pp. 608–612
5. Cloninger, A., Steinerberger, S.: On the dual geometry of Laplacian eigenfunctions. Experimental Mathematics (2018). https://doi.org/10.1080/10586458.2018.1538911
6. Coifman, R.R., Gavish, M.: Harmonic analysis of digital data bases, in Wavelets and Multiscale Analysis: Theory and Applications, J. Cohen and A. I. Zayed, eds., Applied and Numerical Harmonic Analysis, Boston, MA, (2011), Birkhäuser, pp. 161–197

7. Coifman, R.R., Maggioni, M.: Diffusion wavelets. Appl. Comput. Harmon. Anal. **21**, 53–94 (2006)
8. Coifman, R.R., Meyer, Y.: Nouvelles bases orthonormées de $L^2(\mathbb{R})$ ayant la structure du systèm de Walsh, preprint. Yale University, New Haven, CT, Dept. of Mathematics (Aug. 1989)
9. Coifman, R.R., Meyer, Y.: Remarques sur l'analyse de Fourier à fenêtre, Comptes Rendus Acad. Sci. Paris, Série I, 312 (1991), pp. 259–261
10. Coifman, R.R., Wickerhauser, M.V.: Entropy-based algorithms for best basis selection. IEEE Trans. Inform. Theory **38**, 713–718 (1992)
11. Daubechies, I.: Ten Lectures on Wavelets. CBMS-NSF Regional Conference Series in Applied Mathematics, vol. 61. SIAM, Philadelphia, PA, USA (1992)
12. Egilmez, H.E., Ortega, A.: Spectral anomaly detection using graph-based filtering for wireless sensor networks. In: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, (2014), pp. 1085–1089
13. Gavish, M., Nadler, B., Coifman, R.R.: Multiscale wavelets on trees, graphs and high dimensional data: theory and applications to semi supervised learning. In: Proc. 27th Intern. Conf. Machine Learning, J. Fürnkranz and T. Joachims, eds., Haifa, Israel, (2010), Omnipress, pp. 367–374
14. Girault, B., Ortega, A.: What's in a frequency: new tools for graph Fourier transform visualization, (2019), arXiv:1903.08827 [eess.SP]
15. Godsil, C., Royle, G.: Algebraic Graph Theory. Graduate Texts in Mathematics, vol. 207. Springer, New York (2001)
16. Hammond, D.K., Vandergheynst, P., Gribonval, R.: Wavelets on graphs via spectral graph theory. Appl. Comput. Harmon. Anal. **30**, 129–150 (2011)
17. Herley, C., Xiong, Z., Ramchandran, K., Orchard, M.T.: Joint space-frequency segmentation using balanced wavelet packet trees for least-cost image representation. IEEE Trans. Image Process. **6**, 1213–1230 (1997)
18. Irion, J., Saito, N.: The generalized Haar-Walsh transform. In: Proc. 2014 IEEE Workshop on Statistical Signal Processing, (2014), pp. 472–475
19. Irion, J., Saito, N.: Hierarchical graph Laplacian eigen transforms. JSIAM Lett. **6**, 21–24 (2014)
20. Irion, J., Saito, N.: Applied and computational harmonic analysis on graphs and networks, in Wavelets and Sparsity XVI, Proc. SPIE 9597, M. Papadakis, V. K. Goyal, and D. Van De Ville, eds., (2015). Paper # 95971F
21. Irion, J., Saito, N.: Efficient approximation and denoising of graph signals using the multiscale basis dictionaries, IEEE Trans. Signal and Inform. Process. Netw. **3**, 607–616 (2017)
22. Jaffard, S., Meyer, Y., Ryan, R.D.: Wavelets: Tools for Science & Technology. SIAM, Philadelphia, PA (2001)
23. Jennrich, R.I.: A simple general procedure for orthogonal rotation. Psychometrika **66**, 289–306 (2001)
24. Jones, P.W., Maggioni, M., Schul, R.: Manifold parametrizations by eigenfunctions of the Laplacian and heat kernels. Proc. Natl. Acad. Sci. U.S.A. **105**, 1803–1808 (2008)
25. Kaiser, H.F.: The varimax criterion for analytic rotation in factor analysis. Psychometrika **23**, 187–200 (1958)
26. Kohli, D., Cloninger, A., Mishne, G.: LDLE: Low Distortion Local Eigenmaps, (2021), arXiv:2101.11055 [math.SP]
27. Lee, A., Nadler, B., Wasserman, L.: Treelets–an adaptive multi-scale basis for sparse unordered data. Ann. Appl. Stat. **2**, 435–471 (2008)
28. Leus, G., Segarra, S., Ribeiro, A., Marques, A.G.: The dual graph shift operator: Identifying the support of the frequency domain. (2017). arXiv:1705.08987 [cs.IT]
29. Levie, R., Monti, F., Bresson, X., Bronstein, M.M.: CayleyNets: Graph convolutional neural networks with complex rational spectral filters. IEEE Trans. Signal Process. **67**, 97–109 (2018)
30. Li, H.: Natural graph wavelet packet dictionaries. https://github.com/haotian127/NGWP.jl, (2020)
31. Li, H., Saito, N.: Metrics of graph Laplacian eigenvectors. In: Wavelets and Sparsity XVIII, Proc. SPIE 11138, D. Van De Ville, M. Papadakis, and Y. M. Lu, eds., (2019). Paper #111381K
32. Lindberg, M., Villemoes, L.F.: Image compression with adaptive Haar-Walsh tilings. In: Wavelet Applications in Signal and Image Processing VIII, Proc. SPIE 4119, A. Aldroubi, A. F. Laine, and M. A. Unser, eds., (2000), pp. 911–921
33. Lindenbaum, O., Salhov, M., Yeredor, A., Averbuch, A.: Gaussian bandwidth selection for manifold learning and classification. Data Min. Knowl. Discov. **34**, 1676–1712 (2020)
34. Mallat, S.A.: A Wavelet Tour of Signal Processing, 3rd edn. Academic Press, Burlington, MA (2009)
35. Mathai, A.M., Davis, T.A.: Constructing the sunflower head. Math. Biosci. **20**, 117–133 (1974)

36. Mishne, G., Cohen, I.: Multiscale anomaly detection using diffusion maps. IEEE J. Sel. Top. Signal Process. **7**, 111–123 (2012)

37. Mulaik, S.A.: Foundations of Factor Analysis, 2nd edn. CRC Press, Boca Raton (2010)

38. Murtagh, F.: The Haar wavelet transform of a dendrogram. J. Classif. **24**, 3–32 (2007)

39. Nadler, B., Galun, M.: Fundamental limitations of spectral clustering. In: Advances in Neural Information Processing Systems, (2007), pp. 1017–1024

40. Nakatsukasa, Y., Saito, N., Woei, E.: Mysteries around the graph Laplacian eigenvalue 4. Linear Algebra Appl. **438**, 3231–3246 (2013)

41. Ng, A., Jordan, M., Weiss, Y.: On spectral clustering: analysis and an algorithm. Adv. Neural Inf. Process. Syst. **14**, 849–856 (2001)

42. Ortega, A., Frossard, P., Kovačević, J., Moura, J.M.F., Vandergheynst, P.: Graph signal processing: overview, challenges, and applications. Proc. IEEE **106**, 808–828 (2018)

43. Perraudin, N., Ricaud, B., Shuman, D.I., Vandergheynst, P.: Global and local uncertainty principles for signals on graphs. APSIPA Trans. Signal Image Process. **7**, E3 (2018)

44. Ricaud, B., Shuman, D.I., Vandergheynst, P.: On the sparsity of wavelet coefficients for signals on graphs, in Wavelets and Sparsity XV, Proc. SPIE 8858, D. V. D. Ville, V. K. Goyal, and M. Papadakis, eds., (2013). Paper # 88581L

45. Rodieck, R.W.: The First Steps in Seeing. Sinauer Associates Inc., Sunderland, MA (1998)

46. Rudis, B., Ross, N., Garnier, S.: The viridis color palettes. https://cran.r-project.org/web/packages/viridis/vignettes/intro-to-viridis.html, (2018)

47. Saito, N.: The generalized spike process, sparsity, and statistical independence, in Modern Signal Processing, D. Rockmore and J. D. Healy, eds., vol. 46 of MSRI Publications, Cambridge University Press, (2004), pp. 317–340

48. Saito, N.: Applied harmonic analysis on graphs and networks. Bull. Jpn. Soc. Ind. Appl. Math. **25**, 6–15 (2015). in Japanese

49. Saito, N.: How can we naturally order and organize graph Laplacian eigenvectors?, in Proc. 2018 IEEE Workshop on Statistical Signal Processing, (2018), pp. 483–487

50. Saito, N., Woei, E.: Analysis of neuronal dendrite patterns using eigenvalues of graph Laplacians, JSIAM Lett., 1 (2009), pp. 13–16. Invited paper

51. Saito, N., Woei, E.: On the phase transition phenomenon of graph Laplacian eigenfunctions on trees. RIMS Kôkyûroku **1743**, 77–90 (2011)

52. Shao, Y., Saito, N.: The extended generalized Haar-Walsh transform and applications, in Wavelets and Sparsity XVIII, Proc. SPIE 11138, D. Van De Ville, M. Papadakis, and Y. M. Lu, eds., (2019). Paper #111380C

53. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Machine Intell. **22**, 888–905 (2000)

54. 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**, 83–98 (2013)

55. Shuman, D.I., Ricaud, B., Vandergheynst, P.: Vertex-frequency analysis on graphs. Appl. Comput. Harmon. Anal. **40**, 260–291 (2016)

56. Strang, G.: The discrete cosine transform. SIAM Rev. **41**, 135–147 (1999)

57. Szlam, A.D., Maggioni, M., Coifman, R.R., Bremer, J.C. Jr.: Diffusion-driven multiscale analysis on manifolds and graphs: top-down and bottom-up constructions, in Wavelets XI, Proc. SPIE 5914, M. Papadakis, A. F. Laine, and M. A. Unser, eds., (2005). Paper # 59141D

58. The JuliaGeometry Team, JuliaGeometry: Computational Geometry with Julia. https://github.com/JuliaGeometry, (2021)

59. Thiele, C.M., Villemoes, L.F.: A fast algorithm for adapted time-frequency tilings. Appl. Comput. Harmon. Anal. **3**, 91–99 (1996)

60. Vogel, H.: A better way to construct the sunflower head. Math. Biosci. **44**, 179–189 (1979)

61. von Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**, 395–416 (2007)

62. Vondrák, J.: Optimal approximation for the submodular welfare problem in the value oracle model, in STOC'08, pp. 67–74. ACM, New York (2008)

63. Yamagishi, Y., Sushida, T.: Archimedean Voronoi spiral tilings. J. Phys. A **51**, 045203 (2018)