JSIAM Letters Vol.** (****) pp.1– ©**** Japan Society for Industrial and Applied Mathematics

Hierarchical graph Laplacian eigen transforms

Jeff Irion[1] and Naoki Saito[1]

[1] Department of Mathematics, University of California, Davis, CA 95616 USA

E-mail *jlirion@math.ucdavis.edu, saito@math.ucdavis.edu*

Received

Abstract

We describe a new transform that generates a dictionary of bases for handling data on a graph by combining recursive partitioning of the graph and the Laplacian eigenvectors of each subgraph. Similar to the wavelet packet and local cosine dictionaries for regularly sampled signals, this dictionary of bases on the graph allows one to select an orthonormal basis that is most suitable to one's task at hand using a best-basis type algorithm. We also describe a few related transforms including a version of the Haar wavelet transform on a graph, each of which may be useful in its own right.

Keywords graph Laplacian eigenvectors, Fiedler vectors, spectral graph partitioning, a dictionary of orthonormal bases, wavelet-like transforms on graphs

Research Activity Group Wavelet Analysis

1. Introduction

For signal processing on regular domains, wavelets have both a well-developed theory and a proven track record of success. Accordingly, efforts have been made to extend classical wavelets and wavelet techniques to the ever-expanding realm of data on graphs. Such datasets include structural/morphological data (e.g., tracings of neuronal dendrites), traffic and transportation data, and social networks. The motivation for developing these so-called "second generation wavelets" is simple: to determine whether they afford the same advantages offered by classical wavelets for approximation/compression, denoising, and classification in this more general setting.

However, a key difficulty in extending wavelets to graphs is that we lack the notion of "frequency" in general, i.e., we cannot apply the Littlewood-Paley theory directly. Therefore, a common strategy has been to develop wavelet-*like* transforms rather than true generalizations of classical wavelets; see e.g., [1–9]. In this article, we propose a new redundant transform for data on graphs, along with two variations, and then show the basis vectors computed on a particular graph.

2. Definitions and notation

Let G be an undirected connected graph, let $V(G)$ and $E(G)$ denote its vertices and edges, respectively, and let $N := |V(G)|$. Let $W(G) = (W_{ij}) \in \mathbb{R}^{N\times N}$ be the symmetric weight matrix of G, where W_{ij} denotes the edge weight between vertices i and j. In an unweighted (i.e., combinatorial) graph, W_{ij} is either 0 or 1, depending on whether there is an edge between the two vertices. By contrast, in a weighted graph, W_{ij} indicates the proximity of vertices i, j or affinity of information measured at i, j. Let $\boldsymbol{f} = (f(1), \ldots, f(N))^{\mathsf{T}} \in \mathbb{R}^N$ be a data vector, where $f(i)$ is the value measured at the vertex i of the graph. Let $\mathbf{1} := (1, \ldots, 1)^{\mathsf{T}} \in \mathbb{R}^N$.

A standard technique for working with data on a graph is to utilize the eigenvectors of the *Laplacian matrix* of the graph, which is defined as $L(G) := D(G) - W(G)$, where $D(G) = \mathrm{diag}(d_i)$ is the (diagonal) degree matrix with $d_i := \sum_j W_{ij}$. Alternatively, we may use the *random-walk normalized Laplacian*, which is defined as $L_{\mathrm{rw}}(G) := D(G)^{-1}L(G) = I - D(G)^{-1}W(G)$. The eigenvectors of both $L(G)$ and $L_{\mathrm{rw}}(G)$ form a basis of $\mathbb{R}^N$ and can thus be used for representation, approximation, and analysis of data on G. The simple *path graph* P_N consisting of N vertices provides an important insight for the development of our new transform. As pointed out in [10], the eigenvectors of $L(P_N)$ are nothing but the Discrete Cosine Transform (DCT) Type II, which are used in the JPEG image compression standard. In general, it is difficult to know the essential support of the Laplacian eigenvectors a priori, which strongly depends on the structure of the graph: sometimes they are completely global, like those of P_N, while the other times they may be quite localized, as shown in [10]. Hence, it is worth controlling the support of the eigenvectors explicitly.

3. Hierarchical graph Laplacian eigen transform (HGLET)

We now introduce our Hierarchical Graph Laplacian Eigen Transform (HGLET). First, we compute the complete set of eigenvectors of $L(G)$: $\boldsymbol{\phi}^0_{0,0}, \boldsymbol{\phi}^0_{0,1}, \ldots, \boldsymbol{\phi}^0_{0,N-1}$ with corresponding eigenvalues $0 = \lambda^0_{0,0} < \lambda^0_{0,1} \leq \cdots \leq \lambda^0_{0,N-1}$. As this is a multiscale transform, we adopt the notation $(\lambda^j_{k,l}, \boldsymbol{\phi}^j_{k,l})$ for the eigenpairs, with j denoting the level (or depth) of the partition, k denoting the region number on level j, and l indexing the eigenvectors for region k on level j. Then we partition the graph into two disjoint subgraphs (or regions) according to the sign of the Fiedler vector, $\boldsymbol{\phi}^0_{0,1}$. Partitioning the graph in this manner is supported by the theory discussed in [11]. Furthermore, the Fiedler vector of $L(G)$ (or $L_{\mathrm{rw}}(G)$) is the solution of the relaxed RatioCut (or Normalized Cut)

minimization problem; see e.g., [12] and the references therein.

Let G_0^1 and G_1^1 be the two disjoint regions of G obtained by this partitioning process; note $V(G) = V(G_0^1) \cup V(G_1^1)$ but $E(G) \supsetneq E(G_0^1) \cup E(G_1^1)$. From here we repeat the process recursively. The whole process can be summarized as follows:

Algorithm 1 (HGLET)

Step 0: Set $G_0^0 = G$ and $N_0^0 = N = |V(G)|$; initialize $K^0 = 1$ and $K^1 = 0$; set $j = 0$ and $k = 0$.

Step 1: Construct the Laplacian matrix $L(G_k^j)$.

Step 2: Compute its eigenvectors, $\{\boldsymbol{\phi}_{k,l}^j\}_{l=0}^{N_k^j-1}$.

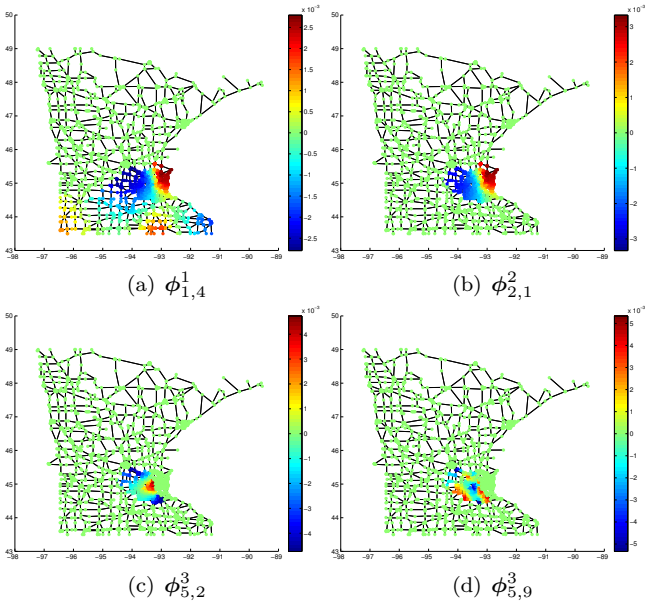
Step 3: If $N_k^j > 1$, **then** partition G_k^j by the sign of the Fiedler vector $\boldsymbol{\phi}_{k,1}^j$ into $G_{K^{j+1}}^{j+1}$ and $G_{K^{j+1}+1}^{j+1}$; set $N_{K^{j+1}}^{j+1} = |V(G_{K^{j+1}}^{j+1})|$, and $N_{K^{j+1}+1}^{j+1} = |V(G_{K^{j+1}+1}^{j+1})|$, and $K^{j+1} = K^{j+1} + 2$; **else** set $G_{K^{j+1}}^{j+1} = G_k^j$, $N_{K^{j+1}}^{j+1} = |V(G_k^j)|$, and $K^{j+1} = K^{j+1} + 1$.

Step 4: If $k+1 < K^j$, **then** set $k = k+1$ and go back to Step 1; **else** go to Step 5.

Step 5: If $|V(G_k^{j+1})| = 1$ for $k = 0, \ldots, K^{j+1}-1$, **then** finish; **else** set $j = j+1$, $k = 0$, $K^{j+1} = 0$, and go back to Step 1.

Several remarks on this algorithm are in order.

- $L(G_k^j)$ in Step 1 above can be replaced by $L_{\mathrm{rw}}(G_k^j)$, which may result in better partitions; see [12].
- Similar to dictionaries of orthonormal bases such as wavelet packet or local cosine dictionaries for regularly-sampled signals, our HGLET yields a highly overcomplete basis set for data measured on the vertices $V(G)$ (after extending each eigenvector $\boldsymbol{\phi}_{k,l}^j$ from its original support $V(G_k^j)$ to $V(G)$ by zeros). There are in fact more than $2^{\lfloor N/2 \rfloor}$ possible bases choosable from this overcomplete set, which allow us to select a basis most suitable for the task at hand via the best-basis type algorithms originally developed for regularly-sampled signals; see e.g., [13].
- The HGLET eigenvectors $\{\boldsymbol{\phi}_{k,l}^j\}_{l=0}^{N_k^j-1}$ for each fixed (j,k) form an orthonormal basis for $\mathbb{R}^{N_k^j}$ if one uses the usual Laplacian matrix $L(G_k^j)$ in Step 1. On the other hand, if one uses the random-walk version $L_{\mathrm{rw}}(G_k^j)$, the resulting eigenvectors are neither mutually orthogonal nor normalized to have unit ℓ^2-norm in general. To generate a set of orthonormal vectors, one only needs to multiply $\sqrt{D(G_k^j)}$ to each such eigenvector; see also [12].
- The actual HGLET transform of a given data vector $\boldsymbol{f} \in \mathbb{R}^N$ can be done either on the fly in Algorithm 1 (i.e., immediately after the orthonormal eigenvectors are computed at each j and k) by taking inner products $\langle \boldsymbol{f}|_{V(G_k^j)}, \boldsymbol{\phi}_{k,l}^j \rangle$, where $\boldsymbol{f}|_{V(G_k^j)} \in \mathbb{R}^{N_k^j}$ is the portion of $\boldsymbol{f}$ supported on $V(G_k^j)$; or one can do the same after Algorithm 1 is completed.



(a) $\boldsymbol{\phi}_{1,4}^1$ (b) $\boldsymbol{\phi}_{2,1}^2$ (c) $\boldsymbol{\phi}_{5,2}^3$ (d) $\boldsymbol{\phi}_{5,9}^3$

Fig. 1. The HGLET eigenvectors computed on the Minnesota road map ($N = 2636$). The random-walk Laplacians were used for this set of experiments with the inverse physical (Euclidean) distances between vertices as edge weights. (a)-(c) show the eigenvectors at different scales covering the densely connected region. (d) shows the eigenfunctions with higher oscillations whose support is the same as that of (c).

- The computational cost of generating the whole set of eigenvectors in the HGLET is clearly $O(N^3)$.
- For an unweighted path graph P_N, the HGLET *exactly* yields a dictionary of the block DCT-II bases; in other words, the HGLET can be viewed as a true generalization of the block DCT-II dictionary.
- Finally, it is easy to see that, over all levels of the partition tree, this scheme yields a total of $N-1$ subgraphs (including the initial graph G) containing two or more vertices; see also Section 4.

Figure 1 shows some HGLET eigenvectors on the Minnesota road network.

4. Variations of HGLET

In this section, we will discuss two variants of the HGLET that are not a member of the HGLET dictionary strictly speaking, i.e., not directly choosable using the best-basis type algorithms.

4.1 Generalized Haar transform

The first one, which we call the *Generalized Haar Transform* (GHT), provides a complete orthonormal basis (i.e., no redundancy) of $\mathbb{R}^N$ comprised of one piecewise constant vector from each subgraph G_k^j containing two or more vertices, along with the constant (i.e., DC) vector on the entire graph G. This variation proceeds as Algorithm 1 except Step 2, which is modified as follows:

Step 2: Compute only the Fiedler vector, $\boldsymbol{\phi}_{k,1}^j$; define

$$\boldsymbol{\psi}_k^j(i) := \begin{cases} 1 & \text{if } \phi_{k,1}^j(i) \geq 0; \\ -r_k^j & \text{if } \phi_{k,1}^j(i) < 0, \end{cases} \quad i = 1, \ldots, N_k^j,$$

where $r_k^j := \dfrac{|\{m \in [1, N_k^j] \,|\, \phi_{k,1}^j(m) \geq 0\}|}{|\{m \in [1, N_k^j] \,|\, \phi_{k,1}^j(m) < 0\}|}$; then set $\boldsymbol{\psi}_k^j = \boldsymbol{\psi}_k^j / \|\boldsymbol{\psi}_k^j\|$.

This modified Algorithm 1 yields a GHT orthonormal basis: $\boldsymbol{\varphi}_0^0 \cup \{\{\boldsymbol{\psi}_k^j\}_{k=0}^{K^j-1}\}_{j=0}^{J}$, where $\boldsymbol{\varphi}_0^0 = \mathbf{1}/\sqrt{N}$ is the constant vector on G, the $\boldsymbol{\psi}_k^j$ vectors are extended by zeros to $V(G_0^0) \setminus V(G_k^j)$, and J is the deepest level of the partitioning (note that J may be larger than $\log_2 N$ in general, unlike the regularly-sampled signals). We note that in this GHT, each of the $N-1$ subgraphs G_k^j with $|V(G_k^j)| \geq 2$ contributes a single $\boldsymbol{\psi}_k^j$ basis vector to the GHT basis, with the initial graph G also contributing $\boldsymbol{\varphi}_0^0$. This is quite a contrast to the general HGLET dictionary case because not all the subgraphs generated in Algorithm 1 contribute their eigenvectors to a basis chosen by the best-basis type algorithm from this dictionary.

The computational cost for generating this GHT is $O(N \log_2 N)$, which should be contrasted with the full HGLET case. This speed up is mainly due to the fact that we only need to compute one eigenvector in Step 2 of the GHT algorithm.

4.2 Orthonormalized hierarchical Fiedler transform

Each eigenvector $\boldsymbol{\phi}_{k,l}^j$ in the HGLET dictionary is a discretized version of a function that is "continuous" *within* its support $V(G_k^j)$. On the other hand, the basis vectors in the GHT are all binary-valued (i.e., "discontinuous") except the DC vector $\boldsymbol{\varphi}_0^0$. Hence it is natural to consider a smoother version of the GHT. Here, we propose the *Orthonormalized Hierarchical Fiedler Transform* (OHFT), which proceeds similarly to the GHT with the following modification and addition:

Step 2: Compute only the Fiedler vector $\boldsymbol{\phi}_{k,1}^j$; then set $\boldsymbol{\psi}_k^j := \boldsymbol{\phi}_{k,1}^j$.

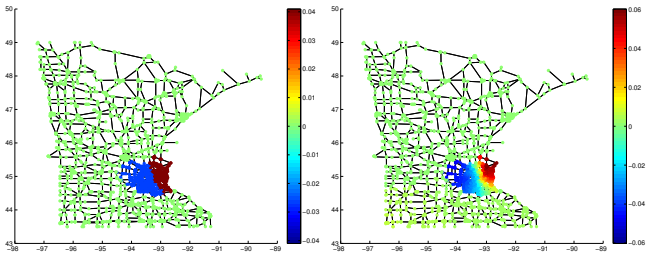
Step 6: Form $\widetilde{\boldsymbol{\psi}}_k^j$ by extending each $\boldsymbol{\psi}_k^j$ (except $j = 0$) by zeros to $V(G_0^0) \setminus V(G_k^j)$; define $\boldsymbol{\varphi}_0^0 := \mathbf{1}/\sqrt{N}$; form a matrix $\Psi := \left[\boldsymbol{\varphi}_0^0 \,\middle|\, \boldsymbol{\psi}_0^0 \,\middle|\, \widetilde{\boldsymbol{\psi}}_0^1 \,\middle|\, \cdots \,\middle|\, \widetilde{\boldsymbol{\psi}}_{K^J-1}^J\right] \in \mathbb{R}^{N \times N}$ where J is the deepest level of the partitions; finally, orthonormalize the columns of Ψ using the QR factorization.

Step 6 is necessary to form an orthonormal basis since the extended vectors $\widetilde{\boldsymbol{\psi}}_k^j$ are not mutually orthogonal in general.

We note that the OHFT provides a single orthonormal basis for $\mathbb{R}^N$, and its basis vectors are continuous within their original support. However, due to the orthogonalization procedure in Step 6, the support of each basis vector is extended beyond its original support by nonzero values, and moreover, this extension by orthogonalization may not provide continuous extension to the outside of the original support. We are currently investigating whether we can provide smoother and continuous extensions while keeping the orthogonality.

The cost of computing the OHFT is $O(N^3)$ due to the orthogonalization procedure in Step 6.

Figure 2 compares a GHT basis vector and the corresponding OHFT basis vector.



(a) $\boldsymbol{\psi}_2^2$ (GHT) (b) $\boldsymbol{\psi}_2^2$ (OHFT)

Fig. 2. Comparison between one of the GHT basis vectors and the corresponding OHFT basis vector on the MN road map. We again used the random-walk Laplacian, similar to Fig. 1.

5. Discussion

The purpose of this article has been to introduce our newly developed HGLET method and its variations.

In this section, let us first briefly discuss the relation of our work to the previous works. First of all, the hierarchical partitioning of a graph using the Fiedler vectors is a natural idea and obviously not new. For example, Simon [14] discussed such a recursive bi-partitioning of a graph. However, his aim was to create hierarchical (bi)partitions of unstructured grids for the purpose of parallel processing, and he was concerned about neither basis constructions nor data analysis.

There are several constructions of wavelet-like transforms on a graph. Diffusion wavelets [1] are based on the bottom-up approach using the diffusion/random walk on a graph, which have been generalized to wavelet packets in [2]. By contrast, our HGLET and its variants utilize the top-down approach, and can be viewed as a simple generalization of the classical block DCT-II dictionary.

The spectral graph wavelet transform (SGWT) [7] directly applies the Littlewood-Paley theory by viewing the eigenvalues and eigenvectors of the *global* graph Laplacian as the "frequencies" and "Fourier modes," respectively. However, for a general graph, the eigenvalue indices cannot be viewed as natural frequencies of the graph, and moreover some eigenvectors of $L(G)$ may be localized [10]. In other words, one cannot explicitly control localization properties of such SGWT wavelets, which may lead to unexpected problems.

Jansen et al. developed a wavelet-like transform for signals on graphs by generalizing the classical lifting scheme [5]. Since their method proceeds vertex-by-vertex, a discrete (e.g., dyadic) notion of scale no longer exists, which is quite a contrast to our HGLETs.

Rustamov recently constructed two different wavelet-like transforms on graphs. The first one [8] is based upon the average-interpolation wavelets and also uses a top-down partitioning, like our HGLETs. However, it is fundamentally different from ours due to the average-interpolation procedure. The second one [9] used the lifting scheme whose update and prediction operators are adaptively learned from a given set of signals so that the resulting wavelet coefficients of a signal belonging to the same signal class become sparse. The aim of this adapted wavelet construction is the same as that of the best basis selected from the full HGLET dictionary, but

these two constructions are quite different.

We also acknowledge that the GHT is similar to those found in [3,4,6]. The differences are that [3] and [4] utilize bottom-up clustering methods whereas our transforms use a top-down partitioning; [6] assumes that the partition tree has already been computed and is provided as an input.

Finally, after our work for this article was completed, we noticed the article by Szlam et al. [15], which provided the closest idea to ours. In that article, the authors proposed a top-down approach using the Laplacian eigenfunctions satisfying the Neumann boundary condition, briefly mentioned the construction of the Haar basis, and then proposed a generalization of the local cosine dictionary to manifolds and graphs. The construction part of their Haar basis seems to be identical to ours, although they did not describe their algorithm in detail; in particular, they never explicitly mentioned the conversion of the Fiedler vectors (or the Neumann eigenfunctions in their terminology) to their Haar basis vectors. As for their construction of the local cosine dictionary on manifolds and graphs, they transported the folding/unfolding operators on the regular lattice to the general manifold and graph setting. Considering our experiences of the local cosine dictionary on the regular lattice [16], such generalized local cosines may not work well in practice. In fact, such folding/unfolding operations may be unnecessary or even harmful for applications. Moreover, they did not notice that the usual graph Laplacian eigenvectors are the true generalization of DCT Type II.

Much work remains to be done in order to fine tune these methods and apply them to cutting-edge problems ranging from data approximation and denoising to classification and regression on graphs. Beyond such applications, we would like to mention two possible extensions of the HGLET. The first one is to develop the *Generalized Haar-Walsh Transform*, which is a generalization of the Haar-Walsh dictionary to the graph setting, and which includes the GHT as just one possible basis. First, we perform a full partitioning of the graph using Fiedler vectors so that all regions at the finest level consist of a single vertex. The basis vectors at this level are simply Kronecker deltas. From here we perform average and difference operations on the basis vectors corresponding to each pair of children regions to generate the basis vectors of their parent region, and we iterate this process from bottom to top until we reach the root level $j = 0$. This process yields an overcomplete basis set that is a generalization of the Haar-Walsh dictionary.

The second extension is to adopt a more flexible graph partitioning scheme. In our HGLET in this article, we have focused only on using Fiedler vectors to split each subgraph into two smaller subgraphs. However, note that only our OHFT has a crucial dependence on the Fiedler vector, whereas the other transforms simply use the Fiedler vector as a means for partitioning the graph. Each of our transforms could just as easily utilize a different partitioning scheme. Furthermore, it is entirely permissible within the general structure of our transforms to allow G and subsequent subgraphs to be split into an arbitrary non-fixed number of subgraphs; ideally this number of partitions of a given subgraph should be the same as the number of clusters in that subgraph if such clusters are clearly formed. Since graph partitioning is a highly evolving field, it is important that our transforms be independent of the particular choice of hierarchical graph partitioning scheme.

We are currently investigating the above extensions and examining the performance of our HGLETs for various applications including simultaneous segmentation and compression of regularly-sampled signals. We hope to report results of more extensive and challenging experiments in our future article.

Acknowledgments

This research was partially supported by the ONR grant N00014-12-1-0177 and the NDSEG fellowship.

References

[1] R. R. Coifman and M. Maggioni, Diffusion wavelets, Appl. Comput. Harm. Anal., **21** (2006), 53–94.

[2] J. C. Bremer, R. R. Coifman, M. Maggioni, and A. Szlam, Diffusion wavelet packets, Appl. Comput. Harm. Anal., **21** (2006), 95–112.

[3] F. Murtagh, The Haar wavelet transform of a dendrogram, J. Classification, **24** (2007), 3–32.

[4] A. Lee, B. Nadler and L. Wasserman, Treelets—an adaptive multi-scale basis for sparse unordered data, Ann. Appl. Stat., **2** (2008), 435–471.

[5] M. Jansen, G. P. Nason, and B. W. Silverman, Multiscale methods for data on graphs and irregular multidimensional situations, J. R. Stat. Soc. Ser. B, **71** (2008), 97–125.

[6] M. Gavish, B. Nadler and R. Coifman, 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 et al. eds., pp. 367–374, Omnipress, Haifa, 2010.

[7] D. K. Hammond, P. Vandergheynst, and R. Gribonval, Wavelets on graphs via spectral graph theory, Appl. Comput. Harm. Anal., **30** (2011), 129–150.

[8] R. M. Rustamov, Average interpolating wavelets on point clouds and graphs, arXiv:1110.2227v1 [math.FA], 2011.

[9] R. M. Rustamov and L. Guibas, Wavelets on graphs via deep learning, in: Advances in Neural Information Processing Systems, Vol. 26, 2013, to appear.

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

[11] M. Fiedler, A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory, Czechoslovak Math. J., **25** (1975), 619–633.

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

[13] N. Saito, Local feature extraction and its applications using a library of bases, in Topics in Analysis and Its Applications: Selected Theses, R. Coifman ed., pp. 269–451, World Scientific Pub. Co., Singapore, 2000.

[14] H. D. Simon, Partitioning of unstructured problems for parallel processing, Comput. Sys. Eng., **2** (1991), 135–148.

[15] A. D. Szlam, M. Maggioni, R. R. Coifman, and J. C. Bremer, Jr., Diffusion-driven multiscale analysis on manifolds and graphs: top-down and bottom-up constructions, in: Wavelets XI, M. Papadakis et al. eds., Proc. SPIE 5914, Paper # 59141D, 2005.

[16] N. Saito and J.-F. Remy, The polyharmonic local sine transform: A new tool for local image analysis and synthesis without edge effect, Appl. Comput. Harm. Anal., **20** (2006), 41–73.